



互联网软件开发技术与实践

CSS基础

《互联网软件开发技术与实践》

课程建设小组

北京大学

二零一九年 北京



北京大学



上次课程回顾

- 了解HTML语言
- 使用HTML开发了“人物简介”页面
 - 是否使用了CSS？



北京大学



CSS是什么

- 层叠样式表(英文全称：Cascading Style Sheets)是一种**用来表现HTML或XML等文件样式**的计算机语言。
- CSS 能够对网页中元素位置的**排版**进行**像素级精确控制**，支持几乎所有的**字体字号样式**，拥有对网页**对象**和**模型样式**编辑的能力。
- 样式通常保存在外部的 **.css 文件**中。只需要编辑一个简单的 CSS 文档就可以指定页面的**布局**和**外观**。





我是红颜色的文字

方法1

我是红颜色的文字

方法2

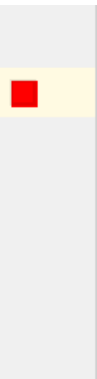
我是红颜色的文字

`.f1{color:#ff0000}`

以上两种方法是否使用了CSS样式？



北京大學



.f1{color:#ff0000}

实验1

css

style.css

index.html

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title></title>
  <link rel="stylesheet" type="text/css" href="css/style.css"/>
</head>
<body>
<p>我是<span style="color: red">红颜色的文字</span></p>
<p>我是<span class="f1">红颜色的文字</span></p>
</body>
</html>
```



localhost:63342/web/css实验/实验1/index.html

我是红颜色的文字

我是红颜色的文字

CSS语法

- CSS 规则由两个主要的部分构成：**选择器**，以及一条或多条**声明**。
 - `selector {declaration1; declaration2; ... declarationN }`
- 每条声明由一个属性和一个值组成。每个属性有一个值。属性和值被冒号分开。
 - `selector {property: value}`

```
■ .f1{color:#ff0000}
```





CSS引用方式

- 外部样式表


```
<head>  
<link rel="stylesheet" type="text/css" href="mystyle.css">  
</head>
```

- 内部样式表

```
<head>  
  
<style type="text/css">  
body {background-color: red}  
p {margin-left: 20px}  
</style>  
  
</head>
```

- 内联样式

```
<p style="color: red; margin-left: 20px">  
This is a paragraph  
</p>
```





CSS选择器

- 类选择器

