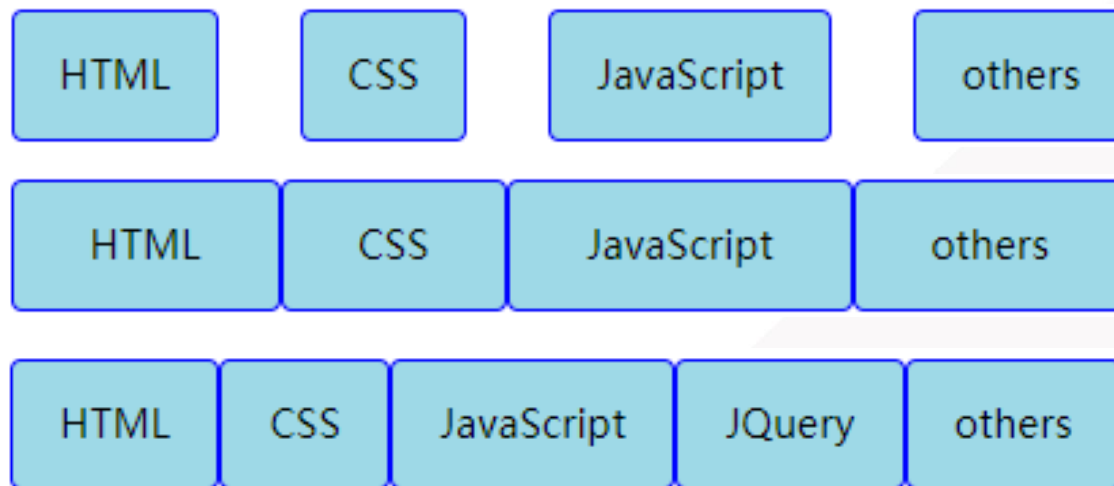


弹性盒子概述



概述

弹性盒子布局 Flexbox Layout



元素可以

- 拉伸以填充额外的空间
- 收缩以适应更小的空间



概述

Flexbox 可以解决如下问题

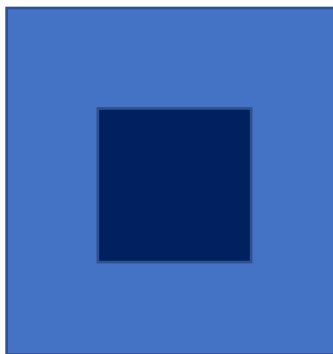
Lorem ipsum dolor		
Lorem ipsum dolor sit amet. Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut?	Lorem ipsum dolor sit amet. Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut? Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut?	Lorem ipsum Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut? Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut? Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut? Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut?

Lorem ipsum dolor		
Lorem ipsum dolor sit amet. Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut?	Lorem ipsum dolor sit amet. Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut? Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut?	Lorem ipsum Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut? Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut? Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut? Lorem ipsum, dolor sit amet consectetur adipisicing elit. Hic culpa iure aut?



概述

Flexbox 可以解决如下问题



垂直居中



间隙的平均分配

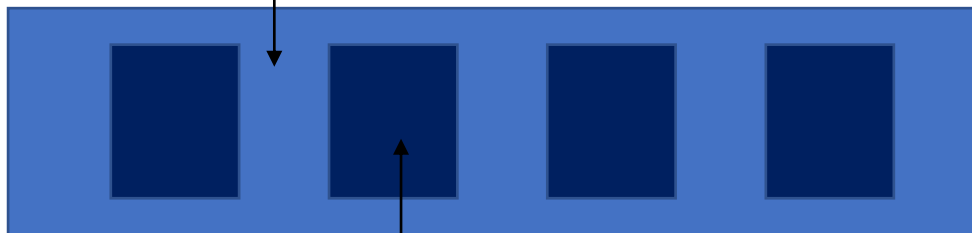


自动占据剩余空间

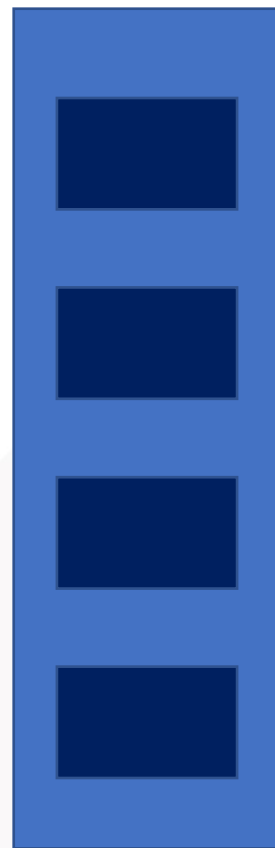
弹性盒子组成

弹性盒子：一维布局
按行水平布局、按列垂直布局

弹性容器 (Flex Container)



弹性元素 (Flex Item)



弹性盒子样式



弹性容器 (Flex Container)

display: flex

flex-direction

flex-wrap

justify-content

align-items

align-content

flex-flow

弹性元素 (Flex Item)

flex-grow

flex-shrink

flex-basis

order

align-self

flex

弹性盒子样式

弹性盒子样式



弹性容器 (Flex Container)

弹性元素 (Flex Item)

display: flex

flex-direction

flex-wrap

justify-content

align-items

align-content



flex-flow


弹性容器样式

1. display属性

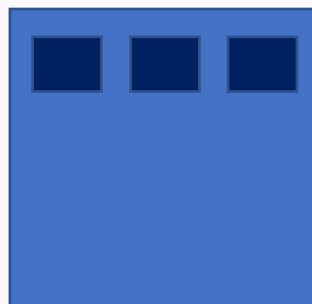
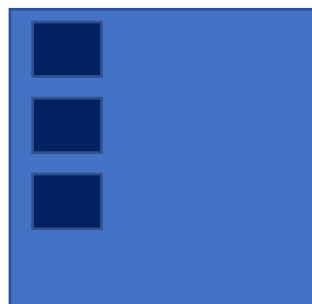
定义弹性容器

```
<div class="flex-container">  
  <div class="flex-item">1</div>  
  <div class="flex-item">2</div>  
  <div class="flex-item">3</div>  
</div>
```

样式添加在这里

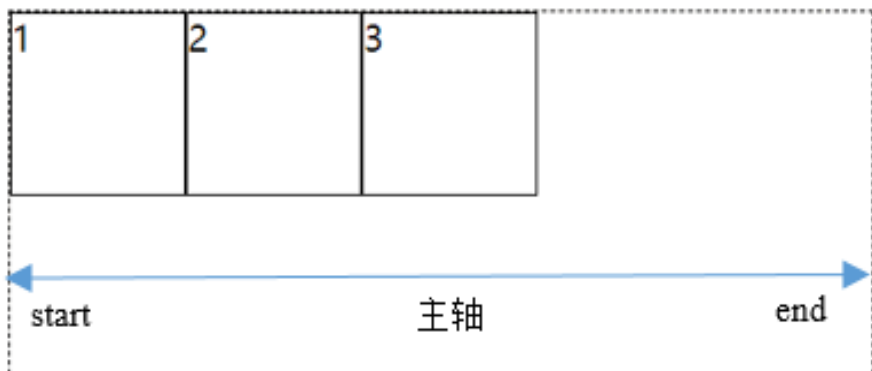


```
.flex-container{  
  display: flex;  
}
```

