第八章 CSS3页面布局

Web程序设计基础

学习目标

了解网页布局方式

掌握DIV+CSS布局技术

理解多列布局方法

掌握CSS3的弹性盒布局

能用HTML5的语义化标签及DIV+CSS网页布局技术设计网页

目录

8.1 网页布局的类型

网页布局是指在页面中如何对标题、导航栏、主要内容、脚注、表单等各种主要构成元素进行一个合理的排版。

固定式

固定宽度网页布局

用固定宽度设计网页布局尺寸，这种布局有一个设置了固定宽度的外包裹，里面的各个模块也是固定宽度。

优点

- 设计师所设计的就是最终用户所看到的
- 设计简单，并且容易定制
- 在所有浏览器中宽度一样，不会受到图片、表单、视频和其他固定宽度内容的影响

缺点

- 高分辨率时，会留下很大的空白
- 屏幕分辨率过小时会出现横向滚动条
- 应该确保至少居中外包裹DIV（margin:0 auto）以保持一种显示平衡，否则对于使用大分辨率的用户，整个页面会被藏到一边去。

8.1 网页布局的类型

固定式　流　式　响应式

流式网页布局

通常被称作液态布局。通常采用**相对于分辨率大小的百分比的方式**自适应不同的分辨率。它不会像固定布局一样出现在左右两侧空白，或是被窗口框切掉，它可以根据浏览器的宽度和屏幕的大小自动调整效果，灵活多变。

优点

- 对用户更加友好，因为它能够部分自适应用户的设置
- 页面周围的空白区域在所有分辨率和浏览器下都是相同的，在视觉上更美观
- 如果设计良好，流动布局可以避免在小分辨率下出现水平滚动条

缺点

- 设计者需要在不同的分辨率下进行测试，才能够看到最终的设计效果
- 不同分辨率下图像或者视频需要不同素材
- 在屏幕分辨率跨度特别大时，内容会过大或者过小，变得难以阅读。

8.1 网页布局的类型

固定式　流　式　响应式

响应式网页布局

这是一种糅合了流式布局，再搭配媒体查询技术使用的网页布局方式。这种方式能根据不同的设备平台对内容、媒体文件和布局结构进行相应的调整与优化，从而使网站在各种环境下都能为用户提供一种最优且相对统一的体验模式。

优点

- 面对不同分辨率设备灵活性强页面周围的空白区域
- 能够快捷解决多设备显示适应问题

缺点

- 兼容各种设备工作量大，效率低下
- 代码累赘，会出现隐藏无用的元素，加载时间加长
- 一定程度上改变了网站原有的布局结构，会出现用户混淆的情况。

8.2 DIV+CSS网页布局

DIV+CSS布局是指将页面中的所有元素都看成一个盒子，通过对盒子进行CSS样式定义，并借助于DIV标签及其CSS定位实现的网页布局。

□ DIV+CSS布局的方法步骤

- 整体上对页面进行**分块**
- 按照分块设计使用div标记并定义其**样式属性**
- 对DIV盒子进行**定位**实现页面布局

1. 创建DIV

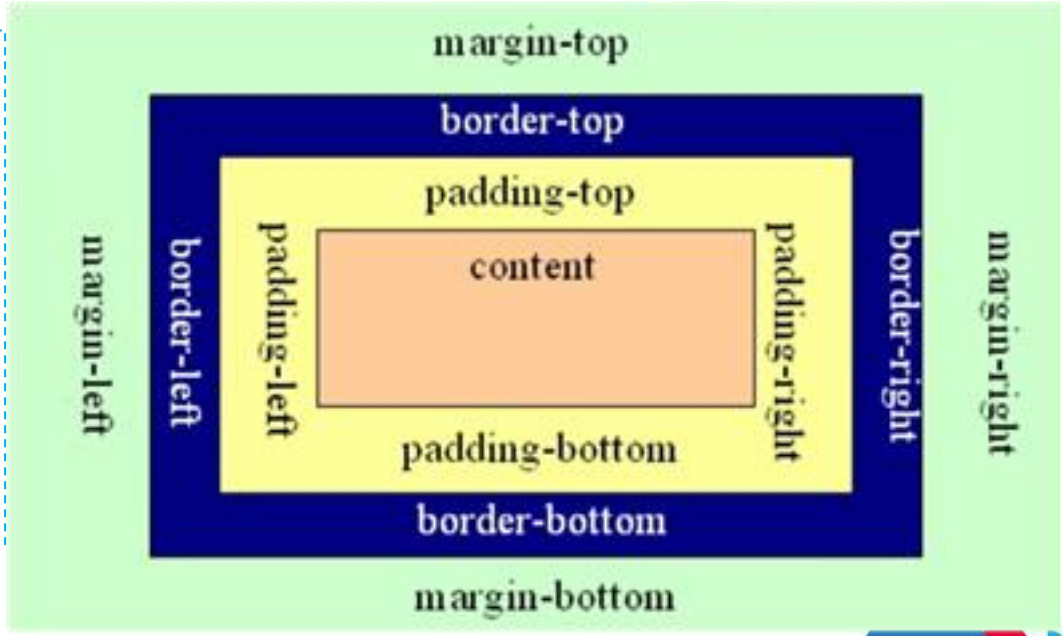
DIV标签是一个块标签，它将页面内容划分为独立的部分，以实现网页的规划和布局。

DIV标签可以加入很多属性，如id、class、align、style等，在CSS布局设计中，为了实现网页内容和表现的分离，应尽量减少align、style属性的编写次数 。因此，常用的属性也只有id和class两种了。

2. CSS盒模型

CSS盒模型是指在页面布局中，为了将页面元素合理有效地组织在一起，形成的一套完整、行之有效的原则和规范。

CSS盒是由**margin**（外边距）、**border**（边框）、**padding**（内边距）和**content**（内容）四个部分组成，此外，还有高度和宽度两个辅助属性。



content内容区

内容区（content）是盒子模型的中心，包含了盒子的主要信息内容，例如文本、图片等。内容区拥有width、height和overflow三个属性，其中overflow属性用于当content中的信息太多，并超出内容区所占的范围时，通过该属性来指定溢出内容的处理方式，属性取值如下：

