三栏布局五种方式

浮动布局

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge"
/>
    <meta name="viewport" content="width=device-width,
initial-scale=1.0" />
    <title>浮动三栏解决方案</title>
  </head>
  <body>
    <section class="layout float">
      <style media="screen">
        .layout.float .left {
          float: left;
          width: 20%;
          height: 50vh;
          background: #6b98aa;
        }
        .layout.float .center {
          background: #c5c57e;
          height: 100vh;
        }
        .layout.float .right {
          float: right;
          width: 20%;
          background: #8181ca;
        }
      </style>
      <h1>三栏布局</h1>
      <article class="left-right-center">
        <div class="left">
          <h2>左边</h2>
```

```
        </div>
        <div class="right">
          <h2>右边</h2>
        </div>
        <div class="center">
          <h2>浮动解决方案</h2>
          <p>
            这是三栏布局的浮动解决方案 这是三栏布局的浮动解决方案
            这是三栏布局的浮动解决方案 这是三栏布局的浮动解决方案
            这是三栏布局的浮动解决方案 这是三栏布局的浮动解决方案
            这是三栏布局的浮动解决方案 这是三栏布局的浮动解决方案
            这是三栏布局的浮动解决方案 这是三栏布局的浮动解决方案
            这是三栏布局的浮动解决方案 这是三栏布局的浮动解决方案
            这是三栏布局的浮动解决方案 这是三栏布局的浮动解决方案
            这是三栏布局的浮动解决方案 这是三栏布局的浮动解决方案
          </p>
        </div>
      </article>
    </section>
  </body>
</html>
```

优点：兼容性好

缺点：浮动元素脱离文档流，要做清除浮动，这个处理不好的话，会带来很多问题，比如父容器高度塌陷等

绝对定位布局

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge"
/>
    <meta name="viewport" content="width=device-width,
initial-scale=1.0" />
    <title>绝对定位三栏解决方案</title>
  </head>
  <body>
    <section class="layout absolute">
```

```
      <style>
        .layout.absolute .left-center-right > div {
          position: absolute;
        }
        .layout.absolute .left {
          left: 0;
          width: 300px;
          height: 50vh;
          background: #6b98aa;
        }
        .layout.absolute .center {
          left: 300px;
          right: 300px;
          height: 80vh;
          background: #c5c57e;
        }
        .layout.absolute .right {
          right: 0;
          width: 300px;
          height: 50vh;
          background: #7e73af;
        }
      </style>
      <h1>三栏布局</h1>
      <article class="left-center-right">
        <div class="left">
          <h2>左边</h2>
        </div>
        <div class="center">
          <h2>绝对定位解决方案</h2>
          这是三栏布局的绝对定位解决方案 这是三栏布局的绝对定位解决方案
          这是三栏布局的绝对定位解决方案 这是三栏布局的绝对定位解决方案
          这是三栏布局的绝对定位解决方案 这是三栏布局的绝对定位解决方案
          这是三栏布局的绝对定位解决方案 这是三栏布局的绝对定位解决方案
          这是三栏布局的绝对定位解决方案 这是三栏布局的绝对定位解决方案
          这是三栏布局的绝对定位解决方案 这是三栏布局的绝对定位解决方案
```

```
        </div>
        <div class="right">
          <h2>右边</h2>
        </div>
      </article>
    </section>
  </body>
</html>
```

优点：快捷，设置很方便，而且也不容易出问题。

缺点：容器脱离了文档流，后代元素也脱离了文档流，高度未知的时候，会有问题，这就导致了这种方法的有效性和可使用性是比较差的。

flexbox布局

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge"
/>
    <meta name="viewport" content="width=device-width,
initial-scale=1.0" />
    <title>flex布局三栏解决方案</title>
  </head>
  <body>
    <section class="layout flexbox">
      <style>
        .layout.flexbox {
          margin-top: 20px;
        }
        .layout.flexbox .left-center-right {
          display: flex;
        }
        /* 左边300px，剩下的宽度由中间和右边各占一份：flex: 1 */
        .layout.flexbox .left {
          width: 300px;
          background: #6b98aa;
        }
```

