

网易微专业之《前端开发工程师》

学习笔记

开始时间：2016.2.26

《页面架构》

CSS Reset

定义：

清除默认样式。
全局样式定义。

Reset First:

- 在项目初期确定
- 样式定义时顺序应放在第一位

布局解决方案

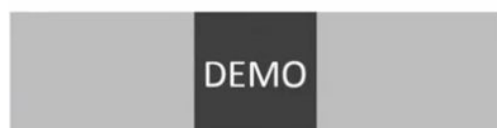
居中布局

水平居中布局（子容器和父容器宽度不定情况）



inline-block + text-align

```
<div class="parent">  
  <div class="child">DEMO</div>  
</div>
```



```
.child{  
  display: inline-block;  
}  
.parent{  
  text-align: center;  
}
```

- 兼容性好，兼容 IE6-7;

- 需要额外代码来修复 text-align 造成问题。

table + margin

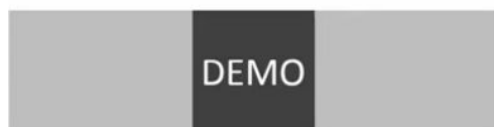
```
<div class="parent">
  <div class="child">DEMO</div>
</div>
```



```
.child{
  display: table;
  margin: 0 auto;
}
```

absolute + transform

```
<div class="parent">
  <div class="child">DEMO</div>
</div>
```



```
.parent{
  position: relative;
}
.child{
  position: absolute;
  left: 50%;
  transform: translateX(-50%);
}
```

- 居中元素不会对其他元素产生影响
- 兼容性较差

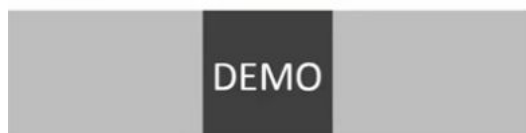
flex + justify-content

```
<div class="parent">
  <div class="child">DEMO</div>
</div>
```



```
.parent{
  display: flex;
  justify-content: center;
}
```

```
<div class="parent">
  <div class="child">DEMO</div>
</div>
```



```
.parent{
  display: flex;
}
.child{
  margin: 0 auto;
}
```

- 只需设置父元素即可实现居中布局
- 低版本不支持

垂直居中（子容器和父容器高度不定情况）



table-cell + vertical-align



```
<div class="parent">
  <div class="child">DEMO</div>
</div>
```

```
.parent{
  display: table-cell;
  vertical-align: middle;
}
```

兼容性较好（IE6-7，需要将结构改为表格结构）

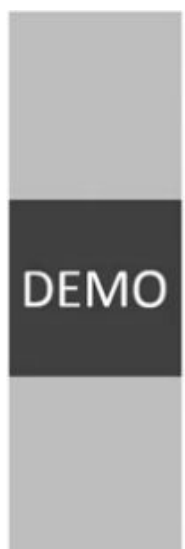
absolute + transform



```
<div class="parent">
  <div class="child">DEMO</div>
</div>
```

```
.parent{
  position: relative;
}
.child{
  position: absolute;
  top: 50%;
  transform: translateY(-50%);
}
```

flex + align-items



```
<div class="parent">
  <div class="child">DEMO</div>
</div>
```

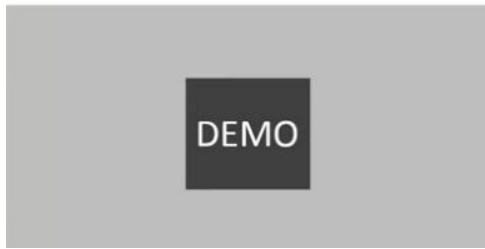
```
.parent{
  display: flex;
  align-items: center;
}
```

