Web前端开发

第13章：动画

阮晓龙

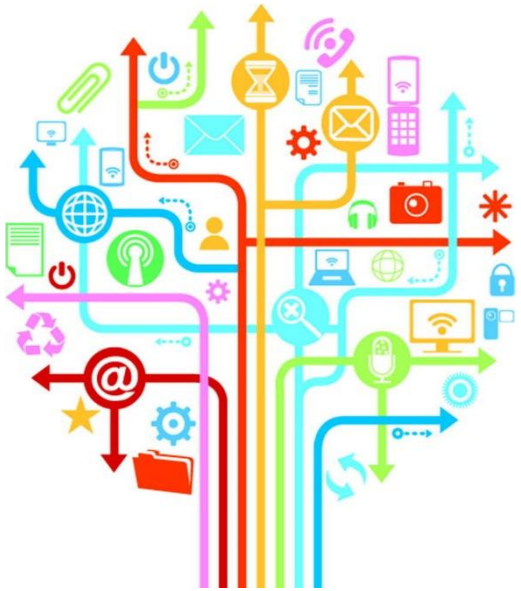
13938213680 / rxl@hactcm.edu.cn
http://web.book.51xueweb.cn
http://www.51xueweb.cn

河南中医药大学管理科学与工程学科

2018.5

本章主要内容

- Web动画
- 使用变形属性
- 使用过渡属性
- 使用动画属性
- 案例：引人入胜的动态照片墙



1.Web动画

- GIF（Graphics Interchange Format，图形互换格式）是为跨平台而开发的动画格式，从1989年GIF89a格式问世之后就变得十分流行，Web前端开发者可通过GIF实现简单的网页动画。
- GIF支持透明背景图像，适合在不同背景的网页上展示。同时许多操作系统中都可以使用GIF，使其展示有着跨平台的优势。另外，GIF图片的“体型”很小，很适合在网上传播。

现场演示：

- 案例13-01：在页面中使用GIF动画

1.Web动画

- Flash是由MacroMedia公司（后被Adobe公司收购）推出的交互式矢量图和Web动画的标准，它以绚丽的视觉效果和丰富的交互体验而著称。
- Flash是作为浏览器的一个扩展出现，为当时的浏览器提供了它们所不具备的功能，正是因为Flash动画在网页中的广泛应用才使得Web前端呈现一种多媒体效果。
- 通过Flash技术，Web前端开发者可以实现动态的视觉效果和多元化的信息传达。

现场演示：

- 案例13-02：在页面中使用Flash动画

1.Web动画

- JavaScript动画是Web前端中另外一种重要的动画形式。
  - 其实现主要基于JavaScript对HTML的操作，通过JavaScript与CSS样式的结合可以实现丰富多彩的动画效果。
- JavaScript动画在Web前端开发中以其良好的兼容性、灵活的控制力占据着Web动画开发的半壁江山。
- JavaScript动画的制作偏向编程，开发使用难度较高，而且其开发需要具有丰富的Web前端开发经验和编程经验。

1.Web动画

- 动画是CSS3的一种新特性，它可以在不借助JavaScript和Flash的情况下使大多数HTML元素动起来。
- CSS动画的出现使的Web前端动画的开发变得更为简单，Web前端开发者只要精通HTML和CSS就可以做出漂亮的动画，甚至可以较为容易地实现对JavaScript来说十分困难的2D、3D效果。

2.使用变形属性

- CSS3变形就是页面元素的一些展示效果的集合，比如平移、旋转、缩放和倾斜效果，每个效果都可通过变形函数（Transform Function）实现。
- 变形函数主要包括有rotate（旋转）、skew（扭曲）、scale（缩放）和translate（移动）以及matrix（矩阵变形）形式，变形属性语法格式如下所示。

```
transform: none|transform-functions;
```

  - none：默认值为，表示不进行变形；
  - transform-function：表示一个或者多个变形函数，以空格分开就是同时对一个元素进行多种变形的属性操作。

现场演示：

- 案例13-03：使用transform属性进行简单变形

2.使用变形属性

- transform-origin属性
  - 该属性用来指定元素的中心点位置，默认情况下，变形的原点在元素的中心点，也就是元素X轴和Y轴的50%处，其语法如下所示。