值	描 述
visible	默认值，溢出的内容不会被修剪，会呈现在元素框之外
hidden	溢出的内容将不可见
scroll	溢出的内容会被修剪，但是可以通过滚动条查看隐藏部分
auto	由浏览器决定如何处理溢出部分

盒子内容区的应用案例

```
<!DOCTYPE html>
<html>
 <head>
    <title>content-overflow属性
    </title>
  <style type="text/css">
     div{
        width:400px;
        height:200px;
        overflow:scroll;
       }
    </style>
 </head>
```

```
<body>
   <div>
      <h3>定义和用法</h3>
      overflow 属性规定当内容溢出元素框时发生的情况。<hr />
      <h4>说明</h4>
      这个属性定义溢出元素内容区的内容会如何处理。如果值为 scroll，不论是否需要，用户代理都会提供一种滚动机制。因此，有可能即使元素框中可以放下所有内容也会出现滚动条。
   </div>
 </body></html>
```

[点击查看案例]

border边框

边框（border）是指围绕元素的内容和内边距的一条或多条线，拥有border-width 、border-style、 border-color 、border-radius、 border-image 、box-shadow等属性。

边框Border的属性可以分别对top（顶部）、right （右边）、bottom （底部）、left（左边） 进行设置，例如：{border-top-width：10px；}。也可以**按“上→右→下→左”**的顺序一起设置，各个属性取值之间用空格分隔，例如：{border-width：10px 20px 30px 40px;}。

- **当边框宽度有3个参数时，将按“上→左+右→下”的顺序作用到边框上**
- **当边框宽度有2个参数时，将按“上+下→左+右”的顺序作用到边框上**
- **当边框宽度有1个参数时，四个方向的边框宽度取值相同**

边框样式的应用案例

border-style属性

border-style属性取值可以为：none（无边框）、dotted（点状线）、dashed（虚线）、solid（实线）、double（双线）、groove（3D凹陷型）、ridge（3D凸起型）、 inset（ 3D凹边）、 outset（ 3D凸边）。

[点击查看案例]

```
<!DOCTYPE html>
<html> <head>
  <title>border属性</title>
  <style type="text/css">
    h3{border-style:solid dashed;}
   div{border-style:solid dashed;}
     p{border-style:outset;
       border-color:gray;
       border-width:8px;}
 #div2{border-style:inset;
       border-color:gray;
       border-width:8px;}
  </style> </head>
```

```
<body>
 <h3>边框与背景</h3>
  <div>上下边框为实线，左右边框为虚线。</div>
  <p>3D凸边型边框，灰色，线宽8像素。</p>
  <div id=div2 >3D凹边型边框，灰色，线宽8像素。</div>
 </body>
</html>
```

border-radius属性

border-radius属性可以设置左上角、右上角、右下角、左下角的圆角样式。需要设置圆角的水平半径HR和垂直半径VR的大小。

- 水平半径与垂直半径相等时，设置一个参数即可，如

  {border-top-left-radius:8px;}

- 水平半径与垂直半径不相等时，需先设置水平半径，再设垂直半径，如

  {border-top-left-radius:8px 20px;}

- 如果要对四个角的属性同时设置，

  如：{border-radius: 10px;}

  {border-radius: 10px/20px;}

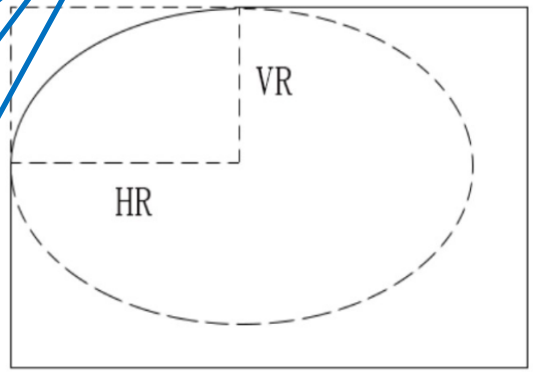
  {border-radius: 10px 20px;}

  {border-radius: 10px 20px 30px;}

  {border-radius: 10px 20px/30px 40px;}

  {border-radius: 10px 20px 30px/40px 50px;}

左上：10px/40px；右上：20px/50px
右下：30px/40px；左下：20px/50px



圆角边框的应用案例

[点击查看案例]

```
<!DOCTYPE html>
<html>
 <head>
  <title>圆角边框</title>
  <style type="text/css">
  div{height:100px;
        width:100px;
        border-style:solid;
        border-width:4px;}
  .Div1{background-color:#CCC;
        border-radius:10px 20px;}
  .Div2{background-color:#999;
   border-radius:10px 20px 30px;}
 .Div3{   background-color:#CCC;
  border-radius:10px 20px / 30px
                        40px;}
.Div4{
   background-color:#999;
   border-radius:10px 20px
                30px / 50px 60px;}
 </style>  </head>
 <body>
   <h3>CSS3圆角边框</h3>
   <div class="Div1"></div><br/>
   <div class="Div2"></div><br/>
   <div class="Div3"></div><br/>
   <div class="Div4"></div>
 </body>
</html>
```

border-image属性

border-image属性用于对元素的图像边框进行统一设置，取值情况相对较复杂。可分别设置border-image-source（图像的路径）、border-image-slice（图像分割方式，设置图像边框向内偏移）、border-image-width（图像边框宽度）、 border-image-repeat（设置图像平铺方式）。也可以统一设置，格式如下：

语法格式

```
border-image : url(图像的路径)
               图像分割方式/图像边框宽度
               图像平铺方式
```

border-image-slice属性

border-image-slice属性用于对背景图像进行分割，边框图片会被4条分隔线分成9个部分，4条分隔线分别从上、右、下、左4条边向图像内部进行偏移，具体偏移多少由border-image-slice属性的值来决定，默认情况下，元素中心区域不填充边框的图像；当提供参数fill时，元素的中心区域将被填充。

