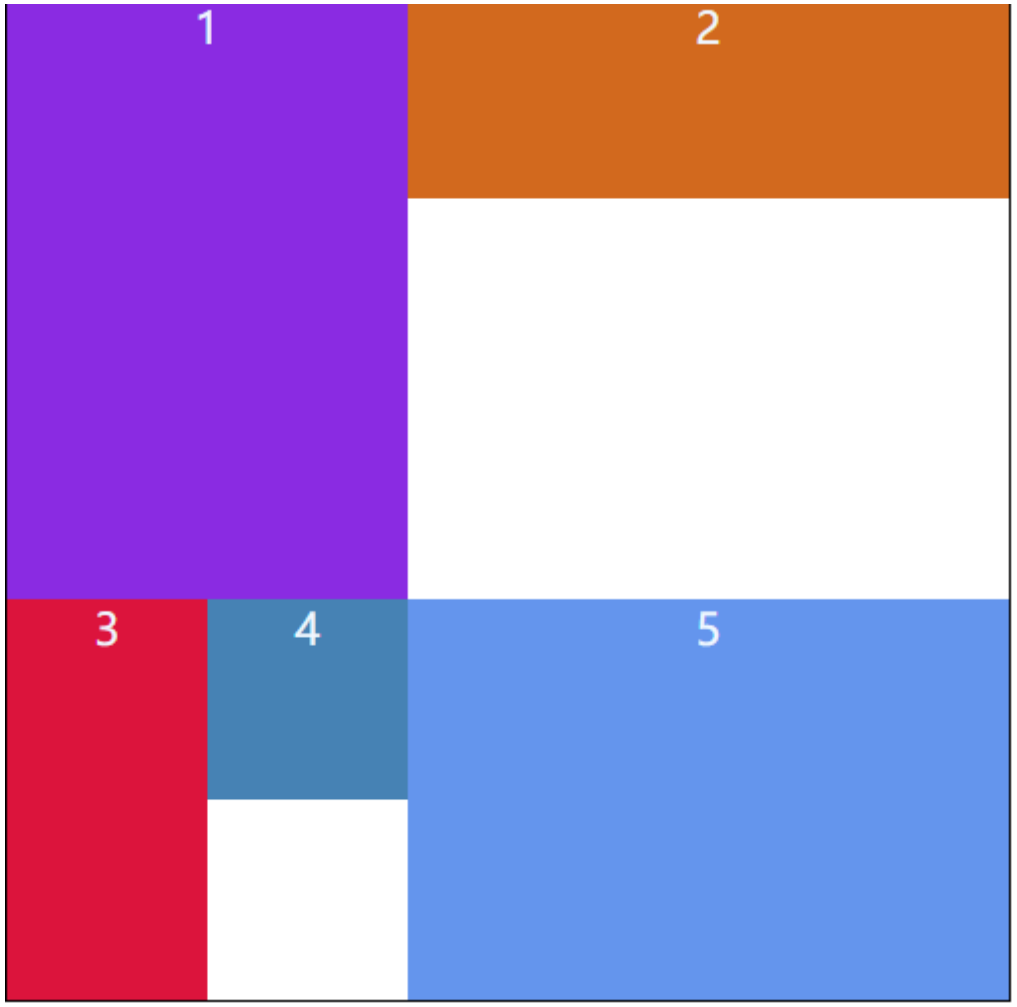
一. Grid网格布局特点：

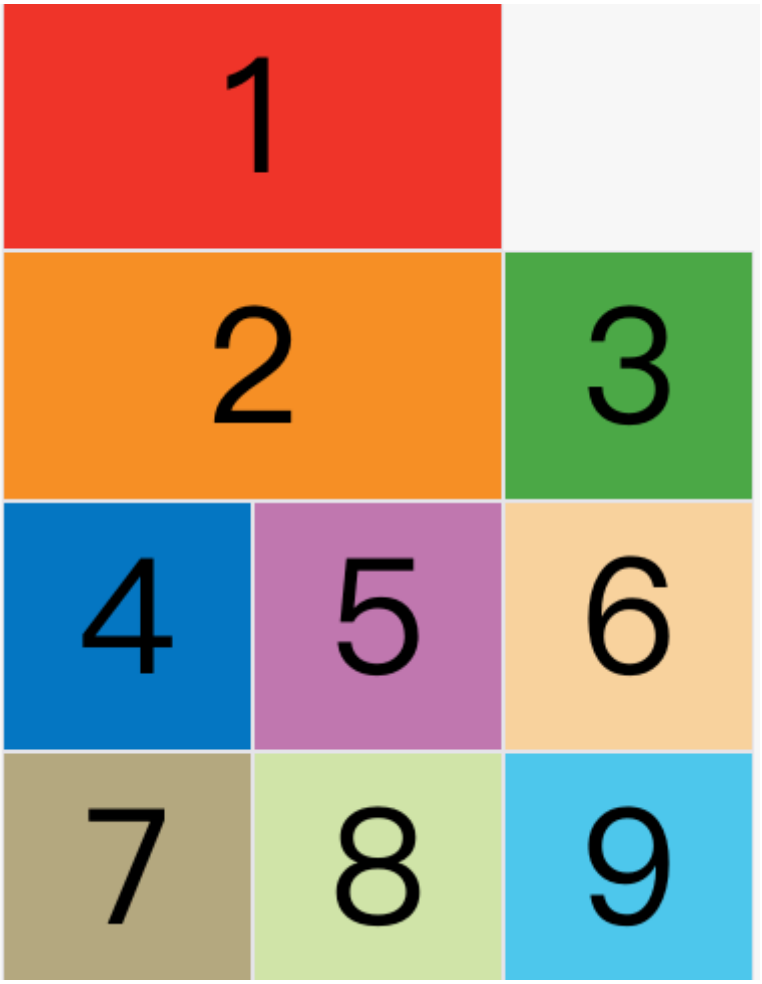
grid布局是将网页划分成一个个网格，可以任意组合不同的网格，做出各种各样的布局