```
.center {text-align: center}
```

```
<h1 class="center">
```

```
This heading will be center-aligned
```

```
</h1>
```

```
<p class="center">
```

```
This paragraph will also be center-aligned.
```

```
</p>
```

- ID选择器

```
#red {color:red;}
```

```
#green {color:green;}
```

```
<p id="red">这个段落是红色。</p>
```

```
<p id="green">这个段落是绿色。</p>
```



北京大学



CSS选择器

- 派生选择器

依据元素在其位置的上下文关系来定义

```
li strong {  
  font-style: italic;  
  font-weight: normal;  
}
```

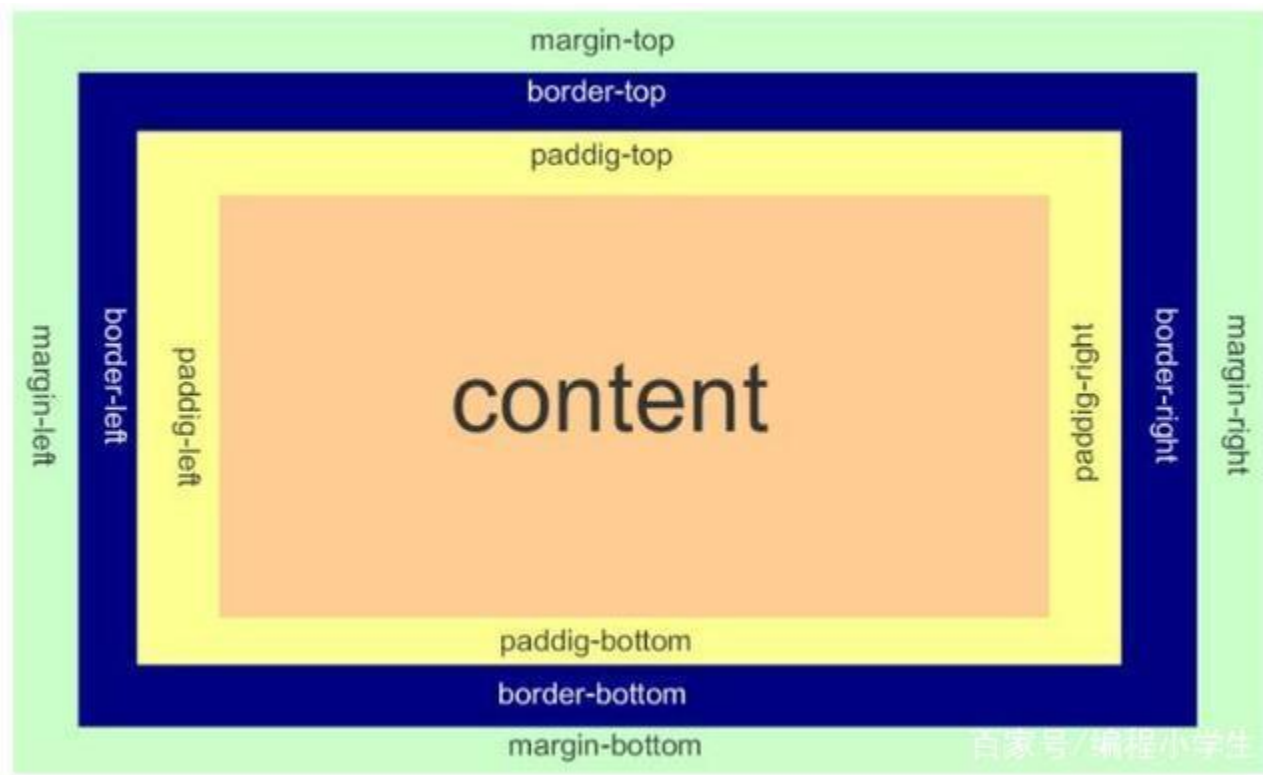
```
<p><strong>我是粗体字，不是斜体字，因为我不在列表当中，所以这个规则对我不起作用</strong></p>  
<ol>  
  <li><strong>我是斜体字。这是因为 strong 元素位于 li 元素内。</strong></li>  
  <li>我是正常的字体。</li>  
</ol>
```




清华大学

CSS 盒模型

- **Margin(外边距)** - 清除边框外的区域，外边距是透明的。
- **Border(边框)** - 围绕在内边距和内容外的边框。
- **Padding(内边距)** - 清除内容周围的区域，内边距是透明的。
- **Content(内容)** - 盒子的内容，显示文本和图像。





```
*{margin:0;padding:0}
```

```
div1 {  
    background-color: #f00;  
    width: 300px;  
    height: 100px;  
}
```

```
div2 {  
    background-color: #00f;  
    width: 300px;  
    border: 25px solid green;  
    padding: 25px;  
    margin: 25px;  
}
```





```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title></title>
  <link rel="stylesheet" type="text/css" href="css/style.css"/>
</head>
<body>
  <h2>盒子模型演示</h2>