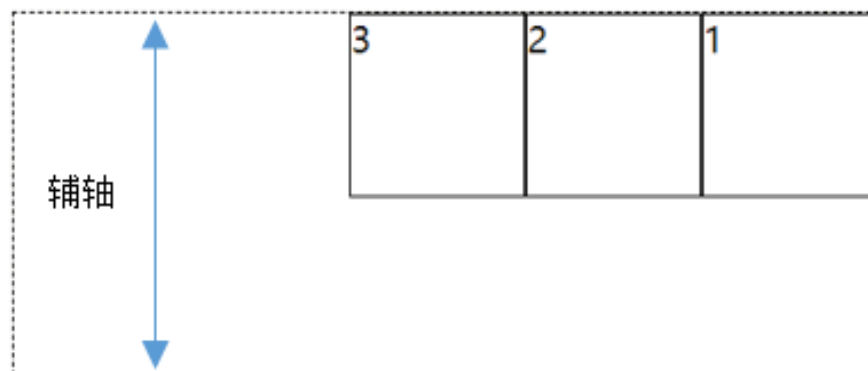


弹性容器样式

2. flex-direction属性——行布局



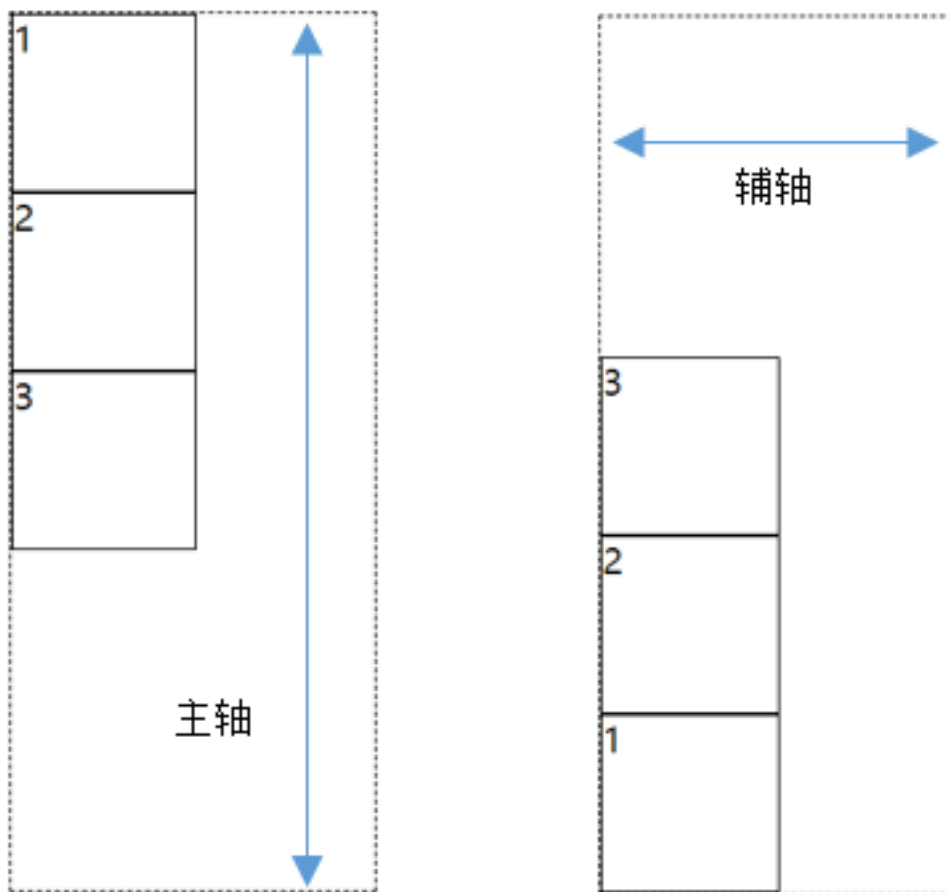
`flex-direction: row`



`flex-direction: row-reverse`

弹性容器样式

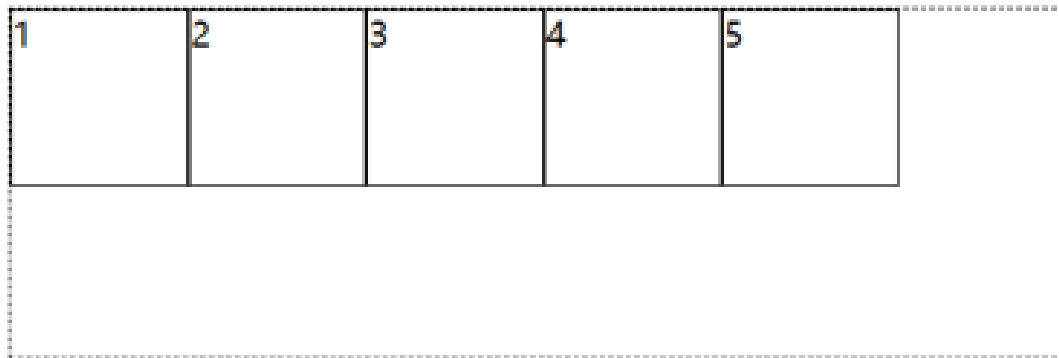
2. flex-direction属性——列布局



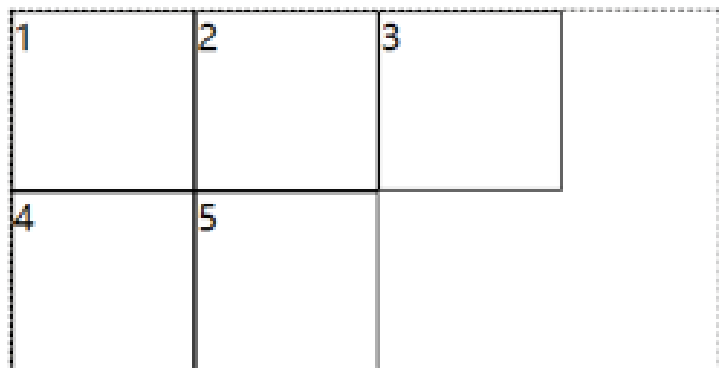
flex-direction: **column** flex-direction: **column-reverse**

弹性容器样式

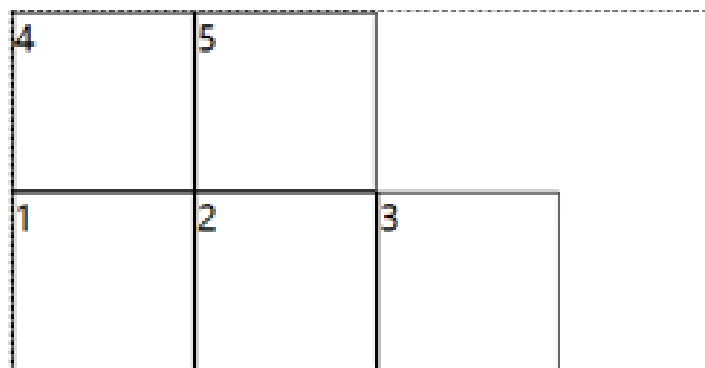
3. flex-wrap 属性



flex-wrap: **nowrap**



flex-wrap: **wrap**



flex-wrap: **wrap-reverse**



弹性容器样式

3. flex-wrap 属性

Lorem ipsum dolor					
Lorem ipsum dolor sit amet. Lorem ipsum, dolor sit amet consectetur adipiscing elit. Hic culpa iure aut?	Lorem ipsum dolor sit amet. Lorem ipsum, dolor sit amet consectetur adipiscing elit. Hic culpa iure aut?	Lorem ipsum, dolor sit amet consectetur adipiscing elit. Hic culpa iure aut?	Lorem ipsum dolor sit amet. Lorem ipsum, dolor sit amet consectetur adipiscing elit. Hic culpa iure aut?	Lorem ipsum dolor sit amet. Lorem ipsum, dolor sit amet consectetur adipiscing elit. Hic culpa iure aut?	Lorem ipsum, dolor sit amet consectetur adipiscing elit. Hic culpa iure aut?

flex-wrap: nowrap

Lorem ipsum dolor		
Lorem ipsum dolor sit amet. Lorem ipsum, dolor sit amet consectetur adipiscing elit. Hic culpa iure aut?	Lorem ipsum dolor sit amet. Lorem ipsum, dolor sit amet consectetur adipiscing elit. Hic culpa iure aut?	Lorem ipsum, dolor sit amet consectetur adipiscing elit. Hic culpa iure aut?
Lorem ipsum dolor sit amet. Lorem ipsum, dolor sit amet consectetur adipiscing elit. Hic culpa iure aut?	Lorem ipsum dolor sit amet. Lorem ipsum, dolor sit amet consectetur adipiscing elit. Hic culpa iure aut?	Lorem ipsum, dolor sit amet consectetur adipiscing elit. Hic culpa iure aut?