二. Grid网格布局与flex的区别：

flex是流式布局，会按照轴进行排列，如果侧轴方向的大小不统一，则侧轴方向就会有空白处例如：



grid网格布局是按照网格结构排列，



三. 基础概念

1.容器和项目

容器： 设置display：grid的元素

项目： 其下的子元素

如：

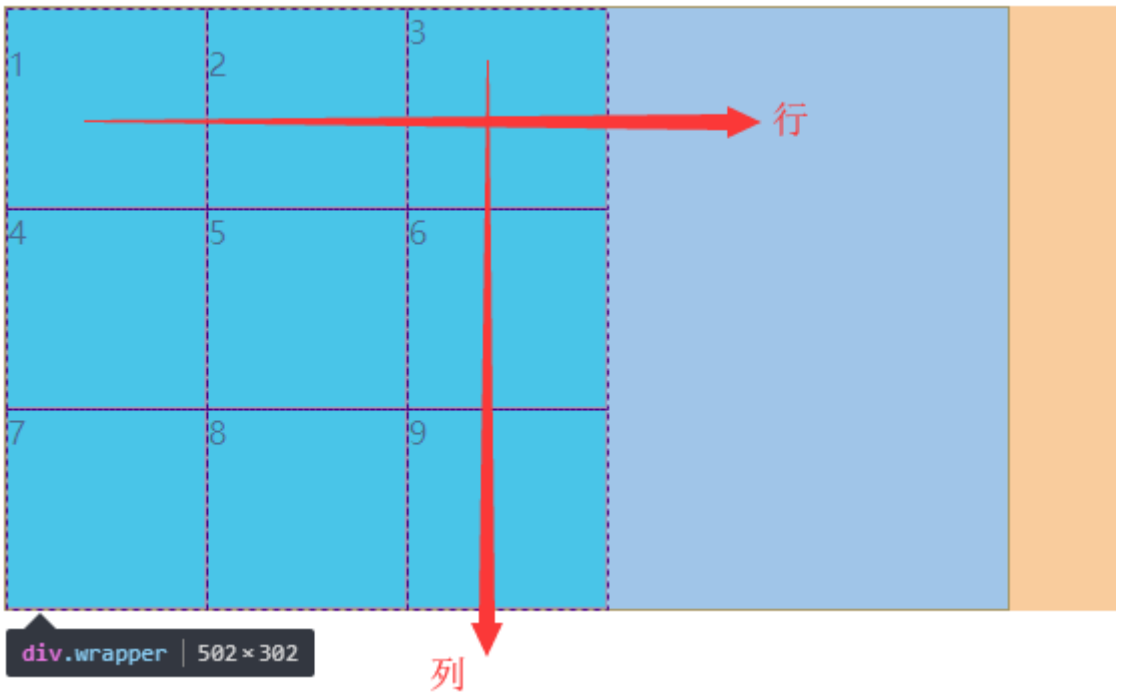
```
    <style>
        .wrapper {
            display: grid;
            border: 1px solid #000;
            grid-template-columns: 100px 100px 100px;
            grid-template-rows: 100px 100px 100px;
            width: 500px;
        }
        .wrapper div {
            background-color: aqua;
            border: 1px solid salmon;

        }
    </style>
</head>
<body>
    <div class="wrapper">
        <div>
            <p>1</p>
        </div>
        <div>
            <p>2</p>
        </div>
        <div>3</div>
        <div>4</div>
        <div>5</div>
        <div>6</div>
        <div>7</div>
        <div>8</div>
        <div>9</div>
    </div>
```

上图代码中wrapper 为容器，wrapper下面的div为项目，p标签是项目内的子元素

2.行和列

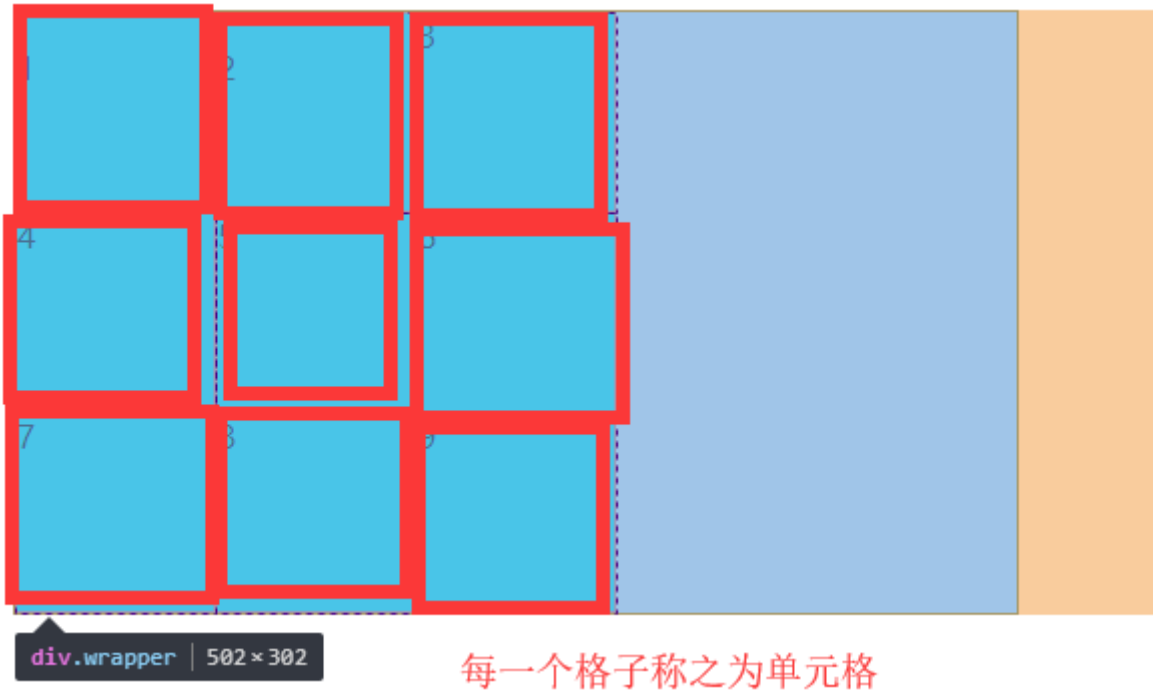
行： 容器内水平区域为行

列： 容器内垂直区域为列



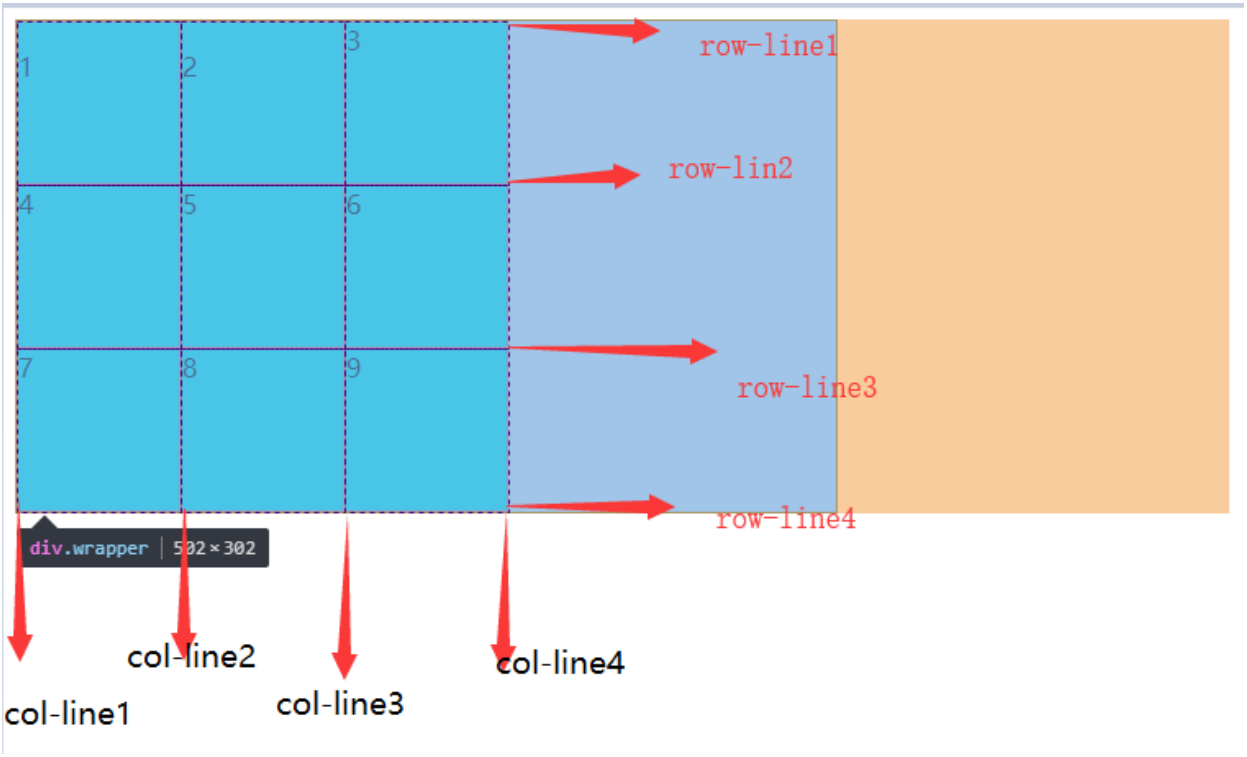
3.单元格

行和列的交叉区域



4. 网格线：

划分网格的线， 水平划分出行，竖直划分出列



四. grid容器内的方法:

1、display: grid/inline-grid; 指定一个容器•采用网格布局

2、grid-template-columns 属性，grid-template-rows 属性

grid-template-columns: 设置列宽

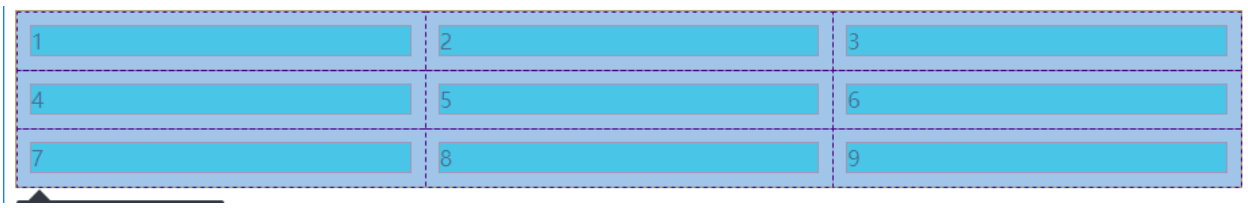
grid-template-rows： 设置行高

案例： 将容器分成3 * 3 的网格结构

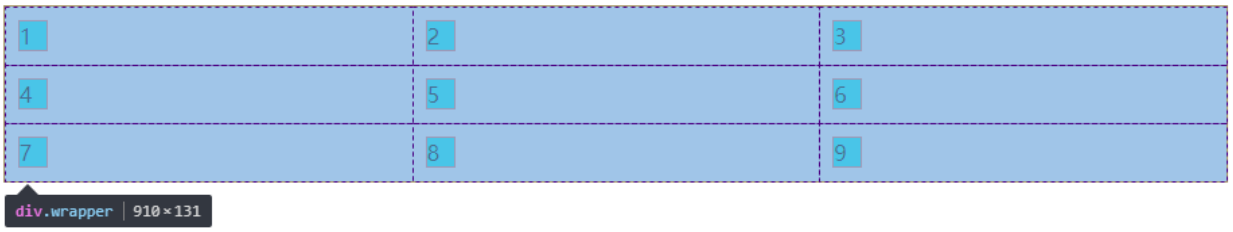
代码：

```
    <style>
        .wrapper {
            display: grid;
            border: 1px solid #000;
            grid-template-columns: 1fr 1fr 1fr;
            grid-template-rows: 1fr 1fr 1fr;
        }
        .wrapper div {
            background-color: aqua;
            border: 1px solid salmon;
            margin: 10px;
        }
    </style>
</head>
<body>
    <div class="wrapper">
        <div>1</div>
        <div>2</div>
        <div>3</div>
        <div>4</div>
        <div>5</div>
        <div>6</div>
        <div>7</div>
        <div>8</div>
        <div>9</div>
    </div>
```

效果：



设置没格子元素的宽度为20px效果为



分析：

3 * 3 的网格结构，默认每个子元素站在一个网格内。

以下代码均设置在容器上即wrapper身上

（1）. grid-template-columns的值:

1） 具体像素 ： 表示每列的宽度

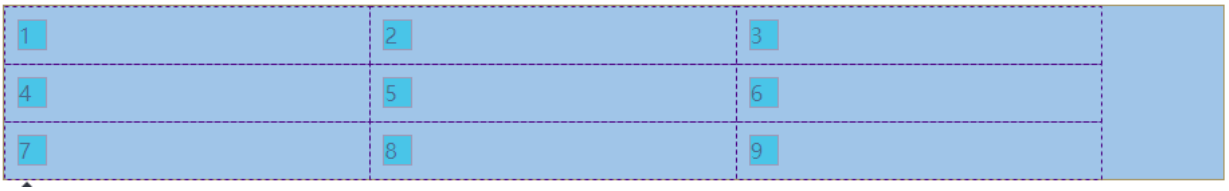
设置grid-template-columns: 100px 100px 100px;

效果：

2）百分比： 表示每列宽度占据容器宽度的百分比

设置grid-template-columns: 30% 30% 30%;

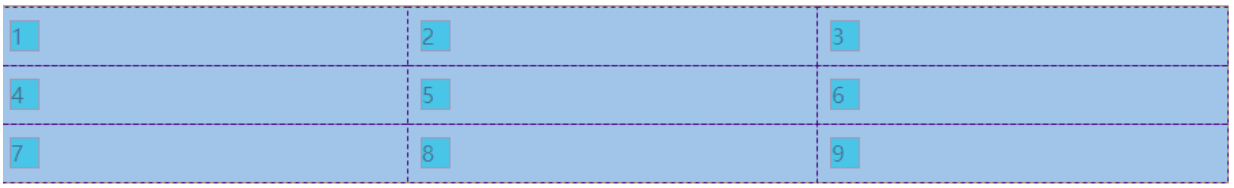
效果：



3）repeat()： 表示设置重复的列宽

设置grid-template-columns: repeat(3, 33.33%);

效果：



分析：

设置grid-template-columns: repeat(3, 33.33%);相当于grid-template-columns: 33.33% 33.33% 33.33%;

设置grid-template-columns: repeat(3, 100px);

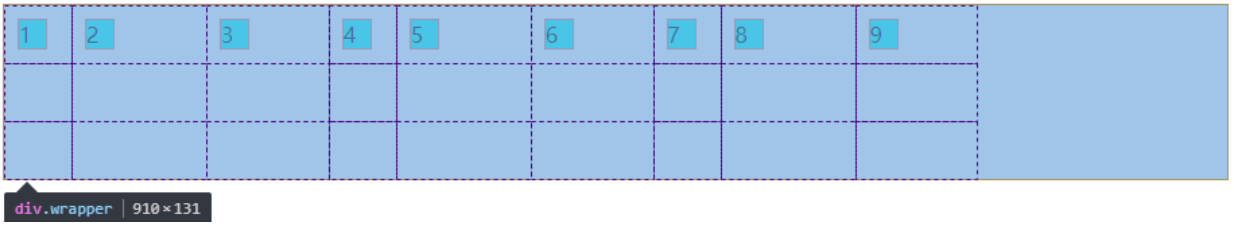
效果

分析：

设置grid-template-columns: repeat(3, 100px);相当于grid-template-columns: 100px 100px 100px;

设置grid-template-columns: repeat(3, 50px 100px 10%);

效果：



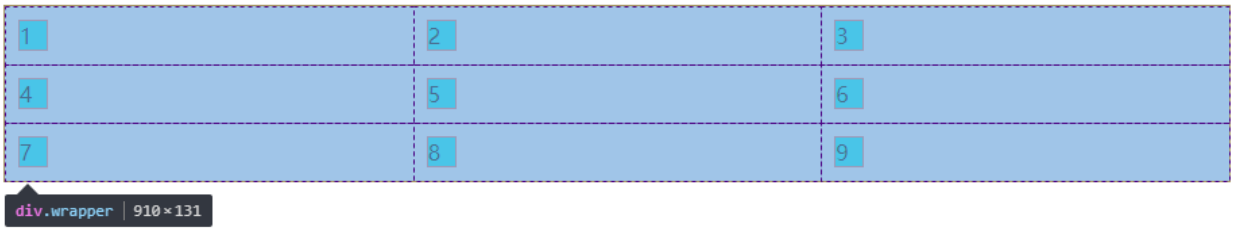
分析：

设置grid-template-columns: repeat(3, 50px 100px 10%);相当于grid-template-columns: 50px 100px 10% 50px 100px 10% 50px 100px 10%; 即有9列