  <p>CSS盒模型本质上是一个盒子，封装周围的HTML元素，它包括：边距，边框，填充，和实际内容。</p>

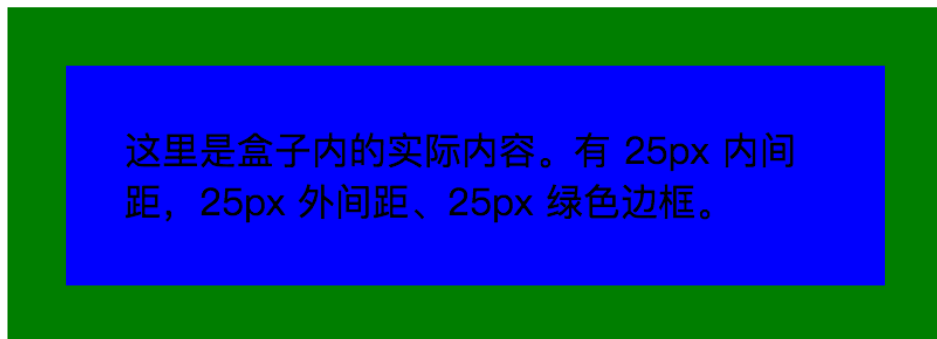
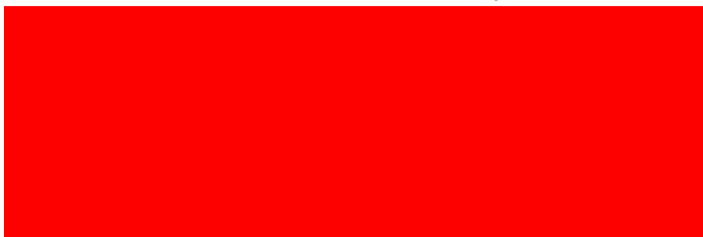
  <div class="div1"></div>
  <div class="div2">这里是盒子内的实际内容。有 25px 内间距，25px 外间距、25px 绿色边框。</div>
  <div class="div1"></div>
</body>
</html>
```



北京大学

盒子模型演示

CSS盒子模型本质上是一个盒子，封装周围的HTML元素，它包括：边距，边框，填充，和实际内容。





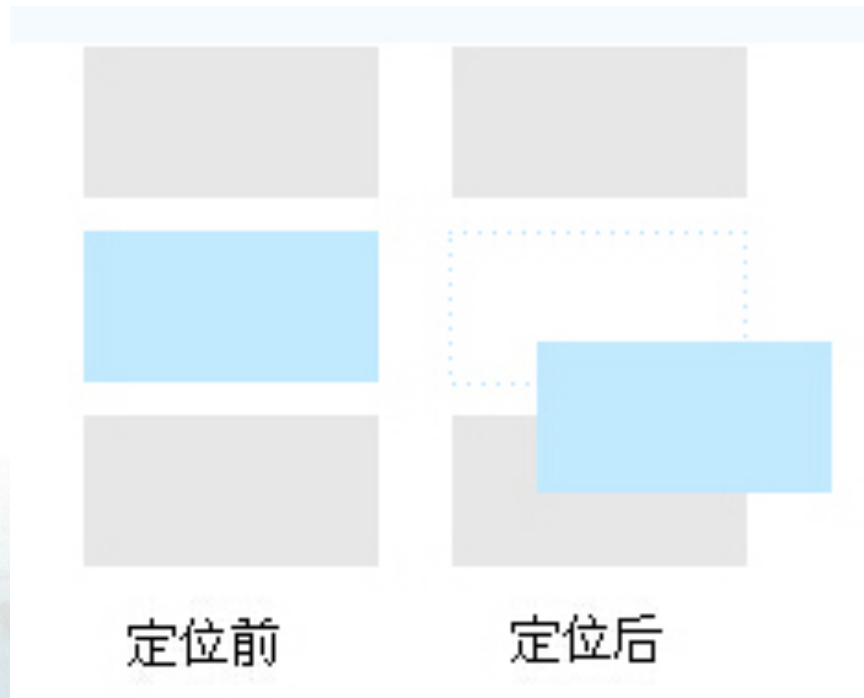
CSS 定位

- 普通流
 - 除非专门指定，否则所有框都在普通流中定位。
 - 普通流中元素框的位置由元素在(X)HTML中的位置决定。
 - 块级框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。



相对定位

- 定位元素的位置相对于它在普通流中的位置进行移动
- 使用相对定位的元素不管它是否进行移动，元素仍要占据它原来的位置



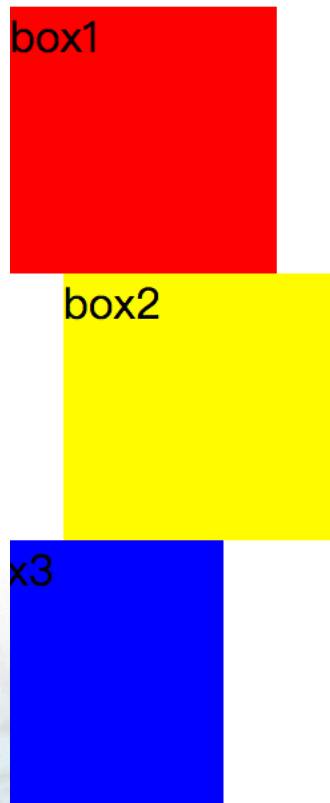
代码示例