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

  - 该属性值可以是百分比、em、px等具体的值，也可以是top、bottom、right、left和center关键词。
  - 2D变形中该属性可以设置一个参数值，也可设置两个属性值。
  - 3D变形中该属性还包括了Z轴的值，设置元素在三维空间中的像素原点。

现场演示：

- 案例13-04：改变元素transform-origin属性

2.使用变形属性

- transform-style属性
  - 该属性是3D变形中的一个重要属性，指定其子元素在3D空间中如何呈现，其语法如下所示。

```
transform-style: flat|preserve-3d;
```

  - 该属性主要包括以下两个值：
    - flat：默认值，表示所有子元素在2D平面呈现。
    - preserve-3d：所有子元素在3D空间中展示。

现场演示：

- 案例13-05：transform-style属性效果

2.使用变形属性

- perspective属性
  - 该属性用于设置观看者的位置，并将可视内容映射到一个视锥上，继而投到2D视平面上。
  - 如果不指定该属性，则Z轴空间中的所有点将平铺到同一个2D视平面中，并且变换结果中将不存在景深概念。
  - 该属性语法如下所示。

```
perspective: number|none;
```

    - none：默认值，表示以无限的视角来看3D物体，但看上去是平的。
    - number：接受一个单位大于0的值，其单位不能为百分比值。

现场演示：

- 案例13-06：perspective属性效果展示

2.使用变形属性

- perspective-origin属性
  - 该属性的使用需要配合perspective属性，主要用来决定观看者的视觉观测点在Z平面的位置，它实际上设置了X轴和Y轴的位置，可设置不同的视觉观测点，其语法如下所示。

```
perspective-origin: x-axis y-axis;
```

  - 第一个值指定视觉观测点在元素X轴上的位置，属性值是长度值、百分比或者以下三个关键词之一。
    - left：在包含框的X轴方向长度的0%。
    - center：中间点。
    - right：长度的100%。

2.使用变形属性

- perspective-origin属性
  - 第二个值指定视觉观测点在元素Y轴上的位置，属性值是长度值、百分比或者以下三个关键词之一。
    - top：在包含框的Y轴方向长度的0%。
    - center：中间点。
    - bottom：长度的100%。

现场演示：

- 案例13-07：perspective-origin属性效果

2.使用变形属性

- backface-visibility属性
  - 该属性决定元素旋转后其背面是否可见，对于未旋转的元素，该元素的正面面向观看者。
  - 如果该元素沿Y轴旋转约180度时会导致元素的背面面对观看者，这时就需要定义元素的背面是否可见，其语法如下所示。

```
backface-visibility: visible|hidden;
```

  - 该属性有以下两个属性值。
    - visible：默认值，反面可见。
    - hidden：反面不可见。

现场演示：

- 案例13-08：backface-visibility属性效果

2.使用变形属性

- 2D位移
  - 在元素变形中需要使用到多种变形函数以实现元素的位移、缩放、旋转等操作。
  - translate()函数为位移函数，可在不影响到X、Y轴上其他元素的情况下将元素从原来的位置移动到另外一个位置，其语法如下所示。

```
translate(x,y)
```

2.使用变形属性

- 2D位移
  - translate()函数可以取一个值，也可取两个值，取值说明如下：
    - x：代表X轴（横坐标）移动的向量长度，当其值为正值时，元素向X轴右方向移动，反之其值为负值时，元素向X轴左方向移动。
    - y：代表Y轴（纵坐标）移动的向量长度，当其值为正值时，元素向Y轴下方向移动，反之其值为负值时，元素向Y轴上方向移动。
  - 如果要将元素仅仅沿着一个方向移动，可使用translate(x,0)和translate(0,y)来实现。
  - 变形函数中提供了一些单个方向移动的简单函数，方便开发者使用。
    - ranslateX(n)：在水平方向上移动一个元素。
    - translateY(n)：在竖直方向上移动一个元素。

2.使用变形属性

- 2D缩放
  - 缩放函数scale()可以使元素根据其中心原点进行缩放，默认值为1，即不缩放。
    - 当scale()的值为0.01到0.99之间，可以让一个元素缩小；
    - scale()的值为任何大于或等于1.01的值时，可以让一个元素放大，其语法如下所示。

```
scale(x,y)
```