4）fr关键字： 表示容器剩余空间的占比

设置 grid-template-columns: 1fr 1fr 1fr;

效果：

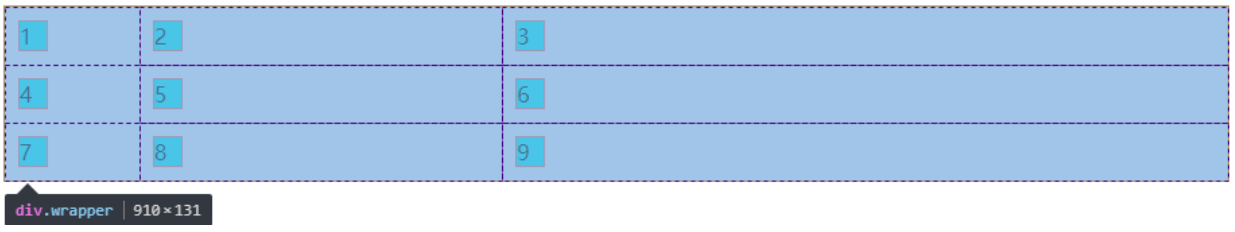


分析：

fr单位用于设置占据剩余区域的比例，如上表示将网格的每列宽度设置为1： 1： 1

设置 grid-template-columns: 100px 1fr 2fr;

效果：



分析：

如上表示第一列的列宽为100px ，剩余区域以1：2的比例分配各第二列和第三列

5）minmax(min, max)： 表示取一个区间值最小为min 最大为max ，若max小于min则取min

设置： `grid-template-columns: 100px minmax(100px, 200px) 200px;`

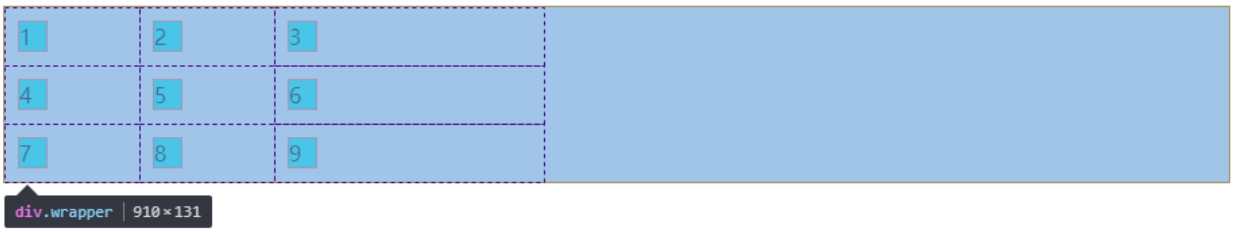
效果:

分析：

如上表示第一列100px　第二列的宽度在100px -- 200px之间根据容器宽度来分配，第三列为200px

设置：`grid-template-columns: 100px minmax(100px, max-content) 200px;`

效果:



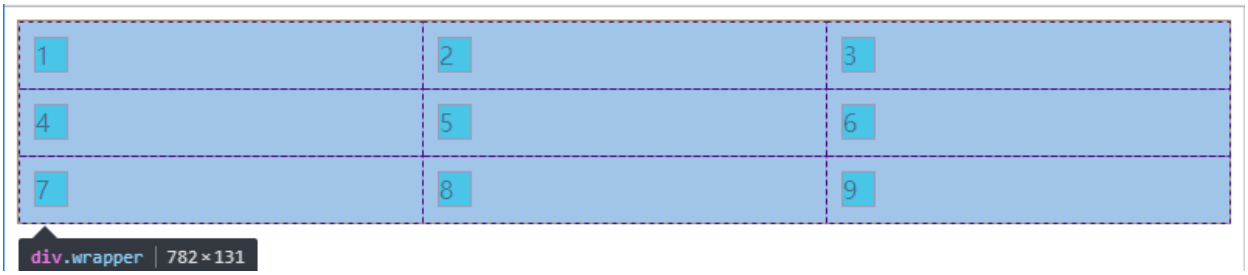
分析：

max-content代表的是元素内容区撑开的宽度，如上2/5/8内容区的宽度为42px 小于了100px则取minmax的最小值100px

6）auto：表示由浏览器自己分配

设置： grid-template-columns: auto auto auto;

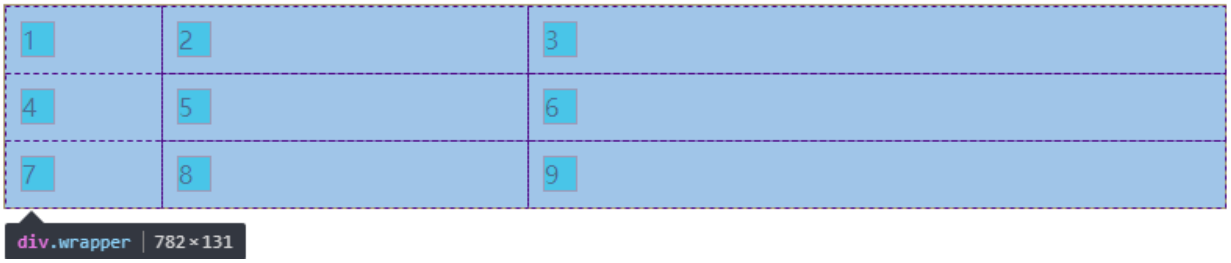
效果：



7）网格线名称

设置： grid-template-columns: [c1] 100px [c2] 30% [c3] 1fr [c4];

