

CSS 详解

---- 老孙

课程目标

- 1、CSS介绍
 - 2、CSS与HTML结合方式
 - 3、CSS的使用
 - 4、CSS3新特性
-

1.css介绍

1.1 什么是CSS？



- CSS是指层叠样式表 cascading style sheets
- 通过CSS可以让我们定义HTML元素如何显示。
- CSS可以让我们原本HTML不能描述的效果，通过CSS描述出来。
- 通过CSS描述我们的html页面，可以让我们的页面更加漂亮，可以提高工作效率。

2.CSS与HTML结合方式

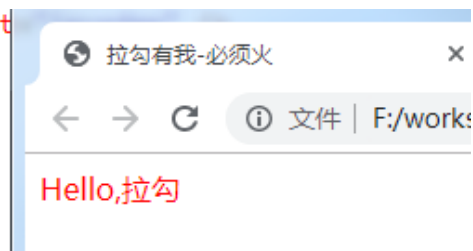
2.1 第一种方式 内联/行内样式

就是在我们的HTML标签上通过style属性来引用CSS代码。

优点:简单方便 ;

缺点:只能对一个标签进行修饰。

```
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<title>拉勾有我-必须火</title>
</head>
<body>
  <div style="color:red">Hello,拉勾</div>
</body>
```



2.2 第二种方式 内部样式表

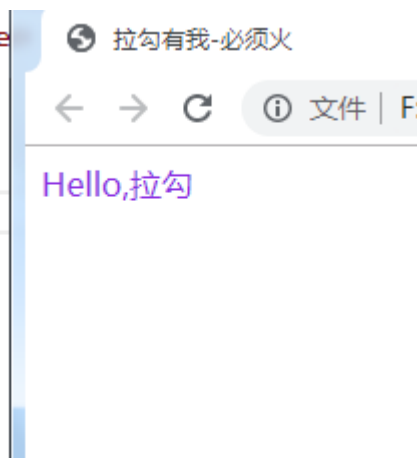
我们通过<style>标签来声明我们的CSS. 通常<style>标签我们推荐写在head和body之间，也就是“脖子”的位置

优点:可以通过多个标签进行统一的样式设置

缺点: 它只能在本页面上进行修饰

语法： 选择器 {属性:值;属性:值}

```
<title>拉勾有我-必须火</title>
</head>
<style>
  div{
    color: blueviolet;
  }
</style>
<body>
  <div>Hello,拉勾</div>
</body>
```

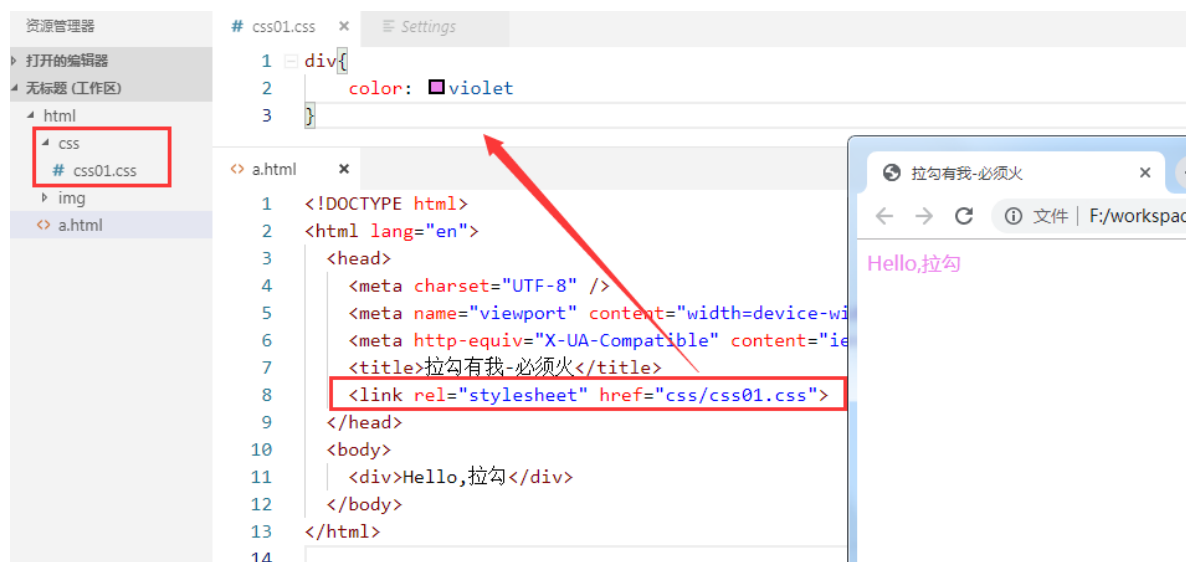


2.3 第三种方式 外部样式表

我们需要单独定义一个CSS文件,注意CSS文件的后缀名就是.css

在项目根目录下，创建css目录，在css目录中创建css文件 css01.css

在<head>中使用<link>标签引用外部的css文件



还可以使用另一种引入css文件的方式：

```
<style>
@import 'css/css01.css'
</style>
```

关于外部导入css使用<link>与@import的区别？

1. 加载顺序不同

- @import方式导入会先加载html，然后才导入css样式，那么如果网络条件不好，就会先看到没有修饰的页面，然后才看到修饰后的页面。
- 如果使用link方式，它会先加载样式表，也就是说，我们看到的直接就是修饰的页面；

2. @import方式导入css样式，它是不支持javascript的动态修改的。而link支持。

三种样式表的优先级：满足就近原则

内联 > 内部 > 外部

3.CSS的使用

3.1 css中选择器

3.1.1 元素(标签)选择器

它可以对页面上相同的标签进行统一的设置，它描述的就是标签的名称。

```
<meta http-equiv="X-UA-Compatible"
<title>拉勾有我-必须火</title>
</head>
<style>
  /* 注释 */
  h2{
    /* 注释 */
    color: hotpink;
  }
</style>
<body>
  <h2>Hello,老孙</h2>
  <h2>Hello,老孙</h2>
  <h2>Hello,老孙</h2>
</body>
</html>
```