- 可设置4个参数，如

```
{border-image-slice: 10 20 30 40;}
```

- 也可设置5个参数，如

```
{border-image-slice: 10 20 30 40 fill ;}
```

- 也可缺省设置，如

```
{border-image-slice: 10;}
{border-image-slice: 10  20;}
{border-image-slice: 10 20 30;}
```

border-image-repeat属性

border-image-repeat属性用于对背景图像的平铺方式，取值可以是stretch（拉伸覆盖）、 repeat （平铺覆盖）、round （取整平铺）。

padding内边距属性

padding属性用于设置内边距，可按照“**上右下左**”的顺序对元素的四个方向的内边距进行设置。

margin外边距属性

margin属性用于设置外边距，可按照“**上右下左**”的顺序对元素的四个方向的外边距进行设置。

图像边框的应用案例

[点击查看案例]

```
<!DOCTYPE html>
<html><head>
   <title>图像边框</title>
   <style type="text/css">
    Div{border:40px solid #CCC;}
    .Div1{width:200px;height:200px;}
    .Div2{width:200px;height:200px;
border-image:url("borderImage.jpg")
            40;}
    .Div3{width:120px;height:120px;
border-image:url("borderImage.jpg")
            40/40px 35px 30px 25px;}
    .Div4{width:120px;height:120px;
border-image:url("borderImage.jpg")
   40 35 30 25/40px 35px 30px 25px;}
    .Div5{width:210px;height:200px;
border-image:url("borderImage.jpg")
            40/40px repeat;}
```

```
.Div6{width:210px;height:200px;
            margin-top:20px;
border-image:url("borderImage.jpg")
            40/40px round;}
 .Div7{width:200px;height:200px;
            margin-top:20px;
border-image:url("borderImage.jpg")
            40/40px stretch;}
</style> </head>
 <body> <h3>图像边框</h3>
    <div class="Div1"><img
src="borderImage.jpg"/></div>
    <div class="Div2"></div>
    <div class="Div3"></div>
    <div class="Div4"></div>
    <div class="Div5"></div>
    <div class="Div6"></div>
    <div class="Div7"></div>
 </body></html>
```

box-shadow属性

box-shadow属性用于对元素的**边框添加一个或多个阴影。属性取值由6个参数构成，格式如下：**

语法格式

```
box-shadow: h-shadow v-shadow [blur] [spread] [color] [inset];
```

- 仅设置h-shadow和 v-shadow 2个参数，如
  {box-shadow: 50px 50px;}
- 设置h-shadow、v-shadow、blur和color4个参数，如
  {box-shadow: 50px 50px 5px black;} blur：模糊距离
- 设置h-shadow、v-shadow、blur、spread和color5个参数，如
  {box-shadow: 10px 6px 10px 5px #AAAAAA;} spread：阴影大小
- 设置h-shadow、v-shadow、blur、spread和color5个参数，如
  {box-shadow: 10px 10px 50px 20px pink inset;}

边框阴影的应用案例

```
<!DOCTYPE html>
<html>
 <head>
  <title>边框阴影</title>
  <style type="text/css">
   body{background-color:#E9E9E9;}
     div{border-style:solid;
         border-width:4px;}
   .imageDiv{width:294px;
       padding:10px 10px 20px 10px;
       border:1px solid #BFBFBF;
       background-color:white;
       box-shadow:10px 6px 10px 5px
                  #AAAAAA ;}
  </style>
 </head>
```

[点击查看案例]

```
<body>
  <h3>CSS3边框阴影</h3>
  <div class="imageDiv">
   <img src="明向校区.jpg"
        alt="明向校门"
      width="284" height="213" />
   <p>
      太原理工大学明向校区校门
   </p>
  </div>
 </body>
</html>
```

3. 网页元素的定位

CSS 为定位和浮动提供了一些属性，利用这些属性，可以建立列式布局，甚至可以将布局的一部分与另一部分重叠。

CSS 有三种基本的定位机制：普通流、浮动和绝对定位。

- **普通流：** 除非专门指定，否则所有框都在普通流中定位。例如块级框是从上到下一个接一个地排列，框之间的垂直距离是由框的垂直外边距计算出来。
- **浮动：** 用float属性来定位
- **绝对定位：** 用position属性来定位

元素的浮动

元素的浮动是指设置了浮动属性的元素会脱离标准文档流的控制，移动到其父元素中指定位置的过程。浮动的盒子可以左右移动，直到它的边缘碰到包含框或另一个浮动盒子框的边缘。

基本语法格式

```
选择器{float:属性值;}
```

属性值	描述
left	元素向左浮动
right	元素向右浮动
none	元素不浮动（默认值）

元素浮动的应用案例1

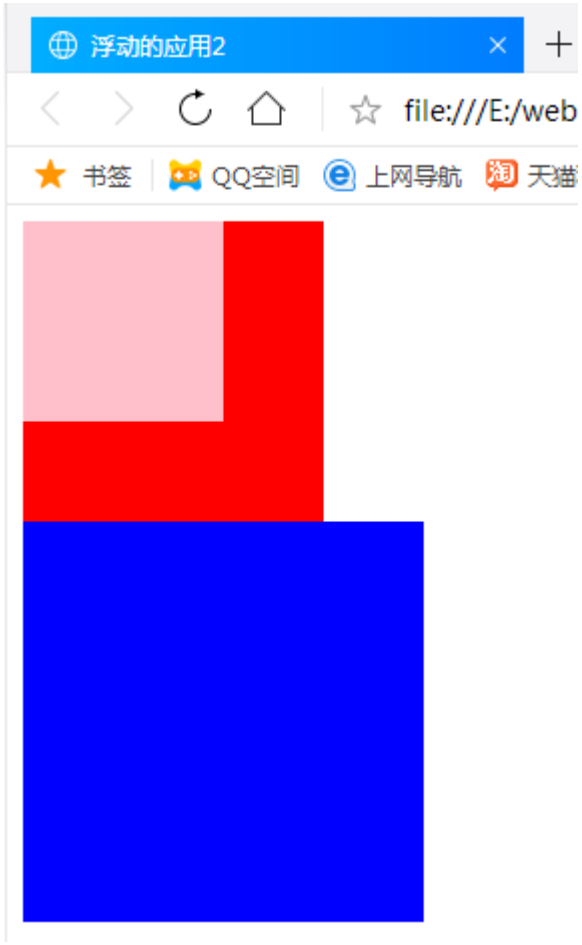
```
<!DOCTYPE html>
<html>
<head>
<title>浮动的应用1</title>
<style type="text/css">
   .one{width:100px;
        height:100px;
        background:pink;
        float: left; }
    .two{width:150px;
        height:150px;
        background:red;
        float: left; }
```

