【第 1284 期】大学没学过数学也要理解 CSS3 transform 中的 matrix

明非前端早读课5 天前

前言

又到月底了，这几天过了就到 6 月份了。今日早读文章由手淘@明非投稿分享。

> *@明非，斜杠实验室成员，就职于手机淘宝互动前端团队。*

正文从这开始～

CSS3 中使用 transform 可以对元素进行变换。其中包含：位移、旋转、偏移、缩放。 transform 可以使用 translate/rotate/skew/scale 的方式来控制元素变换，也可以使用 matrix 的方式来控制元素变换。

比如：

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

通过 transform 属性进行变换。

首先演示使用 translate/rotate/skew/scale 的方式：

```
.box {
      width: 100px;
      height: 100px;
      background: #00C487;
```

```
        transform: translate(10px, 20px) rotate(30deg) scale(1.5, 2);
}
```

也可以使用 matrix 的方式：

```
.box {
        width: 100px;
        height: 100px;
        background: #00C487;
        transform: matrix(0.75, 0.8, -0.8, 1.2, 10, 20);
}
```

查看 demo:https://codepen.io/fanmingfei/pen/pVYdpO

Matrix 的中文是矩阵，是一个数学术语，在计算机科学中，会用矩阵来对象量进行变换，在 CSS3 的 transform 属性中，可以使用矩阵对图像进行变换。

矩阵长什么样子？

矩阵可以分为一个形容词+一个名字，矩是形容词，阵是名词。

如果你喜欢看战争片，不管是古代战争还是现代战争，都需要有阵势，打仗没阵型，等于耍流氓；或者是开一局农药，可能也要考虑各个英雄的站位，各种球类运动、各种棋类都需要有阵型。

阵型中的每一个个体对整体的都会产生影响。比如打王者荣耀射手时候，射手应该猥琐在一个位置输出，站错位置，输掉整个游戏。

那，其实矩阵就是一些列的数字按照矩形排列。

在数学中，矩阵用方括号包裹起来。

上图就是一个矩阵。

CSS3 里的 matrix 如何和矩阵对应呢？

为什么要用矩阵来表示转换呢？因为在计算机科学中，矩阵可以对向量进行转换。矩阵中的每一个数字，对向量的转换都会产生影响。

CSS3 里面可以用矩阵表示 2D 和 3D 转换，这里只讲 2D。

```
selector {
      transform: matrix(a, b, c, d, e, f);
}
```

2D 的转换是由一个 3*3 的矩阵表示的，前两行代表转换的值，分别是 a b c d e f，要注意是竖着排的，第一行代表 x 轴发生的变化，第二行代表 y 轴发生的变化，第三行代表 z 轴发生的变化，因为这里是 2D 不涉及 z 轴，所以这里是 0 0 1。

假设一个问题

创建一个宽高为 200px 的 div，div 里面有一个红色的点，位置是 {x:181px y:50px}。

倘若将这个 div 向右平移 10px，x 轴向下平移 20px，旋转 37°，x 轴缩放 1.5 倍，y 轴缩放 2 倍：

transform: translate(10px, 20px) rotate(37deg) scale(1.5, 2);

那么红色点的变化后的位置在哪里呢?

既然我们知道矩阵可以对向量进行转换那么我们只要把上面的信息转换成矩阵信息，通过矩阵信息可以将我们的原始坐标转换到新的坐标。

缩放 scale(x, y)

缩放对应的是矩阵中的 a 和 d，x 轴的缩放比例对应 a，y 轴的缩放比例对应 d。

transform: scale(x,y);

a=x d=y

所以 scale(1.5, 2) 对应的矩阵是:

transform: matrix(1.5, 0, 0, 2, 0, 0);

如果一个没有元素没有被缩放，默认 a=1 d=1。

平移 translate(10, 20)

平移对应的是矩阵中的 e 和 f，平移的 x 和 y 分别对应 e 和 f。

transform: translate(10, 20)

e=10

f=20

对应： transform: matrix(a, b, c, d10, 20);

结合缩放： transform: matrix(1.5 0, 0, 2, 10, 20);

平移只会影响 e 和 f 值。

旋转 rotate(θdeg)

旋转影响的是 a/b/c/d 四个值，分别是什么呢?

transform: rotate(θdeg)

a=cosθ

b=sinθ

c=-sinθ

d=cosθ

这个是高中学的哦~

如果要计算 30° 的 sin 值：

首先我们要将 30° 转换为弧度，传递给三角函数计算。用 JS 计算就是下面的样子了。

```
// 弧度和角度的转换公式：弧度=π/180×角度
const radian = Math.PI / 180 * 30 // 算出弧度
const sin = Math.sin(radian) // 计算 sinθ
```

```
const cos = Math.cos(radian) // 计算 cos θ
console.log(sin, cos) // 输出 ≈ 0.5, 0.866
```

这样我们算出了 sin 和 cos，分别是 0.5 和 0.866

如果我们不考虑缩放和偏移，只旋转 30°，矩阵应该表示为

transform: rotate(30deg)

a=0.866

b=0.5

c=-0.5

d=0.866

transform: matrix(0.866, 0.5, -0.5, 0.866, 0, 0);

偏移 skew(20deg, 30deg)