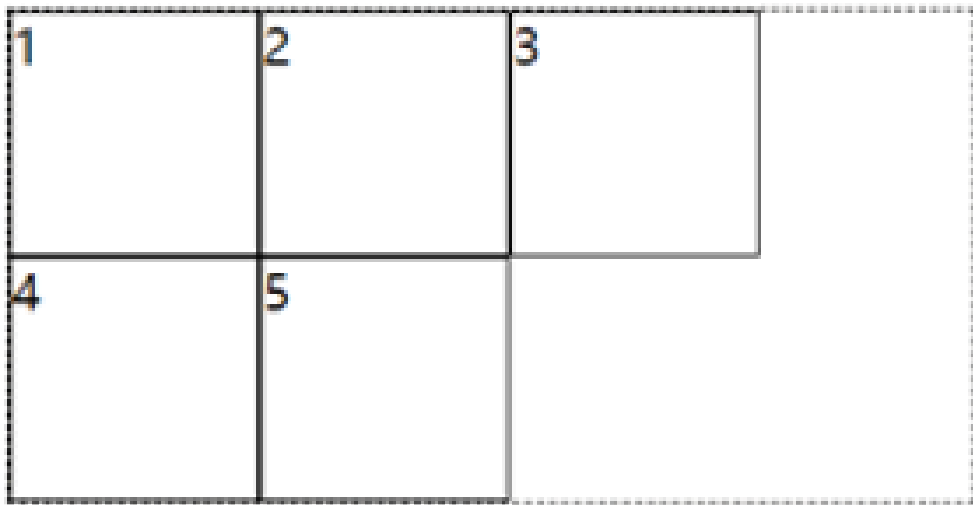
flex-wrap: wrap

弹性容器样式

4. flex-flow属性

flex-flow: flex-direction flex-wrap

flex-flow: row wrap;