[点击查看案例]

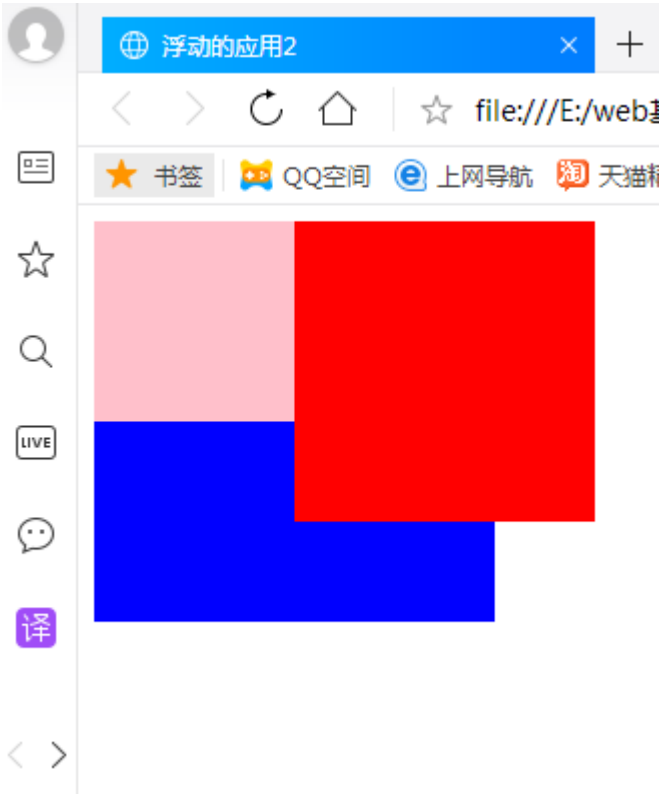
```
   .three{width:200px;
        height:200px;
        background:blue;
        float: left;}
  </style>
 </head>
 <body>
  <div class="one"></div>
  <div class="two"></div>
  <div class="three"></div>
 </body>
</html>
```

元素浮动的应用案例2

只为第1个盒子添加浮动



为第1个和第2个盒子添加浮动



当一个盒子设置了浮动而脱离标准文档流，其后的元素会自动上流。

元素的清除

由于浮动元素不再占用原文档流中的位置，所以会对页面中其他元素的排版产生影响，如果要避免这种影响，就需要对元素清除浮动。

基本语法格式

```
选择器{clear:属性值;}
```

属性值	描述
left	不允许左侧有浮动元素（清除左侧浮动的影响）
right	不允许右侧有浮动元素（清除右侧浮动的影响）
both	同时清除左右两侧浮动的影响

清除元素浮动的应用案例1（清前）

```
!DOCTYPE html>
<html>
<head>
<title>清除浮动前</title>
<style type="text/css">
div{float:left;}
</style>
</head>
<body>
  <div><img src="pic11.jpg"/>
  </div>
  <div><img src="pic22.jpg"/>
  </div>
  <div><img src="pic33.jpg"/>
  </div>
```

[点击查看案例]

```
<p>水墨画：由水和墨经过调配水和墨的浓度所画出的画，是绘画的一种形式，更多时候，水墨画被视为中国传统绘画，也就是国画的代表。也称国画，中国画。基本的水墨画，仅有水与墨，黑与白色，但进阶的水墨画，也有工笔花鸟画，色彩缤纷。后者有时也称为彩墨画。</p>
</body>
</html>
```

清除元素浮动的应用案例1（清后）

```
!DOCTYPE html>
<html>
 <head>
  <title>清除浮动前</title>
  <style type="text/css">
   div{float:left;}
   p{clear:left}
  </style>
</head>
<body>
   <div><img src="pic11.jpg"/>
   </div>
   <div><img src="pic22.jpg"/>
   </div>
   <div><img src="pic33.jpg"/>
   </div>
```

[点击查看案例]

```
<p>水墨画：由水和墨经过调配水和墨的浓度所画出的画，是绘画的一种形式，更多时候，水墨画被视为中国传统绘画，也就是国画的代表。也称国画，中国画。基本的水墨画，仅有水与墨，黑与白色，但进阶的水墨画，也有工笔花鸟画，色彩缤纷。后者有时也称为彩墨画。</p>
</body>
</html>
```

清除元素浮动的应用案例2（清前）

如果出现嵌套应用，父元素没有设置高度，而子元素设置了浮动，将对父元素的效果产生影响。例如：

```
<!DOCTYPE html>
<html>
 <head>
  <title>子元素浮动对父元素的
         影响</title>
  <style type="text/css">
   .one,.two,.three{
         float:left;
         width:100px;
         height:100px;
         margin:10px;
         background: red;
        }
```

[点击查看案例]

```
   .box{border:1px solid #ccc;
         background: blue;}
  </style></head>
 <body>
  <div>
    <div class="box">
      <div class="one">div1</div>
      <div class="two">div2</div>
      <div class="three">div3</div>
    </div>
  </div>
 </body>
</html>
```