效果：



分析： 中括号内的名称为分隔线的名称

(2) . grid-template-rows的值

与grid-template-columns的值相同

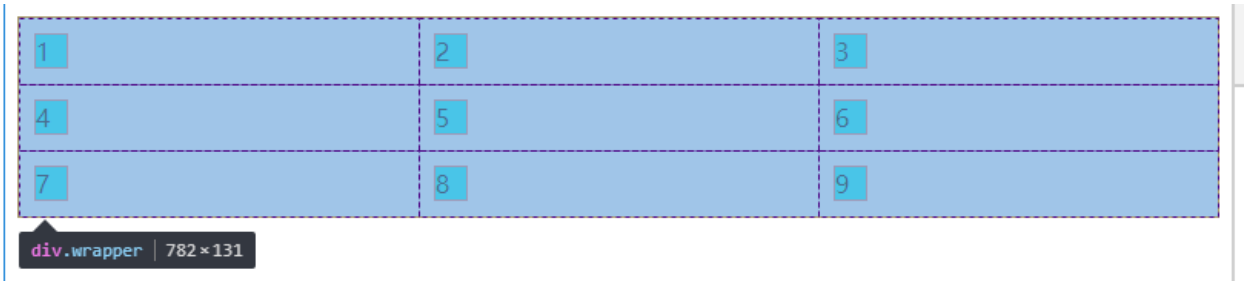
3. grid-gap 设置列与列行与行之间的间距

由grid-row-gap （行与行之间的间距）和 grid-column-gap（列与列之间的间距）两个子属性构成

案例：

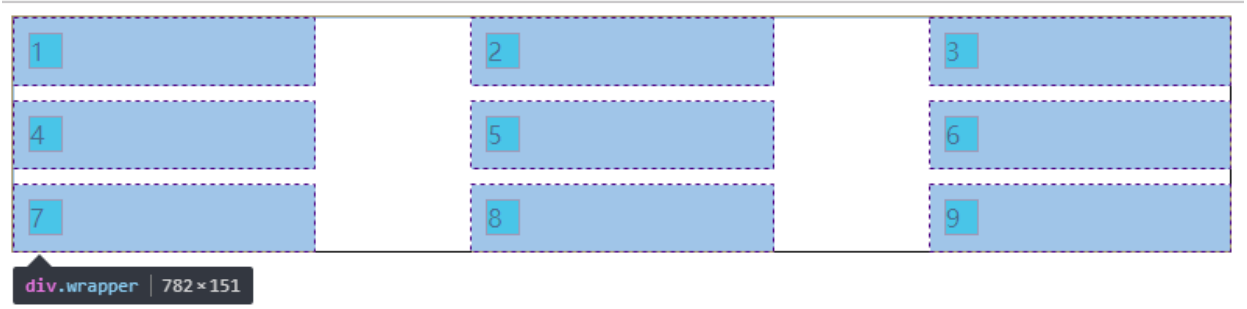
代码：

效果：



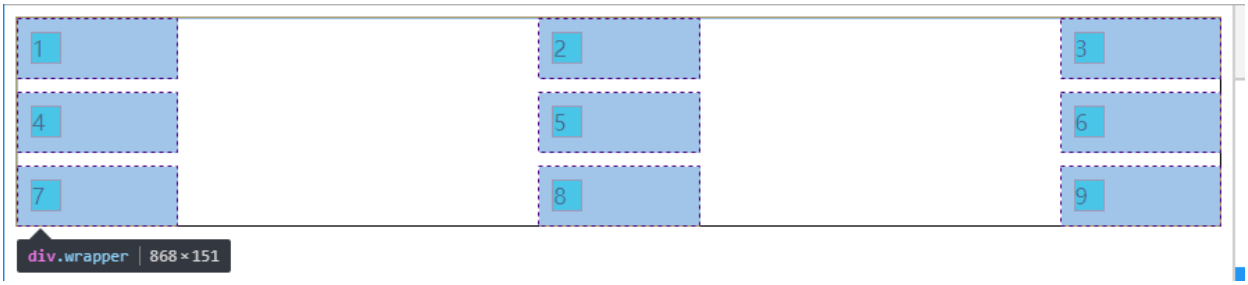
（1）设置grid-row-gap: 10px; grid-column-gap: 100px;

效果：



（2）设置grid-row-gap: 10px; grid-column-gap: 30%;

效果：



分析： 设置百分比表示容器的百分比大小如上容器的宽度为868px ， grid-column-gap: 30%;即 868 * 30% = 260.4px 每两列之间的间隔即为260.4px，

4. justify-content 设置表格区域在容器水平方向上的位置

取值有：1. center 水平居中

2. start 沿容器水平方向起始位置对齐
3. end 沿容器水平方向终止位置对齐
4. space-between 每两列之间的空隙相同

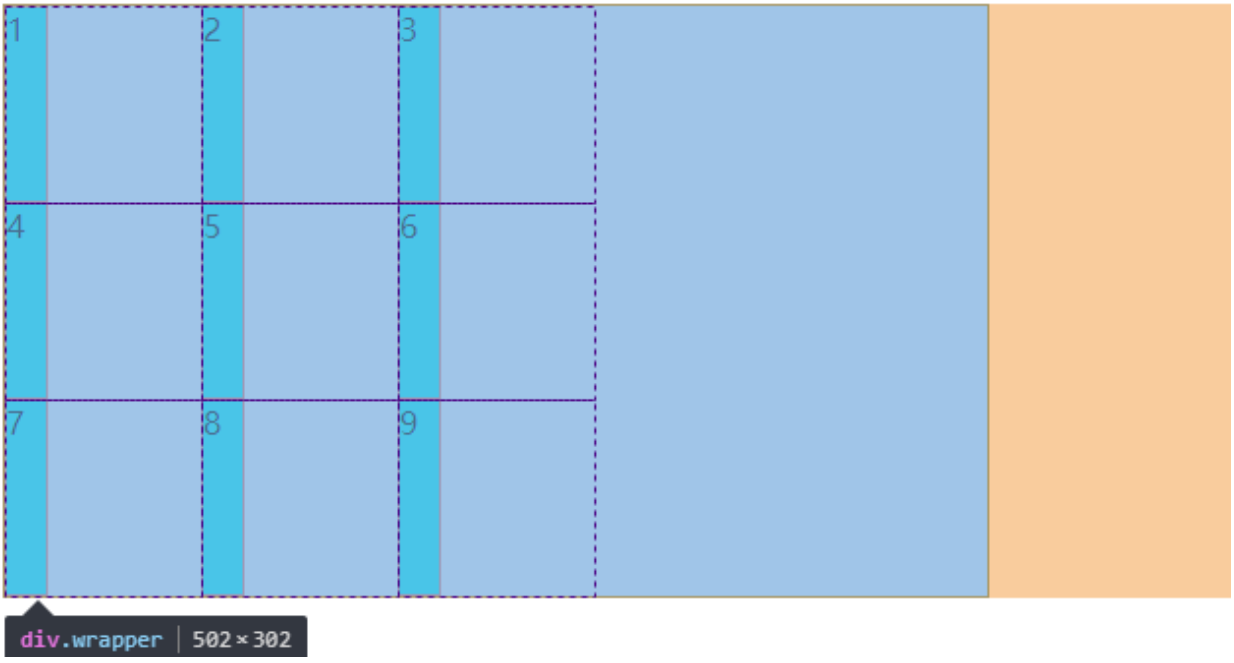
5. space-around　　每列的左右两边的空隙相同
6. space-evenly　　每列之间与容器与列之间的空隙一样

案例：

代码：

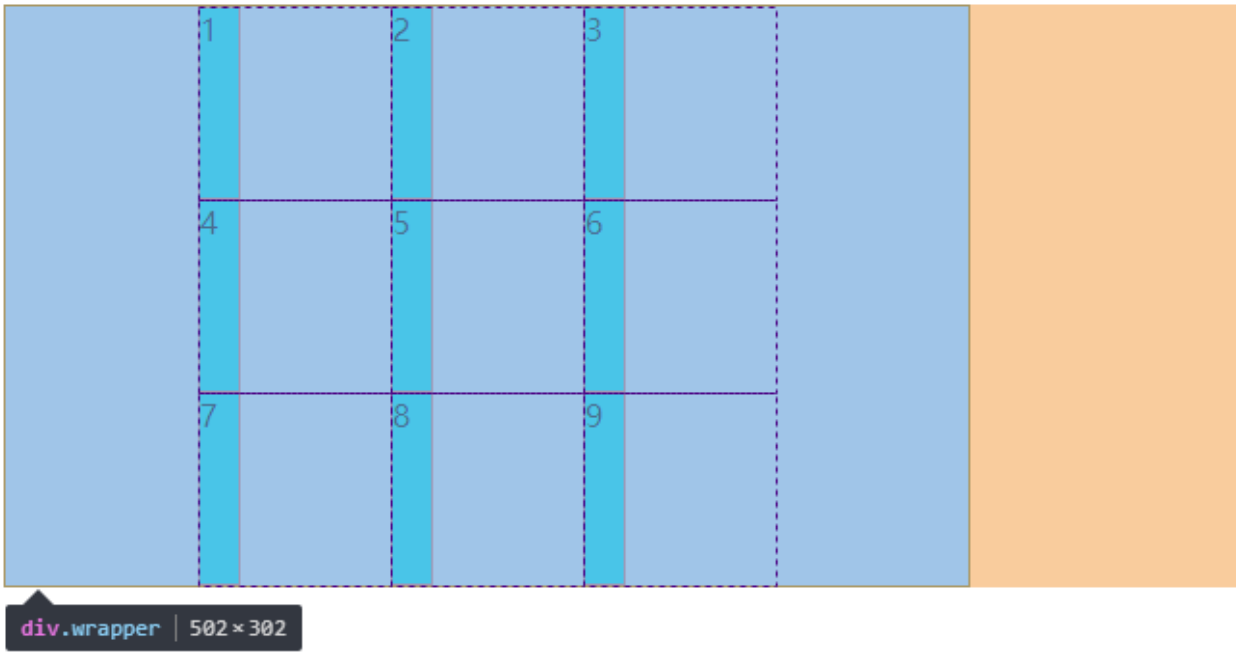
```
    <style>
        .wrapper {
            display: grid;
            border: 1px solid #000;
            grid-template-columns: 100px 100px 100px;
            grid-template-rows: 100px 100px 100px;
            width: 500px;
        }
        .wrapper div {
            background-color: aqua;
            border: 1px solid salmon;
            width: 20px;
        }
    </style>
</head>
<body>
    <div class="wrapper">
        <div>1</div>
        <div>2</div>
        <div>3</div>
        <div>4</div>
        <div>5</div>
        <div>6</div>
        <div>7</div>
        <div>8</div>
        <div>9</div>
    </div>
```