```
<div class="box1">box1</div>
<div class="box2">box2</div>
<div class="box3">box3</div>
```

← → ↻ ⓘ localhost:63342/web/css实

相对定位演示

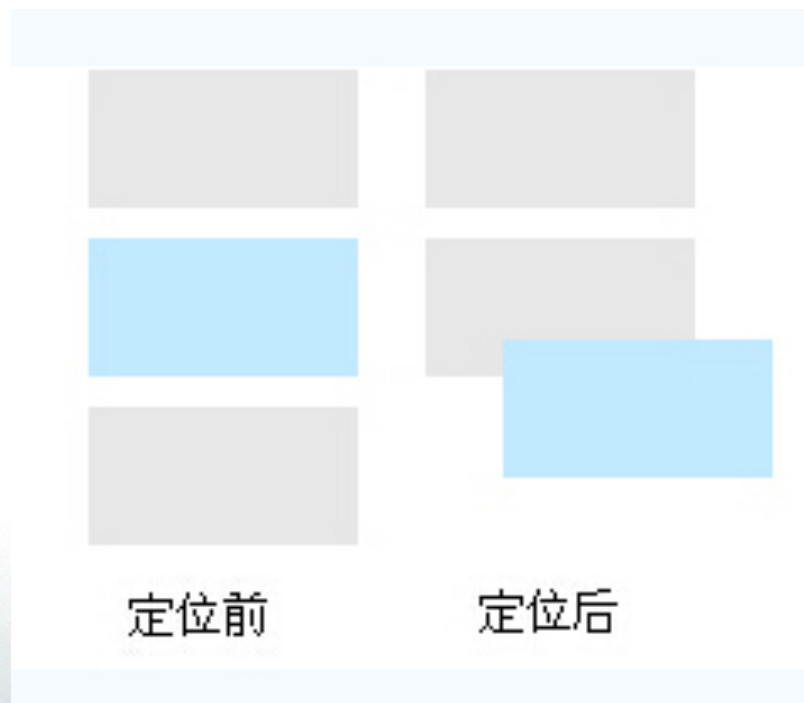
定位元素的位置相对于它在普通流中的位置进行



```
*{margin:0;padding:0}
.box1{
  background-color: red;
  width:100px;
  height:100px;
}
.box2{
  background-color: yellow;
  width:100px;
  height:100px;
  position: relative;
  left: 20px;
}
.box3{
  background-color: blue;
  width:100px;
  height:100px;
  position: relative;
  right: 20px;
}
```


绝对定位

- 绝对定位的元素位置是相对于距离它最近的那个**已定位的祖先**(相对/绝对)元素决定的
- 与相对定位相反，绝对定位使元素与文档流无关，因此不占据空间



```
*{margin:0;padding:0}
.relative {
    position: relative;
    width: 600px;
    height: 400px;
    background-color: #f00;
}
.absolute {
    position: absolute;
    top: 120px;
    right: 0;
    width: 300px;
    height: 200px;
    background-color: #00f;
}
```