清除元素浮动的应用案例2（清前）

Clear属性只能实现清除左右浮动的影响，此时就要考虑别的方法，例如：可在浮动元素之后添加空标记**来解决这个问题。**

[点击查看案例]

```
<!DOCTYPE html>
<html>
 <head>
  <title>子元素浮动对父元素的
         影响</title>
  <style type="text/css">
   .one,.two,.three{
         float:left;
         width:100px;
         height:100px;
         margin:10px;
         background: red;}
   .four{clear:both}
```

```
   .box{border:1px solid #ccc;
         background: blue;}
  </style></head>
<body>
 <div>
   <div class="box">
    <div class="one">div1</div>
    <div class="two">div2</div>
    <div class="three">div3</div>
    <div class="four"></div>
   </div>
 </div>
</body></html>
```

元素的定位

通过**CSS定位设置**可以实现网页元素的精确定位。元素的定位属性主要包括定位模式和边偏移两部分。

定位模式基本语法

```
选择器{position:属性值；}
```

属性值	描述
relative	相对定位，相对于其原文档流的位置进行定位
absolute	绝对定位，相对于其上一个已经定位的父元素进行定位
fixed	固定定位，相对于浏览器窗口进行定位

定位模式只能决定元素以哪一种方式定位，要确定元素的具体位置，还需要通过**边偏移**属性进行具体说明。

边偏移基本语法格

选择器{边偏移属性:属性值；}

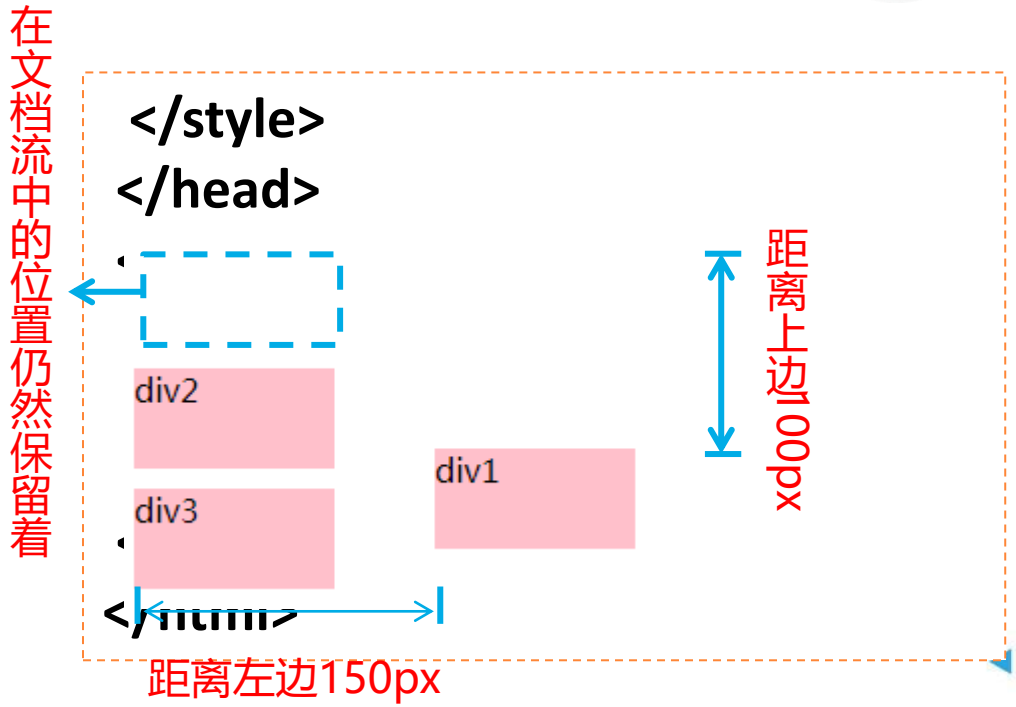
边偏移属性	描述
top	顶端偏移量，定义元素相对于其父元素上边线的距离
bottom	底部偏移量，定义元素相对于其父元素下边线的距离
left	左侧偏移量，定义元素相对于其父元素左边线的距离
Right	右侧偏移量，定义元素相对于其父元素右边线的距离

相对定位relative的应用案例

相对定位是将元素相对于它在标准文档流中的位置按指定偏移移动到新位置，移动后元素形状不变，原本位置所占空间也继续保留。

```
<!DOCTYPE html>
<html>
 <head>
  <title>相对定位的应用</title>
  <style type="text/css">
    div{width:100px;
        height:50px;
        background: pink;
        margin-bottom: 10px;}
  .div1{position: relative;
         left: 150px;
         top:100px;}
  </style>
 </head>
 <body>
  <div></div>
  <div class="div2">div2</div>
  <div class="div3">div3</div>
  <div class="div1">div1</div>
 </body>
</html>
```

[点击查看案例]



绝对定位absolute的应用案例1

绝对定位是将元素依据最近的已经定位（绝对、固定或相对定位）的父元素进行定位，若所有父元素都没有定位，则依据body根元素（浏览器窗口）进行定位。 ，移动后元素形状不变，原本位置所占空间不保留。

```
<!DOCTYPE html>
<html><head>
  <title>绝对定位的应用</title>
  <style type="text/css">
  .father{
        margin:0 auto;
        width:300px;
        height:200px;
        background:yellow;}
  .div1,.div2,.div3{
        width:100px;
        height:50px;
        background: pink;
        margin-bottom: 10px;}
  .div1{ position: absolute;
        left: 150px;
        top:100px;}
  </style></head>
  <body>
   <div class="father">
    <div class="div1">div1</div>
    <div class="div2">div2</div>
    <div class="div3">div3</div>
   </div>
  </body>
</html>
```

[点击查看案例]

绝对定位absolute的应用案例2

[点击查看案例]

```
<!DOCTYPE html>
<html><head>
   <title>绝对定位的应用</title>
   <style type="text/css">
   .father{
          margin:0 auto;
          width:300px;
          height:200px;
          background:yellow;
          position:relative;
          left:0;
          top:0;}
   .div1,.div2,.div3{
          width:100px;
          height:50px;
          background: pink;
          margin-bottom: 10px;}
   .div1{ position: absolute;
          left: 150px;
          top:100px;}
   </style></head>
   <body>
     <div class="father">
       <div class="div1">div1</div>
       <div class="div2">div2</div>
       <div class="div3">div3</div>
     </div>
   </body>
</html>
```

