一、盒子的CSS排版方法

- 标准流方式排版

是“标准文档流”的，是指在不使用其他与排列和定位相关的特殊CSS规则时各种页面元素默认的排列规则。

  - 块级元素(**block level**)：将从上到下一个接一个地排列，左右撑满。
  - 内联元素(**inline**)：相邻元素之间横向排列，到最右端自动换行。

- 浮动排版
  - 使用**float**属性进行浮动设置。
- 定位排版
  - 使用**position**属性指定定位类型。

1. 盒子的浮动排版

- 在标准流排版中，一个块级元素在水平方向会自动伸展，直到包含它的父级元素的边界；在垂直方向上和兄弟元素依次排列，不能并排。如果排版时需要改变块级元素的这种默认排版，需要使用浮动排版或定位排版。
- 在浮动排版中，块级元素的宽度不再自动伸展，而是根据盒子里放置的内容决定其宽度，要修改该宽度，可设置元素的宽度和内边距。
- 浮动的盒子可以向左或向右移动，直到它的外边缘碰到包含框或另一个浮动框的边框为止。

浮动框不在文档的标准流中，所以文档的标准流中的块框表现得就像浮动框不存在一样。

(1) 盒子的浮动设置

- 盒子的浮动使用CSS属性float来设置。
- float属性取值：
  - none：盒子不浮动
  - left：盒子浮在其父级元素的左边
  - right：盒子浮在其父级元素的右边
- float属性的值指出了盒子是否浮动及如何浮动。当该属性等于left或right引起对象浮动时，盒子将被视作块级元素(block-level)，即display属性等于block。
- 默认情形下，此时盒子的宽度不再伸展，将根据盒子里面放的内容来决定其宽度。同时， 浮动元素将脱离标准流，此时文档流中的块级元素表现得就像浮动元素不存在一样，所以如果不正确设置外边距，将会发生文档流中的元素和浮动元素重叠现象。

(2)盒子的浮动清除

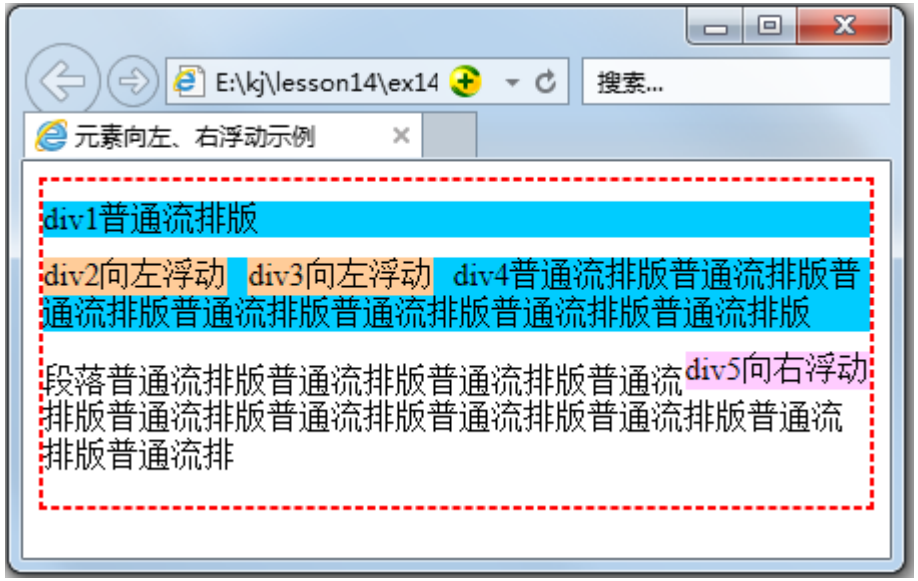
- 在排版页面元素时，可以指定元素的旁边不允许出现浮动元素，这个功能可通过设置**clear**属性来实现。
- **clear**属性取值：

值	描述
left	在左侧不允许浮动元素。
right	在右侧不允许浮动元素。
both	在左右两侧均不允许浮动元素。
none	默认值。允许浮动元素出现在两侧。

