一、盒子模型

css盒模型用来封装周围的html元素，它包括内容（content）、内边框（padding）、边框（border）、外边距（margin）

- 内容(content)

显示文本和图像

css:div{

width:100px;

height:100px;

background-color:black;

} html:

我是内容

- 内边框（padding）

不让内容贴着左上角展示，撑大盒子

div{padding:50px;}

padding中一个值代表上下左右都是50px

div{padding:50px,10px;}

第一个值上下，第二个值左右

padding-left左边距

padding-right右边距

padding-top上边距

padding-bottom下边距

- 边框(border)

围绕在内边距和内容外的边框

div{border:5px solid blue;}

第一个值代表边框粗细，第二个值代表边框为实线，第三个值代表边框颜色

div{border-radius:10px;}

实现边框圆角的效果，值代表弧度

- 外边距(margin)

透明

div{margin:50px 10px;}

第一个值上下，第二个值左右

margin-right:50px;右边距

margin-left:50px;左边距

margin-top:50px;上边距

margin-bottom:50px;下边距

margin-top塌陷问题：当父元素包裹着一个子元素的时候，当给子元素设置margin-top属性时，此时只是想让子元素的边框距离父元素边框有一段距离，而却出现父元素顶端距离body这个边框出现了位移 解决方式：在父元素中加上overflow:hidden;

二、 基础布局方式

1. 空间居中布局

不管容器的大小，项目总是占据中心点

.container{

display:grid;

place-items:center;

} place-items:<align-items> <justify-items>;

align-items属性控制垂直位置，justify-items属性控制水平位置

这两个属性的值一致时，就可以合并写成一个值

左上角布局place-items:start; 左上角布局place-items:end;右下角布局

- 并列式布局

多个项目并列，如果宽度不够，放不下的项目就自动换行

css: .container{

display:flex;

flex-wrap:wrap;

justify-content:center; }

.item{

flex:0 1 150px;

margin:5px;

}

项目的初始宽度是150 px，且不可以扩大，但是当容器宽度不足150 px时，项目可以缩小

第一个值flex-grow:指定如果有多余宽度，项目是否可以扩大

第二个值flex-shrink指定如果宽度不足，项目是否可以缩小

第三个值 flex-basis项目的初始宽度flex:1 1 150 px项目始终会占满所有宽度

- 两栏式布局

css:.main{

height:800px;

display:grid;

grid-template-columns:200px auto;

grid-template-rows:100%;

}

.left{

background-color:lightblue;

width:200px;