```
<div class="relative"></div>
<div class="absolute"></div>
```



北京大學



```
<div class="relative"></div>  
<div class="absolute"></div>
```



localhost:63342/web/css实验/实验_绝对定位/index.html



绝对定位演示

定位的元素位置是相对于距离它最近的那个已定位的祖先(相对/绝对)元素决定的。





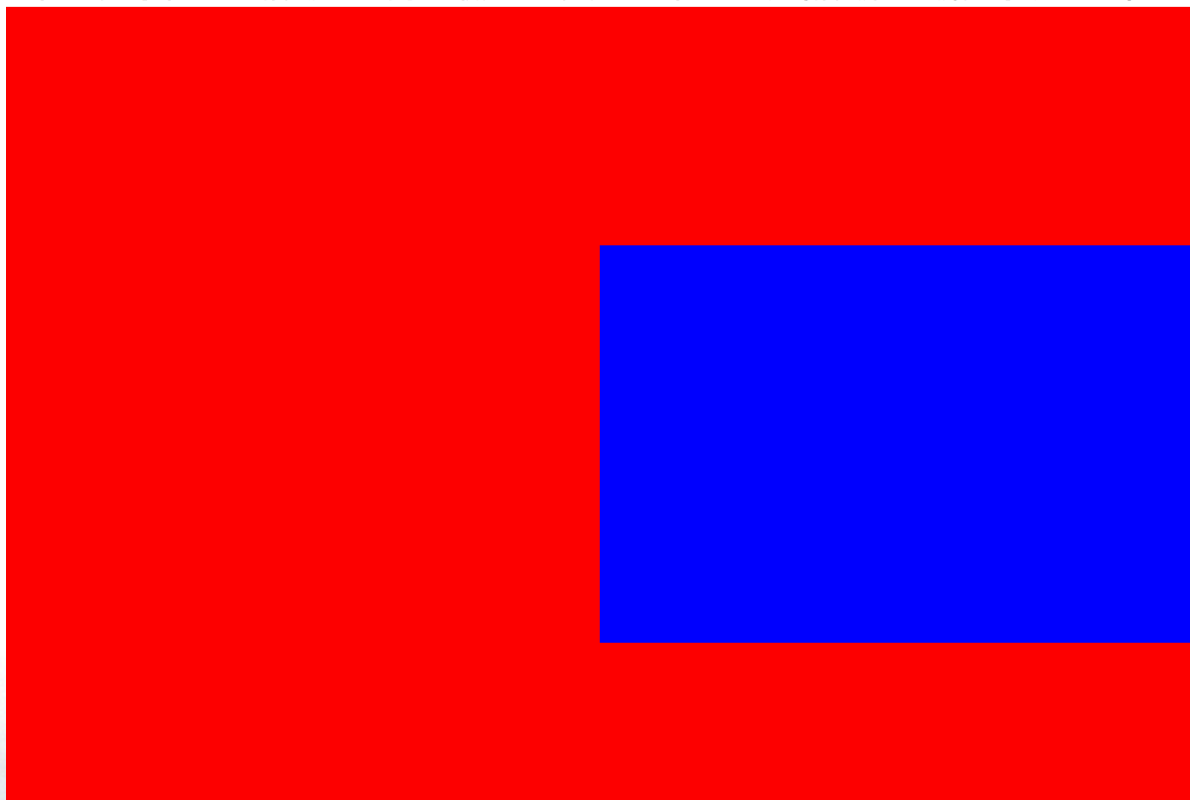
```
<div class="relative">  
  <div class="absolute"></div>  
</div>
```

← → ↻ ⓘ localhost:63342/web/css实验/实验_绝对定位_2/index.html



绝对定位演示

定位的元素位置是相对于距离它最近的那个已定位的祖先(相对/绝对)元素决定的。





CSS浮动

- 浮动的框脱离普通流
- 浮动的框可以在左右移动，直到它的外边框边缘碰到包含框或另一个浮动框的边缘
- 如果包含块太窄，无法容纳水平排列的浮动元素，那么其他浮动块向下移动
- 行内元素会围绕着浮动框排列



浮动演示

浮动的框脱离普通流;浮动的框可以在左右移动,直到它的外边框边缘碰到包含框或另一个浮动框的边缘;如果包含块太窄,无法容纳水平行内元素会围绕着浮动框排列



```
<div class="div1">div1</div>
<div class="div2">div2</div>
<div class="div3">div3</div>
<div class="div4">div4</div>
```