- 设置了**clear**属性的元素将下沉到浮动元素的后面

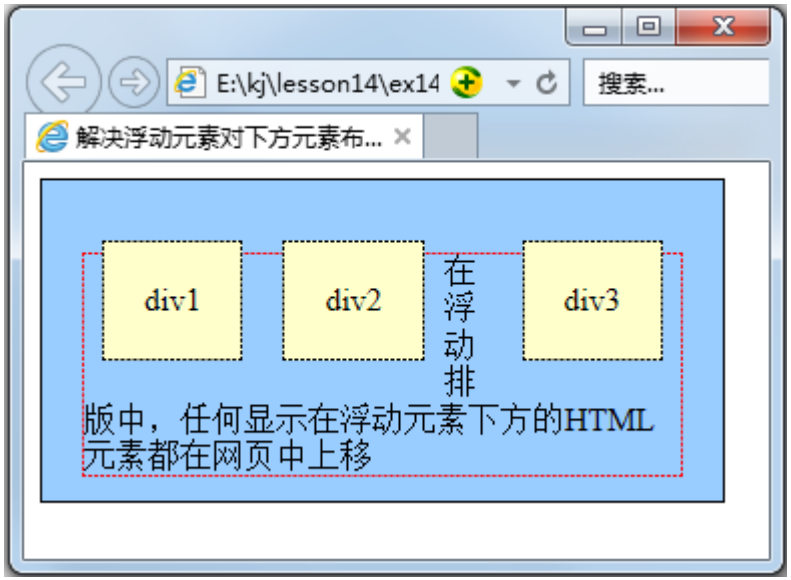
元素浮动示例

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>元素向左、右浮动示例</title>
<style>
div{margin-top:10px;}
.father{
    margin:0px;
    border:2px dashed red;
}
.son1,.son4{
    background-color:#0CF;
}
.son2,.son3{
    float:left;
    margin-right:10px;
    background-color:#FC9;
}
.son5{
    float:right;
    background-color:#FCF;
}
</style>
</head>
<body>
   <div class="father">
    <div class="son1">div1普通流排版</div>
    <div class="son2">div2向左浮动</div>
    <div class="son3">div3向左浮动</div>
    <div class="son4">div4普通流排版普通流排版普通流排版普通流排版普通流排版普通流排版</div>
    <div class="son5">div5向右浮动</div>
    <p>段落普通流排版普通流排版普通流排版普通流排版普通流排版普通流排版普通流排版普通流排版普通流排</p>
  </div>
</body></html>
```



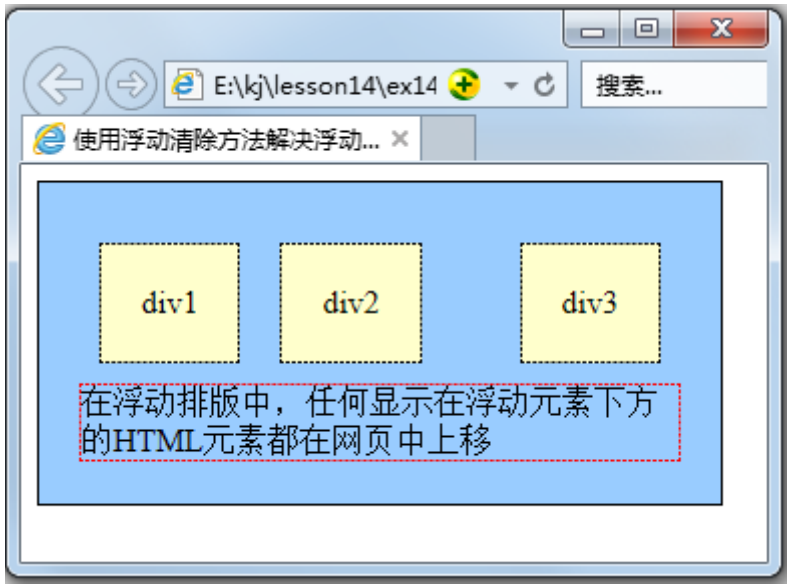
浮动元素对下方元素布局的影响示例

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>浮动元素对下方元素布局的影响示例</title>
<style>
.father{
    width:300px;
    height:120px;
    padding:20px;
    background:#9CF;
    border:1px solid black;
}
.son1,.son2,.son3{
    padding:20px;
    margin:10px;
    background:#FFFFCC;
    border:1px dashed black;
}
.son1,.son2{float:left;}
.son3{float:right;}
p{
    border:1px dashed red;
}
</style>
</head>
<body>
  <div class="father">
  <div class="son1">div1</div>
  <div class="son2">div2</div>
  <div class="son3">div3</div>
  <p>在浮动排版中，任何显示在浮动元素下方的HTML元素都在网页中上移</p>
</div>
</html>
</html>
```