固定定位fixed的应用案例

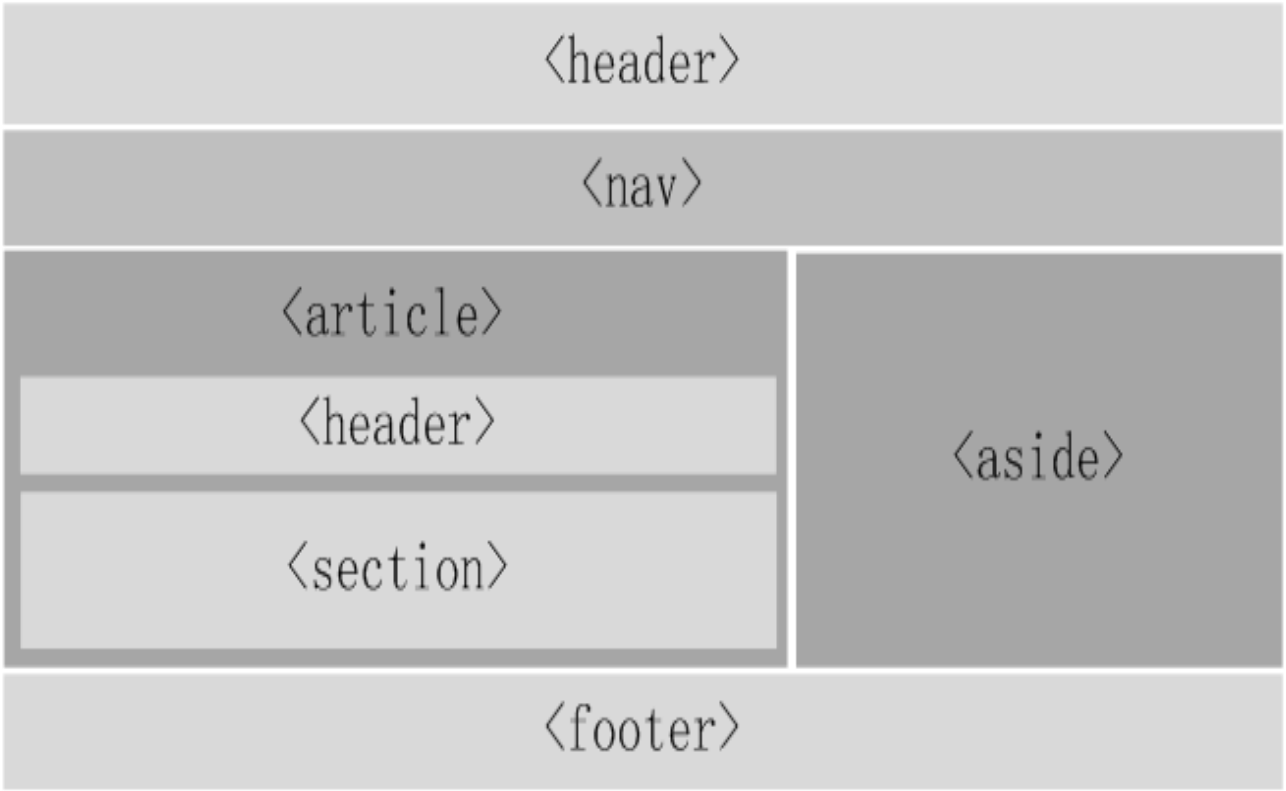
固定定位是绝对定位的一种特殊形式，它以浏览器窗口作为参照物来定义网页元素。

[点击查看案例]

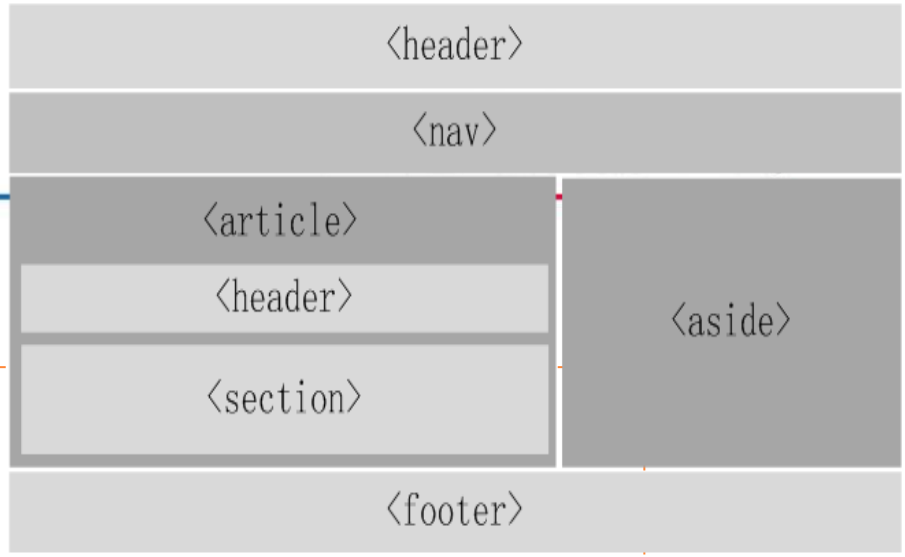
```
<!DOCTYPE html>
<html><head>
  <title>绝对定位的应用</title>
  <style type="text/css">
  .father{
        margin:0 auto;
        width:300px;
        height:200px;
        background:yellow;
        position:relative;
        left:0;
        top:0;}
  .div1,.div2,.div3{
        width:100px;
        height:50px;
        background: pink;
        margin-bottom: 10px;}
   .div1{position: fixed;
        left: 150px;
        top:100px;}
  </style></head>
  <body>
   <div class="father">
    <div class="div1">div1</div>
    <div class="div2">div2</div>
    <div class="div3">div3</div>
   </div>
  </body>
</html>
```