  - scale()函数的语法和translate()函数非常相似，可以设置一个值，也可以设置两个值，只有一个值时，第二个值默认和第一个值相等。

2.使用变形属性

- 2D缩放
  - scale()函数的取值具体说明如下所示。
    - x：指定横向坐标方向的缩放向量，如果值为0.01~0.99之间，会让元素在X轴方向缩小，如果值大于或者等于1.01，元素在X轴方向放大。
    - y：指定纵向坐标方向的缩放向量，如果值为0.01~0.99之间，会让元素在Y轴方向缩小，如果值大于或者等于1.01，元素在Y轴方向放大。
  - 如果要使元素仅仅沿着X轴或者Y轴方向缩放，而不是同时缩放的话，可以将函数设置为scale(x,1)或scale(1,y)。
    - sacleX(n)：相当于scale(x,1)。元素只在X轴缩放元素，默认值为1。
    - sacleY(n)：相当于scale(1,y)。元素只在Y轴缩放元素，默认值为1。

2.使用变形属性

- 2D旋转
  - 旋转方法roate()可以通过设定的角度参数对元素实现基于中心原点的2D旋转。
  - 在二维空间内进行操作，设置角度值，用来指定元素旋转的幅度。
    - 如果这个值为正值，元素相对中心原点顺时针旋转。
    - 如果这个值为负值，则元素相对中心原点逆时针旋转。
  - roate()的具体语法如下所示。

```
rotate(angle)
```

2.使用变形属性

- 2D倾斜
  - 倾斜函数skew()能够让元素倾斜显示，可以将一个元素以其中心原点位置围绕着X轴和Y轴按照一定的角度倾斜。
  - 与rotate()方法的旋转不同是rotate()方法只是旋转元素，不会改变元素的形状，其语法如下所示。

```
skew(x-angle,y-angle)
```

    - x-angle：指定元素水平方向（X轴方向）倾斜的角度。
    - y-angle：指定元素竖直方向（Y轴方向）倾斜的角度。

2.使用变形属性

- 2D倾斜
  - 在X轴和Y轴倾斜之外还可使用skewX()和skewY()方法让元素只在水平或者竖直方向倾斜。
    - skewX(angle)：相当于skew(x-angle,0)。按指定的角度沿X轴指定一个倾斜变形。skewX()使元素以其中心点为基点，并在水平方向（X轴）进行倾斜变形。
    - skewY(angle)：相当于skew(0, y-angle)。按指定的角度沿Y轴指定一个倾斜变形。skewY()使元素以其中心点为基点，并在垂直方向（Y轴）进行倾斜变形。

2.使用变形属性

- 2D矩阵
  - CSS3中transform让操作变形时变得很简单，如位移函数translate()、缩放函数scale()、旋转函数roate()和倾斜函数skew()，这些函数很简单，也很方便使用，是CSS动画中常用的方法，但CSS3中同时提供了一个矩阵函数matrix()。
  - 在CSS中的所有变形方法都可通过matrix()方法来实现，matrix()函数是所有变形函数的基础，使用它可以开发出更多的变形样式，而不仅仅局限于位移、缩放、旋转、倾斜等，具体操作如下。

```
matrix(n,n,n,n,n,n)
```

现场演示：

- 案例13-09：用2D变形方法制作立方体

2.使用变形属性

- 3D位移
  - CSS中的3D位移主要由四个函数实现：translate3d()、translateX()、translateY()和translateZ()。
  - translate3d()函数可以使一个元素在三维空间上移动，变形的原理是通过设置三维向量的坐标，定义元素在每个方向上移动了多少，从而实现元素在3D空间中的位移，具体语法如下所示。

```
translate3d(x,y,z)
```

    - x：表示横向坐标位移向量的长度。
    - y：表示纵向坐标位移向量的长度。
    - z：表示Z轴位移向量的长度。

2.使用变形属性

- 3D位移
  - translateX()、translateY()和translateZ()函数则是translate3d()函数的简化形式，分别负责元素在3D空间中沿X、Y、Z轴方向位移的功能，其具体语法如下所示。

```
translateX(x)
translateY(y)
translateZ(z)
```

  - 在translateZ(z)中，取值z指的是Z轴的向量位移长度。使用translateZ()方法可以让元素在Z轴进行位移。
    - 当其值为负值时，元素在Z轴上越移越远，导致元素在越来越小。
    - 当其值为正值时，元素在Z轴上越移越近，导致元素在越来越大。