元素浮动清除示例

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>使用浮动清除方法解决浮动元素对下方元素布局的影响示例</title>
<style>
.father{
    width:300px;
    height:120px;
    padding:20px;
    background:#9CF;
    border:1px solid black;
}
.son1,.son2,.son3{
    padding:20px;
    margin:10px;
    background:#FFFFCC;
    border:1px dashed black;
}
.son1,.son2{float:left;}
.son3{float:right;}
p{
    border:1px dashed red;
    clear:both;
}
</style>
</head>
<body>
  <div class="father">
  <div class="son1">div1</div>
  <div class="son2">div2</div>
  <div class="son3">div3</div>
  <p>在浮动排版中，任何显示在浮动元素下方的HTML元素都在网页中上移</p>
</div>
</html>
```



2. 盒子的定位

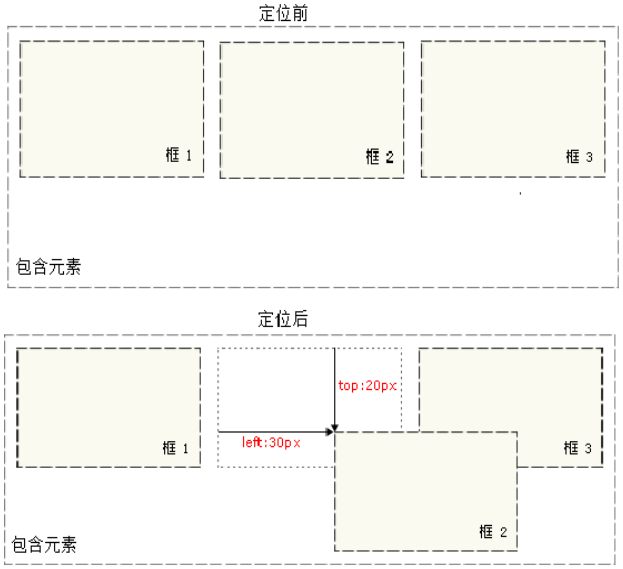
- 所谓“定位”：指的将某个元素放到某个位置。
- 使用以下CSS属性可进行元素定位

属性	属性值	描 述
position	static|relative|absolute|fixed	把元素放置到一个默认的丨相对的丨绝对的丨固定的位置中
top	value|%	指定定位元素在垂直方向上与参照元素上边界之间的偏移，正值表示向下偏移，负值表示向上偏移，0值表示不偏移
right	value|%	指定定位元素在水平方向上与参照元素右边界之间的偏移，正值表示向左偏移，负值表示向右偏移，0值表示不偏移
left	value|%	指定定位元素在水平方向上与参照元素左边界之间的偏移，正值表示向右偏移，负值表示向左偏移，0值表示不偏移
bottom	value|%	指定定位元素在垂直方向上与参照元素下边界之间的偏移，正值表示向上偏移，负值表示向下偏移，0值表示不偏移
overflow	visible	默认值，当元素的内容溢出其区域时内容会呈现在元素框之外
	hidden	当元素的内容溢出其区域时，溢出内容不可见
	auto	当元素的内容溢出其区域时，浏览器会显示滚动条
	scroll	不管元素的内容是否溢出其区域，浏览器都会显示滚动条
	inherit	规定从父元素继承overflow属性的值
z-index	number(负数、0、正数)	设置元素的堆叠顺序，值大的元素堆叠在值小的元素上面