效果：



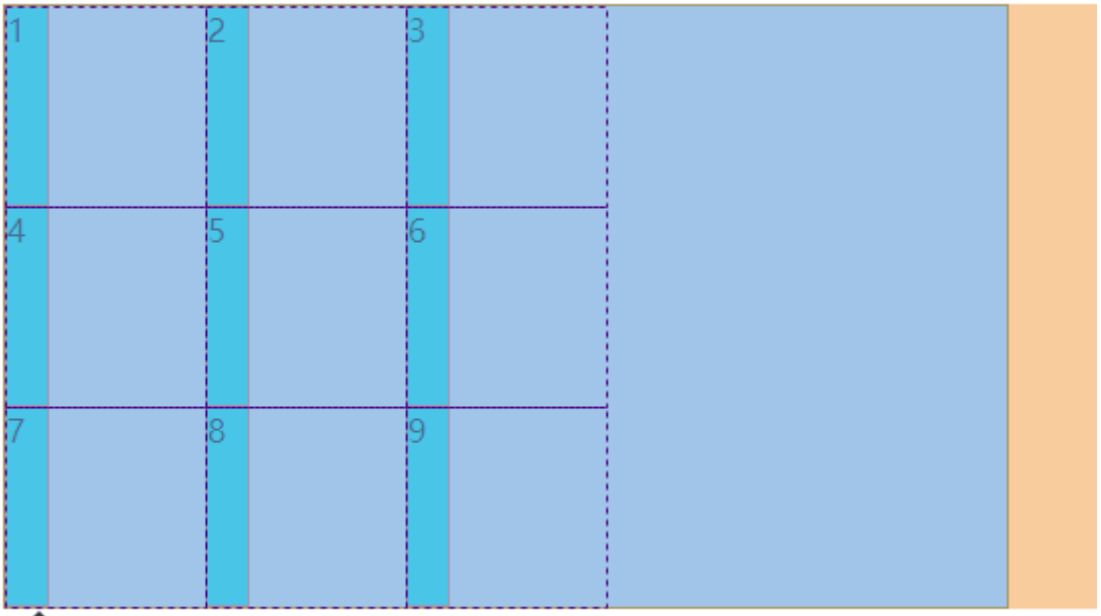
（1）设置justify-content: center; 表格在容器的水平方向上居中对齐

效果：



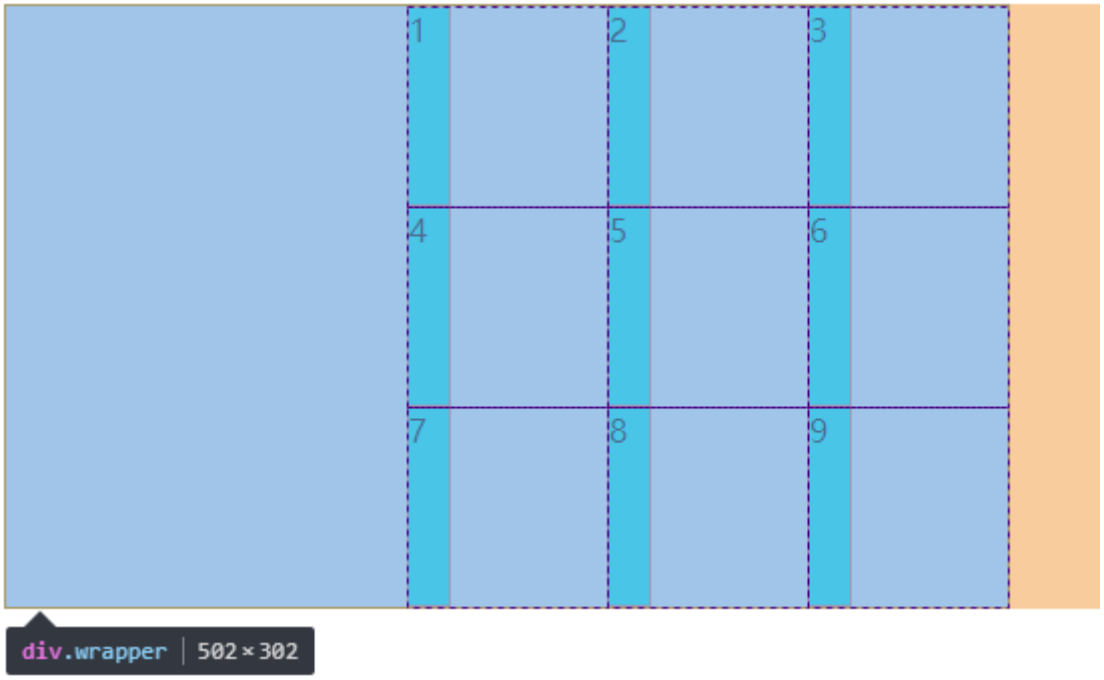
（2）设置justify-content:start; 表格在容器水平方向的左侧对齐