水平-垂直居中



inline-block + text-align + table-cell + vertical-align

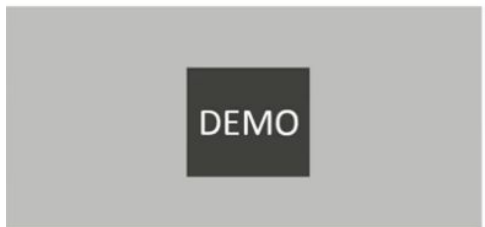
```
<div class="parent">
  <div class="child">DEMO</div>
</div>
```



```
.parent{
  text-align: center;
  display: table-cell;
  vertical-align: middle;
}
.child{
  display: inline-block;
}
```

absolute + transform

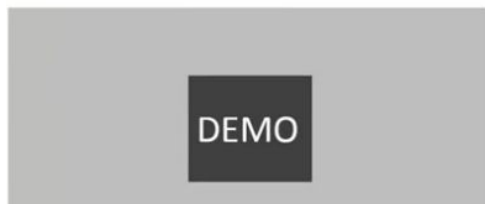
```
<div class="parent">
  <div class="child">DEMO</div>
</div>
```



```
.parent{
  position: relative;
}
.child{
  position: absolute;
  left: 50%;
  top: 50%;
  transform: translate(-50%, -50%);
}
```

flex + justify-content + align-items

```
<div class="parent">
  <div class="child">DEMO</div>
</div>
```



```
.parent{
  display: flex;
  justify-content: center;
  align-items: center;
}
```

解决方案思路：

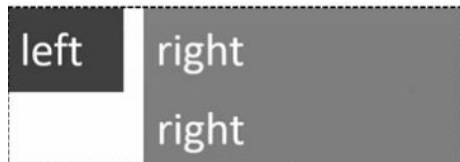
- 掌握 CSS 属性和值的特性
- 对问题进行分解

多列布局



float + margin

```
<div class="parent">
  <div class="left">
    <p>left</p>
  </div>
  <div class="right">
    <p>right</p>
    <p>right</p>
  </div>
</div>
```

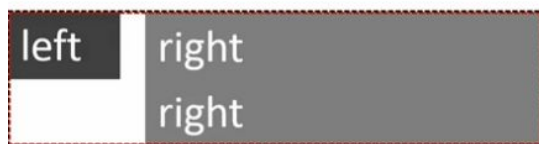


```
.left{
  float: left;
  width: 100px;
}
.right{
  margin-left: 120px;
}
```

- IE6 不兼容，会产生 3px 的 bug.
- 清除浮动时也会出现某些问题。

float + margin + (fix)

```
<div class="parent">
  <div class="left"><p>left</p></div>
  <div class="right-fix">
    <div class="right">
      <p>right</p><p>right</p>
    </div>
  </div>
</div>
```



```
.left{
  float: left; width: 100px;
  position: relative;
}
.right-fix{
  float: right; width: 100%;
  margin-left: -100px;
}
.right{
  margin-left: 120px;
}
```

- 兼容性好，兼容 IE6.

float + overflow

```
<div class="parent">
  <div class="left">
    <p>left</p>
  </div>
  <div class="right">
    <p>right</p>
    <p>right</p>
  </div>
</div>
```



```
.left{
  float: left;
  width: 100px;
  margin-right: 20px;
}
.right{
  overflow: hidden;
}
```

table



table-layout: auto | fixed

设置或检索表格的布局算法。

- auto: 默认的自动算法。布局将基于各单元格的内容，换言之，可能你给某个单元格定义宽度为 100px，但结果可能并不是 100px。表格在每一单元格读取计算之后才会显示出来，速度很慢
- fixed: 固定布局的算法。在这算法中，水平布局是仅仅基于表格的宽度，表格边框的宽度，单元格间距，列的宽度，而和表格内容无关。也就是说，内容可能被裁切
- 通常 fixed 算法会比 auto 算法高效，尤其是对于那些长表格来说。fixed 算法使得表格可以像其它元素一样一行一行的渲染。

flex



适用于小范围的布局使用，复杂布局不建议使用。



```
<div class="parent">
  <div class="left">
    <p>left</p>
  </div>
  <div class="center">
    <p>center</p>
  </div>
  <div class="right">
    <p>right</p>
    <p>right</p>
  </div>
</div>
```

```
.left,.center{
  float: left;
  width: 100px;
  margin-right: 20px;
}
.right{
  overflow: hidden;
}
```

left	center	right
		right



float + overflow

```
<div class="parent">
  <div class="left">
    <p>left</p>
  </div>
  <div class="right">
    <p>right</p>
    <p>right</p>
  </div>
</div>
```

```
.left{
  float: left;
  width: 200px;
  margin-right: 20px;
}
.right{
  overflow: hidden;
}
```