Hello,老孙
Hello,老孙
Hello,老孙

3.1.2 类选择器

类选择器在使用时使用"."来描述，它描述的是元素上的class属性值

```

<meta http-equiv="X-UA-Compatible"
<title>拉勾有我-必须火</title>
</head>
<style>
  #a{
    color: red;
  }
  .b{
    color: greenyellow;
  }
</style>
<body>
  <h2 id="a">Hello,老孙</h2>
  <h2 class="b">Hello,老孙</h2>
  <h2>Hello,老孙</h2>
</body>
</html>

```

Hello,老孙

Hello,老孙

Hello,老孙

3.1.3 id选择器

它只能选择一个元素，使用 "#" 引入，引用的是元素的id属性值。

id选择器，比类选择器更具有唯一性

```

<meta name="viewport" content="wid
<meta http-equiv="X-UA-Compatible"
<title>拉勾有我-必须火</title>
</head>
<style>
  #a{
    color: red;
  }
</style>
<body>
  <h2 id="a">Hello,老孙</h2>
  <h2>Hello,老孙</h2>
  <h2>Hello,老孙</h2>
</body>

```

Hello,老孙

Hello,老孙

Hello,老孙

3.1.4 选择器组

逗号表示，谁和谁。

例如，我有手机，你有手机，他也有手机，一条一条写太麻烦，就可以合并编写

```

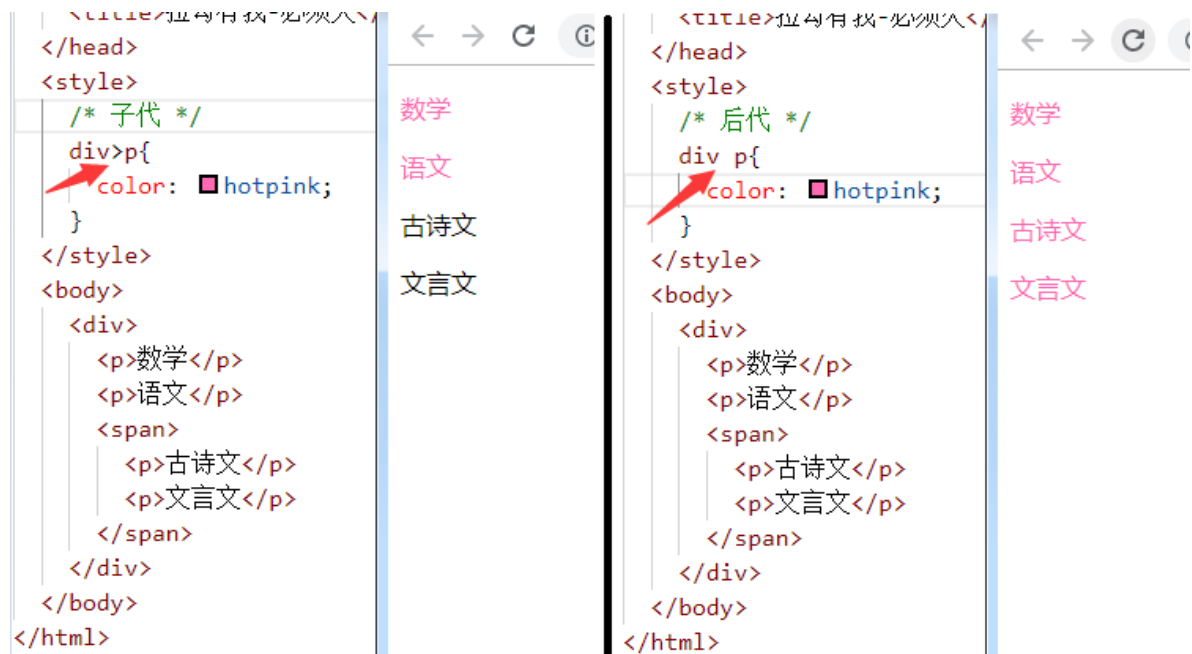
我,你,他{
  手机
}

```



3.1.5 派生选择器

- 子代：父子关系（隔代不管）
- 后代：父子孙，曾孙，从孙...



3.1.6 CSS伪类

- CSS伪类可对css的选择器添加一些特殊效果
- 伪类属性列表：
 - :active 向被激活的元素添加样式。
 - :hover 当鼠标悬浮在元素上方时，向元素添加样式。
 - :link 向未被访问的链接添加样式。
 - :visited 向已被访问的链接添加样式。
 - :first-child 向元素的第一个子元素添加样式。

超链接的伪类：要遵守使用顺序，爱恨原则 LoVeHAtE , lVhA

```
a:link {color: #FF0000} /* 未访问的链接 */
a:visited {color: #00FF00} /* 已访问的链接 */
a:hover {color: #FF00FF} /* 鼠标移动到链接上 */
a:active {color: #0000FF} /* 选定的链接 */
```

```
</head>
<style>
  li:first-child{
    color: hotpink;
  }
</style>
<body>
  <ul>
    <li>斗罗大陆</li>
    <li>凡人修仙传</li>
    <li>拉勾教育</li>
  </ul>
</body>
```

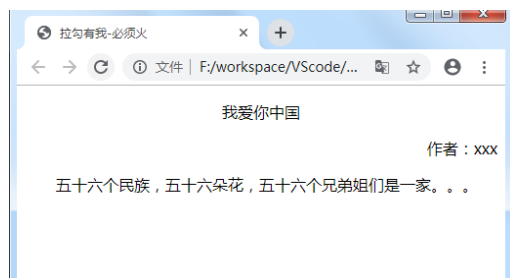
- 斗罗大陆
- 凡人修仙传
- 拉勾教育

3.2 CSS基本属性

3.2.1 文本属性

- 指定字体：font-family：value;
- 字体大小：font-size：value;
 - px：像素
 - em：倍数
- 字体加粗：font-weight：normal/bold;
- 文本颜色：color：value;
- 文本排列：text-align：left/right/center;
- 文字修饰：text-decoration：none/underline;
- 行高：line-height：value;
- 首行文本缩进：text-indent：value（2em）;

```
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title>拉勾有我-必须火</title>
</head>
<style>
  .title{text-align: center;}
  .author{text-align: right;}
  .content{text-indent: 2em;}
</style>
<body>
  <p class="title">我爱你中国</p>
  <p class="author">作者：xxx</p>
  <p class="content">五十六个民族，五十六朵花，五十六个兄弟姐妹们是一家。。。</p>
</body>
```