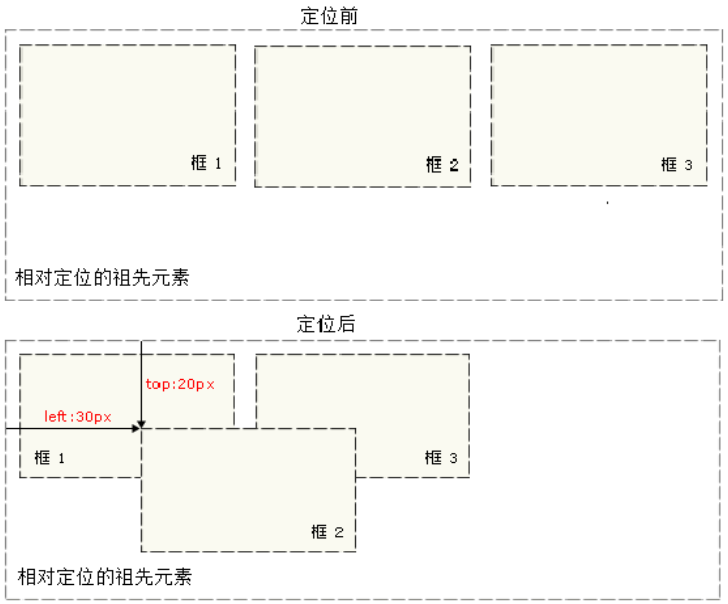
- **position**取不同的值，将实现不同类型的定位方式：

✓ **static**：静态定位，这是默认的属性值，元素将按照标准流进行布局，一般不需要设置。

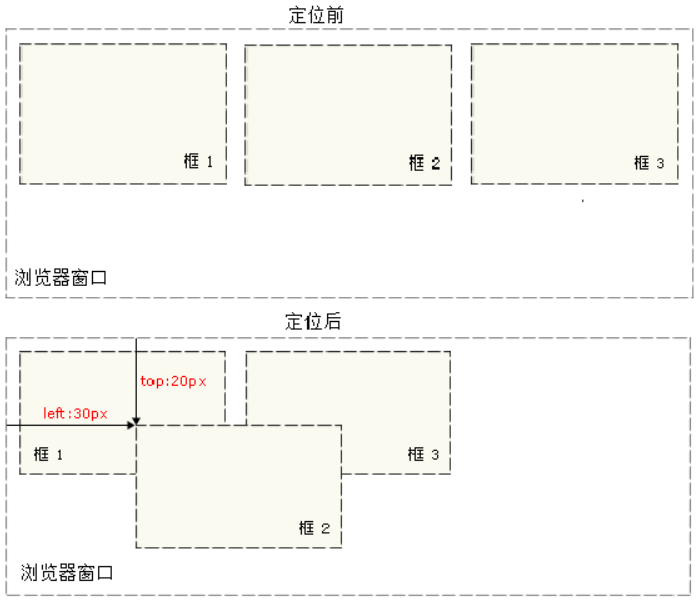
✓ **relative**：相对定位。设置元素相对于它在标准流中的位置进行偏移。不管元素是否偏移，它原来所占的空间仍然保留，没有脱离文档流。相对定位移动元素时有可能会导致它覆盖其他的元素。示意图如右图所示：



- **absolute**：绝对定位。绝对定位的元素会基于相对于距离它最近的那个已定位(相对/绝对)的祖先元素偏移某个距离，如果元素没有已定位的祖先元素，那么它的偏移位置将相对于最外层的包含框。绝对定位的元素脱离文档流，原来所占的空间不保留。元素定位后生成一个块级框，而不论原来它在普通流中生成何种类型的框。绝对定位移动元素时有可能会导致它覆盖其他的元素。示意图如右图所示：

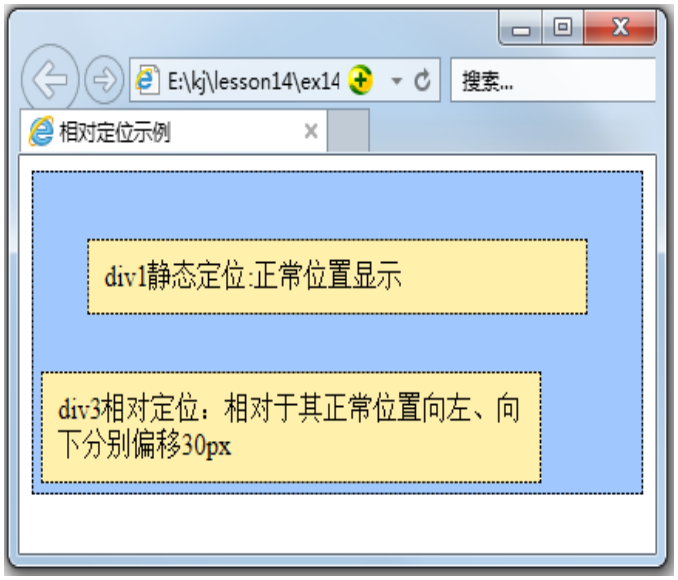


✓**fixed**：固定定位。固定定位的元素相对于浏览器窗口偏移某个距离，且固定不动，当拖动浏览器窗口的滚动条时，依然保持对象位置不变。和绝对定位类似，固定定位的元素脱离文档流，原来所占的空间不保留示意图如下图所示：



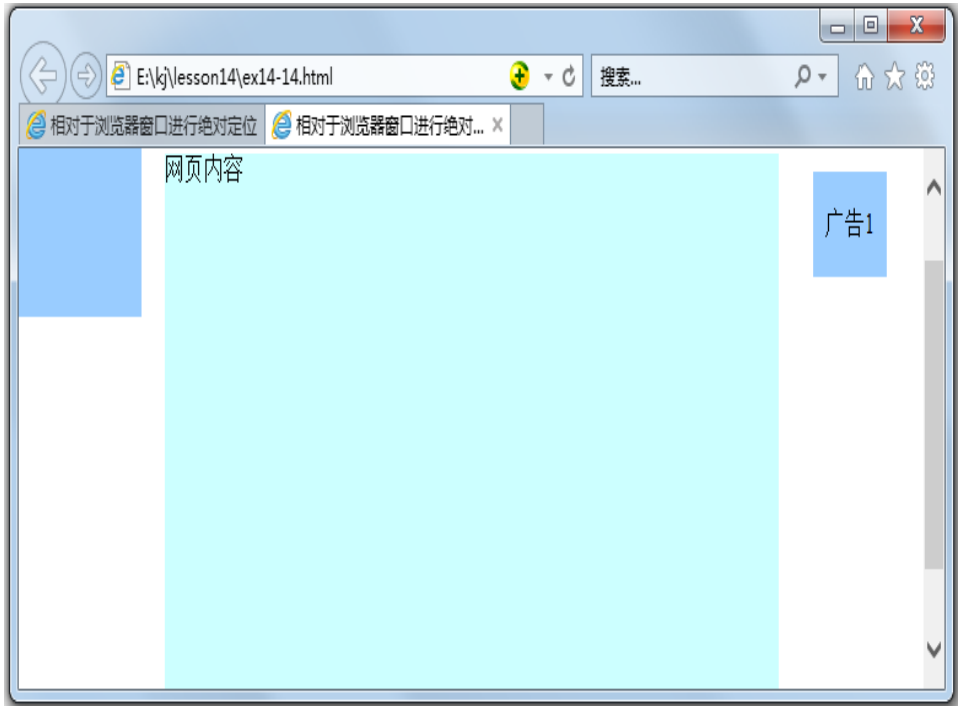
相对定位示例

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>相对定位示例</title>
<style>
#father{
    padding:35px;
    background-color:#a0c8ff;
    border:1px dashed #000000;
}
#son1{
    padding:10px;
    background-color:#fff0ac;
    border:1px dashed #000000;
}
#son2{
    padding:10px;
    position:relative;
    left:-30px;
    top:30px;
    background-color:#fff0ac;
    border:1px dashed #000000;
}
</style>
</head>
<body>
  <div id="father">
    <div id="son1">div1静态定位:正常位置显示</div>
    <div id="son2">div3相对定位：相对于其正常位置向左、向下分别偏移30px</div>
  </div>
</body>
</html>
```



