CSS3

一、CSS3属性选择器

选择符	简介
E[att]	选择具有att属性的E元素
E[att="val"]	选择具有att属性且属性值等于val的E元素
E[att^="val"]	匹配具有att属性、且值以val开头的E元素
E[att$="val"]	匹配具有att属性、且值以val结尾的E元素
E[att*="val"]	匹配具有att属性、且值中含有val的E元素

类选择器、属性选择器、伪类选择器，权重为 10

二、结构伪类选择器

选择符	简介
E:first-child	匹配父元素中的第一个子元素E
E:last-child	匹配父元素中最后一个E元素
E:nth-child(n)	匹配父元素中的第n个子元素E
E:first-of-type	指定类型E的第一个
E:last-of-type	指定类型E的最后一个
E:nth-of-type(n)	指定类型E的第n个

nth-child（n）

> n可以是数字，关键字和公式
>
> n如果是数字，就是选择第n个
>
> 常见的**关键词 even 偶数 odd 奇数**
>
> 常见的公式如下（如果n是公式，则从0开始计算）
>
> 但是 第0个元素或者超出了元素的个数会被忽略）

公式	取值
2n	偶数
2n+1	奇数
5n	5　10　15 ...
n+5	从第5个开始（包含第五个）到最后
-n+5	前5个（包含第5个）...

- 结构伪类选择器就是选择第n个
- Nth-child从所有子级开始算的，可能不是同一种类型
- Nth-of-type 是指定同一种类型的子级，比如 ul li:nth-of-type(2) 是选择第2个li
- 关于nth-child（n） 我们要知道n从0开始计算的，要记住常用的公式
- 如果是无无序列表，我们肯定用 nth-child 更多

三、CSS3 伪元素选择器

选择符	简介
::before	在元素内部的前面插入内容
::after	在元素内部的后面插入内容

> - **before 和 after 必须有 content 属性**
> - **before 在内容的前面，after 在内容的后面**
> - before 和 after 创建一个元素，但是属于行内元素。
> - 因为在 dom 里面看不见刚才创建的元素，所以我们称为伪元素
> - 伪元素和标签选择器一样，权重为 1

伪元素字体图标

```
p::before {
          position: absolute;
          right: 20px;
          top: 10px;
          content: '\ea50';
          font-size: 20px;
}
```

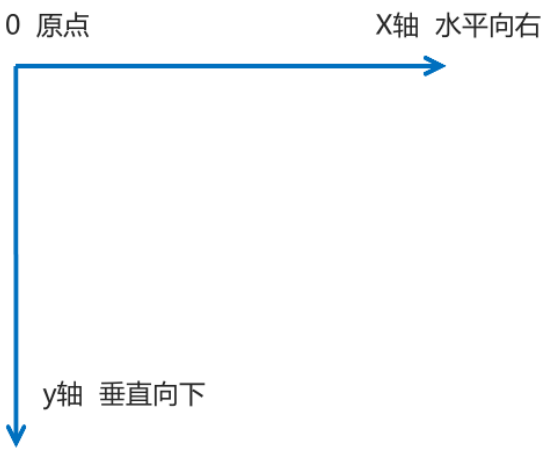
四、CSS3 2D转换

转换（transform）是CSS3中具有颠覆性的特征之一，可以实现元素的位移、旋转、缩放等效果

> - 移动：translate
> - 旋转：rotate

- 缩放：scale

1、二维坐标系



2、2D 转换之移动 translate

2D移动是2D转换里面的一种功能，可以改变元素在页面中的位置，类似定位。

语法

```
transform: translate(x,y); 或者分开写
transform: translateX(n);
transform: translateY(n);
```

1. 定义 2D 转换中的移动，沿着 X 和 Y 轴移动元素
2. translate最大的优点：不会影响到其他元素的位置
3. **translate中的百分比单位是相对于自身元素的 translate:(50%,50%);**
4. **对行内标签没有效果**

3、2D 转换之旋转 rotate

2D旋转指的是让元素在2维平面内顺时针旋转或者逆时针旋转。

语法

```
transform:rotate(度数)
```

1. rotate里面跟度数， 单位是 deg 比如 rotate(45deg)
2. 角度为正时，顺时针，负时，为逆时针
3. 默认旋转的中心点是元素的中心点

案例：三角形

```
p::before {
            content: '';
            position: absolute;
            right: 20px;
            top: 10px;
            width: 10px;
            height: 10px;
            border-right: 1px solid #000;
            border-bottom: 1px solid #000;
            transform: rotate(45deg);
}
```