left	right
	right

```
<div class="parent">
  <div class="left">
    <p>left</p>
  </div>
  <div class="right">
    <p>right</p>
    <p>right</p>
  </div>
</div>
```

```
.left{
  float: left;
  margin-right: 20px;
}
.right{
  overflow: hidden;
}
.left p{width: 200px;}
```

left	right
	right

table


```
<div class="parent">
  <div class="left">
    <p>left</p>
  </div>
  <div class="right">
    <p>right</p>
    <p>right</p>
  </div>
</div>
```



```
.parent{
  display: table; width: 100%;
  table-layout: fixed;
}
.left,.right{
  display: table-cell;
}
.left{
  width: 200px;
  padding-right: 20px;
}
```

设置宽度由内容决定时，代码如下：

```
<div class="parent">
  <div class="left">
    <p>left</p>
  </div>
  <div class="right">
    <p>right</p>
    <p>right</p>
  </div>
</div>
```



```
.parent{
  display: table; width: 100%;
}
.left,.right{
  display: table-cell;
}
.left{
  width: 0.1%;
  padding-right: 20px;
}
.left p{width: 200px;}
```

flex

```
<div class="parent">
  <div class="left">
    <p>left</p>
  </div>
  <div class="right">
    <p>right</p>
    <p>right</p>
  </div>
</div>
```



```
.parent{
  display: flex;
}
.left{
  width: 200px;
  margin-right: 20px;
}
.right{
  flex: 1;
}
```

```
<div class="parent">
  <div class="left">
    <p>left</p>
  </div>
  <div class="right">
    <p>right</p>
    <p>right</p>
  </div>
</div>
```



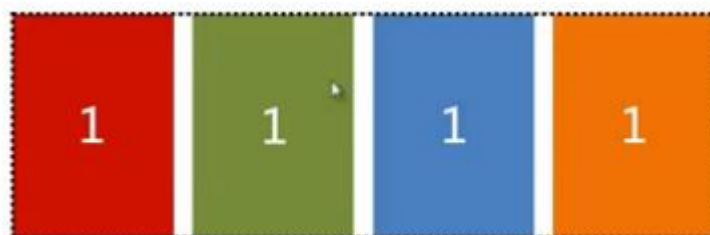
```
.parent{
  display: flex;
}
.left{
  margin-right: 20px;
}
.right{
  flex: 1;
}
.left p{width: 200px;}
```



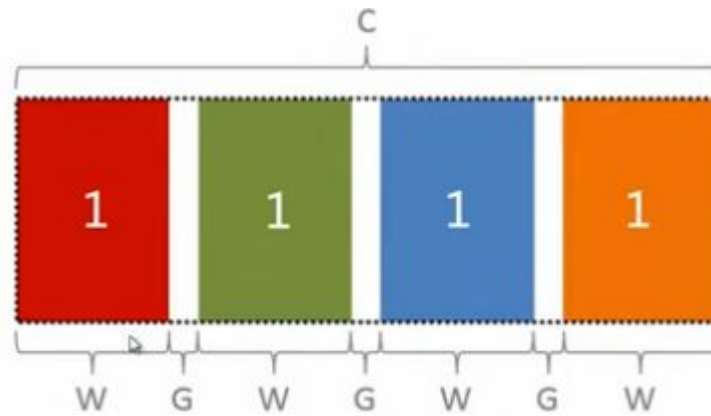
```
<div class="parent">
  <div class="left">
    <p>left</p>
  </div>
  <div class="center">
    <p>center</p>
  </div>
  <div class="right">
    <p>right</p>
    <p>right</p>
  </div>
</div>
```



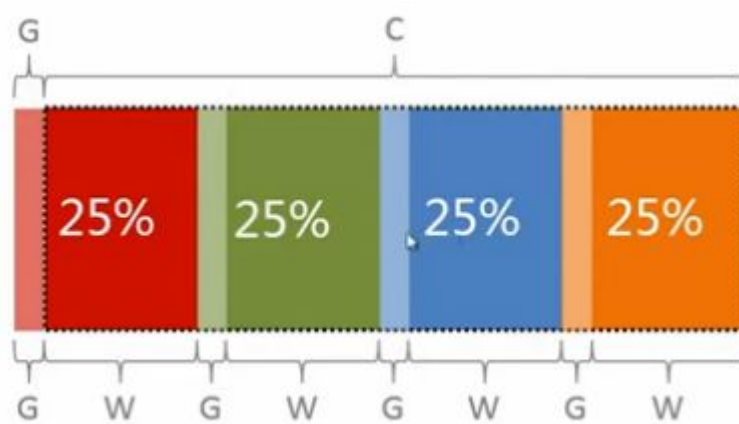
```
.left,.center{
  float: left;
  margin-right: 20px;
}
.right{
  overflow: hidden;
}
.left p,.center p{
  width: 100px;
}
```