3.2.2 背景属性

CSS 允许应用纯色作为背景，也允许使用背景图像创建相当复杂的效果。

- background-color 设置元素的背景颜色。
- background-image 把图像设置为背景。

```
background-image: url('img/1.jpg');
```

- background-repeat 设置背景图像的墙纸效果，是否及如何重复
 - repeat：在垂直方向和水平方向重复，为重复值
 - repeat-x：仅在水平方向重复
 - repeat-y：仅在垂直方向重复
 - no-repeat：仅显示一次
- background-position 设置背景图像的起始位置
- 1：控制水平方向 x轴：正值，向右移动；负值，向左移动
- 2：控制垂直方向 y轴：正值，向下移动；负值，向上移动

```
/* 图片向左移动50px，向下移动100px （可以为负值） */
background-position:50px 100px;
```

- background-attachment 背景图像是否固定或者随着页面的其余部分滚动
 - 默认值是 scroll：默认情况下，背景会随文档滚动
 - 可取值为 fixed：背景图像固定，并不会随着页面的其余部分滚动，常用于实现称为水印的图像

```
background-attachment: fixed;
```

3.2.3 列表属性

CSS列表属性作用如下：

- 设置不同的列表项标记为有序列表
- 设置不同的列表项标记为无序列表
- 设置列表项标记为图像

有两种类型的列表：

- 无序列表 - 列表项标记用特殊图形（如小黑点、小方框等）
- 有序列表 - 列表项的标记有数字或字母

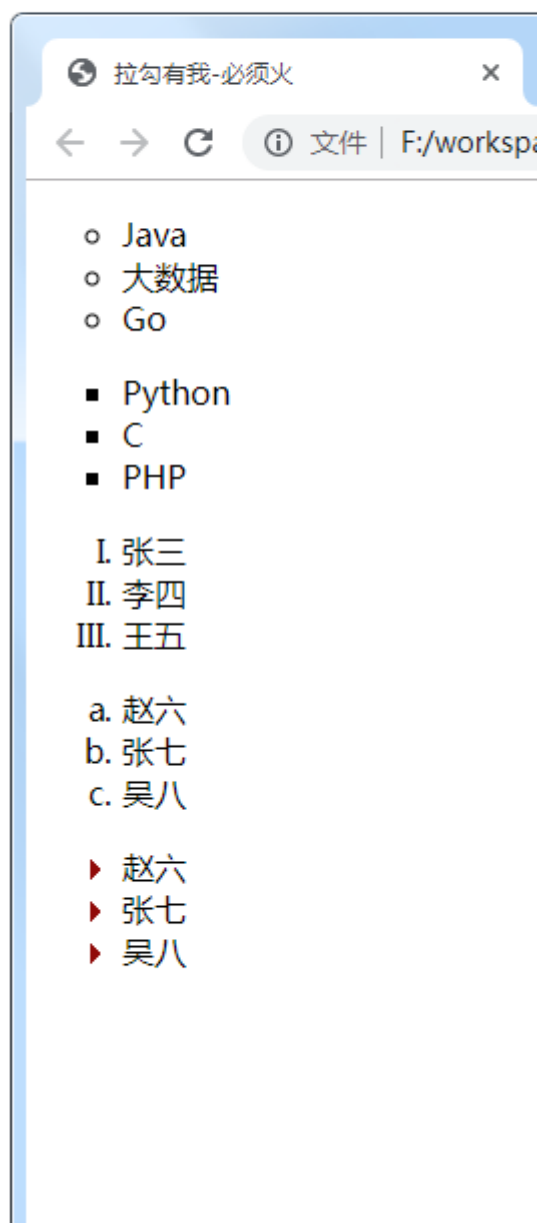
使用CSS，可以列出进一步的样式，并可用图像作列表项标记。

- none：无标记。（去除标记）
- disc：默认。标记是实心圆。
- circle：标记是空心圆。
- square：标记是实心方块。
- decimal：标记是数字。
- decimal-leading-zero：0开头的数字标记。(01, 02, 03, 等。)
- lower-roman：小写罗马数字(i, ii, iii, iv, v, 等。)
- upper-roman：大写罗马数字(I, II, III, IV, V, 等。)
- lower-alpha：小写英文字母The marker is lower-alpha (a, b, c, d, e,等。)
- upper-alpha：大写英文字母The marker is upper-alpha (A, B, C, D, E,等。)

```

<style>
  ul.a {list-style-type: circle;}
  ul.b {list-style-type: square;}
  ol.c {list-style-type: upper-roman;}
  ol.d {list-style-type: lower-alpha;}
  ol.e {list-style-image: url('img/list-img-1.gif');}
</style>
<body>
  <ul class="a">
    <li>Java</li>
    <li>大数据</li>
    <li>Go</li>
  </ul>
  <ul class="b">
    <li>Python</li>
    <li>C</li>
    <li>PHP</li>
  </ul>
  <ol class="c">
    <li>张三</li>
    <li>李四</li>
    <li>王五</li>
  </ol>
  <ol class="d">
    <li>赵六</li>
    <li>张七</li>
    <li>吴八</li>
  </ol>
  <ol class="e">
    <li>赵六</li>
    <li>张七</li>
    <li>吴八</li>
  </ol>
</body>