上面的题目中没有出现出现偏移值，偏移值也是由两个参数组成，x 轴和 y 轴，分别对应矩阵中的 c 和 b。是 x 对应 c，y 对应 b， 这个对应并不是相等，需要对 skew 的 x 值 和 y 值进行 tan 运算。

transform: skew(20deg, 30deg);

b=tan30°

c=tan20°

注意 y 对应的是 c，x 对应的是 b。

transform: matrix(a, tan(30deg), tan(20deg), d, e, f)

使用 JS 来算出 tan20 和 tan30

```
// 先创建一个方法，直接返回角度的 tan 值
function tan (deg) {
      const radian = Math.PI / 180 * deg
      return Math.tan(radian)
}
const b = tan(30)
const c = tan(20)
console.log(b, c) // 输出 ≈ 0.577, 0.364
```

b=0.577 c=0.364

transform: matrix(1, 0.577, 0.364, 1, 0, 0)

旋转+缩放+偏移+位移怎么办？

如果我们既要旋转又要缩放又要偏移，我们需要将旋转和缩放和偏移和位移多个矩阵相乘，**要按照 transform 里面 rotate/scale/skew/translate 所写的顺序相乘。**

这里我们先考虑旋转和缩放，需要将旋转的矩阵和缩放的矩阵相乘

实在是用语言解释不清楚如何去乘，用一张图解释吧：

这里我用小写字母代表第一个矩阵中的值，大写字母代表第二个矩阵里的值

将我们的已经得到的矩阵带入到公式

得出：

transform: rotate(30) scale(1.5 2);

转换为 matrix 表示为：

transform: matrix(1.299, 0.75, -1, 1.732, 0, 0);

找到这次转换的矩阵

div 的 transform 值如下

transform: translate(10px, 20px) rotate(37deg) scale(1.5, 2);

translate(10px, 20px)

x 平移 10px，y 平移 20px，所以 e=10，f=20。

rotate(37deg)

sin37° ≈ 0.6

cos37° ≈ 0.8

根据 a 对应 cos b，对应 sin，c 对应 -sin，d 对应 cos 的值

得到：

a=0.8，b=0.6，c=-0.6，d=0.8

scale(1.5, 2)

x 轴缩放 1.5，y 轴缩放 2，所以 a=1.5，d=2

结合

transform: translate(10px, 20px) rotate(37deg) scale(1.5, 2);

我们使用 位移矩阵 *旋转矩阵* 缩放矩阵（根据 transform 中的变换类型书写的顺序）

可以使用矩阵计算器进行计算

从左往右依次计算

所以最终得到矩阵

matrix(1.2, 0.9, -1.2 1.6, 10, 20)

验证一下

transform: matrix(1.2, 0.9, -1.2 1.6, 10, 20)

和

transform: translate(10px, 20px) rotate(37deg) scale(1.5, 2);

效果是一样的

如何对一个坐标进行矩阵变换

我们已经知道了这个矩阵，如何通过矩阵对一个坐标进行变化，找到这个坐标变化后的位置呢?

我们用之前得出的变换矩阵去乘以这一个坐标组成的 3*1（三排一列）矩阵。

上面已经介绍过如何进行矩阵乘法了，这里在介绍一遍

上图中左右两个矩阵颜色相同的位置相乘后相加，每一行都进行这样的计算:

得到一个 3*1 的矩阵，第一行是转换后的 x 值，第二行是转换后的 y 值，第三行是转换后的 z 值（2d 不考虑 z 值）。

前面讲到，矩阵的第一行影响 x，第二行影响 y，也体现在这个地方。

假设我们的坐标是(50, 80)，这里还没有针对我们提出的问题上面的点进行计算。

我们把坐标写成矩阵的形式，设置 z 轴是 1:

然后进行乘法计算：

通过我们计算出来的矩阵变换得到新的位置(46, 172)

继续刚刚问题

坐标是需要基于一个坐标系存在的，我们需要找到正确的坐标系才能算出准确的坐标。 在 CSS transform 中，有个属性是 transform-origin，来设置变换所基于的点，默认是 transform-origin: 50% 50%，基于中间元素的中心点。我们需要以这个点建立坐标系。

在网页中，坐标系是 x 轴向右，y 轴向下。

转换前：

转换后：

根据题目我们知道，这个点相对于绿色 div 左上角的坐标是(181, 50) 绿色 div 的宽高为 200 基于绿色 div 中心点建立的坐标系，这个点的坐标是(81, -50)

将坐标代入公式进行计算：

得到坐标约为(167, 13)

再将这个坐标转换成页面坐标系(267,113)

最终我们得到了这个点在经过转换后的坐标

总结

矩阵在计算机图形学中运用非常多，就像我们经常用的 PhotoShop，虽然是设计软件，但它的图形也是依赖各种数学能力进行计算后展现的。我们玩的游戏、看的 3D 电影，其实都和数学息息相关，学好这些知识，才能真正的成为发明者，即便不成为发明者，在应用层理解这些，让我们能做的事情更多。

关于本文

作者：@明非

原文：https://fanmingfei.com/posts/CSS3_Transform_Matrix_Intro.html

招聘：对前端图形感兴趣或者对手淘感兴趣的童鞋，可以发邮件到 mingfei.fmf@alibaba-inc.com

最后，为你推荐

【第 1249 期】使用 CSS 来做素数的判定与筛选

阅读原文

微信扫一扫

关注该公众号