```
        /* 中间区域的高度设置，会把左右都拉伸，如何实现左中右不登
高？ */
        .layout.flexbox .center {
          flex: 1;
          height: 60vh;
          background: #c5c57e;
        }
        .layout.flexbox .right {
          flex: 1;
          background: #7e73af;
        }
      </style>
      <h1>三栏布局</h1>
      <article class="left-center-right">
        <div class="left">
          <h2>左边</h2>
        </div>
        <div class="center">
          <h2>flexbox解决方案</h2>
          <p>
            flexbox解决方案flexbox解决方案flexbox解决方案
flexbox解决方案
            flexbox解决方案flexbox解决方案flexbox解决方案
flexbox解决方案
            flexbox解决方案flexbox解决方案flexbox解决方案
flexbox解决方案
            flexbox解决方案flexbox解决方案flexbox解决方案
flexbox解决方案
          </p>
        </div>
        <div class="right">
          <h2>右边</h2>
        </div>
      </article>
    </section>
  </body>
</html>
```

优点：css3里新出的一个，它就是为了解决上述两种方式的不足出现的，是比较完美的一个。

缺点：IE10开始支持，但是IE10的是 `-ms` 形式的

表格布局

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge"
/>
    <meta name="viewport" content="width=device-width,
initial-scale=1.0" />
    <title>表格布局三栏解决方案</title>
  </head>
  <body>
    <section class="layout table">
      <style>
        .layout.table .left-center-right {
          width: 100%;
          height: 100px;
          display: table;
        }
        .layout.table .left-center-right > div {
          display: table-cell;
        }
        .layout.table .left {
          width: 300px;
          background: #6b98aa;
        }
        .layout.table .center {
          background: #c5c57e;
          height: 50vh;
        }
        .layout.table .right {
          width: 300px;
          background: #7e73af;
        }
      </style>
      <h1>三栏布局</h1>
      <article class="left-center-right">
```

```
        <div class="left">
          <h2>左边</h2>
        </div>
        <div class="center">
          <h2>表格布局解决方案</h2>
          表格布局解决方案表格布局解决方案表格布局解决方案布局解决方
案
          表格布局解决方案表格布局解决方案表格布局解决方案布局解决方
案
          表格布局解决方案表格布局解决方案表格布局解决方案布局解决方
案
          表格布局解决方案表格布局解决方案表格布局解决方案布局解决方
案
          表格布局解决方案表格布局解决方案表格布局解决方案布局解决方
案
        </div>
        <div class="right">
          <h2>右边</h2>
        </div>
      </article>
    </section>
  </body>
</html>
```

优点：兼容性很好，在flex布局不兼容的时候，可以尝试表格布局。当内容溢出时会自动撑开父元素。

缺点：无法设置栏边距；对seo不友好；当其中一个单元格高度超出的时候，两侧的单元格也是会跟着一起变高的，然而有时候这并不是我们想要的效果。

网格布局

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge"
/>
```

```
    <meta name="viewport" content="width=device-width,
initial-scale=1.0" />
    <title>Document</title>
  </head>
  <body>
    <section class="layout grid">
      <style>
        .layout.grid .left-center-right {
          width: 100%;
          display: grid;
          grid-template-rows: 100px;
          grid-template-columns: 300px auto 300px;
        }
        .layout.grid .left {
          width: 300px;
          background: #6b98aa;
        }
        .layout.grid .center {
          background: #c5c57e;
          height: 50vh;
        }
        .layout.grid .right {
          background: #7e73af;
        }
      </style>
      <h1>三栏布局</h1>
      <article class="left-center-right">
        <div class="left">
          <h2>左边</h2>
        </div>
        <div class="center">
          <h2>网格布局解决方案</h2>
          网格布局解决方案网格布局解决方案网格布局解决方案网格布局解
决方案网格布局解决方案
          网格布局解决方案网格布局解决方案网格布局解决方案网格布局解
决方案网格布局解决方案
          网格布局解决方案网格布局解决方案网格布局解决方案网格布局解
决方案网格布局解决方案
        </div>
        <div class="right">
          <h2>右边</h2>
        </div>
```

```
      </article>
    </section>
  </body>
</html>
```

复制

优点：CSS新标准，创建网格布局最强大和最简单的工具

缺点：兼容性不好。IE10+上支持，而且也仅支持部分属性