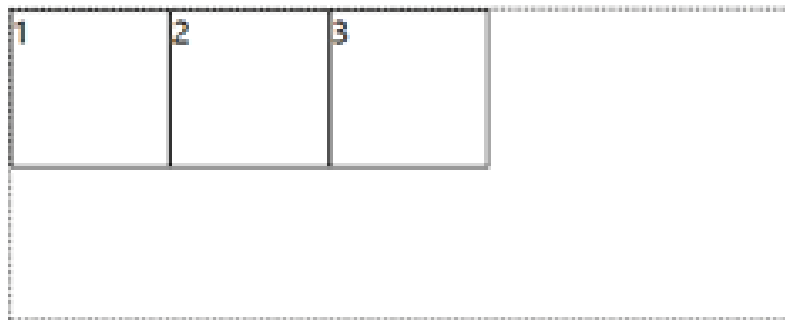


弹性容器样式

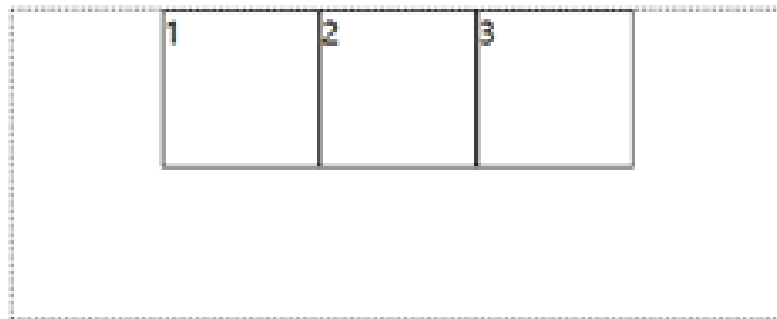
5. justify-content属性

元素在主轴上的对齐方式

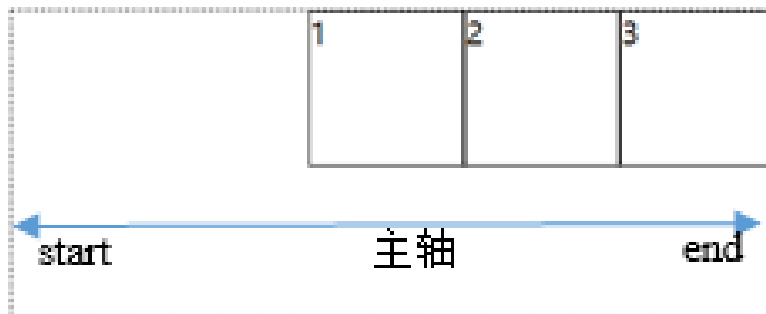
justify-content: **flex-start**



justify-content: **center**



justify-content: **flex-end**

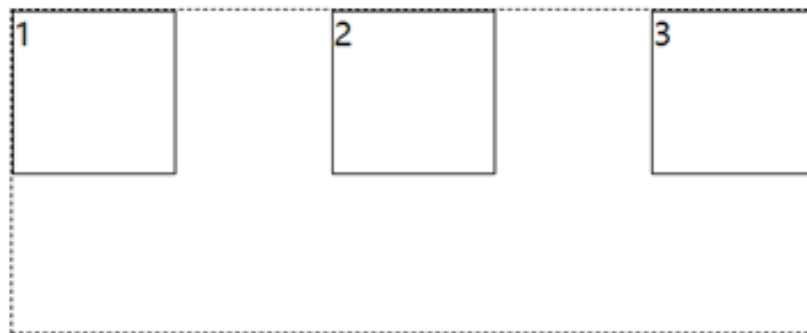


弹性容器样式

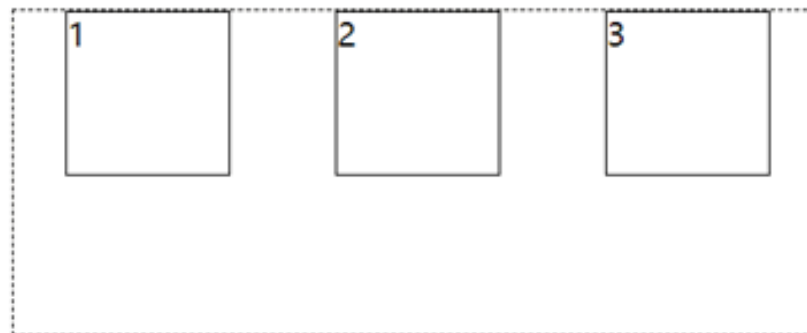
5. justify-content属性

元素在主轴上的对齐方式

justify-content: space-between



justify-content: space-around

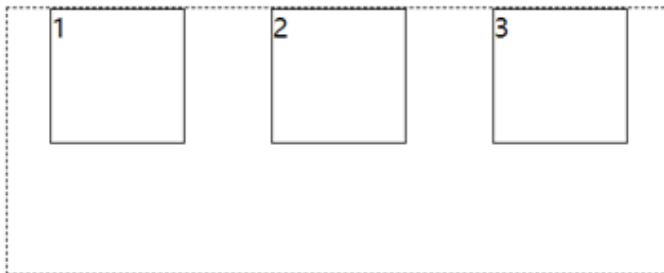


弹性容器样式

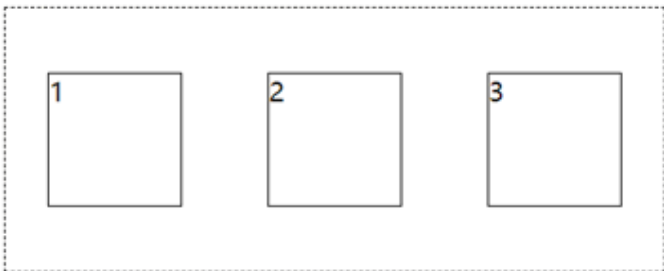
6. align-items属性

元素在辅轴上的对齐方式

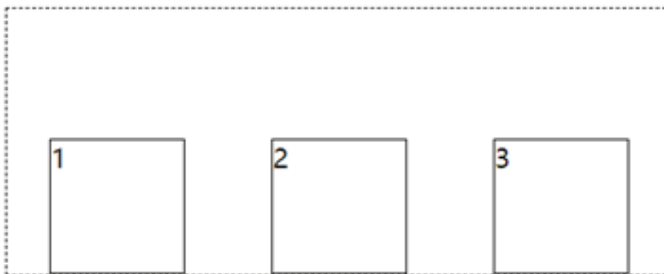
align-items: **flex-start**



align-items: **center**



align-items: **flex-end**



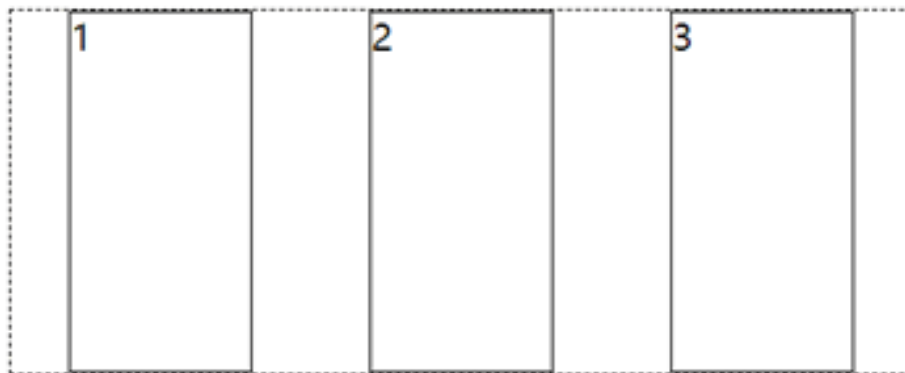
弹性容器样式

6. align-items属性

元素在辅轴上的对齐方式

align-items: stretch

注意：去掉元素高度



```
<div class="flex-container">
```

```
  <div class="flex-item">1</div>
```

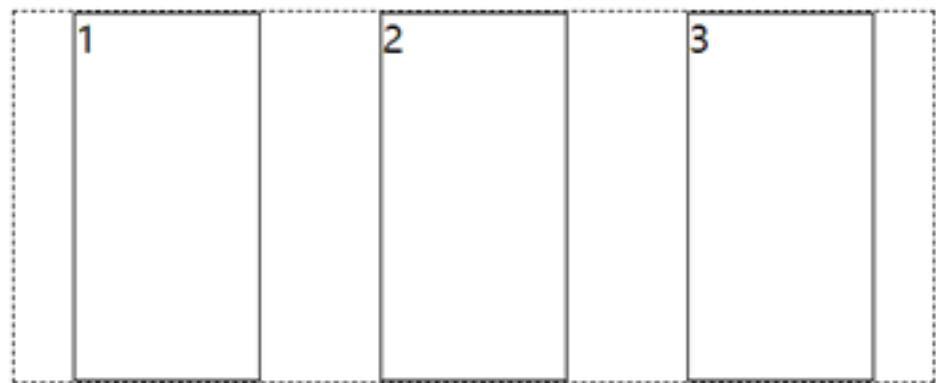
```
  <div class="flex-item">2</div>
```

```
  <div class="flex-item">3</div>
```

```
</div>
```

弹性容器样式

```
.flex-container{  
  display: flex;  
  flex-direction: row;  
  justify-content: space-around;  
  align-items: stretch;  
  
  width: 500px;  
  height: 200px;  
  border: 1px dashed;  
  
}
```



```
.flex-item{  
  
  width: 100px;  
  border: 1px solid;  
  font-size: 20px;  
  
}
```

弹性容器样式

7. align-content属性 设置多行元素在容器中的整体对齐方式

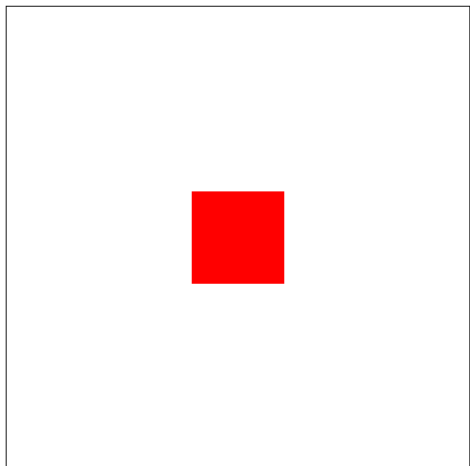
align-content : flex-start | flex-end | center

space-between | space-around | stretch

只有一行或者一列，该属性不起作用



弹性盒子布局练习



```
<div class="flexbox">  
  <div class="flexitem">  
  </div>  
</div>
```

弹性元素样式

弹性盒子样式



弹性容器 (Flex Container)

弹性元素 (Flex Item)

flex-grow

flex-shrink

flex-basis

order

align-self

flex

弹性元素样式

```
<div class="flex-container">  
  <div class="flex-item">1</div>  
  <div class="flex-item">2</div>  
  <div class="flex-item">3</div>  
</div>
```

.flex-item{



样式添加在这里

}

弹性元素样式

1. flex-grow属性

元素被拉大的比例，按比例分配容器剩余空间

- (1) 默认值为0: 元素不占用剩余空间
- (2) 取值为n: 元素占据剩余空间若干份中的n

份

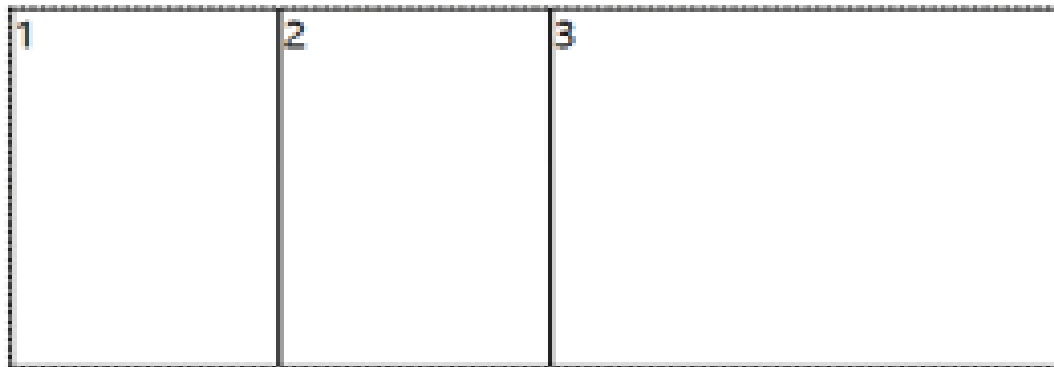
弹性元素样式

```
.flex-container{  
  display: flex;  
  flex-direction: row  
  align-items: stretch;  
}  
.flex-item{  
  border: 1px solid;  
}
```

未规定width,height

```
div:nth-child(1){  
  flex-grow: 1;  
}
```

```
div:nth-child(3){  
  flex-grow: 2;  
}
```



```
div:nth-child(2){  
  flex-grow: 1;  
}
```

1: 1: 2分配剩余空间

弹性元素样式

2. flex-shrink属性

元素被压缩的比例

默认为1，表示弹性元素默认等比例压缩

0则表示不压缩

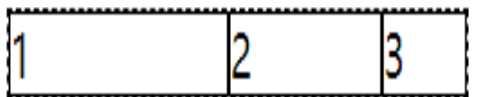
3. flex-basis属性

元素在主轴上的默认尺寸，其优先级高于width属性。

```
.flex-container{  
    display: flex;           /*弹性容器*/  
    width: 50%;  
}
```

弹性元素样式

```
.flex-item{  
    flex-basis: 120px;  
    border: 1px solid;  
    font-size: 20px;  
}  
div:nth-child(1){  
    flex-shrink:0;  
}  
div:nth-child(2){  
    flex-shrink:1;  
}  
div:nth-child(3){  
    flex-shrink:2;  
}
```