float

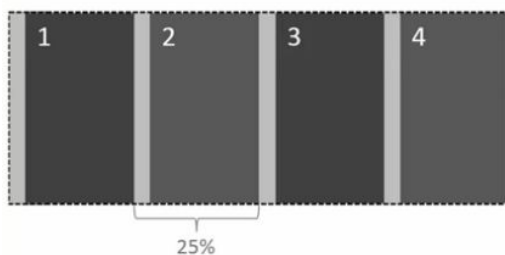


$$C = W*N + G*(N-1) \rightarrow C + G = (W+G)*N$$



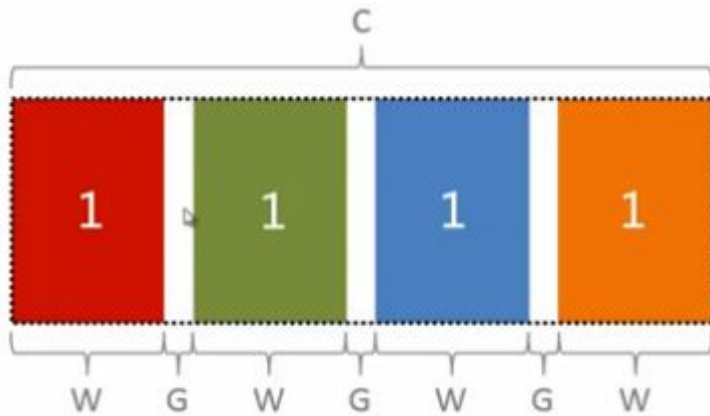
解决方案:

```
<div class="parent">
  <div class="column"><p>1</p></div>
  <div class="column"><p>2</p></div>
  <div class="column"><p>3</p></div>
  <div class="column"><p>4</p></div>
</div>
```



```
.parent{
  margin-left: -20px;
}
.column{
  float: left;
  width: 25%;
  padding-left: 20px;
  box-sizing: border-box;
}
```

table



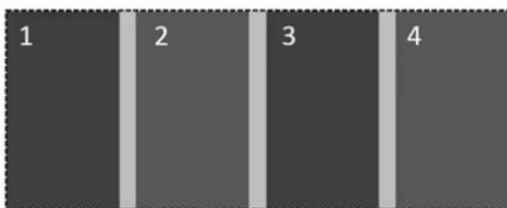
```
<div class="parent-fix">
  <div class="parent">
    <div class="column"><p>1</p></div>
    <div class="column"><p>2</p></div>
    <div class="column"><p>3</p></div>
    <div class="column"><p>4</p></div>
  </div>
</div>
```



```
.parent-fix{
  margin-left: -20px;
}
.parent{
  display: table;
  width:100%;
  table-layout: fixed;
}
.column{
  display: table-cell;
  padding-left: 20px;
}
```

flex

```
<div class="parent">
  <div class="column"><p>1</p></div>
  <div class="column"><p>2</p></div>
  <div class="column"><p>3</p></div>
  <div class="column"><p>4</p></div>
</div>
```



```
.parent{
  display: flex;
}
.column{
  flex: 1;
}
.column+.column{
  margin-left:20px;
}
```



等高布局:

table


```
<div class="parent">
  <div class="left">
    <p>left</p>
  </div>
  <div class="right">
    <p>right</p>
    <p>right</p>
  </div>
</div>
```



```
.parent{
  display: table; width: 100%;
  table-layout: fixed;
}
.left,.right{
  display: table-cell;
}
.left{
  width: 100px;
  padding-right: 20px;
}
```

flex

```
<div class="parent">
  <div class="left">
    <p>left</p>
  </div>
  <div class="right">
    <p>right</p>
    <p>right</p>
  </div>
</div>
```



```
.parent{
  display: flex;
}
.left{
  width: 100px;
  margin-right: 20px;
}
.right{
  flex: 1;
}
```