4. CSS页面布局实例

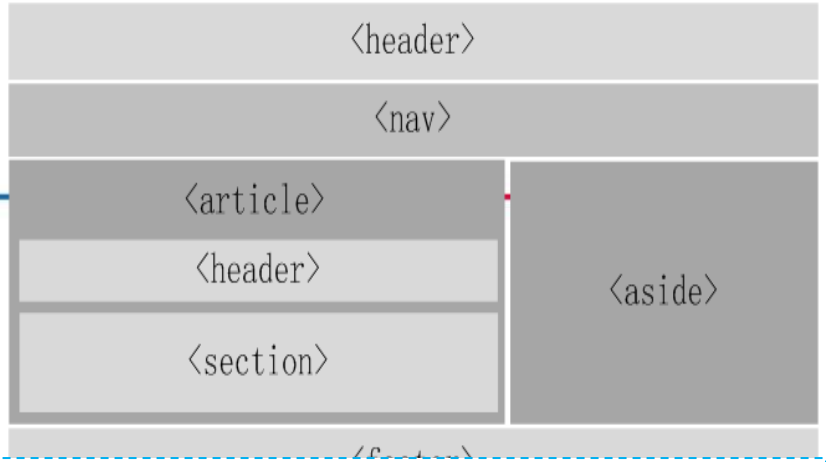
这里结合HTML5的语义化标签，使用DIV+CSS网页布局技术设计下图所示的布局。



结构设计




```
<body>
  <div id="Div0">
    <header>头部</header>
    <nav>菜单</nav>
    <div id="Div1">
      <article>
        <header id="Header1">  >文章的标题</header>
        <section>文章中的节</section>
      </article>
      <aside>侧边栏</aside>
    </div>
    <footer>底部</footer>
  </div>
</body>
```



样式设计

- **网页页面的设计**

  一般屏幕分辨率1024*768以上，适当留白，这里对总盒子的宽选择980，高度auto，适当边框（1px）、圆角（10px)设置

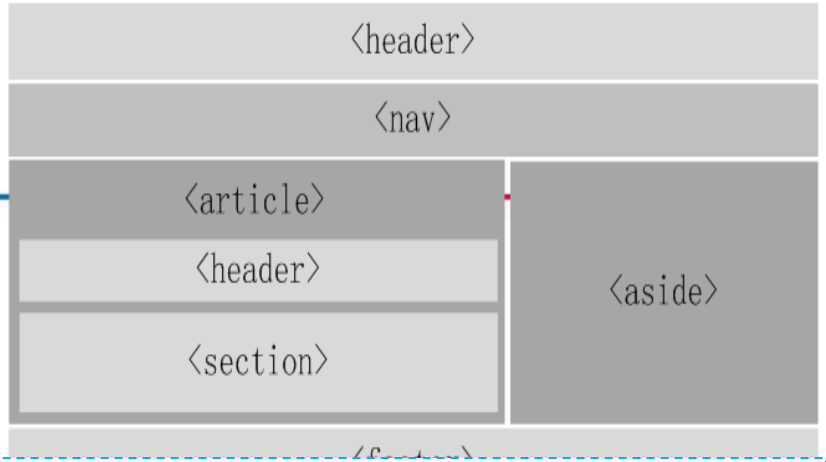
- **头部样式设计**

  宽度应满足≤总盒子宽度-2×总盒子边框线-2×总盒子内边距

  这里头部宽选择960，高度100，适当边框（1px）、圆角(10px)外边距（20px auto)设置

- **导航样式设计**

  高度可以适当减少（50Px)，其他同头部设计



样式设计

- **中部盒子页面的尺寸设计**

  此区域要放“文章页面”和“侧栏页面”，其宽度同上（960px）高度按页面内容设置，这里选（420px）

- **文章区和侧栏的设计**

  高度应满足≤中部盒子高度-2×中部盒子边框线-2×中部盒子内边距，宽度应满足：文章区宽+侧栏宽≤960- 2×中部盒子边框线

  这里高度选择400，文章区宽（600px)侧栏宽（300px）适当边框（1px）、圆角（10px)外边距（20px auto)，浮动设置

- **底部样式设计**

  宽高可以参考头部

设计结果

[点击查看案例]

```
<!DOCTYPE HTML>
<html>
 <head>
  <title>CSS页面布局实例</title>
  <link rel="stylesheet"
        type="text/css"
        href="mystyle1.css" / >
 </head>
 <body>
   <div id=“Div0">
    <header>头部</header>
    <nav>菜单</nav>
    <div id=“Div1">
      <article>
    <header id=“Header1" >文章
     的标题</header>
    <section>文章中的节
    </section>
   </article>
  <aside>侧边栏</aside>
 </div>
 <footer>底部</footer>
</div>
</body>
</html>
```

样式表

8.3 设计多列布局

使用float或position属性布局两栏或多栏内容时，会因为内容高度不同而使得底部很难对齐，CSS3中columns可以解决这个问题。

多列显示语法格式

选择器{columns :列宽度 列数；}

属性	值	描 述
columns	长度值，auto	默认值auto，用于设置列宽和列数。
columns-gap	长度值	默认值1em，用于设置列间距。
columns-rule	宽度 样式 颜色	用于设置列间边框的宽度、样式和颜色
columns-span	1或者all	默认值1，指定某个元素应该跨越多少列。