不压缩 压缩1 压缩2

弹性元素样式

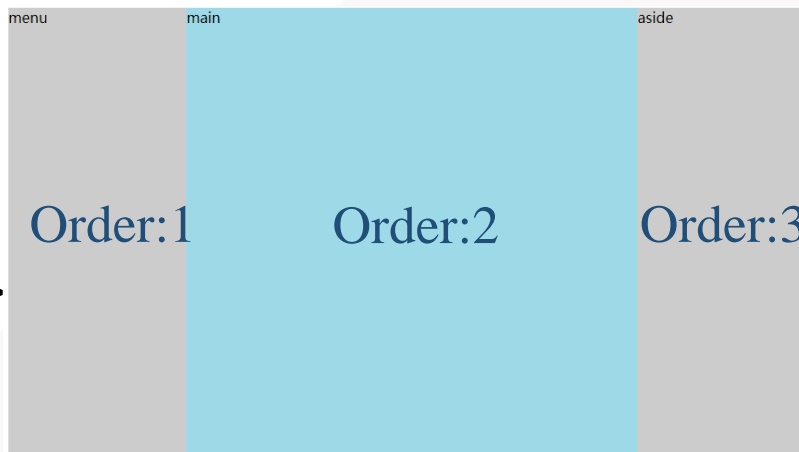
4. flex属性

flex: flex-grow flex-shrink flex-basis;

5. order属性

子元素在弹性容器中的排列顺序，数值越小排名越靠前

```
<div class="flex-container">  
  <div class="flex-item">main</div>  
  <div class="flex-item">menu</div>  
  <div class="flex-item">aside</div>  
</div>
```

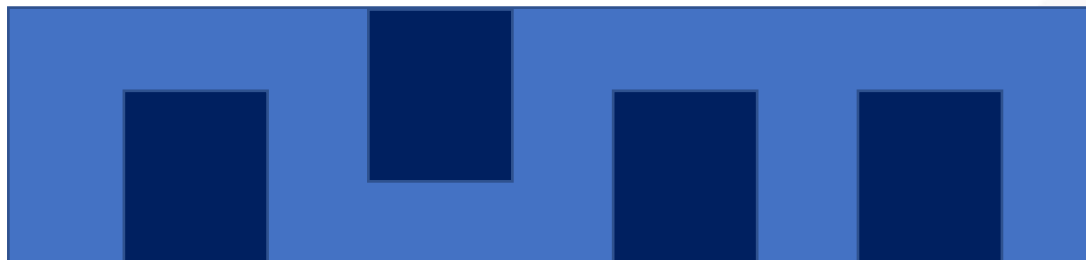


弹性元素样式

6. align-self属性

单个弹性元素在辅轴上的对齐方式，align-items是全部元素

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





弹性盒子布局练习

Web前端开发

HTML

超文本标记语言，用于构建网页结构，定义网页包括的内容。

CSS

层叠样式表，用于构建网页布局、外观，美化页面。

JavaScript

JavaScript，脚本语言，用于构建网页行为，与用户进行交互，使用户获得更好的体验。

概述

弹性盒子flexbox 一维布局

HTML

CSS

JavaScript

个人中心

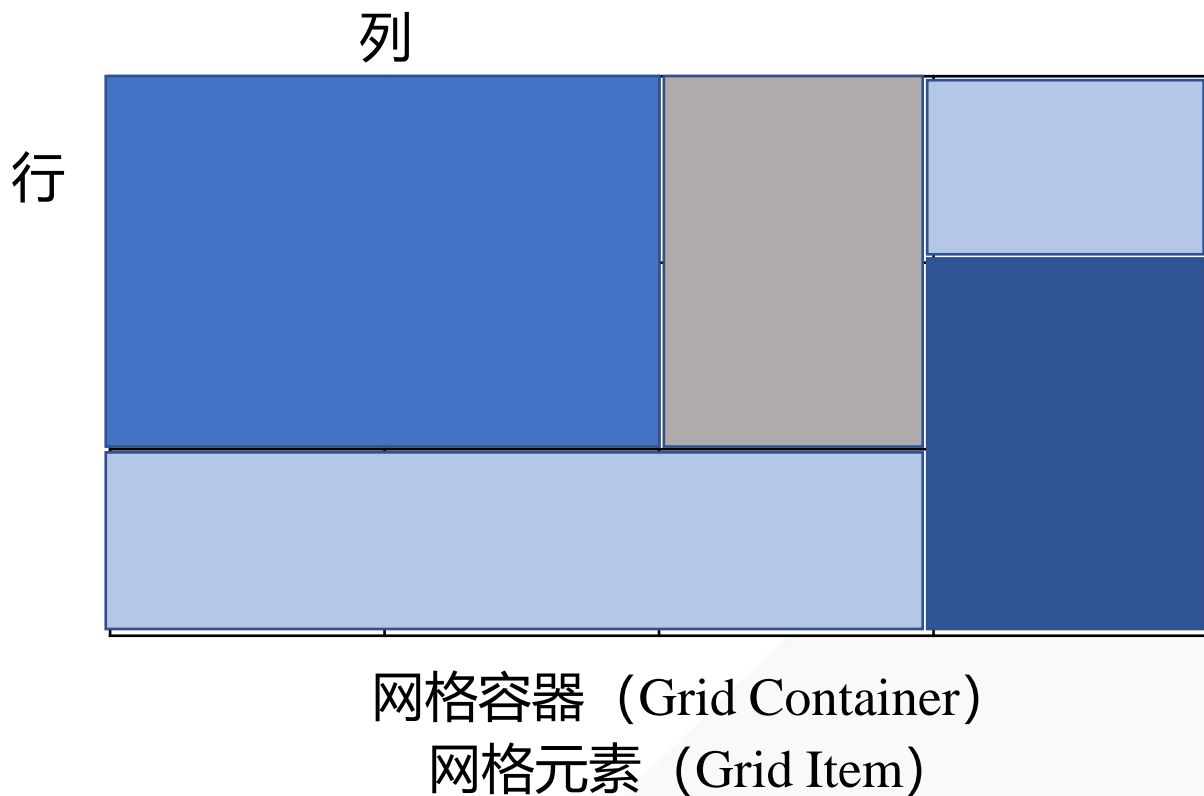
网格布局grid 二维布局





网格布局

网格布局 (Grid Layout)





网格布局

网格容器样式



1. display属性：定义网格容器

```
.grid-container{  
    display: grid;  
}
```

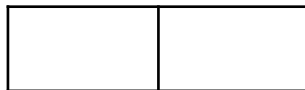


网格布局

2. grid-template-columns和grid-template-rows属性

划分网格的行和列，取值：px、%、auto、fr、repeat()函数

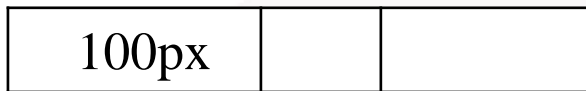
grid-template-columns: 1fr 1fr;



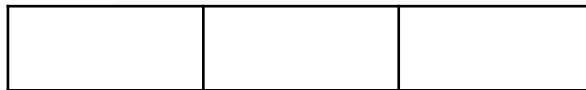
grid-template-columns: 1fr 2fr;



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



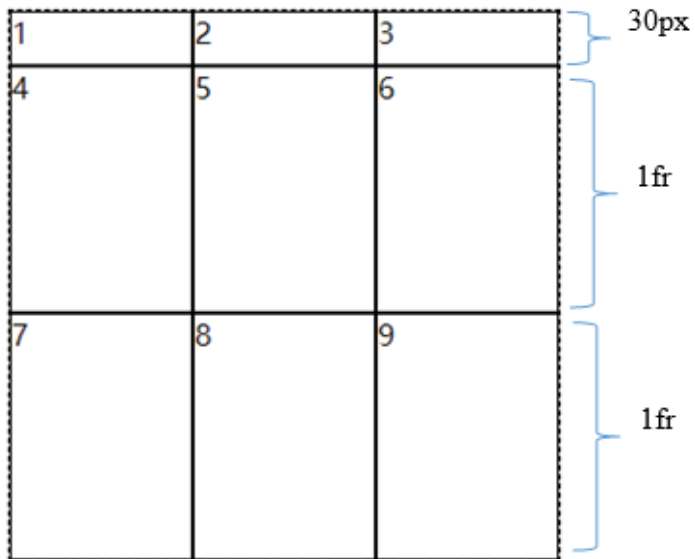
grid-template-columns: repeat(3, 1fr);





网格布局

完成如下网格



```
<div class="grid-container">
```

```
  <div class="grid-item">1</div>
```

```
  .....
```

```
  <div class="grid-item">9</div>
```

```
</div>
```

```
.grid-item{
```