```





3.2.4 边框属性

CSS边框属性允许你指定一个元素边框的样式和颜色。

在四边都有边框

红色底部边框

圆角边框

```
<style>
  div{
    border-width: 20px;
    border-color: green;
    border-style: outset;
  }
</style>
<body>
  <div>hello</div>
</body>
```

border-style取值：

none: 默认无边框

dotted: 定义一个点线边框

dashed: 定义一个虚线边框

solid: 定义实线边框

double: 定义两个边框。两个边框的宽度和 border-width 的值相同

groove: 定义3D沟槽边框。效果取决于边框的颜色值

ridge: 定义3D脊边框。效果取决于边框的颜色值

inset: 定义一个3D的嵌入边框。效果取决于边框的颜色值

outset: 定义一个3D突出边框。效果取决于边框的颜色值

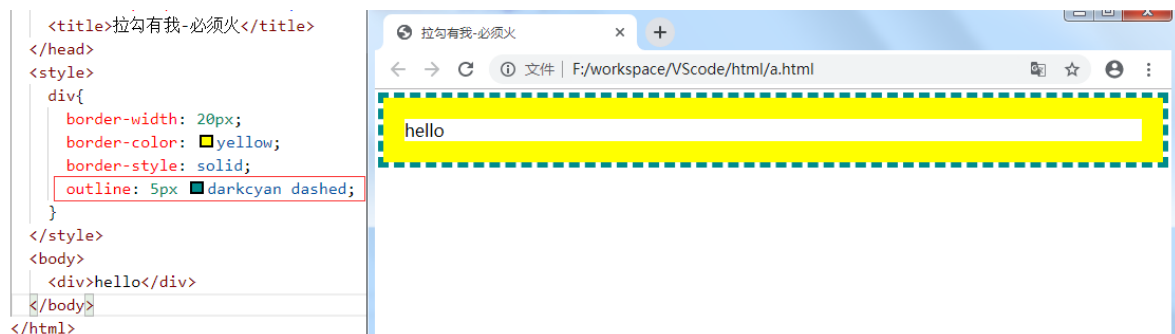
3.2.5 轮廓属性

轮廓 (outline) 是绘制于元素周围的一条线，位于边框边缘的外围，可起到突出元素的作用。

轮廓和边框的区别：

- 边框 (border) 可以是围绕元素内容和内边距的一条或多条线；
- 轮廓 (outline) 是绘制于元素周围的一条线，位于边框边缘的外围，可起到突出元素的作用。

CSS outline 属性规定元素轮廓的样式、颜色和宽度。



3.2.6 盒子模型

所有HTML元素可以看作盒子，在CSS中，"box model"这一术语是用来设计和布局时使用。

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。

盒子模型允许我们在其它元素和周围元素边框之间的空间放置元素。

下面的图片说明了盒子模型(Box Model)：



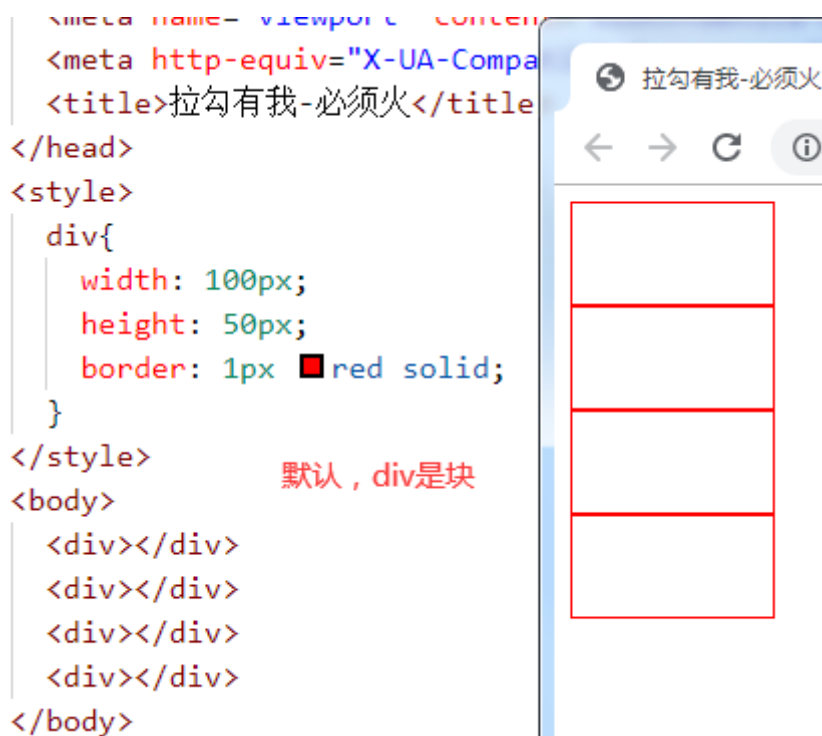
- margin(外边距) - 盒子与盒子之间的距离
- border(边框) - 盒子的保护壳
- padding(内边距/填充) - 内填充，盒子边与内容之间的距离
- content(内容) - 盒子的内容，显示的文本或图像



