过渡（transition）只关注起始状态和最终状态，首次绘制未结束不会触发

|  |  |  |  |
| --- | --- | --- | --- |
| 属性 | 可取值 | 说明 | 备注 |
| transition-property  默认值：all | all  所有可以动画的属性， | 当第一个有三个属性，逗号隔开，速度只有两个的时候，时间从头循环对应属性 | 读取快，渲染速度慢，如果在：hover中有属性值，盒子中也指定了属性值，显示按照hover先变化 离开了再按照盒子的变化 |
| transition-duration | 数字+单位 | 速度的大小（注：都要带单位）duration只要有值，指定属性变化的动画都是按照这个歌值 |
| **transition-timing-function**  **坑：**   1. 组合变化顺序要一样 2. 没办法拿到过渡的每一个坑（tween）   3.元素首次没有绘制完不能过渡 | ease： | （加速然后减速）默认值，ease函数等同于贝塞尔曲线(0.25, 0.1, 0.25, 1.0) |  |
| linear：（匀速） | 同于贝塞尔曲线(0.0, 0.0, 1.0, 1.0). |  |
| ease-in：(加速) | 等同于贝塞尔曲线(0.42, 0, 1.0, 1.0). |  |
| ease-out：（减速） | 等同于贝塞尔曲线(0, 0, 0.58, 1.0). |  |
| ease-in-out：(加速后减速） | 等同于贝塞尔曲线(0.42, 0, 0.58, 1.0) |  |
| cubic-bezier： | 贝塞尔曲线，就是看先变快匀速缩小再加速 |  |
| step-start(数字，star)： | 等同于s teps(1,start)直接一步到终点，再等五秒 |  |
| step-end(数字，end)： | 等同于steps(1,end)等五秒再一步到终点 |  |
| **steps(<integer>,[,[start|end]]?)** | 第一个参数：必须为正整数，指定函数的步数  第二个参数：指定每一步的值发生变化的时间点（默认值end） |  |
| transition-delay | 数字+单位  属性值  值以秒（s）或毫秒（ms）为单位，表明动画过渡效果将在何时开始。取值为正时会延迟一段时间来响应过渡效果；取值为负时会导致过渡立即开始 | 等待的时间，再慢慢过渡到终点。  默认值：0s  你可以指定多个延迟时间，每个延迟将会分别作用于你所指定的相符合的css属性。如果指定的时长个数小于属性个数，那么时长列表会重复。如果时长列表更长，那么该列表会被裁减。两种情况下，属性列表都保持不变 |  |
| **transitionend** | 函数方法  属性值：值以秒（s）或毫秒（ms）为单位，表明动画过渡效果将在何时开始。取值为正时会延迟一段时间来响应过渡效果；取值为负时会导致过渡立即开始 | 这个函数是只能用dom2的形式进行绑定函数，addEvenlistener(transitionend,function{在这里面写过渡完成时发生的时间}）（每一个拥有过渡的属性在其完成过渡时都会触发一次transitionend事件） | 在transition完成前设置 display: none，事件同样不会被触发  放在完成后也不会触发，除非在完成后加一个定时器里面加一个display：none才会被触发 |

**2D变形**（transform）

|  |  |  |
| --- | --- | --- |
| 可取值 | 说明 |  |
| rotate（200deg）旋转 | **正值**:顺时针旋转 rotate(360deg)  **负值**:逆时针旋转 rotate(-360deg)  只能设**单值**。正数表示**顺时针旋**转，负数表示逆时针旋转 |  |
| translate(200px)平移 | **X方向平移**:transform: translateX(tx)  **Y方向平移**:transform: translateY(ty)  **二维平移：**transform: translate(tx[, ty])； 如果ty没有指定，它的值默认为0。 | 可设单值，也可设双值。  正数表示XY轴正向位移，负数为反向位移。设单值表示只X轴位移，Y轴坐标不变，  例如transform: translate(100px);  等价于transform:translate(100px,0); |
| transform:skewX(45deg);  度数越大，拉得越长 | **X方向倾斜:transform: skewX(angle)**  skewX(45deg):参数值以deg为单位 代表与y轴之间的角度  **Y方向倾斜:transform: skewY(angle)**  skewY(45deg):参数值以deg为单位 代表与x轴之间的角度  **二维倾斜:transform: skew(ax[, ay]);**  如果ay未提供，在Y轴上没有倾斜  skew(45deg,15deg):参数值以deg为单位 第一个参数代表与y轴之间的角度 | 第二个参数代表与x轴之间的角度  单值时表示只X轴扭曲，Y轴不变，如transform: skew(30deg);等价于 transform: skew(30deg, 0);考虑到可读性，不推荐用单值，应该用transform: skewX(30deg);。  skewY表示只Y轴扭曲，X轴不变  正值:拉正斜杠方向的两个角  负值:拉反斜杠方向的两个角 |
| transform:scale(2);  就是坐标放大缩小 | X方向缩放:transform: scaleX(sx);  Y方向缩放:transform: scaleY(sy);  二维缩放 :transform: scale(sx[, sy]); (如果sy 未指定，默认认为和sx的值相同) | 要缩小请设0.01～0.99之间的值，要放大请设超过1的值。  例如缩小一倍可以transform: scale(.5);  放大一倍可以transform: scale(2);  如果只想X轴缩放，可以用scaleX(.5)相当于scale(.5, 1)。  同理只想Y轴缩放，可以用scaleY(.5)相当于scale(1, .5)  正值:缩放的程度  负值:不推荐使用（有旋转效果）  单值时表示只X轴,Y轴上缩放粒度一样，如transform: scale(2);等价于transform: scale(2,2); |
| transform-origin  默认值是中点 | transform-origin CSS属性让你更改一个元素变形的基点。  给数字就是参照值是左上角  给关键字方向英文参照中心点 | 按理在盒子的左上角,一个值数值就到右下角移动 |