多列布局columns的应用案例

[点击查看案例]

```
<!doctype html>
<html><head>
   <meta charset="gb2312">
   <title>CSS3多列布局范例</title>
   <link rel="stylesheet“
         href="CSS/mystyle.css">
 </head>
 <body>
   <div class="news">
     <h2>大势所趋 HTML5成Web开发者最关心的技术</h2>
     <p><b>摘要：</b>移动浪潮的发展也推动了Web技术大幅前进，愈来愈多的新技术不断出现，比如HTML5里面的电池API、振动API等有关移动的新功能不断增加。据研究人员发现，这些新技术也给开发者们带来了不少烦恼。最近，在Stack Exchange上出现了一个比较热门的问题：Web开发者最头疼的问题是什么？结果并不是大家通常认为的兼容性问题，而是关于HTML5。</p>
     <p>在所有与前端开发相关的技术中，如JavaScript、CSS3、HTML5等，Web开发者们私下最经常讨论的技术领域是哪一个呢？理解这个问题就可以更好地了解当前在Web技术领域，大家所面临的挑战是什么？<img  src="images/html5.png" width="150" height="120" alt="html5" style="float:left;">为了一探究竟，英属哥伦比亚大学就“Web开发者经常挖掘的问题”展开了研究。他们在Stacke Exchange上检查了50多万个与JavaScript、HTML5、CSS等与Web有关的技术问题，对这些问题的答案进行了分析，他们总结出了哪些问题是开发者们最感兴趣和经常询问的，总结出了Web开发者们目前所面临的挑战。</p>
     <h3>JavaScript仍然称王 CSS和HTML5在迅速崛起</h3>
     <p>纵观最流行的三大Web技术（JavaScript、HTML5、CSS），JavaScript仍然是开发者们最关心的话题。<img src="images/js.jpg" width="150" height="120" alt="js" style="float:left; margin:10px;">但在Stack Exchange上，HTML5、CSS有关的问题数量也在迅速上升。研究人员表示：“在未来，他们希望这些话题的份额能够持续攀高。”</p>
     <h3>浏览器兼容性问题越来越少</h3>
     <p>研究人员发现，跨浏览器兼容性问题仍然是个热门话题，但随着时间的推移（自2009年以来），该主题的问题数量显著下降。这主要得益于更多更好的JavaScript库，比如jQuery以及浏览器厂商对W3C标准的遵守。</p>
     <h3>HTML5和JavaScript新功能给开发者带来不少烦恼</h3>
     <p>在Stack Exchange上，有关HTML5和JavaScript新功能和新元素（如localStorage）问题数量一直以来都比较稳定，并且有不断上升的趋势。研究人员表示，即使是专家级别的程序员也会对新添加到HTML5和JavaScript的功能感到困惑，对于这些新功能，他们需要更好的API支持。/p>
     <h3>Web技术在移动领域的重要性愈加凸显</h3>
     <p>最后，研究人员发现，自2009年以来，关于HTML5的大部分问题都与移动有关，比如设备检测、本地存储等，并且有关CSS3的viewport meta标签问题的数量也日益增多。这些发现充分说明了Web技术在移动领域的地位更加重要。</p>
     （编译：张红月 责编：夏梦竹）来自：InforWorld
   </div>
 </body>
</html>
```

样式表

8.4 CSS3的弹性盒布局

弹性盒子是 CSS3 的一种新的布局模式。CSS3 弹性盒是一种当页面需要适应不同的屏幕大小以及设备类型时确保元素拥有恰当的布局方式。引入弹性盒布局模型的目的是提供一种更加有效的方式来对一个容器中的子元素进行排列、对齐和分配空白空间。

弹性盒基本格式

```
选择器{display :flex;  }
```

display 属性用于设置显示的类型。有很多属性：none(不显示）、block(块级元素）、inline（内联元素）、inline-block(行内块元素)、list-item(列表显示）、table（块级表格）、inline-table （内联表格）、 flex（弹性布局显示）等。

属性	值	描 述
display	flex	指定弹性容器的内容显示为弹性样式。
flex-flow	方向值 换行设置	默认值row nowrap，是flex-direction 和 flex-wrap 属性的复合属性。
flex-direction	row |row-reverse| Column|column-reverse	用于设置弹性盒子中子元素的排列方向，属性值可以是：行排列、反向行排列、列排列、反向列排列，默认row
flex-wrap	nowrap 或wrap	默认值nowrap，设置弹性盒子的子元素超出父元素容器时是否换行。
flex	扩展量 收缩量 基准值	定义弹性盒子中子元素如何分配空间，可同时设置flex-grow, flex-shrink, flex-basis三个子属性。默认取值：0 1 auto
order	顺序值	定义弹性盒子中子元素排列顺序，数值越小排列越靠前，默认值0
justify-content	flex-start|flex-end| Center|space-between|space-around	定义弹性盒子中子元素在横轴上的对齐方式，默认取值：flex-start
align-items	Stretch|centerflex-start| flex-end|baseline	定义弹性盒子中子元素在纵轴上的对齐方式。默认取值：stretch
align-content	flex-start|flex-end| Center|space-between|space-around	设置各行的对齐方式，默认取值：flex-start
align-self	Stretch|center|flex-start|	定义子某个元素单独在纵轴方向上的对齐方式，

弹性盒子flex的应用案例

[点击查看案例]

```
<!doctype html>
<html>
 <head>
  <meta charset="gb2312">
  <title>弹性盒子范例</title>
  <link rel="stylesheet"
        href="CSS/mystyle.css">
 </head>
 <body>
  <div class="flex-container">
   <div class="flex_item_left">flex item left</div>
   <div class="flex_item_center">flex item
center</div>
   <div class="flex_item_right“>flex item right</div>
  </div>
  </body>
</html>
```

样式表

8.5 设计导航栏

在网页布局中，导航栏是页面不可或缺的重要部分，导航栏一般位于页面上方或两侧最突出的位置，由一系列的超链接热点所组成。制作网页导航栏，需要借助于HTML中的无序列表标签ul和li。

- **用ul和li制作导航要注意两个问题：**
  - 去掉项目符号，可添加属性

    text-decoration: none

    或者 list-style-type： none
  - 将列表项的改为行排列，即内联行显示，可添加属性

    display:inline;

    或者 display:inline-block;

设置导航的应用案例

[点击查看案例]

```
<!doctype html>
<html><head>
   <meta charset="utf-8">
   <title>简单的DIV+CSS导航栏</title>
   <link rel="stylesheet" href="CSS/mystyle.css">
 </head>
 <body>
    <nav class="top_menu">
      <ul>
        <li><a href="index.html">首页</a></li>
        <li><a href="#">HTML教程</a></li>
        <li><a href="#">CSS基础</a></li>
        <li><a href="#">CSS开发工具</a></li>
        <li><a href="#">CSS特效</a></li>
     </ul>
   </nav>
 </body></html>
```

样式表

本章结束