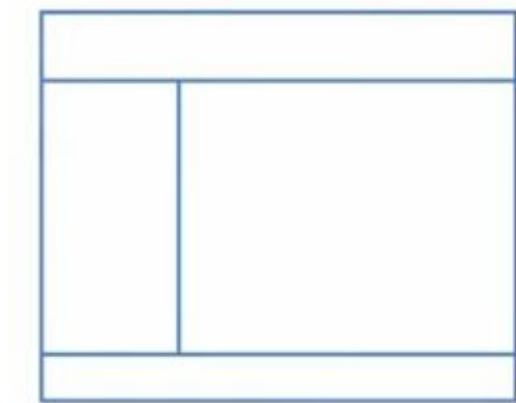
float

```
<div class="parent">
  <div class="left">
    <p>left</p>
  </div>
  <div class="right">
    <p>right</p>
    <p>right</p>
  </div>
</div>
```



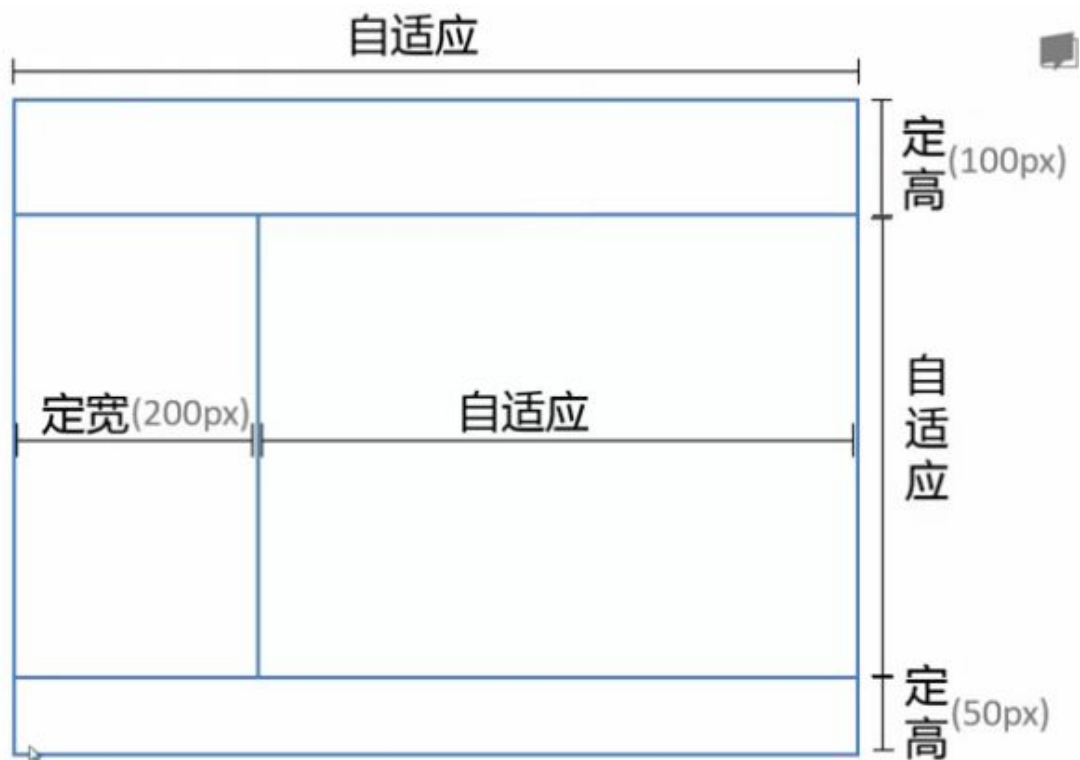
```
.parent{
  overflow: hidden;
}
.left,.right{
  padding-bottom: 9999px;
  margin-bottom: -9999px;
}
.left{
  float: left; width: 100px;
  margin-right: 20px;
}
.right{
  overflow: hidden;
}
```