}

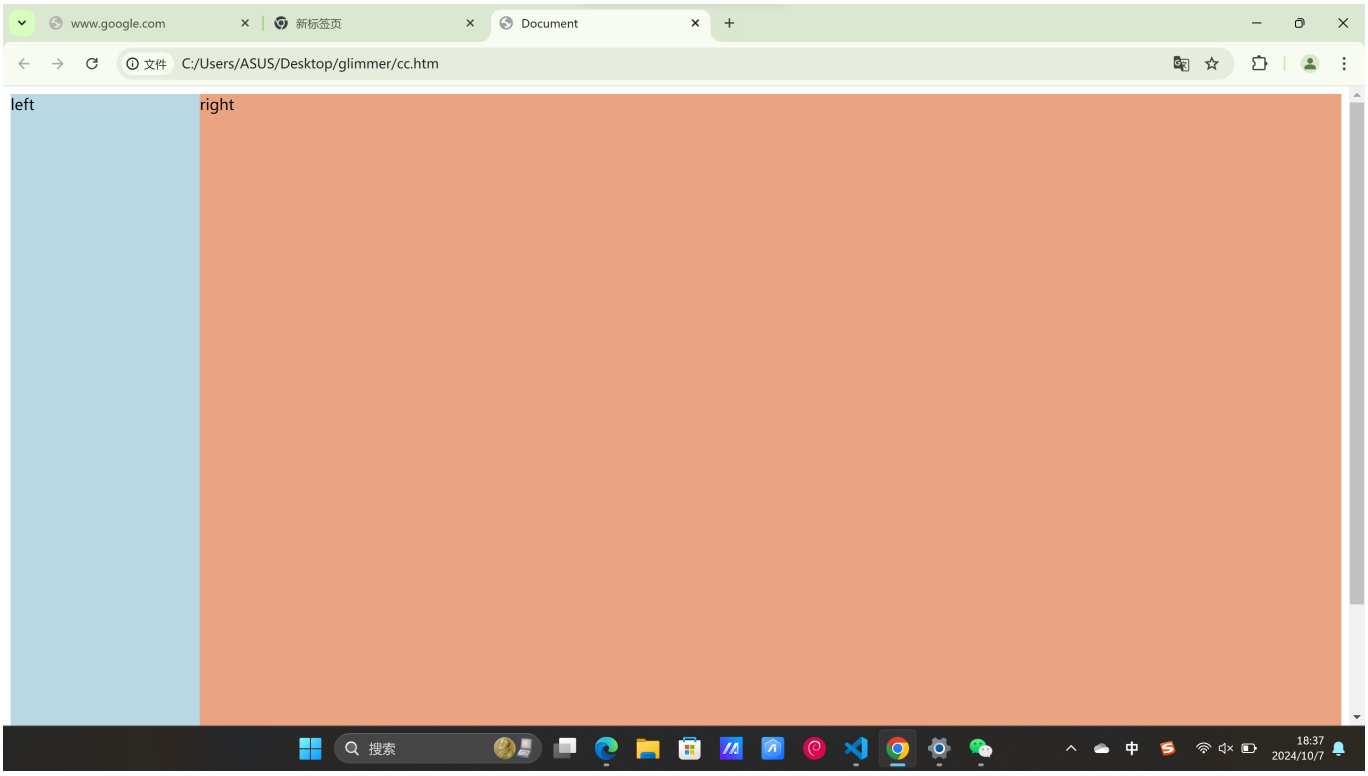
.right{

background-color:lightsalmon; }

html:

left

right



- 三明治布局

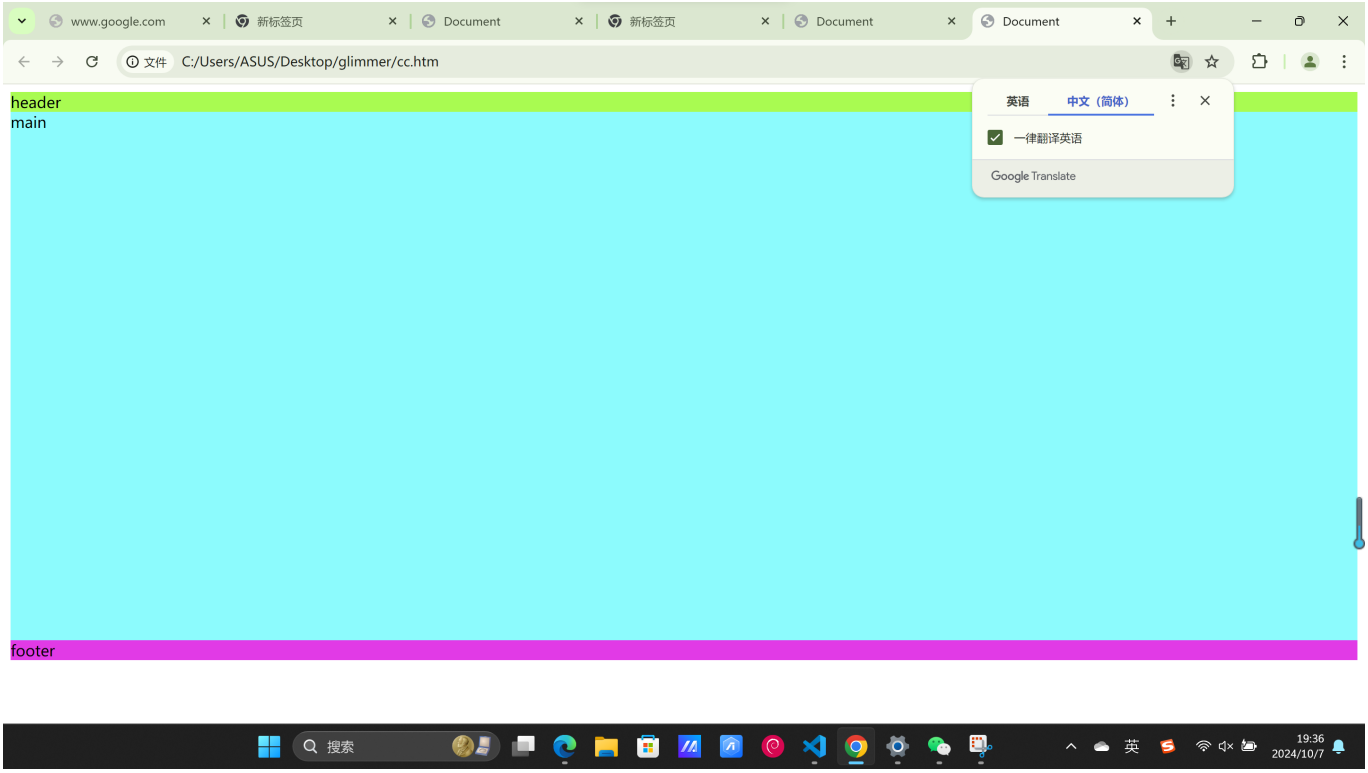
Header页眉、Main内容区、Footer Content页脚

这个布局会根据设备宽度自动适应，并且不管内容具有多少内容，页脚始终在容器底部(粘性页脚) .container{

display:grid;