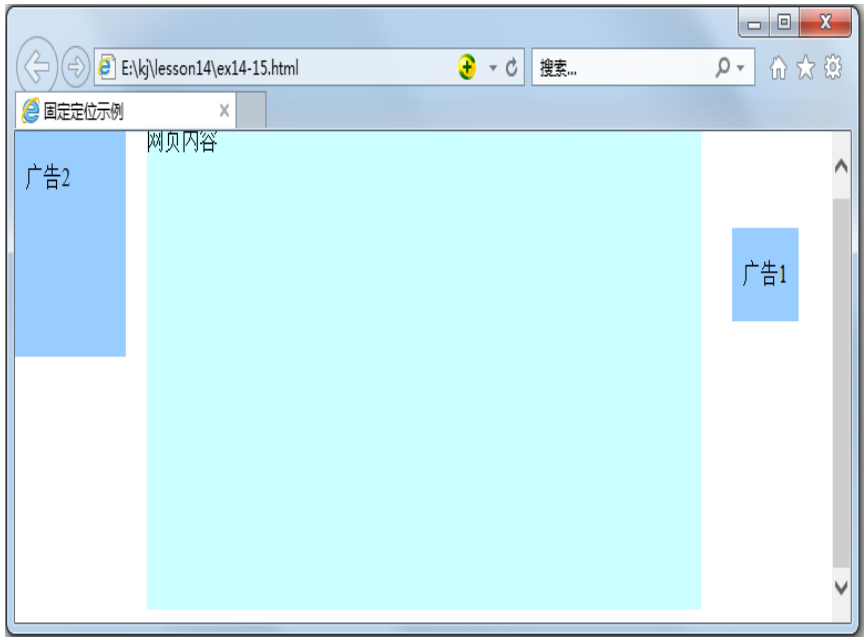
绝对定位示例

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>相对于浏览器窗口进行绝对定位</title>
<style>
#content{width:500px;
     height:300px;
     margin:50px auto;
     background:#CFF;}
#ad1{
     position:absolute;
     top:60px;
     right:30px;
     background:#9CF;
     padding:20px 10px;
}
#ad2{
     position:absolute;
     top:0px;
     left:0px;
     width:80px;
     height:100px;
     background:#9CF;
     padding:20px 10px;
}
</style>
</head>
<body>
  <div id="container">
    <div id="content">网页内容</div>
    <div id="ad1">广告1</div>
    <div id="ad2">广告2</div>
  </div>
</body></html>
```



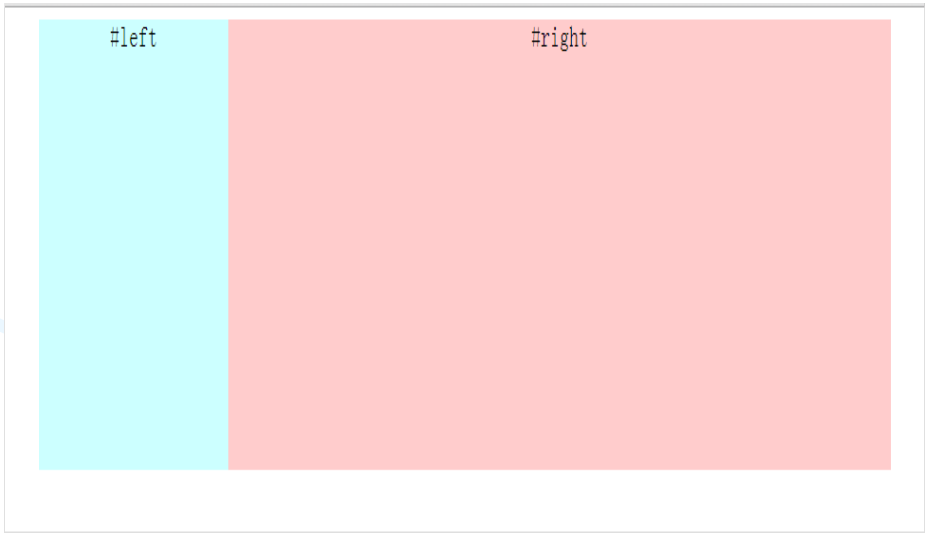
固定定位示例

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>固定定位示例</title>
<style>
#content{width:500px;
     height:300px;
     margin:0 auto;
     background:#CFF;}
#ad1{
     position:fixed;
     top:60px;
     right:30px;
     background:#9CF;
     padding:20px 10px;
}
#ad2{
     position:fixed;
     top:0px;
     left:0px;
     width:80px;
     height:100px;
     background:#9CF;
     padding:20px 10px;
}
</style>
</head>
<body>
  <div id="container">
    <div id="content">网页内容</div>
    <div id="ad1">广告1</div>
    <div id="ad2">广告2</div>
  </div>
</body></html>
```



课堂作业

- 两种方式实现左右两列页面，如下图所示。
  - 浮动布局
  - 定位布局

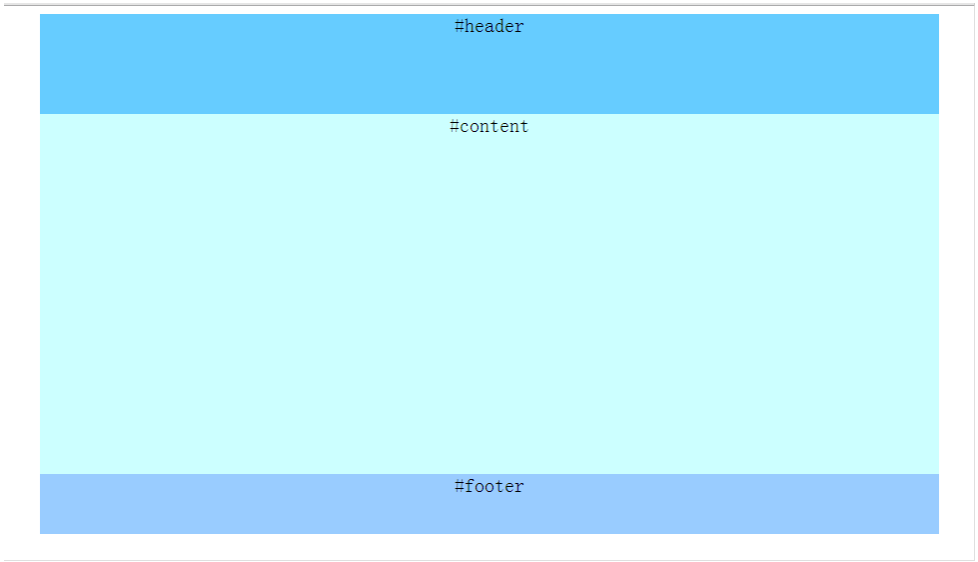


二、经典布局版式

- 布局网页就是把要出现在网页中的各个元素进行定位。布局网页的方式有表格布局和**CSS**布局两种。表格布局已被逐渐摒弃，**CSS**布局是**WEB**标准推荐的网页布局方式。**DIV+HTML5**结构标签**+CSS**是目前经典的网页布局解决方案。
- 常见的网页布局版式有以下几种：
  - 上中下一栏版式
  - 左右两栏版式
  - 左右两栏+页眉+页脚版式
  - 左右宽度固定中间自适应的左中右三栏版式
  - 左右宽度固定中间自适应的左中右三栏+页眉+页脚版式

1. 上中下一栏版式

- 上中下一栏版式用于网页结构的排版，该版式将网页分成上中下三块内容，如下图所示，其中网页的页眉为页面的头部内容，主体内容为页面的中间内容，页脚为页面的页脚内容。