全屏布局



特点：

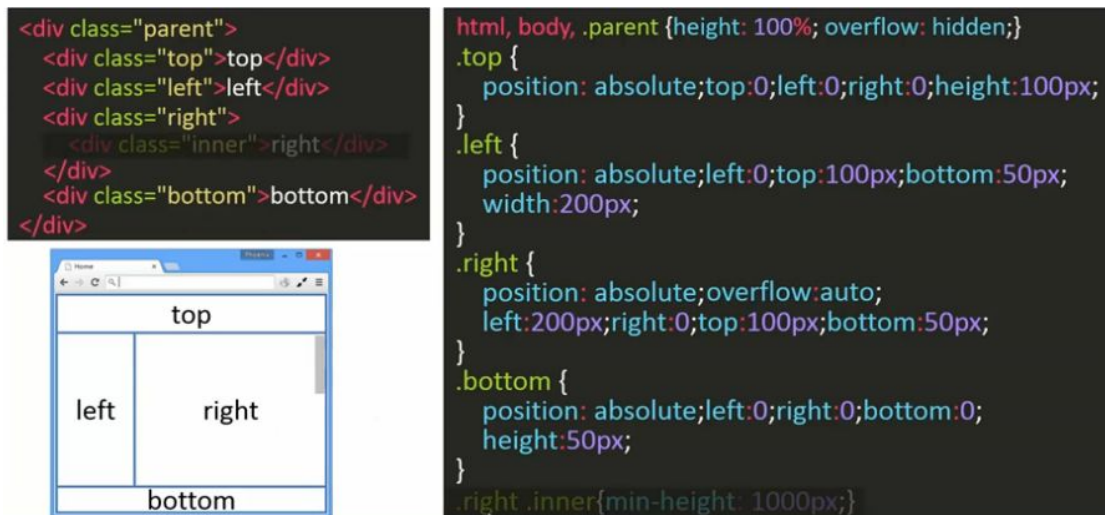
- 页面撑满浏览器窗口，当浏览器窗口变大时，页面也跟着变大。
- 滚动条出现在内容区域



实现方案：

- position
- Flex

Position

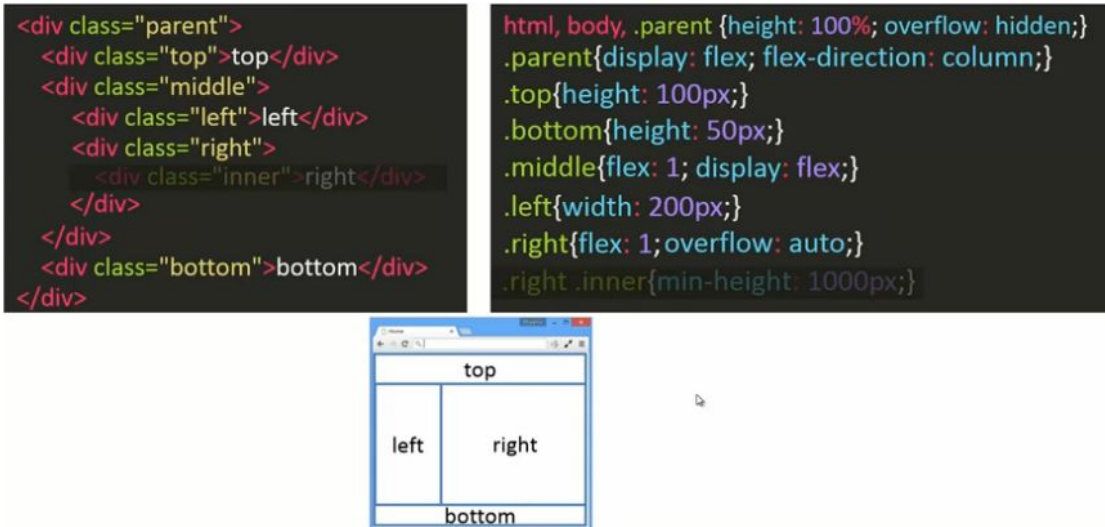


Position(兼容)