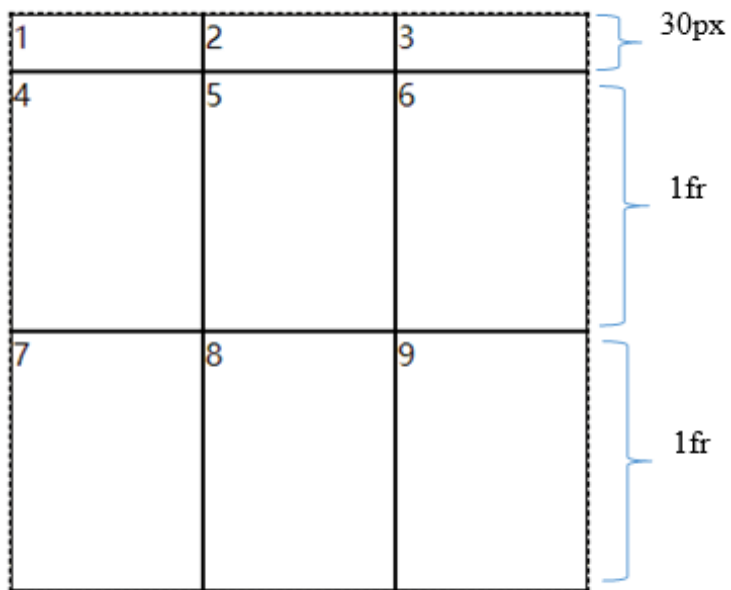
```
  border: 1px solid;
```

```
}
```



网格布局

完成如下网格



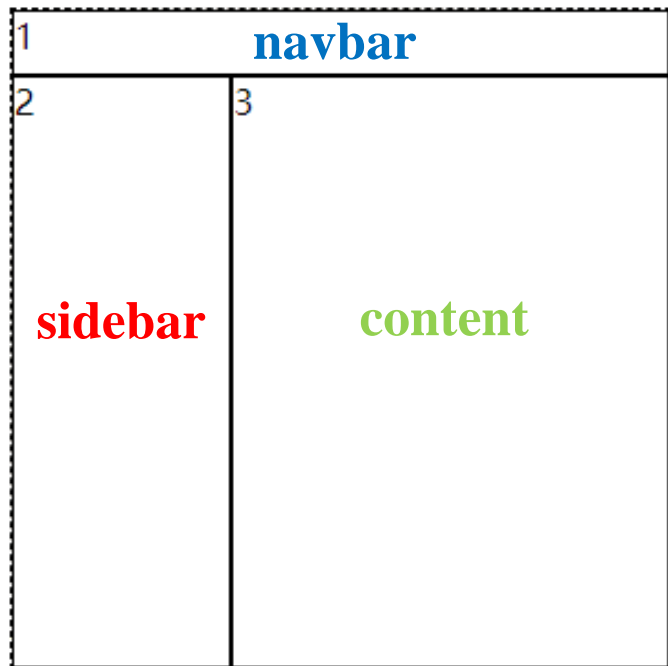
```
.grid-container{  
    width: 300px;  
    height: 300px;  
    border:1px dashed;  
  
    display: grid;  
    grid-template-rows:30px 1fr 1fr;  
    grid-template-columns:1fr 1fr 1fr;  
}
```



网格布局

3. grid-template-areas属性

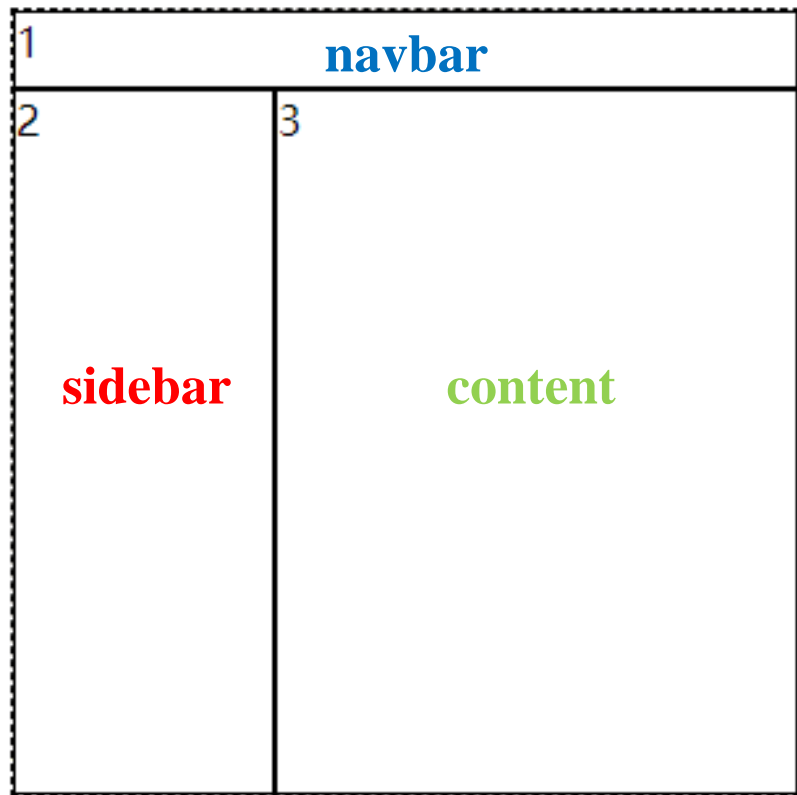
命名单元格，具有相同区域名称的单元格被划分为一个区域



```
<div class="grid-container">  
  <div class="grid-item">1</div>  
  <div class="grid-item">2</div>  
  <div class="grid-item">3</div>  
</div>
```



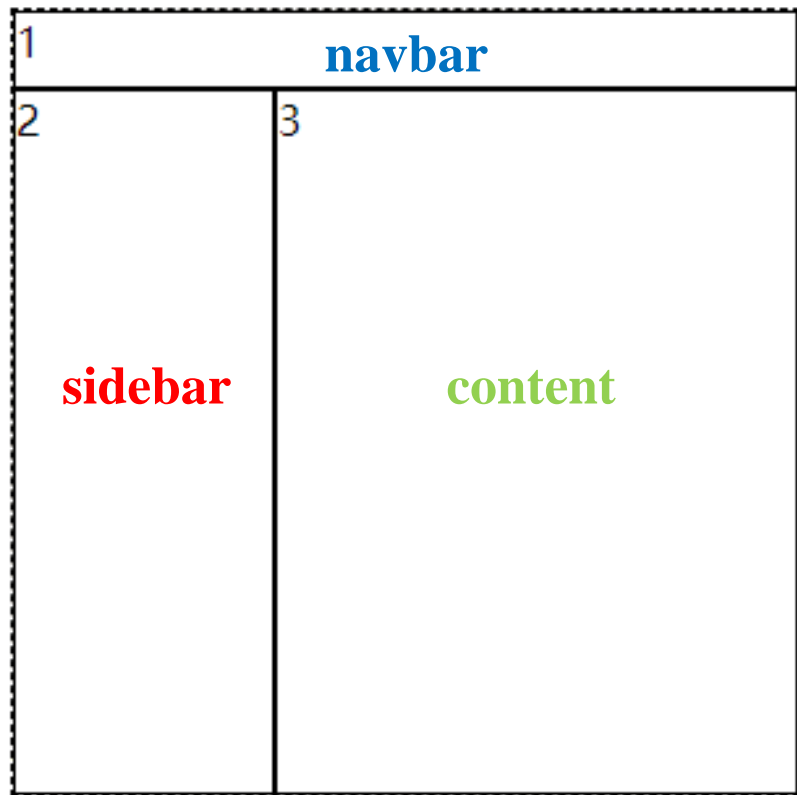
网格布局



```
.grid-container{  
    display: grid;  
    grid-template-rows:30px 1fr 1fr;  
    grid-template-columns:1fr 1fr 1fr;  
  
    grid-template-areas:  
        "navbar navbar navbar"  
        "sidebar content content"  
        "sidebar content content";  
  
    width: 300px;  
    height: 300px;  
    border:1px dashed;  
}
```