4、2D 转换中心点 transform-origin

设置元素转换的中心点

语法

```
transform-origin: x y;
```

1. 注意后面的参数 x 和 y 用空格隔开
2. x y 默认转换的中心点是元素的中心点 (50% 50%)
3. 还可以给x y 设置 像素 或者 方位名词 （top bottom left right center）

5、2D 转换之缩放scale

缩放，顾名思义，可以放大和缩小。 只要给元素添加上了这个属性就能控制它放大还是缩小。

语法

```
transform:scale(x,y);
```

1. 注意其中的x和y用逗号分隔
2. transform:scale(1,1)： 宽和高都放大一倍，相对于没有放大
3. transform:scale(2,2)： 宽和高都放大了2倍
4. transform:scale(2)： 只写一个参数，第二个参数则和第一个参数一样，相当于 scale(2,2)
5. transform:scale(0.5,0.5)：缩小
6. sacle缩放最大的优势：可以设置转换中心点缩放，默认以中心点缩放的，而且不影响其他盒子

2D转化综合写法

1. **同时使用多个转换，其格式为：transform: translate() rotate() scale() ...等，**
2. 其顺序会影转换的效果。（先旋转会改变坐标轴方向）
3. **当我们同时有位移和其他属性的时候，记得要将位移放到最前**

五、CSS3动画

动画（animation）是CSS3中具有颠覆性的特征之一，可通过设置多个节点来精确控制一个或一组动画，常 用来实现复杂的动画效果。

相比较过渡，动画可以实现更多变化，更多控制，连续自动播放等效果。

1、动画的基本使用

制作动画分为两步：

1. 先定义动画
2. 再使用（调用）动画

1.1 用keyframes 定义动画（类似定义类选择器）

```
@keyframes 动画名称 {
    0%{
        width:100px;
    }
    100%{
        width:200px;
    }
}
```

动画序列

- 0% 是动画的开始，100% 是动画的完成。这样的规则就是动画序列。
- 在 @keyframes 中规定某项 CSS 样式，就能创建由当前样式逐渐改为新样式的动画效果。
- 动画是使元素从一种样式逐渐变化为另一种样式的效果。您可以改变任意多的样式任意多的次数。
- 请用百分比来规定变化发生的时间，**或用关键词 "from" 和 "to"，等同于 0% 和 100%。**

1.2 元素使用动画

```
div {
        width: 200px;
        height: 200px;
        background-color: aqua;
        margin: 100px auto;
        /* 调用动画 */
        animation-name: 动画名称;
        /* 持续时间 */
        animation-duration: 持续时间;
    }
```

2、动画常用属性

属性	描述
@keyframes	规定动画。
animation	所有动画属性的简写属性，除了animation-play-state属性。
animation-name	规定@keyframes动画的名称。（必须的）
animation-duration	规定动画完成一个周期所花费的秒或毫秒，默认是0。（必须的）
animation-timing-function	规定动画的速度曲线，默认是“ease”。
animation-delay	规定动画何时开始，默认是0。
animation-iteration-count	规定动画被播放的次数，默认是1，还有infinite
animation-direction	规定动画是否在下一周期逆向播放，默认是“normal“,alternate逆播放
animation-play-state	规定动画是否正在运行或暂停。默认是"running",还有"paused"。
animation-fill-mode	规定动画结束后状态，保持forwards回到起始backwards

3、动画简写属性

animation：动画名称 持续时间 运动曲线 何时开始 播放次数 是否反方向 动画起始或者结束的状态;

```
animation: myfirst 5s linear 2s infinite alternate;
```

1. 简写属性里面不包含 animation-play-state
2. 暂停动画：animation-play-state: puased; 经常和鼠标经过等其他配合使用
3. 想要动画走回来，而不是直接跳回来：animation-direction： alternate
4. 盒子动画结束后，停在结束位置： animation-fill-mode： forwards

4、速度曲线细节

animation-timing-function：规定动画的速度曲线，默认是“ease”

值	描述
linear	动画从头到尾的速度是相同的。匀速
ease	默认。动画以低速开始，然后加快，在结束前变慢。
ease-in	动画以低速开始。
ease-out	动画以低速结束。
ease-in-out	动画以低速开始和结束。
steps()	指定了时间函数中的间隔数量（步长）