- 该布局版式的页面结构代码：

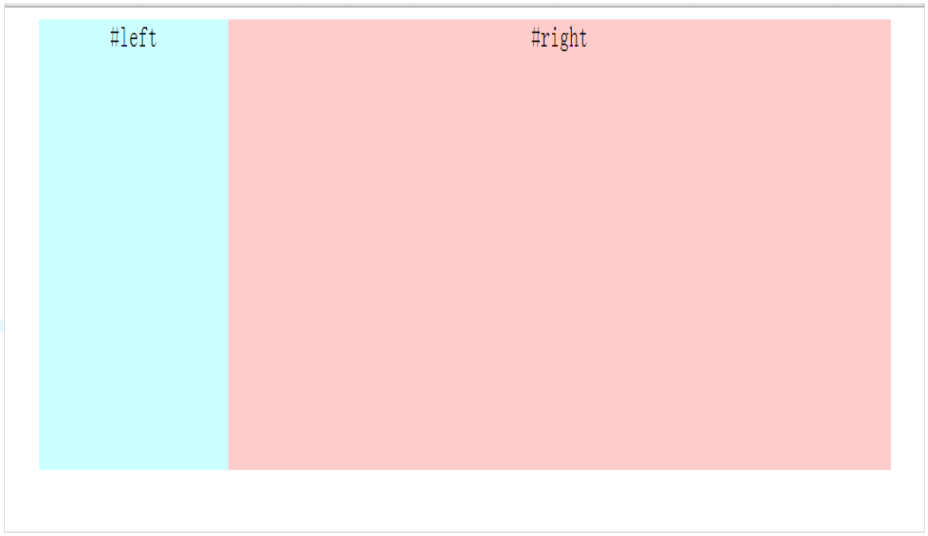
```
<div id="wrap">
    <div id="header">header</div>
    <div id="main">content</div>
    <div id="footer">footer</div>
</div>
```

- 该布局版式的CSS代码：

```
body {
    text-align: center;
    font-size: 20px;
}
.wrap {
    margin: 0 auto;/*设置元素居中显示*/
    width: 900px;/*在此设置宽度固定，可以设置百分数实现宽度自适应父窗口*/
}
#header {
    height: 100px;
    background: #6cf;
}
#main {
    height: 360px;
    background: #cff;
}
#footer {
    height: 60px;
    background: #9CF;
}
```

2. 左右两栏版式

- 左右两栏版式用于对网页内容的排版，排版的该部分内容在网页中分成左右两栏，版式结构如下图所示。为了便于控制左右两栏的宽度以及居中显示，在它们的外面再加一个父**DIV**，然后对这个父**DIV**设置水平居中和宽度样式。



- 该布局版式的页面结构代码：

```
<body>
  <div class="wrap">
     <aside id="left">#left</aside>
     <section id="right">#right</section>
  </div>
</body>
```

- 混合浮动+普通流排版CSS代码：

```
body {
    text-align: center;
    font-size: 20px;
}
.wrap {
    margin: 0 auto;/*水平居中设置*/
    width: 900px;/*在此设置宽度固定，可以设置百分数实现宽度自适应父窗口*/
}
#left{
    float: left;/*向左浮动*/
    width: 200px;
    height: 300px;
    background: #cff;
}
#right {
    height: 300px;
    background: #fcc;
    margin-left: 200px;/*在左边给浮动元素腾出200px的空间*/
}
```

- 纯粹浮动排版CSS代码：

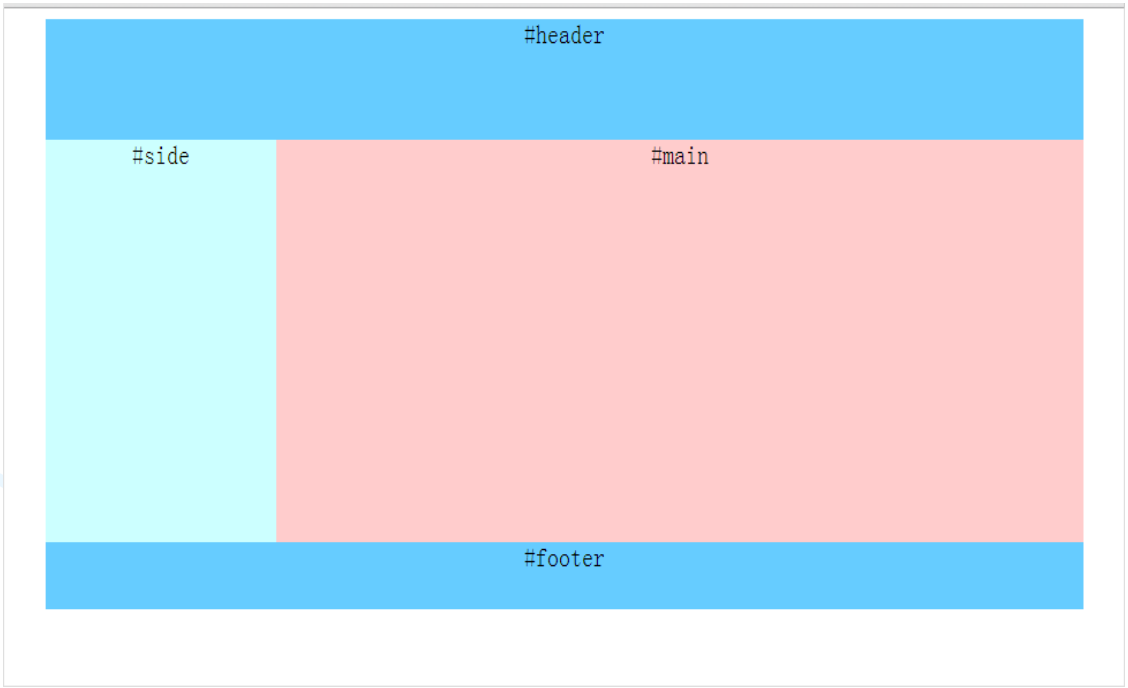
```
 body {
     text-align: center;
     font-size: 20px;
}
.wrap {
     margin: 0 auto;/*水平居中设置*/
     width: 900px;
}
.clearfix:after {/*设置父元素高度自适应*/
     content: "";
     display: block;
     clear: both;
     visibility: hidden;
}
#left {
     float: left;/*向左浮动*/
     width: 200px;
     height: 300px;
     background: #cff;
}
#right {
     float: right;/*向右浮动*/
     Width: 700px;
     height: 300px;
     background: #fcc;
}
```