网格布局



```
.grid-item{  
    border: 1px solid;  
}  
.grid-item:nth-child(1){  
    grid-area : navbar;  
}  
.grid-item:nth-child(2){  
    grid-area : sidebar;  
}  
.grid-item:nth-child(3){  
    grid-area : content;  
}
```

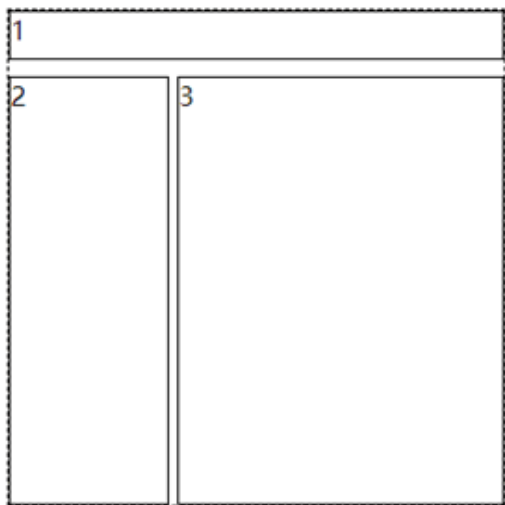



网格布局

4. grid-gap 属性

grid-gap: grid-row-gap grid-column-gap

grid-row-gap: 行间距, grid-column-gap: 列间距



```
.grid-container{  
    grid-row-gap:10px;  
    grid-column-gap:5px;  
}
```



5. justify-items、align-items、place-items 属性

justify-items、align-items、place-items属性用于设置网格元素在网格中的位置。

(1) justify-items: 单元格内容的水平位置

justify-items : start | end | center | 默认stretch;

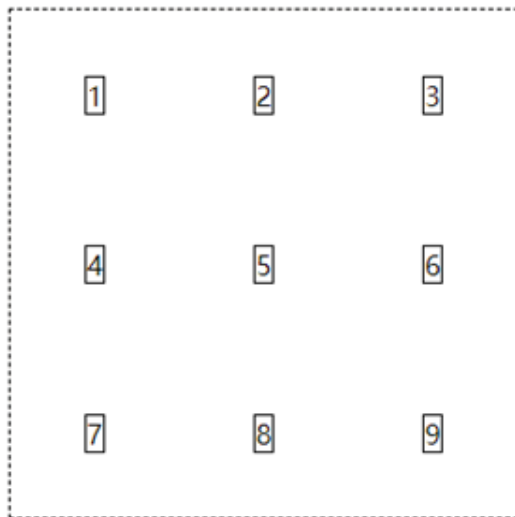
(2) align-items: 单元格内容的垂直位置

align-items : start | end | center | 默认stretch;

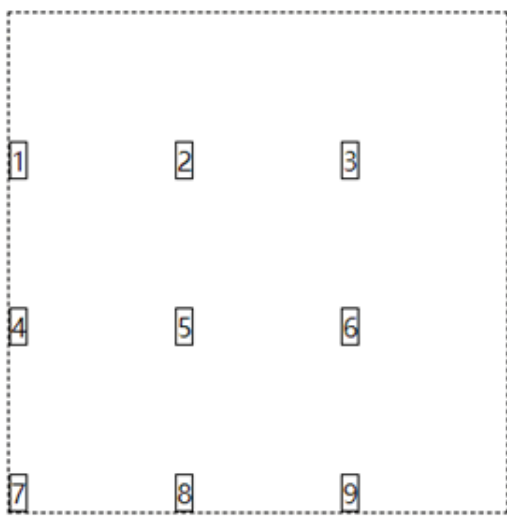
place-items: align-items属性 justify-items属性;



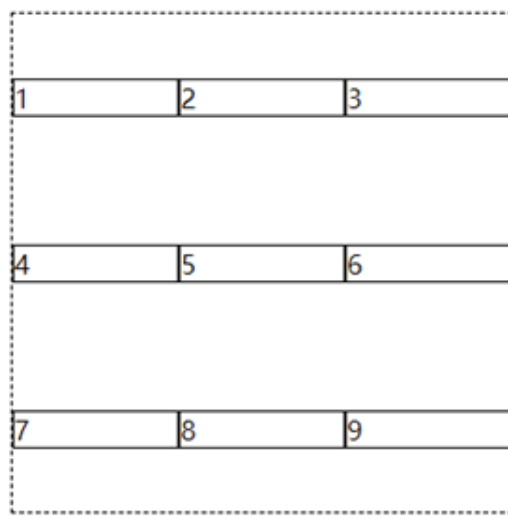
网格布局



justify-items:center;
align-items:center;



justify-items:start;
align-items:end;



justify-items:stretch;
align-items:center;



6. justify-content、align-content、place-content 属性

justify-content、align-content、place-content 属性用于设置整个内容区域在容器里面的位置。

(1) justify-content: 整个内容区域在容器里面的水平位置

justify-content: start | end | center | stretch | space-around | space-between | space-evenly;

(2) align-content: 整个内容区域的垂直位置

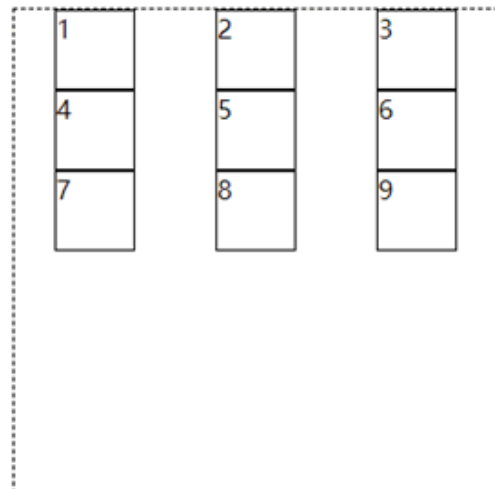
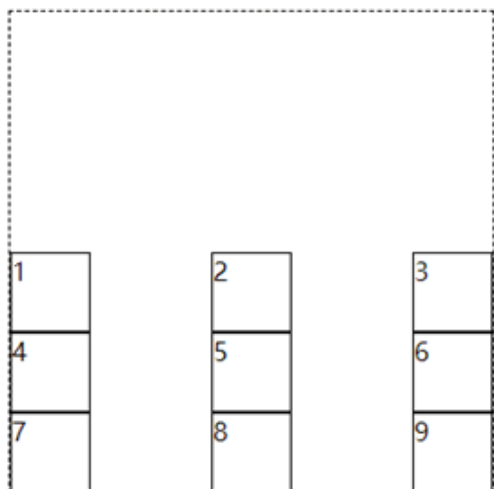
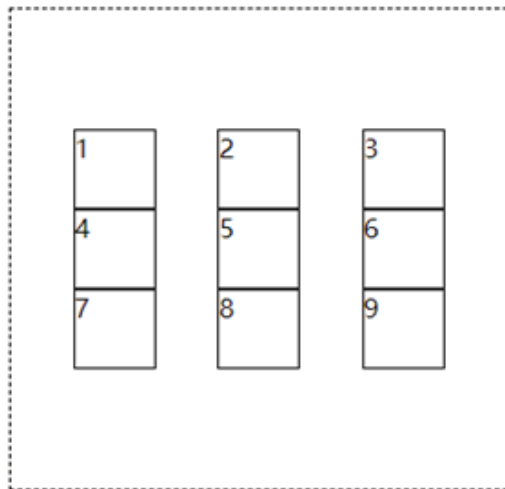
align-content: start | end | center | stretch | space-around | space-between | space-evenly;

(3) place-content :justify-content属性 align-content属性



网格布局

justify-content:space-between;
align-content:end;



justify-content:space-evenly;
align-content:center;

justify-content:space-around;
align-content:start;



网格布局

7. grid-auto-rows | grid-auto-columns | grid-auto-flow属性

这些属性表示当定义的行或列数量不够时，网格元素的自动排列方式。

- (1) grid-auto-columns属性：多出的网格元素的自动列宽。
- (2) grid-auto-rows属性：多出的网格元素的自动行高。
- (3) grid-auto-flow属性：按照先水平方向排列还是垂直方向排列。