六、CSS3 3D转换

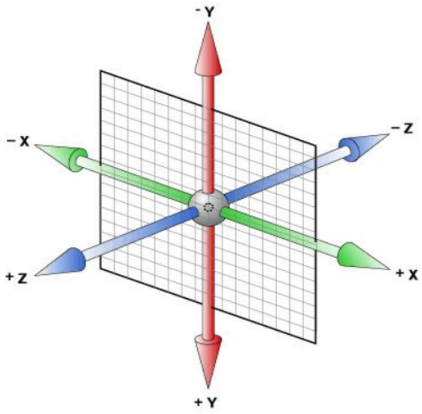
有什么特点

- 近大远小。
- 物体后面遮挡不可见

1、三维坐标系

三维坐标系其实就是指立体空间，立体空间是由3个轴共同组成的。

- x轴：水平向右　　注意：x 右边是正值，左边是负值
- y轴：垂直向下　　注意：y 下面是正值，上面是负值
- z轴：垂直屏幕　　注意：往外面是正值，往里面是负值



2、3D移动 translate3d

3D移动在2D移动的基础上多加了一个可以移动的方向，就是z轴方向。

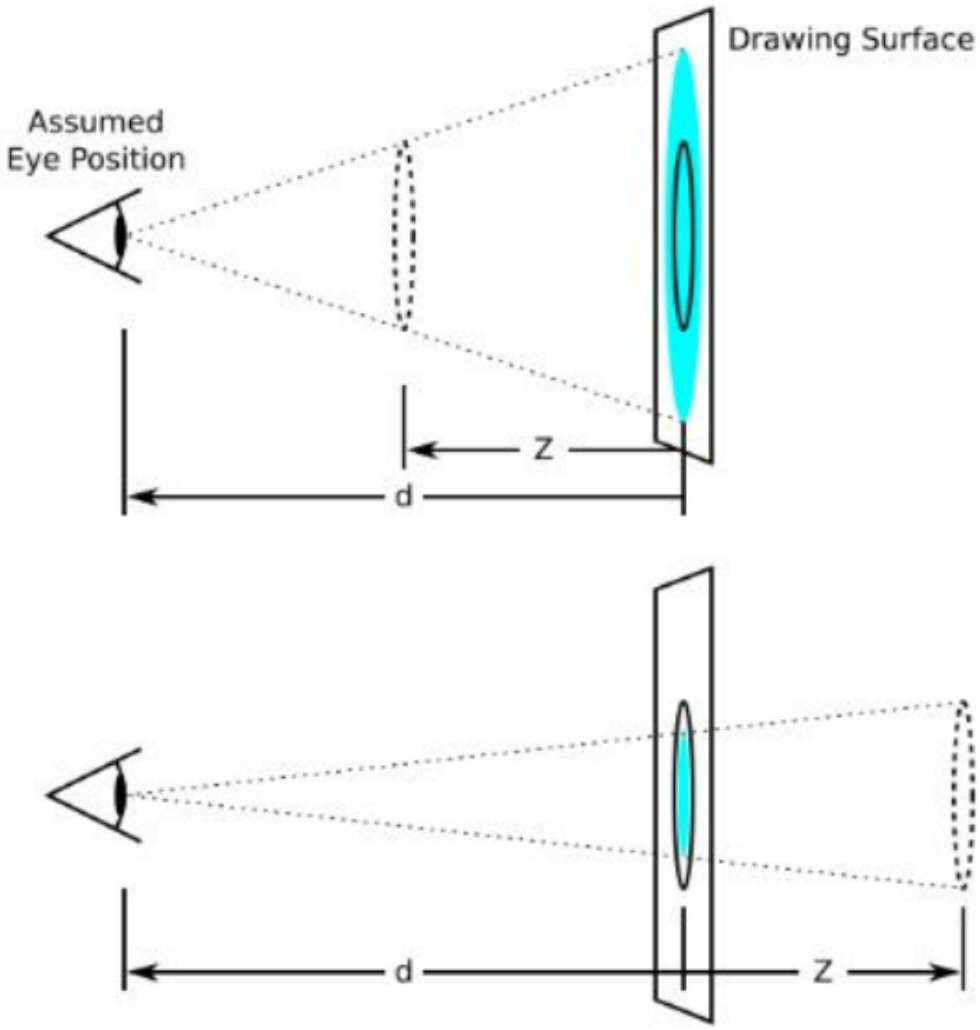
1. translform:translateX(100px)：仅仅是在x轴上移动
2. translform:translateY(100px)：仅仅是在Y轴上移动
3. translform:translateZ(100px)：仅仅是在Z轴上移动（注意：translateZ一般用px单位）
4. transform:translate3d(x,y,z)：其中 x、y、z 分别指要移动的轴的方向的距离