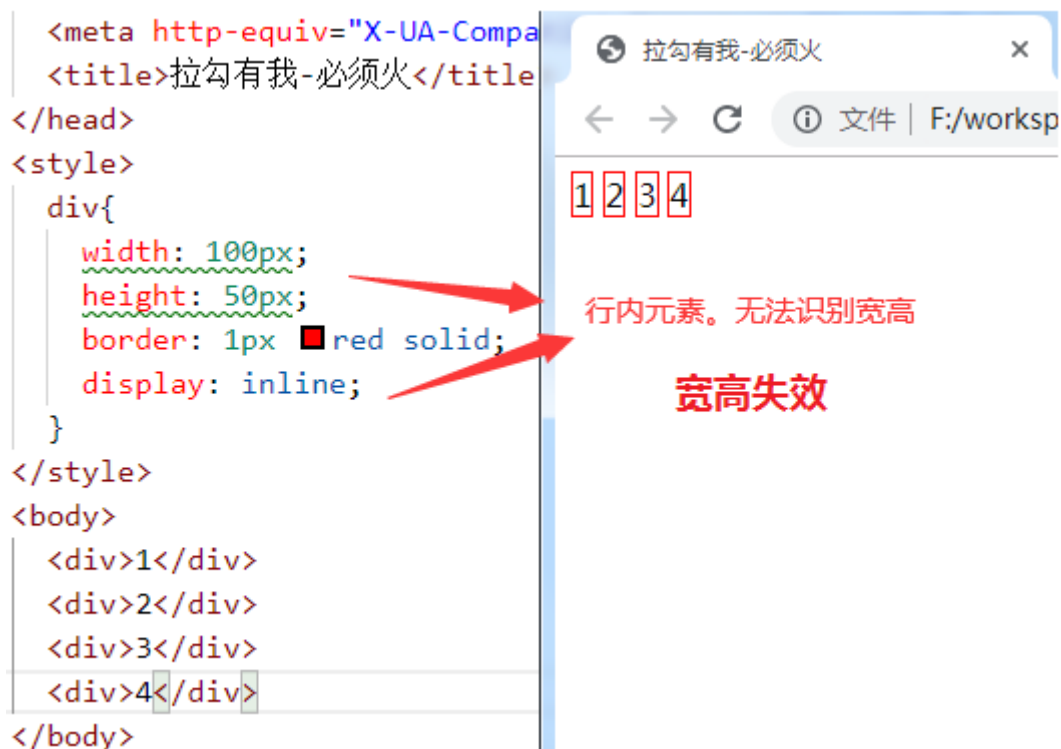
3.3 CSS定位

3.3.1 默认定位

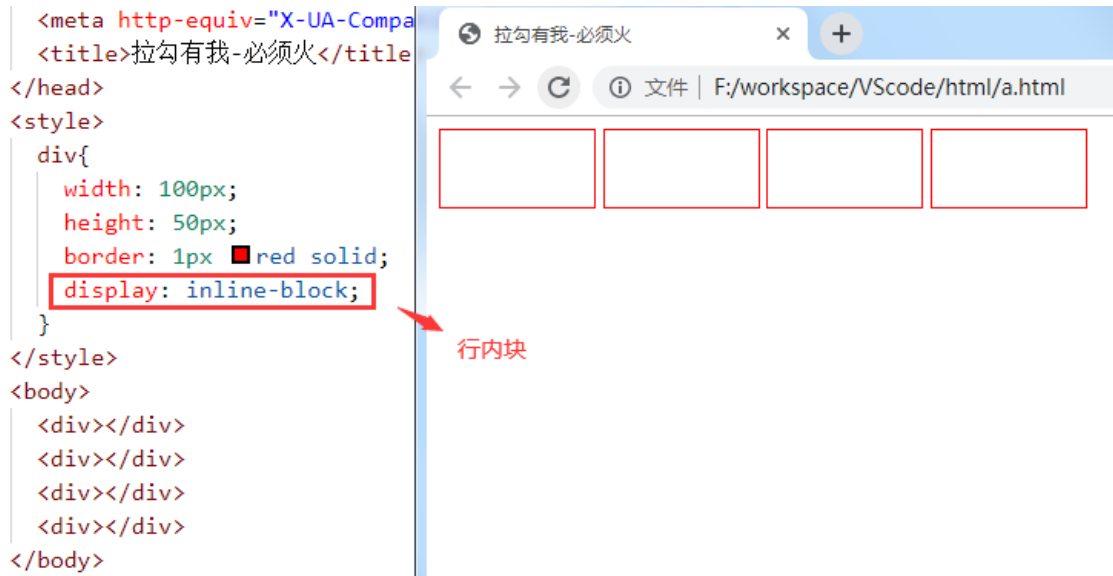
- 块级元素：h1~h6, p, div 等，自上而下，垂直排列（自动换行）；可以改变宽高



- 行内元素：a,b,span,等，从左向右，水平排列（不会换行）；不能改变宽高



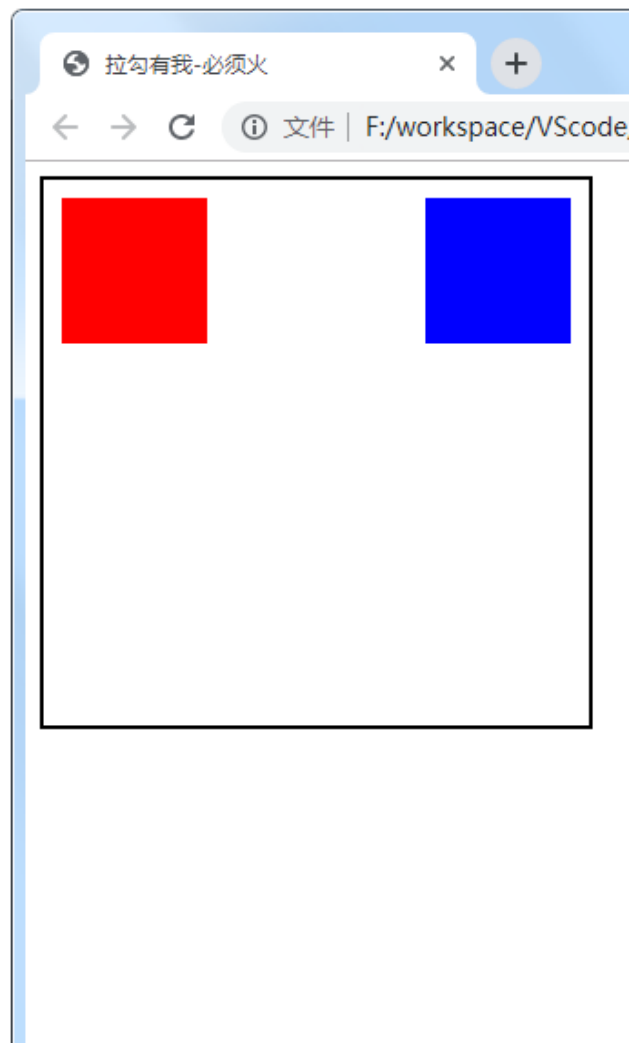
- 行内块元素：input,img等，从左向右，水平排列（自动换行）；可以改变宽高



3.3.2 浮动定位

- 让元素“飞”起来。不仅可以靠着左边或右边。还可以消除“块级”的霸道特性（独自占一行）。
- float取值：
 - none：不浮动
 - left：贴着左边 浮动
 - right：贴着右边 浮动