```
*{margin:0;padding:0}
.div1 {
  width: 200px;
  height: 100px;
  background-color: #f00;
}
.div2 {
  width: 100px;
  height: 200px;
  background-color: #00f;
}
.div3 {
  width: 20px;
  height: 100px;
  background-color: #0f0;
}
.div4 {
  width: 50px;
  height: 150px;
  background-color: #0af;
}
```

浮动演示

浮动的框脱离普通流;浮动的框可以在左
行内元素会围绕着浮动框排列



浮动演示

浮动的框脱离普通流;浮动的框可以在左
行内元素会围绕着浮动框排列



```
<div class="div1">  
<div class="div2">  
<div class="div3">  
<div class="div4">
```

```
*{margin:0;padding:0}
```

```
.div1 {
```

```
float: left;
```

```
width: 200px;
```

```
height: 100px;
```

```
background-color: #f00;
```

```
}
```

```
.div2 {
```

```
width: 100px;
```

```
height: 200px;
```

```
background-color: #00f;
```

```
}
```

```
.div3 {
```

```
width: 20px;
```

```
height: 100px;
```

```
background-color: #0f0;
```

```
}
```

```
.div4 {
```

```
width: 50px;
```

```
height: 150px;
```

```
background-color: #0af;
```

```
}
```

浮动演示

浮动的框脱离普通流;浮动的框可以在左右移动,直到它的外边距与另一个浮动的框或另一个非浮动的框相遇;行内元素会围绕着浮动框排列



```
<div class="div1">div1</div>
<div class="div2">div2</div>
<div class="div3">div3</div>
<div class="div4">div4</div>
```

浮动演示

浮动的框脱离普通流;浮动的框可以在左右移动,直到它的外边距与另一个浮动的框或另一个非浮动的框相遇;行内元素会围绕着浮动框排列



div3



```
*{margin:0;padding:0}
.div1 {
  float: right;
  width: 200px;
  height: 100px;
  background-color: #f00;
}
.div2 {
  width: 100px;
  height: 200px;
  background-color: #00f;
}
.div3 {
```



```
*{margin:0;padding:0}
```

```
.div1 {  
  float: right;  
  width: 200px;  
  height: 100px;  
  background-color: red;  
}
```

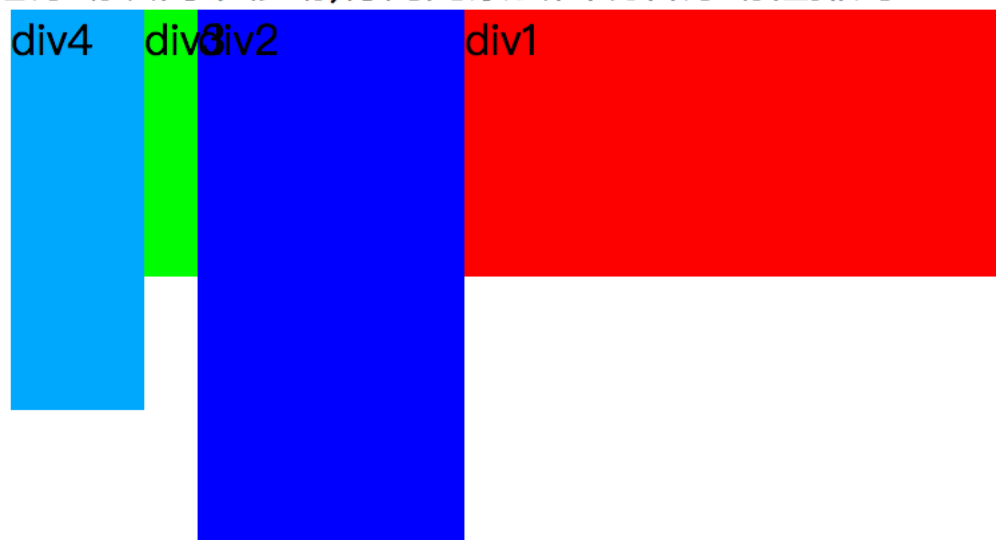
```
.div2 {  
  float: right;  
  width: 100px;  
  height: 200px;  
  background-color: blue;  
}
```