因为z轴是垂直屏幕，由里指向外面，所以默认是看不到元素在z轴的方向上移动

3、透视 perspective

在2D平面产生近大远小视觉立体，但是只是效果二维的

- 如果想要在网页产生3D效果需要透视（理解成3D物体投影在2D平面内）。
- 模拟人类的视觉位置，可认为安排一只眼睛去看
- 透视我们也称为视距：视距就是人的眼睛到屏幕的距离
- 距离视觉点越近的在电脑平面成像越大，越远成像越小
- 透视的单位是像素



> 透视写在被观察元素的父盒子上面的
>
> 1. d：就是视距，视距就是一个距离人的眼睛到屏幕的距离。
> 2. z：就是 z轴，物体距离屏幕的距离，z轴越大（正值） 我们看到的物体就越大。

4、translateZ

translform:translateZ(100px)：仅仅是在Z轴上移动。

有了透视，就能看到translateZ 引起的变化了

> - translateZ：近大远小
> - translateZ：往外是正值
> - translateZ：往里是负值

5、3D旋转 rotate3d

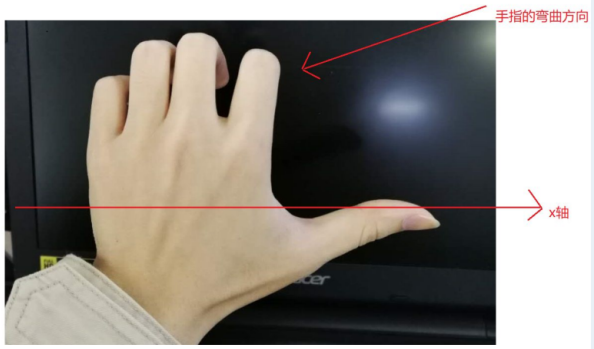
3D旋转指可以让元素在三维平面内沿着 x轴，y轴，z轴或者自定义轴进行旋转。

语法

1. transform:rotateX(45deg)：沿着x轴正方向旋转 45度
2. transform:rotateY(45deg)：沿着y轴正方向旋转 45deg
3. transform:rotateZ(45deg)：沿着Z轴正方向旋转 45deg
4. transform:rotate3d(x,y,z,deg)：沿着自定义轴旋转 deg为角度（了解即可）

左手准则

- 左手的手拇指指向 x轴的正方向
- 其余手指的弯曲方向就是该元素沿着x轴旋转的方向

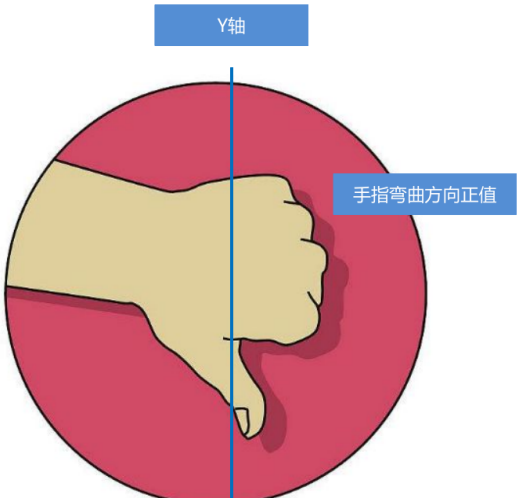


6.5 3D旋转 rotate3d

对于元素旋转的方向的判断 我们需要先学习一个左手准则。

左手准则

- 左手的手拇指指向 y轴的正方向
- 其余手指的弯曲方向就是该元素沿着y轴旋转的方向（正值）



> transform:rotate3d(x,y,z,deg)： 沿着自定义轴旋转 deg为角度（了解即可）
>
> xyz是表示旋转轴的矢量，是标示你是否希望沿着该轴旋转，最后一个标示旋转的角度。
>
> - transform:rotate3d(1,0,0,45deg) 就是沿着x轴旋转 45deg
> - transform:rotate3d(1,1,0,45deg) 就是沿着对角线旋转 45deg

6、3D呈现 transfrom-style

> 控制子元素是否开启三维立体环境。。
>
> transform-style: flat 子元素不开启3d立体空间 默认的
>
> transform-style: preserve-3d; 子元素开启立体空间
>
> **代码写给父级，但是影响的是子盒子**
>
> 这个属性很重要，后面必用