- IE6 不支持
- Hack 方案: <http://nec.netease.com/library/141027>

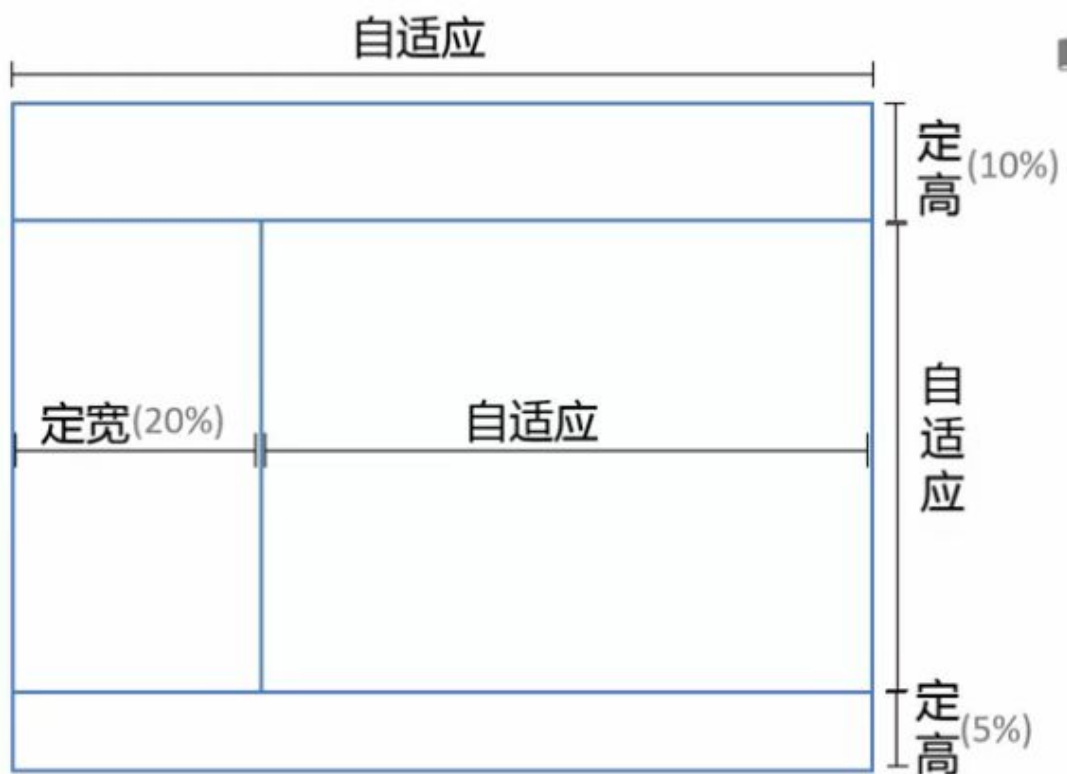


Flex



Flex(兼容)

- IE9 及以下不兼容

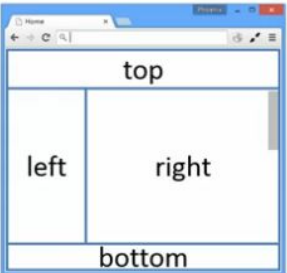


实现方案:

- position
- Flex

Position

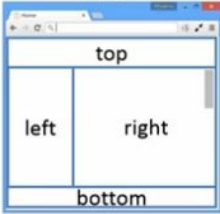
```
<div class="parent">
  <div class="top">top</div>
  <div class="left">left</div>
  <div class="right">
    <div class="inner">right</div>
  </div>
  <div class="bottom">bottom</div>
</div>
```