grid-template-rows:auto 1fr auto;}

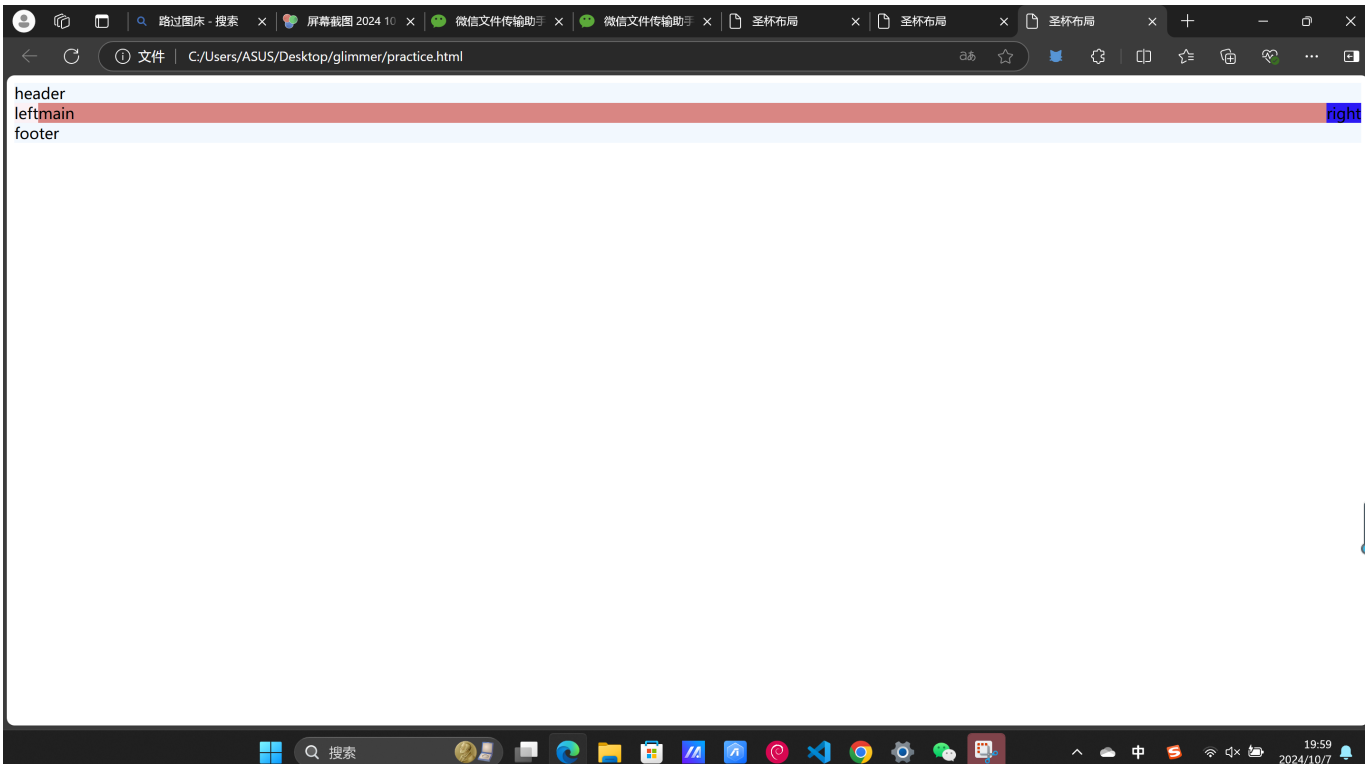
auto本来的内容高度 1fr剩余所有高度



- 圣杯布局

css:

```
.container{
    display:grid;
    grid-template:auto 1fr auto/auto 1fr auto;
    background-color: antiquewhite;
}
header{
    background-color:aliceblue;
      width:100%;
}
.left{
    background-color:lavenderblush;
}
.main{
    background-color: lightcoral;
}
.right {
    background-color: blue;
}
footer{
    background-color: aliceblue;
    width:100%;
}
html:
<header>header </header>
 <div class="container">
    <div class="left">left</div>
    <div class="main">main</div>
    <div class="right">right</div>
</div>
<footer class="footer">footer</footer>
```



三、怎么居中

1. 行内元素水平居中

   父元素加上text-align:center;

2. 块级元素水平居中

   margin：0 auto；

3. 一行文字垂直居中

   line-height:页面高；

4. 子元素在父元素中水平垂直居中（子元素宽高已知）

- 子元素相对定位

  css:

.parent{width:500px;

height:500px;

background-color:red;

} .child{ width:200px;

height:200px;

background-color:pink;

position:relative;

top:150px;

left:150px;

}

150=(500-200)/ 2

- 父元素设置padding

.parent{ padding:150px;

box-sizing:border-box;

}

宽度500px会包含它的border和padding，内容区的实际宽度是width减去border+padding的值

- 子元素添加margin

.parent{overflow:hidden;}

(解决margin-top塌陷问题) .child{margin:150px;}

- 父相子绝1

.parent{position:relative;}

.child{

position:absolute;

top:150px;

left:150px;

}

- 父相子绝2

.parent{position:relative;} .child{

position:absolute;

top:50%;

left:50%;

margin-left:-100px;

margin-top:-100px;

}

注：margin-left取负值指向左挪100px(100=200/2)

margin-top取负值指向上挪100px(100=200/2)

- 父相子绝3

.parent{position:relative;}

.child{

position:absolute;

top:0;

left:0;

bottom:0;

right:0;

margin:auto;

} 注：margin:auto;默认只能水平居中，垂直方向不行

top/left/bottom/right四个都加上也只让小方块在左上（只有top/left生效与顺序无关），但加上这4行能让margin:auto垂直方向生效

- 单元格法

.parent{

display:table-cell;

text-align:center;

vertical-align:middle;}

.child{display:inline-block;}

注：把div转化为行内块级元素，让text-align生效

四、怎么控制元素位置

属性position:static静态定位（默认）、relative相对定位、 absolute绝对定位、fixed固定定位

设置后三种定位后，可以使用4个方向值调整位置left,top,right,bottom

绝对定位和固定定位会脱离文档流

固定定位：

不管页面如何滚动，固定定位都不动

绝对定位：

每设置一个绝对定位，就有一层可用绝对定位实现元素之间的压盖 *注：设置定位之后，相对定位和绝对定位，它是相对于具有定位的父级元素进行位置调整，如果父级元素不存在定位，则继续向上逐级寻找，直到顶层文档（小盒子的容器有position属性，小盒子相对容器left或top移动，小盒子的容器无position属性，小盒子相对文档left或top移动）*

z-index:

该属性设置元素的堆叠顺序拥有更高堆叠顺序的元素，总是处于堆叠顺序较低的元素的前边

五、媒体查询

能使页面在不同终端设备下达到不同的效果，媒体查询会根据设备的大小自动识别加载不同的样式

meta name="viewport" 用户在网页上的可见区域

content="width=device-width当前宽度为设备宽度, initial-scale=1.0允许用户缩放到的默认缩放是1，即不缩放

- 不同颜色

  手机@media screen and (max-width:768px){ .box{background-color:green;} }

  平板@media screen and (min-width:768px) and (max-width:996px){ .box{background-color:blue;} }

  电脑@media screen and (min-width:996px){ .box{background-color:red;} }

- 显示与隐藏

  html: 哈哈

  啦啦

  手机@media screen and (max-width:768px){.p1,.p2{display:none;}}在手机上隐藏p1，p2

  平板@media screen and (min-width:768px) and (max-width:996px)

  {

  .p1{display:none;}

  .p2{display:block;}

  }

  平板上隐藏p1显示p2

  电脑@media screen and (min-width:996px)

  {

  .p1,.p2{

  display:block;

  }

```
}
```

- 横屏竖屏orientation:portrait/landscape;竖屏/横屏