```
.div3 {  
  float: right;  
  width: 20px;  
  height: 100px;  
  background-color: green;  
}
```

```
.div4 {  
  float: right;  
  width: 50px;  
  height: 150px;  
  background-color: #0af;  
}
```

浮动演示

浮动的框脱离普通流;浮动的框可以在左右移动,直到它的外边框边缘碰到包含框或另一个浮动框的边缘;如果包含块太窄,无法容纳水平排列的浮动元素,那么其他浮动块向下移动;行内元素会围绕着浮动框排列



北京大学



localhost:63342/web/css实验/实验_浮动_1/index.html



浮动演示

浮动的框脱离普通流;浮动的框可以在左右移动,直到它的外边框边缘碰到包含框或另一个浮动框的边缘;如果包含块太窄,无法容纳水平排列的浮动元素,那么其他浮动块向下移动;行内元素会围绕着浮动框排列

div2

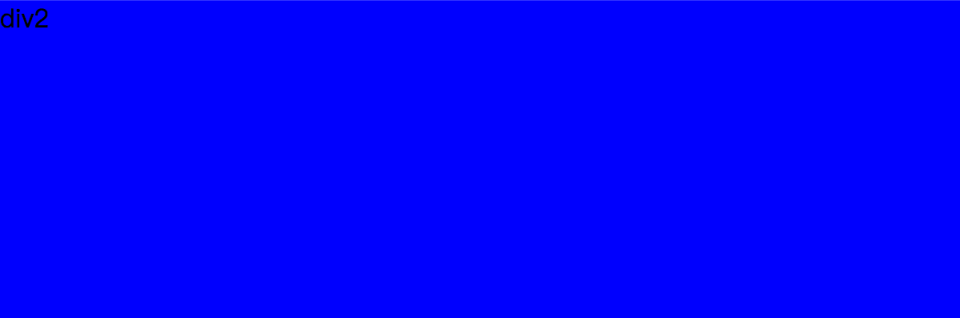
div1

div4

div3

浮动演示

浮动的框脱离普通流;浮动的框可以在左右移动,直到它的外边框边缘碰到包含框或另一个浮动框的边缘;如果包含块太窄,无法容纳水平排列的浮动元素,那么其他浮动块向下移动;行内元素会围绕着浮动框排列



如果屏幕宽度逐渐变窄,会是什么效果?

```
*{margin:0;padding:0}
.div1 {
    float: right;
    width: 200px;
    height: 100px;
    background-color: #f00;
}
.div2 {
    float: left;
    width: 600px;
    height: 200px;
    background-color: #00f;
}
.div3 {
    float: right;
    width: 300px;
    height: 100px;
    background-color: #0f0;
}
.div4 {
    float: right;
    width: 50px;
    height: 150px;
    background-color: #0af;
}
```



```
.div5 {  
  width: 10000px;  
  height: 50px;  
  background-color: #000;  
}
```

```
<div class="div1">div1</div>  
<div class="div2">div2</div>  
<div class="div5">div5</div>  
<div class="div3">div3</div>  
<div class="div4">div4</div>
```

localhost:63342/web/css实验/实验_浮动_2/index.html

浮动演示

浮动的框脱离普通流;浮动的框可以在左右移动,直到它的外边框边缘碰到包含框或另一个浮动框的边缘;如果包含块太窄,无法容纳水平排列的浮动元素,那么其他浮动块向下移动;行内元素会围绕着浮动框排列



北京大学