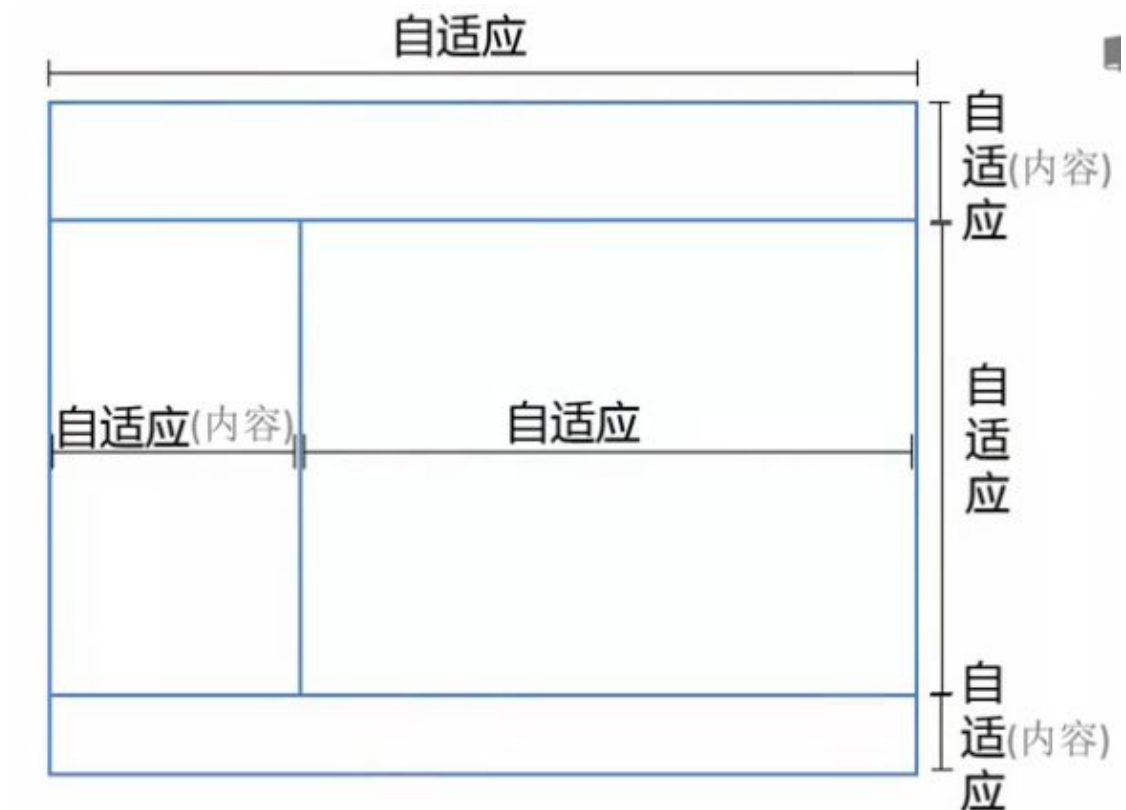
```
html, body, .parent {height: 100%; overflow: hidden;}
.top {
  position: absolute; top: 0; left: 0; right: 0; height: 10% ;
}
.left {
  position: absolute; left: 0; top: 10% ; bottom: 5% ;
  width: 20% ;
}
.right {
  position: absolute; overflow: auto;
  left: 20% ; right: 0; top: 10% ; bottom: 5% ;
}
.bottom {
  position: absolute; left: 0; right: 0; bottom: 0;
  height: 5% ;
}
.right .inner {min-height: 1000px;}
```

Flex

```
<div class="parent">
  <div class="top">top</div>
  <div class="middle">
    <div class="left">left</div>
    <div class="right">
      <div class="inner">right</div>
    </div>
  </div>
  <div class="bottom">bottom</div>
</div>
```



```
html, body, .parent {height: 100%; overflow: hidden;}
.parent {display: flex; flex-direction: column;}
.top {height: 10% ;}
.bottom {height: 5% ;}
.middle {flex: 1; display: flex;}
.left {width: 20% ;}
.right {flex: 1; overflow: auto;}
.right .inner {min-height: 1000px;}
```



实现方案

- Position ×
- Flex ✓
- Grid ✓

Flex (自适应)

```
<div class="parent">
  <div class="top">top</div>
  <div class="middle">
    <div class="left">left</div>
    <div class="right">
      <div class="inner">right</div>
    </div>
  </div>
  <div class="bottom">bottom</div>
</div>
```

```
html, body, .parent {height: 100%; overflow: hidden;}
.parent{display: flex; flex-direction: column;}
.top{height: 10%;}
.bottom{height: 5%;}
.middle{flex: 1; display: flex;}
.left{width: 20%;}
.right{flex: 1; overflow: auto;}
.right .inner{min-height: 1000px;}
```

Grid (自适应)

本笔记由西风潇潇编写，欢迎浏览博客访问更多内容：<http://www.xifengxx.com>

参考：<http://www.w3cplus.com/css3/grid.html>

<https://www.w3.org/TR/2012/WD-css3-grid-layout-20120322/>

总结：

方案	兼容性	性能	自适应
• Position	好	好	部分自适应
• Flex	较差	差	可自适应
• Grid	差	较好	可自适应