案例-两面翻转的盒子

1. 搭建HTML结构

```
<div class="box">
                <div class="front">黑马程序员</div>
                <div class="back">pink老师等你</div>
</div>
```

- box父盒子里面包含前后两个子盒子
- box 是翻转的盒子 front是前面盒子 back是后面盒子

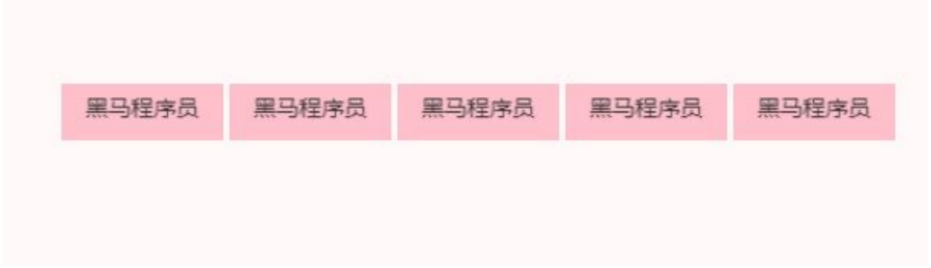
2. CSS样式

① box指定大小，切记要添加3d呈现

② back盒子要沿着Y轴翻转180度

③ 最后鼠标经过box 沿着Y旋转180deg

案例-3D导航栏



1. 搭建HTML结构

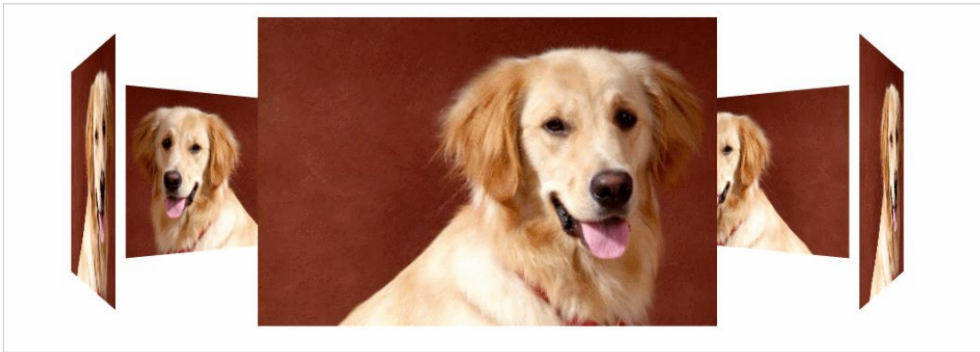
```
<ul>
    <li>
            <div class="box">
                <div class="front">黑马程序员</div>
                <div class="bottom">pink老师等你</div>
            </div>
    </li>
</ul>
```

- li 做导航栏
- .box 是翻转的盒子 front是前面盒子 bottom是底下盒子

2. CSS样式

① li设置大小，加透视和 3d呈现
② front 需要前移 17.5像素
③ bottom 需要下移 17.5像素 并且要沿着x轴翻转 负90度
④ 鼠标放到box 让盒子旋转90度

案例-旋转木马



1. 搭建HTML结构

```
<section>
    <div></div>
    <div></div>
    <div></div>
    <div></div>
    <div></div>
    <div></div>
</section>
```

- 里面的6个div 分别是 6个狗狗图片
- 注意最终旋转是section标签 旋转

2. CSS样式

① 给body添加 透视效果 perspective: 1000px;
② 给section 添加 大小，一定不要忘记添加 3d呈现效果控制里面的6个div
    - 别忘记子绝父相，section要加相对定位
③ 里面6个div 全部绝对定位叠到一起，然后移动不同角度旋转和距离
    - 注意：旋转角度用rotateY 距离 肯定用 translateZ来控制
④ 给section 添加动画animation ，让它可以自动旋转即可

七、浏览器私有前缀

浏览器私有前缀是为了兼容老版本的写法，比较新版本的浏览器无须添加。

1、私有前缀

1. -moz-：代表 firefox 浏览器私有属性
2. -ms-：代表 ie 浏览器私有属性
3. -webkit-：代表 safari、chrome 私有属性
4. -o-：代表 Opera 私有属性

```
-moz-border-radius: 10px;
-webkit-border-radius: 10px;
-o-border-radius: 10px;
border-radius: 10px;
```