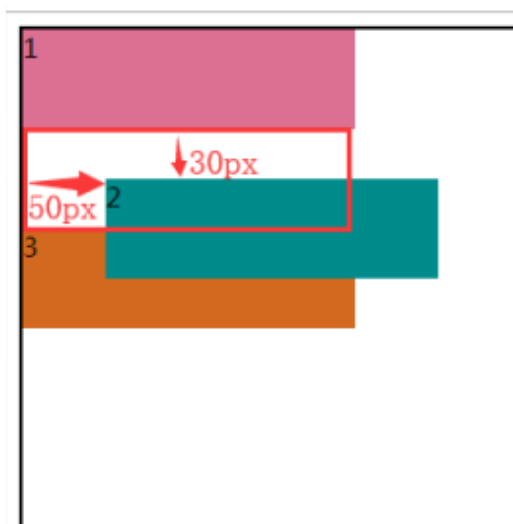
```
<style>
  .da{
    width: 300px;
    height: 300px;
    border: 2px solid black;
  }
  .a,.b{
    width: 80px;
    height: 80px;
  }
  .a{
    background: red;
    float: left;
    margin: 10px;
  }
  .b{
    background: blue;
    float: right;
    margin: 10px;
  }
</style>
<body>
  <div class="da">
    <div class="a"></div>
    <div class="b"></div>
  </div>
</body>
```



3.3.3 相对定位

和原来的位置进行比较，进行移动定位（偏移）

```
<style>
  .da {
    width: 300px;
    height: 300px;
    border: 2px solid black;
  }
  .a,.b,.c {
    width: 200px;
    height: 60px;
  }
  .a {
    background: palevioletred;
  }
  .b {
    background: darkcyan;
    position: relative;
    top: 30px;
    left: 50px;
  }
  .c {
    background: chocolate;
  }
</style>
<body>
  <div class="da">
    <div class="a">1</div>
    <div class="b">2</div>
    <div class="c">3</div>
  </div>
</body>
```



3.3.4 绝对定位

本元素与已定位的祖先元素的距离

- 如果父级元素定位了，就以父级为参照物；
- 如果父级没定位，找爷爷级，爷爷定位了，以爷爷为参照物。
- 如果爷爷没定位，继续向上找，都没定位的话，body是最终选择。

```
<div class="yeye">
  <div class="father">
    <div class="a">1</div>
    <div class="b">2</div>
    <div class="c">3</div>
  </div>
</div>
```

- 以**父节点**作为参照物

```
.yeye {
  width: 400px;
  height: 250px;
  background: ■ olive;
  padding: 20px;      爷爷没定位
}

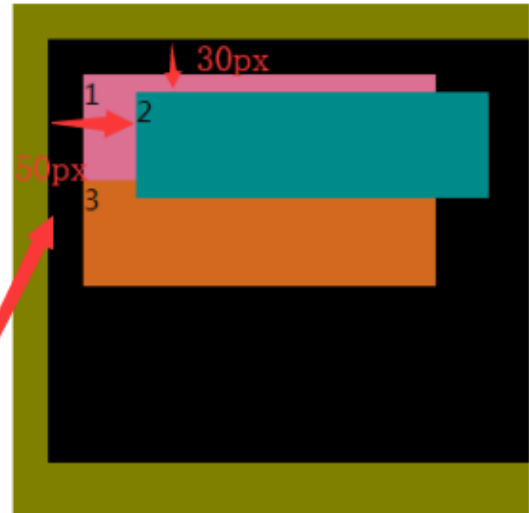
.father {
  width: 300px;
  height: 200px;
  background: ■ black;
  padding: 20px;
  position: relative;  爸爸定位
}

.a,
.b,
.c {
  width: 200px;
  height: 60px;
}

.a {
  background: ■ palevioletred;
}

.b {
  background: ■ darkcyan;
  position: absolute;
  top: 30px;
  left: 50px;
}

.c {
  background: ■ chocolate;
}
```

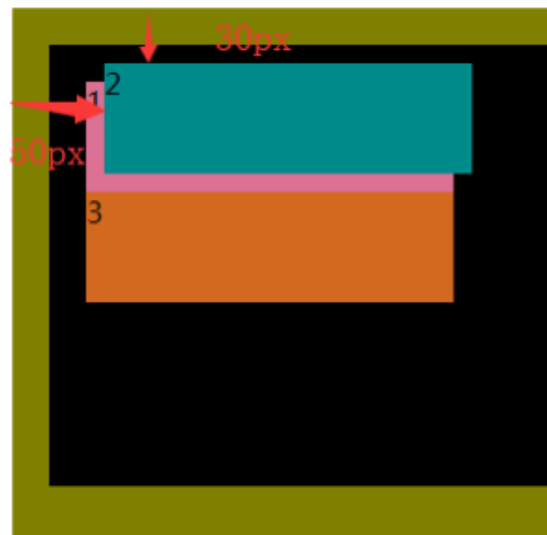


- 以**爷爷节点**作为参照物

```

.yeye {
  width: 400px;
  height: 250px;
  background: ■olive;
  padding: 20px;
  position: relative; 爷爷定位
}
.father {
  width: 300px;
  height: 200px;
  background: ■black;
  padding: 20px;
}
.a,
.b,
.c {
  width: 200px;
  height: 60px;
}
.a {
  background: ■palevioletred;
}
.b {
  background: ■darkcyan;
  position: absolute;
  top: 30px;
  left: 50px;
}
.c {
  background: ■chocolate;
}