效果：



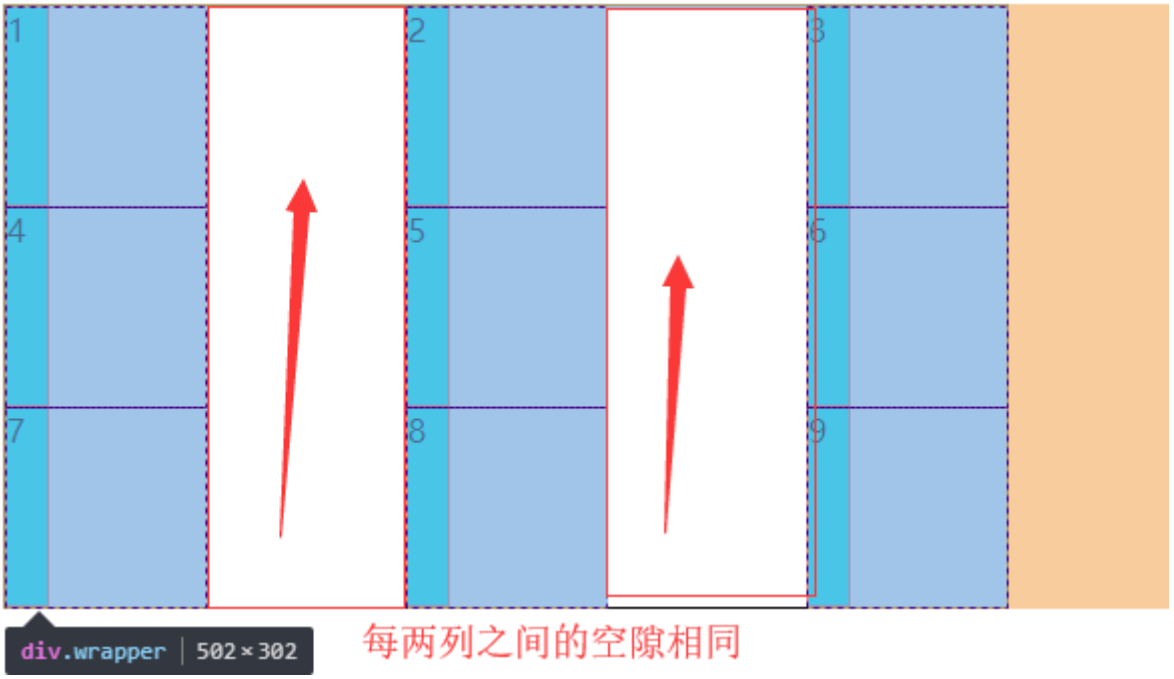
（3）设置justify-content:end；表格在容器水平方向的右侧对齐

效果：



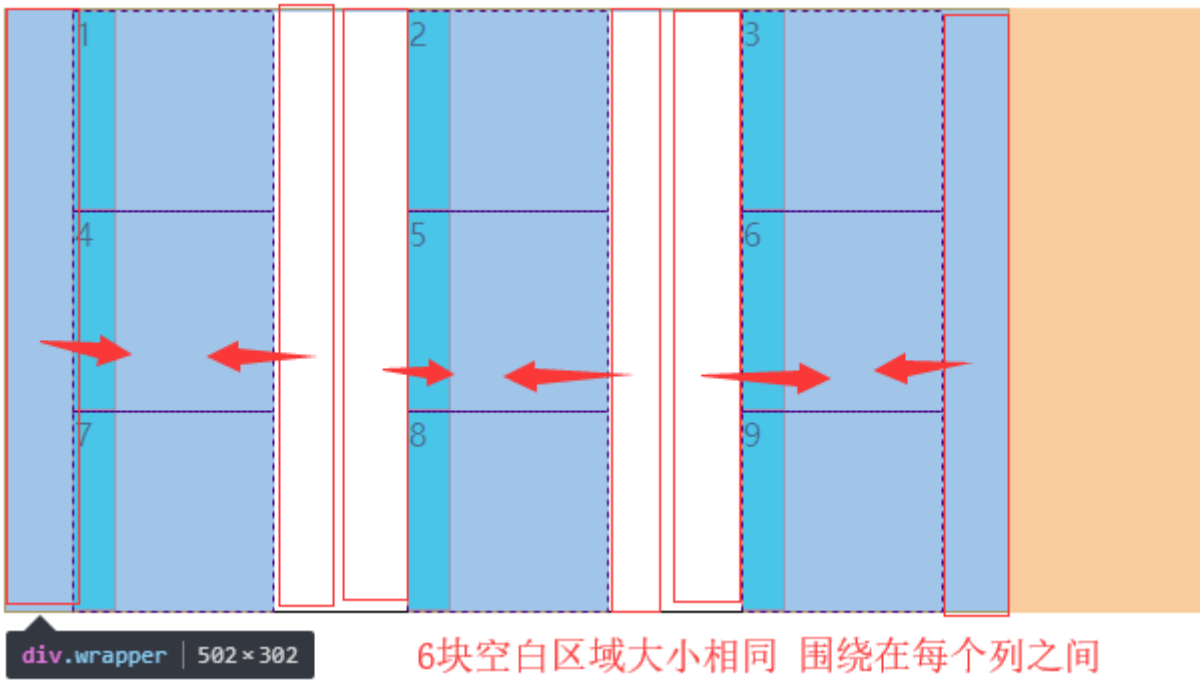
（4）设置justify-content: space-between；容器剩余的部分均放在表格每两列之间

效果：



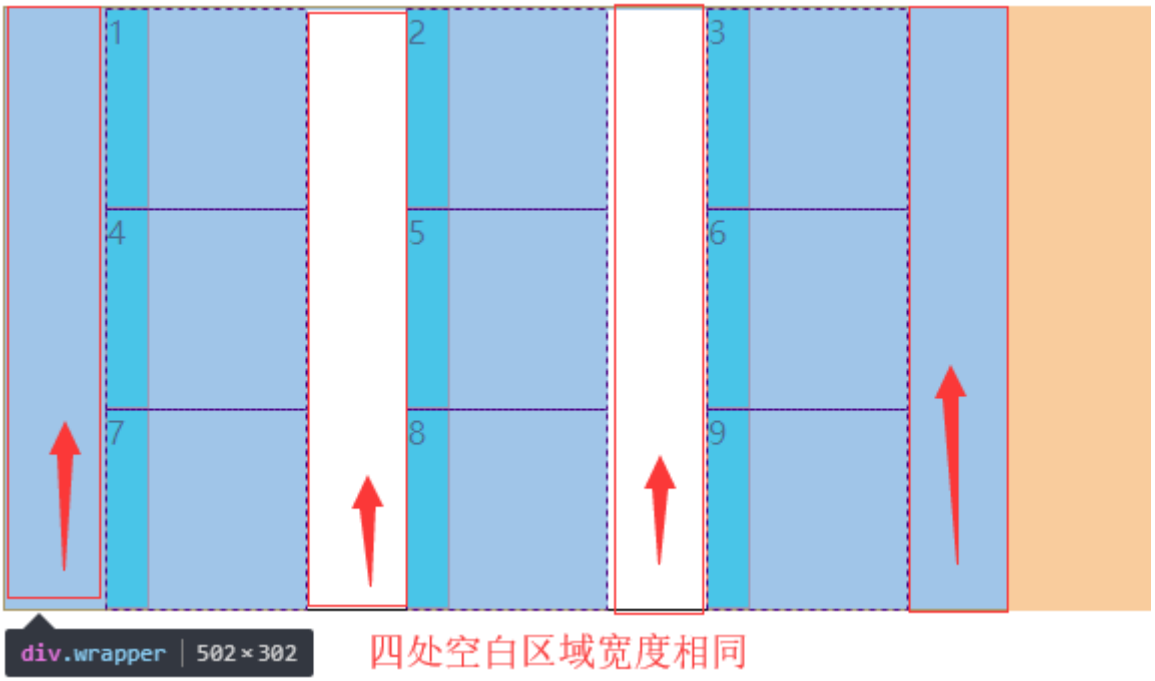
（5）设置justify-content: space-around; 容器剩余的部分均放在表格每列的两端即每列两边空出相同的空白区