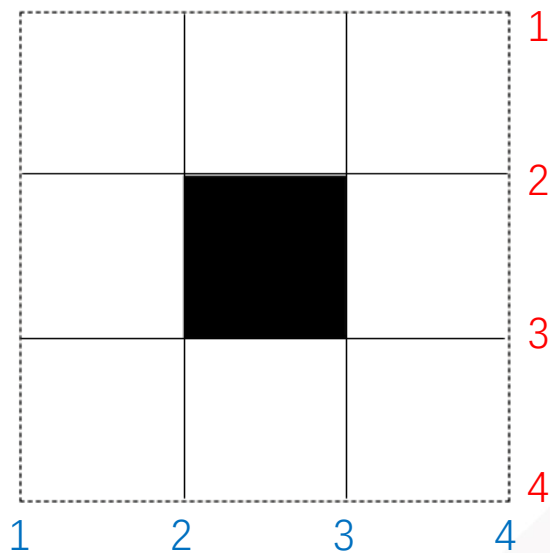


网格布局

网格元素样式



1. grid-column-*, grid-row-*属性



grid-column-start:2;
grid-column-end:3;

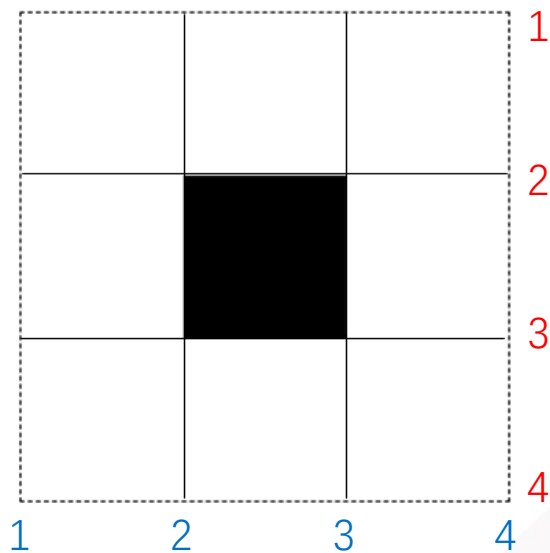
grid-row-start:2;
grid-row-end:3;



网格布局

网格元素样式

1. grid-column-*, grid-row-*属性



```
.grid-item{  
    grid-row-start:2;  
    grid-row-end:3;  
    grid-column-start:2;  
    grid-column-end:3;  
}
```

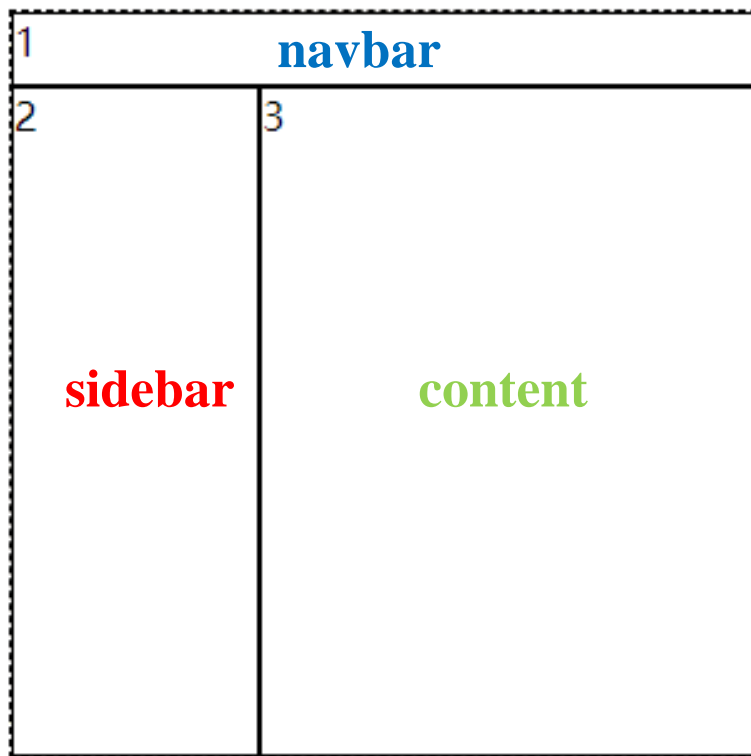
```
.grid-item{  
    grid-row:2/3;  
    grid-column:2/3;  
}
```

```
.grid-item{  
    grid-area:2/2/3/3;  
}
```




网格布局

grid-area属性

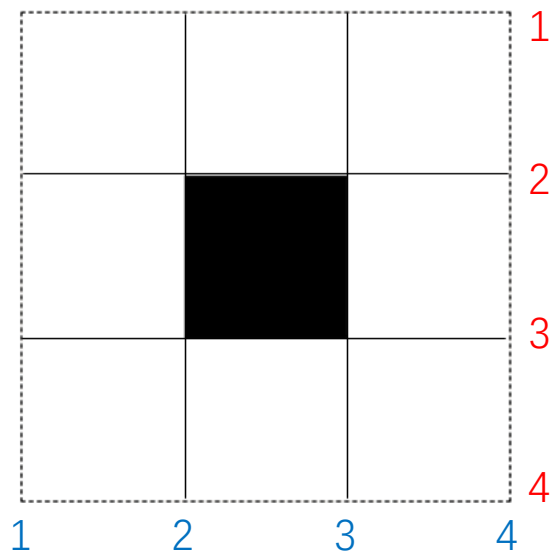


```
.grid-item:nth-child(1){  
    grid-area : navbar;  
}  
.grid-item:nth-child(2){  
    grid-area : sidebar;  
}  
.grid-item:nth-child(3){  
    grid-area : content;  
}
```



网格布局

grid-area属性



```
.grid-item{  
  grid-area:2/2/3/3;  
}
```



网格布局

网格元素样式



2. justify-self、align-self、place-self 属性

单个网格元素的对齐方式。

(1) justify-self属性：单元格内容的水平位置，用法同justify-items属性。

justify-self: start | end | center | stretch;

(2) align-self属性：单元格内容的垂直位置，用法同align-items属性。

align-self: start | end | center | stretch;

place-self: align-self属性 justify-self属性;



案例





案例



```
<div class="grid-container">  
  <div class="grid-item">1</div>  
  .....  
  <div class="grid-item">11</div>  
</div>
```



案例

```
.grid-container{
```

```
display: grid;
```

```
justify-content: center;
```

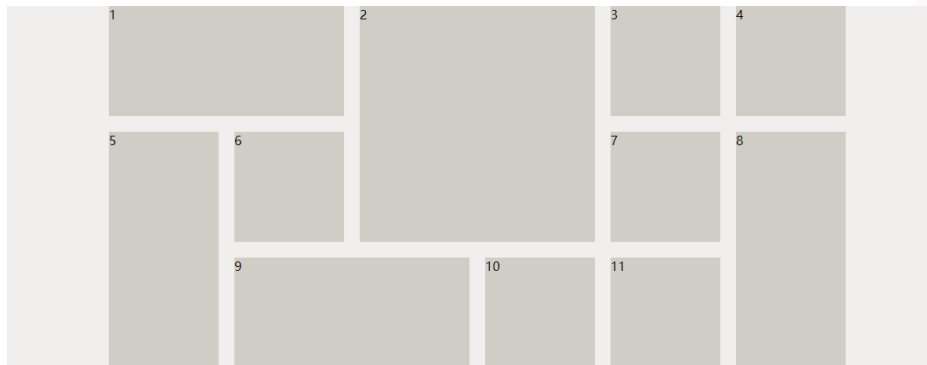
```
align-content: center;
```

```
grid-template-columns: repeat(6,140px);
```

```
grid-template-rows: repeat(3,140px);
```

```
grid-gap: 20px;
```

```
}
```





案例



```
.grid-item:nth-child(1){  
  grid-column: 1 / span 2;  
  grid-row: 1 / 2;  
}
```

```
.grid-item:nth-child(2){  
  grid-column: 3 / span 2;  
  grid-row: 1 / span 2;  
}
```