```



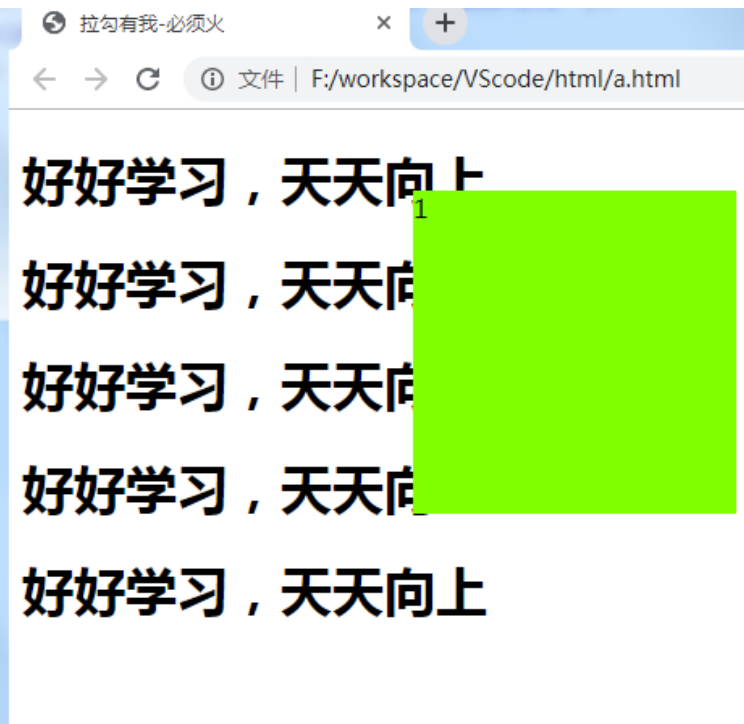
3.3.5 固定定位

将元素的内容固定在页面的某个位置，当用户向下滚动页面时元素框并不随着移动


```

<meta http-equiv="X-UA-Compat
<title>拉勾有我-必须火</title>
</head>
<style>
.a {
  width: 200px;
  height: 200px;
  background: ■ chartreuse;
  position: fixed;
  top: 50px;
  left: 250px;
}
</style>
<body>
<div class="a">1</div>
<h1>好好学习，天天向上</h1>
<h1>好好学习，天天向上</h1>
<h1>好好学习，天天向上</h1>
<h1>好好学习，天天向上</h1>
<h1>好好学习，天天向上</h1>

```



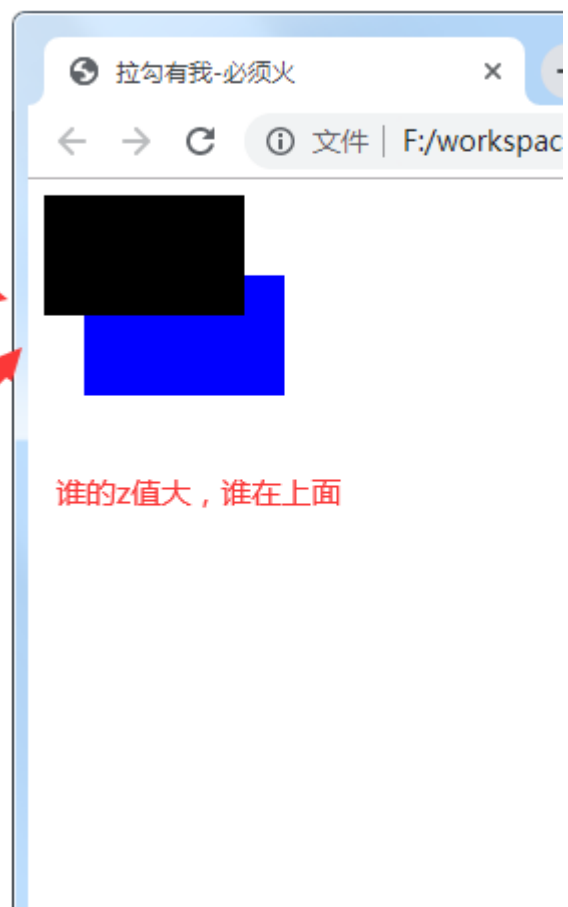
3.3.6 z-index

如果有重叠元素，使用z轴属性，定义上下层次。

```

<style>
.a {
  width: 100px;
  height: 60px;
  background: ■ black;
  position: relative;
  z-index: 4;
}
.b {
  width: 100px;
  height: 60px;
  background: ■ blue;
  position: relative;
  top: -20px;
  left: 20px;
  z-index: 2;
}
</style>
<body>
<div class="a">1</div>
<div class="b">2</div>
</body>

```



注意：

- z轴属性，要配合相对或绝对定位来使用。
- z值没有额定数值（整型就可以，具体用数字几，悉听尊便）



4. CSS3

4.1 圆角

`border-radius` : 左上 右上 右下 左下;

`border-radius` : 四个角;

`border-radius` : 50%; 圆形

<code>border-radius : 1px 10px 20px 30px;</code>	
<code>border-radius : 55px;</code>	

4.2 盒子阴影

`box-shadow` : 1 2 3 4 5;

1 : 水平偏移

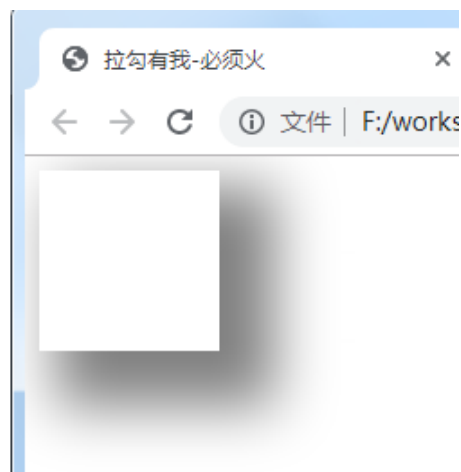
2 : 垂直偏移

3 : 模糊半径

4 : 扩张半径

5 : 颜色

```
<title>拉勾有我-必须火</title>
</head>
<style>
  div {
    width: 100px;
    height: 100px;
    box-shadow: 20px 20px 30px 10px grey;
  }
</style>
<body>
  <div></div>
</body>
```



4.3 渐变

4.3.1 线性渐变

`background: linear-gradient` ([方向/角度], 颜色列表);

```
<style>
  div{
    width: 200px;
    height: 60px;
```

```

        margin: 10px;
    }
    .a1 {
        background: linear-gradient(red,black);
    }
    .a2 {
        background: linear-gradient(red,black,pink, green);
    }
    .a3 {
        background: linear-gradient(to left,red,black);
    }
    .a4 {
        background: linear-gradient(to top left,red,black);
    }
    .a5 {
        background: linear-gradient(30deg,red,black);
    }
</style>
<body>
    <div class="a1"></div>
    <div class="a2"></div>
    <div class="a3"></div>
    <div class="a4"></div>
    <div class="a5"></div>
</body>

```

4.3.2 径向渐变

以圆心向外发散

```
background: radial-gradient(颜色列表);
```

```

<style>
    div {
        width: 200px;
        height: 200px;
        margin: 10px;
    }
    .a1 {
        background: radial-gradient(red, black);
    }
    .a2 {
        background: radial-gradient(red, black, pink, green);
    }
    .a3 {
        border-radius: 50%;
        background: radial-gradient(red, black);
    }
</style>

<body>
    <div class="a1"></div>
    <div class="a2"></div>
    <div class="a3"></div>