效果：



（6）设置justify-content: space-evenly; 容器剩余的部分放在表格每列的两端并且每列的两边和列与列之间的空白区域宽度相同

效果：



5. align-content 设置表格区域在容器垂直方向上的位置

值与justify-content相同，含义也相同，只不过align-content是在竖直方向上的对齐方式 （请参看justify-content的值）

6. align-items 设置每个单元格内垂直方向上的对齐方式

取值：

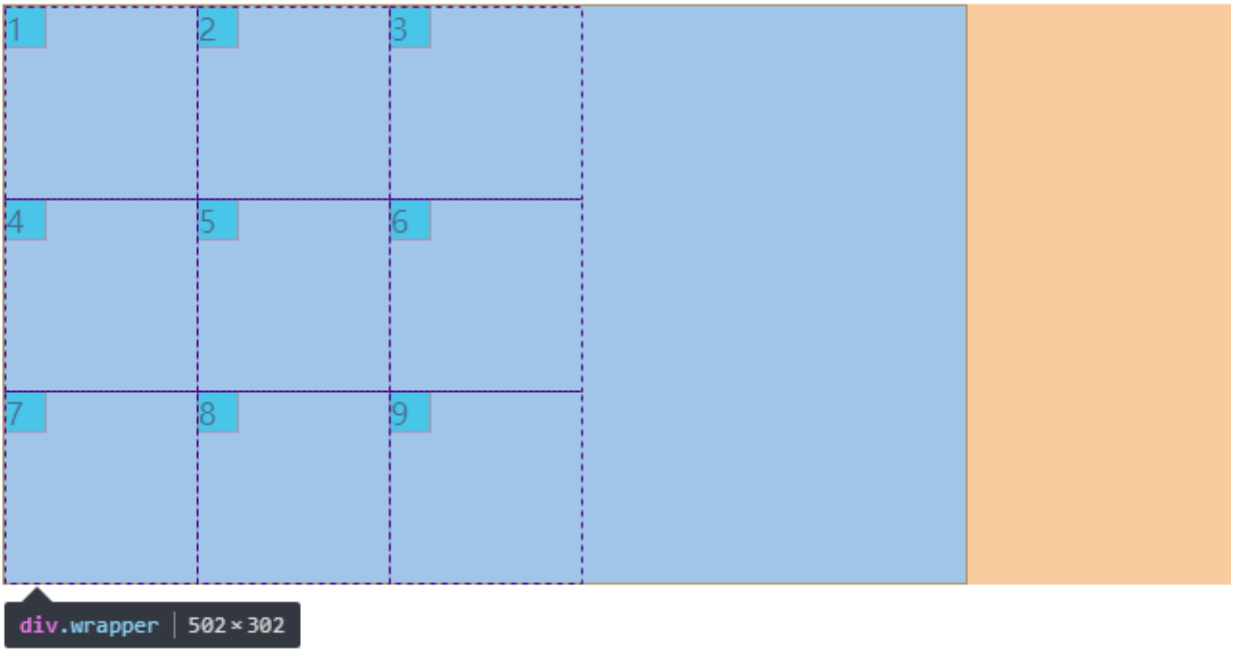
1. center
2. start
3. end

案例：

代码：

```
    <style>
        .wrapper {
            display: grid;
            border: 1px solid #000;
            grid-template-columns: 100px 100px 100px;
            grid-template-rows: 100px 100px 100px;
            width: 500px;
        }
        .wrapper div {
            background-color: aqua;
            border: 1px solid salmon;
            width: 20px;
            height: 20px;
        }
    </style>
</head>
<body>
    <div class="wrapper">
        <div>1</div>
        <div>2</div>
        <div>3</div>
        <div>4</div>
        <div>5</div>
        <div>6</div>
        <div>7</div>
        <div>8</div>
        <div>9</div>
    </div>
```

初始效果：

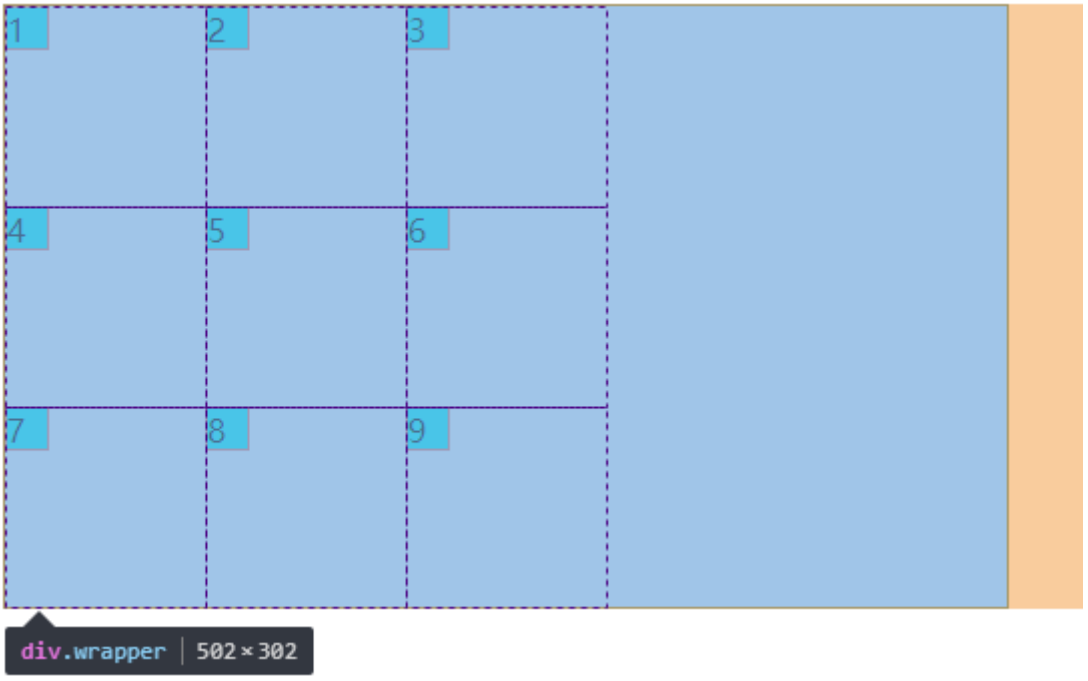


（1）设置align-items: center；子元素在每个单元格内垂直居中

效果：

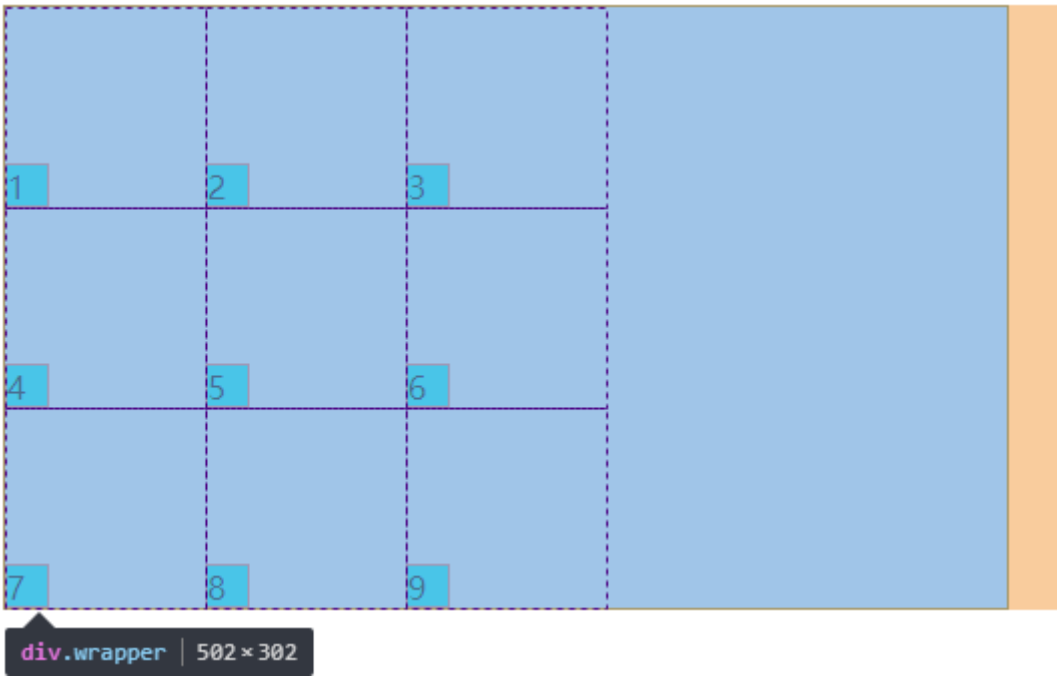
（2）设置align-items: start;子元素在每个单元格的垂直方向的上方

效果



（3）设置align-items: end;子元素在每个单元格的垂直方向的下方

效果



7. justify-items 设置每个单元格内水平方向上的对齐方式

取值与align-items相同，含义也相近只不过justify-items表示的是水平方向上的对齐方式（参考align-items）