```
<div class="div1">div1</div>  
<div class="div2">div2</div>  
<div class="div3">div3</div>  
<div class="div4">div4</div>  
<div class="div5">div5</div>
```

什么效果？



北京大学



清除浮动

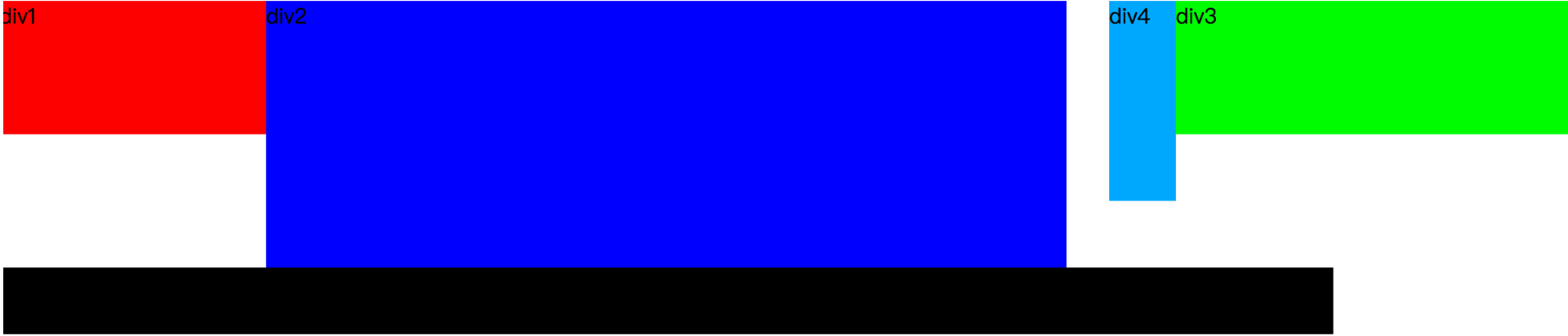
- 元素浮动之后，周围的元素会重新排列，为了避免这种情况，使用 **clear** 属性。
- **clear** 属性指定元素两侧是否能出现浮动元素。
- 语法：**clear : none | left | right | both**
 - **none** : 默认值。允许两边都可以有浮动对象
 - **left** : 不允许左边有浮动对象
 - **right** : 不允许右边有浮动对象
 - **both** : 不允许有浮动对象





浮动演示

浮动的框脱离普通流;浮动的框可以在左右移动,直到它的外边框边缘碰到包含框或另一个浮动框的边缘;如果包含块太窄,无法容纳水平排列的浮动元素,那么其他浮动块向下移动;行内元素会围绕着浮动框排列



```
.div5 {  
    clear: both;  
    width: 1000px;  
    height: 50px;  
    background-color: #000;  
}
```

```
<div class="div1">div1</div>  
<div class="div2">div2</div>  
<div class="div3">div3</div>  
<div class="div4">div4</div>  
<div class="div5">div5</div>
```

浮动演示

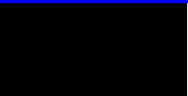
浮动的框脱离普通流;浮动的框可以在左右移动,直到它的外边框边缘碰到包含框或另一个浮动框的边缘;如果包含块太窄,无法容纳水平排列的浮动元素,那么其他浮动块向下移动;行内元素会围绕着浮动框排列



```
.clear{clear: both}
*{margin:0;padding:0}
.div1 {
    float: left;
    width: 200px;
    height: 100px;
    background-color: #f00;
}
```

```
<div class="div1">div1</div>
<div class="div2">div2</div>
<div class="div3">div3</div>
<div class="div4">div4</div>
<div class="clear"></div>
<div class="div5">div5</div>
```





如何实现？



北京大学



```
.clear {clear: both; }
```

```
*{margin:0;padding:0}
```

```
.div1 {
```

```
    float: right;
```

```
    width: 200px;
```

```
    height: 100px;
```

```
    background-color: #f00;
```

```
}
```

```
.div2 {
```

```
    float: left;
```

```
    width: 600px;
```

```
    height: 200px;
```

```
    background-color: #00f;
```

```
}
```

```
<div class="div1">div1</div>
```

```
<div class="div2">div2</div>
```

```
<div class="clear"></div>
```

```
<div class="div5">div5</div>
```

```
<div class="div3">div3</div>
```

```
<div class="div4">div4</div>
```



北京大學



Q&A

本讲结束 !



北京大学