- 定位排版CSS代码：

```
body {
    text-align:center;
    font-size:20px;
}
.wrap {
    position:relative;/*设置相对定位，便于子元素相对它进行绝对定位*/
    margin:0 auto;/*水平居中设置*/
    width:900px;
}
#left {
    position:absolute;/*相对父元素绝对定位*/
    top:0px;
    left:0px;
    width:200px;
    height:300px;
    background:#cff;
}
#right{
    position:absolute;/*相对父元素绝对定位*/
    top:0px;
    right:0px;
    width:700px;
    height:300px;
    background:#fcc;
}
```

3. 左右两栏+页眉+页脚版式

- 左右两栏+页眉+页脚版式用于对网页结构的排版。该版式将网页内容划分为页眉、主体和页脚三块内容，同时主体又划分为左、右两栏内容，结构如下图所示。



- 该布局版式的页面结构代码：

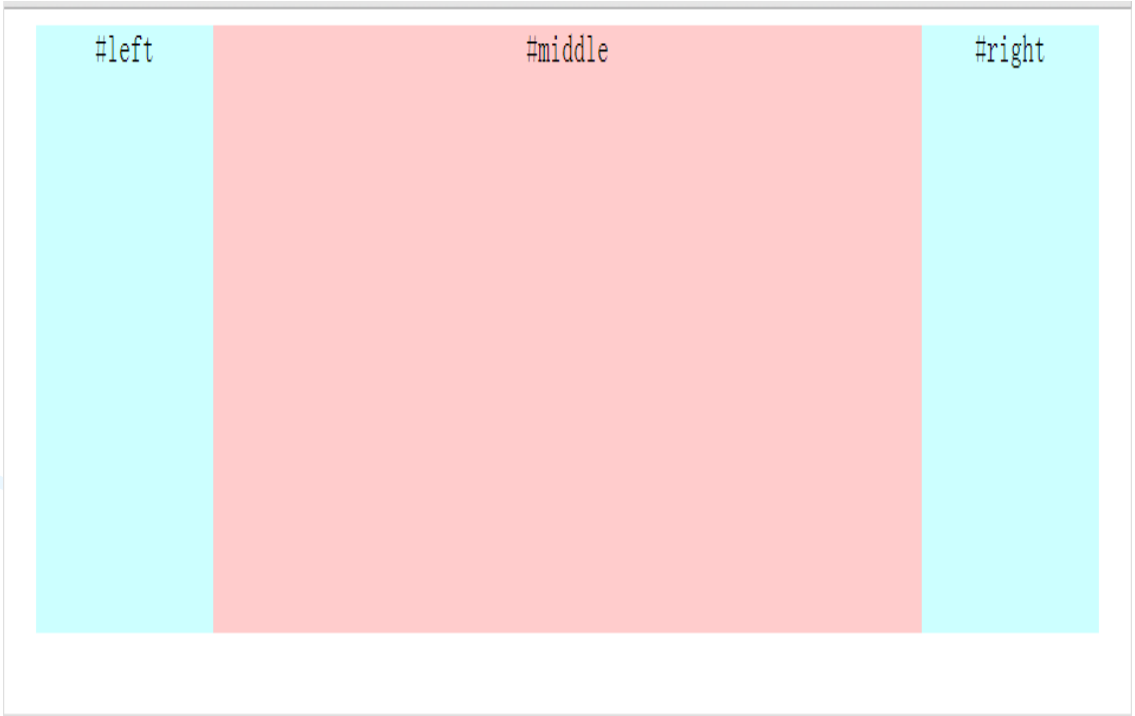
```
<body>
  <header id="header" class="wrap">#header</header>
  <section id="content" class="wrap">
     <aside id="side">#side</aside>
     <article id="main">#main</article>
  </section>
  <footer id="footer" class="wrap">#footer</footer>
</body>
```

- 该布局版式的CSS代码：

```
body {
    text-align: center;
    font-size: 20px;
}
.wrap {
    margin: 0 auto; /*水平居中设置*/
    width: 900px;
}
     #header {
    height: 90px;
    background: #6cf;
}
#content {
    height: 300px;
}
#side {
    float: left;/*向左浮动*/
    height: 300px;
    width: 200px;
    background: #cff;
}
#main {
    height: 300px;
    margin-left: 200px;/*为浮动元素腾出200px的宽度*/
    background: #FCC;
}
#footer {
    height: 50px;
    background: #6cf;
}
```

4. 左右宽度固定中间自适应的左中右三栏版式

- 该版式用于对网页内容的排版，排版的该部分内容在网页中分成左中右三栏，版式结构如下图所示。该版式和两栏版式一样，使用了容器**DIV**来控制三栏内容的居中和宽度。



- 左、右浮动+中间静态排版的页面结构代码：

```
<body>
  <div class="wrap">
    <aside id="left">#left</aside>
    <aside id="right">#right</aside>
    <!--#middle必须放在#left和#right元素之后-->
    <section id="middle">#middle</section>
  </div>
</body>
```