```

```
</body>
```

4.4 背景

4.4.1 背景位置

background-origin：指定了背景图像的位置区域

- border-box：背景贴边框的边
- padding-box：背景贴内边框的边
- content-box：背景贴内容的边

```
<style>
  div {
    background: url("img/1.jpg") no-repeat;
    width: 200px;
    height: 80px;
    margin: 20px;
    border: 10px dashed black;

    padding: 20px;
  }
  .a {
    background-origin: border-box;
  }
  .b {
    background-origin: padding-box;
  }
  .c {
    background-origin: content-box;
  }
</style>

<body>
  <div class="a"></div>
  <div class="b"></div>
  <div class="c"></div>
</body>
```

4.4.2 背景裁切

background-clip:

- border-box 边框开切
- padding-box 内边距开切
- content-box 内容开切

```
<style>
  div {
    width: 200px;
    height: 80px;
```

```

border: 10px dashed red;
background-color: darkcyan;
margin: 20px;
padding: 20px;
}
.a {
background-clip: border-box;
}
.b {
background-clip: padding-box;
}
.c {
background-clip: content-box;
}
}
</style>

<body>
<div class="a"></div>
<div class="b"></div>
<div class="c"></div>
</body>

```

4.4.3 背景大小

background-size:

- cover 缩放成完全覆盖背景区域最小大小
- contain 缩放成完全适应背景区域最大大小

```

<style>
div {
background: url("img/1.jpg") no-repeat;
width: 200px;
height: 100px;
border: 2px solid red;
margin: 20px;
}
.a {
background-size: cover; /* 完全覆盖 */
}
.b {
background-size: contain; /* 完全适应 */
}
</style>

<body>
<div class="a"></div>
<div class="b"></div>
</body>

```

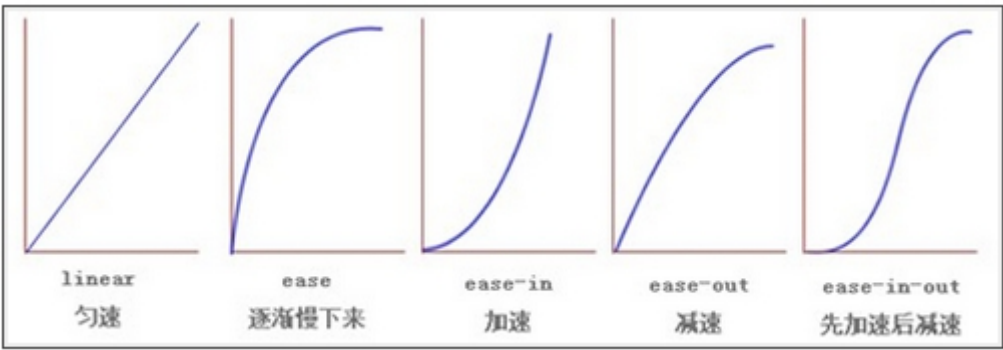
4.5 过渡动画

4.5.1 过渡

从一个状态到另一个状态，中间的“缓慢”过程；
缺点是，控制不了中间某个时间点。

transition { 1 2 3 4 }

- 1：过渡或动画模拟的css属性
- 2：完成过渡所使用的时间（ 2s内完成 ）
- 3：过渡函数。。。



- 4：过渡开始出现的延迟时间

```
transition: width 2s ease 1s;
```

目前，css3只开发出部分的过渡属性，下图所示：

background-color↕	background-position↕	border-bottom-color↕	border-bottom-width↕
border-left-color↕	border-left-width↕	border-right-color↕	border-right-width↕
border-spacing↕	border-top-color↕	border-top-width↕	bottom↕
clip↕	color↕	font-size↕	font-weight↕
height↕	left↕	letter-spacing↕	line-height↕
margin-bottom↕	margin-left↕	margin-right↕	margin-top↕
max-height↕	max-width↕	min-height↕	min-width↕
opacity↕	outline-color↕	outline-width↕	padding-bottom↕
padding-left↕	padding-right↕	padding-top↕	right↕
text-indent↕	text-shadow↕	vertical-align↕	visibility↕
width↕	word-spacing↕	z-index↕	↕

```
<style>
div{
width: 100px;
```

```

        height: 50px;
        border: 2px solid red;
    }

    .a{
        transition: width 2s linear 1s;    /*1秒过后，div在2秒内匀速缓慢的变宽*/
    }

    div:hover{ width: 300px;}    /*触发div变宽*/

</style>
<body>

    <div class="a">Hello, 拉勾</div>

</body>

```

4.5.2 动画

从一个状态到另一个状态，过程中每个时间点都可以控制。

- 关键帧：@keyframes 动画帧 { from{} to{} } 或者{ 0%{} 20%{}... }
- 动画属性：animation{ 1, 2, 3, 4, 5 }
 - 1：动画帧
 - 2：执行时间
 - 3：过渡函数
 - 4：动画执行的延迟（可省略）
 - 5：动画执行的次数

需求1：一个元素从左向右移动，3秒内执行2次

```

<style>
    div{
        width: 700px;
        border: 1px solid red;
    }

    @keyframes x{
        from{ margin-left: 0px;}
        to{ margin-left: 550px;}
    }

    img{
        animation: x 3s linear 2;
    }
</style>
<body>
    <div>
        
    </div>
</body>

```

需求2：一个元素从左向右移动，3秒内执行完成。无限次交替执行

infinite：无限次

alternate：来回执行（交替，一去一回）

```
<style>
.wai{
  width: 600px;
  height: 100px;
  border: 2px solid red;
}
.nei{
  width: 40px;
  height: 80px;
  margin: 5px;
  background: red;
}

.nei{
  animation: x 3s linear infinite alternate;
}

@keyframes x{
  0%{ margin-left: 0px; }
  25%{ background: yellowgreen; }
  50%{ background: goldenrod; }
  75%{ background: palevioletred;}
  100%{
    background: coral;
    margin-left: 550px;
  }
}
</style>

<body>
  <div class="wai">
    <div class="nei"></div>
  </div>
</body>
```