变化组合时，变换函数的执行计算是从右往左

**矩阵**

|  |  |  |  |
| --- | --- | --- | --- |
| 属性 | 矩形表示 | 说明 | 备注 |
| rotate(θ) | graphic | 即等价于矩阵变换函数 matrix(cosθ, sinθ, -sinθ, cosθ, 0, 0)。 |  |
| translate(X, Y) | graphic | 即等价于使用矩阵变换函数 matrix(1, 0, 0, 1, X, Y)。 |  |
| skew(α, β)， | graphic | 即等价于使用矩阵变换函数 matrix(1, tanβ, tanα,1, 0, 0)。 |  |
| scale(scaleX, scaleY) | graphic | 即等价于使用矩阵变换函数 matrix(scaleX, 0, 0, scaleY, 0, 0) |  |

3D(transfrom)

|  |  |  |
| --- | --- | --- |
| 属性 | 说明 |  |
| **transfrom:rotateX(角度deg)** | 绕着X轴转 |  |
| **transfrom:rotateY(角度deg)** | 绕着Y轴转 |  |
| **transfrom:rotate(角度deg)** | 相当于rotate（角度） |  |
| **transfrom:rotate3d(x,y,z,angle)** | x, y, z分别接受一个数值(number),用来计算矢量方向(direction vector)，矢量方向是三维空间中的一条线, 从坐标系原点到x, y, z值确定的那个点，元素围绕这条线旋转angle指定的值 |  |
| **transfrom:translateX(length)** | 靠着X轴位移 |  |
| **transfrom:translateY(length)** | 靠着Y轴位移 |  |
| **transfrom:translateZ(length)** | 是3D Transformaton特有的， |  |
| **translate3d(translateX,translateY,translateZ);** | 第三个值不能设置百分比 z轴没有厚度之说 |  |
| **transform: scaleZ(number)** | 如果只设置scaleZ(number)，你会发现元素并没有被扩大或压缩，scaleZ(number)需要和translateZ(length)配合使用，number乘以length得到的值，是元素沿Z轴移动的距离，从而使得感觉被扩大或压缩 |  |
| **transform: scale3d(scaleX,scaleY,scaleZ);** | 三者合一的写法 |  |
| **perspective：200px**  **近大远小** | 景深：镜头到人的距离 ，不可继承，作用与后代元素，会和后代元素叠加  在我们CSS3中，perspective用于激活一个3D空间，属性值就是景深大小（默认none无景深） | 景深越大，元素离我们越远，效果就不好， |
| **transform: perspective(depth)** | depth的默认值是none，可以设置为一个长度值，这个长度是沿着Z轴距离坐标原点的距离。1000px被认为是个正常值，若使用perspective()函数，那么他必须被放置在transform属性的首位，如果放在其他函数之后，则会被忽略 |  |
| **perspective-origin：** | 同perspective属性，也是设置在父元素上，对子元素起作用。 这个属性来设置你在X, Y轴坐标确定的那个点来看这个元素，Z轴是被perspective属性设置的 |  |
| **灭点** | 指的是立体图形各条边的延伸线所产生的相交点。透视点的消失点灭点：景深越大灭点越小，变形越大 |  |
| **transform-style**  **不可继承，值作用于子元素** | 这个属性指定了子元素如何在空间中展示，只有两个属性值：flat（默认）和preserve-3d float 表示所有子元素在2D平面呈现  preserve-3d 表示所有子元素在3D平面呈现， | 如果被扁平化，则子元素不会独立的存在于三维空间。因为该属性不会被（自动）继承，所以必须为元素所有非叶后代节点设置该属性 |
| **backface-visibility**  **隐藏背面可分正面和背面** | backface-visibility属性用来设置，是否显示元素的背面，默认是显示的。  backface-visibility: keyword;  keyword有两个值，hidden和visible，默认值是visible。 |  |