左、右浮动+中间静态排版的CSS代码：

```
body {
    text-align:center;
    font-size:20px;
}
.wrap {
    margin:0 auto; /*水平居中对齐*/
    width:900px;
}
#left {
    float:left;/*向左浮动*/
    width:150px;
    height:300px;
    background:#cff;
}
#right {
    float:right;/*向右浮动*/
    width:150px;
    height:300px;
    background:#cff;
}
#middle{
    height:300px;
    background:#fcc;
    margin:0 150px;/*在左、右两侧分别为浮动元素腾出150px的宽度*/
}
```

- “双飞翼”布局的页面结构代码：

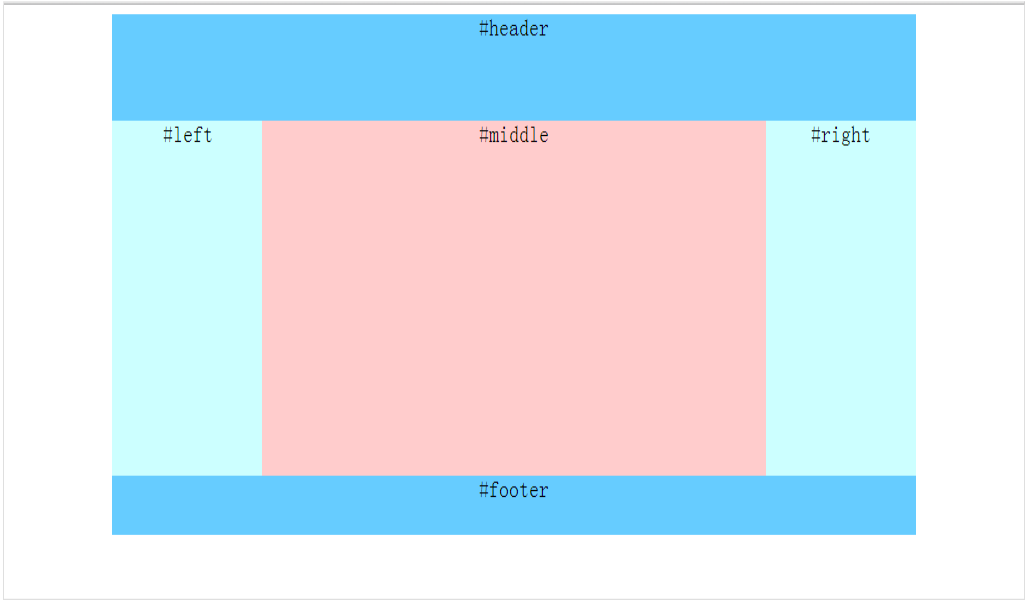
```
<body>
  <div class="wrap">
    <!--#middle必须放在#left和#right元素前面-->
    <div id="midContainer">
      <section id="middle">#middle</section>
    </div>
    <aside id="left">#left</aside>
    <aside id="right">#right</aside>
  </div>
</body>
```

“双飞翼”布局的CSS代码：

```css
body {
    text-align:center;
    font-size:20px;
}
.wrap {
   /*页面内容占浏览器窗口宽度的80%，如果希望页面占满整个浏览器窗口，可以不用设该属性*/
    width: 80%;
    margin: 0 auto;
}
#midContainer {
    width: 100%;/*自适应窗口大小*/
    float: left;
}
#middle {
    height: 300px;
    background:#fcc;
    margin: 0 150px;/*为左、右栏腾出空间*/
}
#left {
    float: left;
    width: 150px;
    height: 300px;
    background: #cff;
    margin-left: -100%;/*父元素的100%,使左栏上移一行且从该行的右边移到左边*/
}
#right {
    float: left;
    width: 150px;
    height: 300px;
    background: #cff;
    margin-left:-150px;/*使右栏从下面移上来*/
}
```

5. 左右宽度固定中间自适应的左中右三栏+页眉+页脚版式

- 该版式用于对网页结构的排版，该版式将网页内容划分为页眉、主体和页脚三块内容，同时主体又划分为左、中、右三栏内容，版式结构如下图所示。



- “双飞翼”布局的页面结构代码：

```
<body>
  <header id="header" class="wrap">#header</header>
  <div class="wrap clearfix">
    <div id="midContainer">
      <section id="middle">#middle</section>
    </div>
    <aside id="left">#left</aside>
    <aside id="right">#right</aside>
  </div>
  <footer id="footer" class="wrap">#footer</footer>
</body>
```

- “双飞翼”布局的CSS代码：

```
body {
    text-align: center;
    font-size: 20px;
    min-width: 700px;/*当内容宽度小于700px时会显示滚动条，否则自适应父窗口宽度*/
}
.wrap {
    width: 80%;/*页面内容占浏览器窗口宽度的80%*/
    margin: 0 auto;
}
#header {
    height: 90px;
    background: #6cf;
}
.clearfix:after {/*设置父元素高度自适应*/
    content: "";
    clear: both;
    display: block;
    visibility: hidden;
}
```

- “双飞翼”布局的CSS代码(续前)：

```
#midContainer {
    width: 100%;/*自适应窗口大小*/
    float: left;
}
#middle {
    height: 300px;
    background: #fcc;
    margin: 0 150px;/*为左、右栏腾出空间*/
}
#left {
    float: left;
    width: 150px;
    height: 300px;
    background: #cff;
    margin-left: -100%;/*父元素的100%,使左栏从右边移到左边*/
}
#right {
    float: left;
    width: 150px;
    height: 300px;
    background: #cff;
    margin-left: -150px;/*使右栏从下面移上来*/
}
#footer {
    height: 50px;
    background: #6cf;
}
```