2.使用变形属性

- 3D缩放
  - CSS中的3D缩放函数主要有scale3d()、scaleX()、scaleY()和scaleZ()四个，可以控制元素在三维空间上的缩放。
  - scale3d()可以控制一个元素在X、Y、Z轴方向的缩放向量，对元素进行3D的缩放变形，其具体语法如下所示。

```
scale3d(x,y,z)
```

  - scaleX()、scaleY()和scaleZ()是scale3d()函数的简化形式，其语法如下所示。

```
translateX(x)
translateY(y)
translateZ(z)
```

2.使用变形属性

- 3D旋转
  - 在CSS中有着四个旋转函数用于实现元素的3D旋转，分别是：rotate3d()、rotateX()、rotateY()和rotateZ()。
  - rotate3d()函数可以通过设置元素在X、Y、Z轴的方向向量和一个旋转角度来控制元素在3D空间中的旋转，其语法如下所示。

```
translate3d(x,y,z)
```

    - x：0~1的数值，用来描述元素围绕X轴旋转的向量值。
    - y：0~1的数值，用来描述元素围绕Y轴旋转的向量值。
    - z：0~1的数值，用来描述元素围绕Z轴旋转的向量值。
    - angle：角度值，用来指定元素在3D空间旋转的角度，如果其值为正值，元素顺时针旋转，反之元素逆时针旋转。

2.使用变形属性

- 3D旋转
  - 同样rotateX()、rotateY()和rotateZ()是rotate3d()的简化形式，用于设置元素在仅在X、Y、Z方向上的旋转角度，其语法如下所示。

```
rotateX(angle)
rotateY(angle)
rotateZ(angle)
```

  - angle是一个角度值，其值可以为正值也可以为负值。
    - 如果值为正值，元素顺时针旋转。
    - 如果值为负值，元素逆时针旋转。
  - rotateX()、rotateY()和rotateZ()的每个方法的具体功能：
    - rotateX(angle)方法的功能等同于rotate3d(1,0,0,angle)。
    - rotateY(angle)方法的功能等同于rotate3d(0,1,0,angle)。
    - rotateZ(angle)方法的功能等同于rotate3d(0,0,1,angle)。

2.使用变形属性

- 3D矩阵
  - CSS中的3D矩阵比2D矩阵复杂，从二维到三维是从4到9，而在矩阵里是3*3变成4*4，即16。
  - 对于3D矩阵而言，本质上很多东西与2D矩阵是一致的，只是复杂程度不一样而已，其语法如下所示。

```
matrix3d(n,n,n,n,n,n,n,n,n,n,n,n,n,n,n,n)
```

现场演示：

- 案例13-10：使用3D变形方法制作立方体

2.使用变形属性

- 时钟
  - 时钟是日常生活中一种很常见的工具，使用CSS3中的变形元素可以很好的实现这种功能。
  - 下面使用CSS3实现动态的时钟效果，全部动画的实现不依赖Flash和JavaScript。

现场演示：

- 案例13-11：制作时钟

3.使用过渡属性

- CSS 过渡（transition）是通过设定元素起点的状态和结束点的状态，在一定的时间区间内实现元素平滑地过渡或变化，是一种补间动画的机制。
  - 通过过渡属性可以让元素属性的改变过程持续一段时间，而不立即生效，从而形成的一种动画效果。
  - 通过transition可以设置哪个属性发生动画效果、何时开始动画 、动画持续多久以及如何动画，其语法如下所示。

```
transition: property duration timing-function delay;
```

  - 该属性是一个综合属性，可设置多个属性值，实现元素的动画过渡。

现场演示：

- 案例13-12：transition属性效果展示

3.使用过渡属性

- transition属性的工作机制是为元素设置两套样式用于用户与页面的交互，在过渡属性未触发时是一种样式，触发后又是一种样式。
- 在这个过程中需要指定元素触发后需要改变的属性，这个值主要通过其子属性transition-porperty来指定，其语法如下所示。

```
transition-property: none|all|property;
```

  - transition-porperty属性的取值说明如下所示。
    - none：没有指定任何样式。
    - all：默认值，表示指定元素所有支持该属性的样式。
    - property：指定样式名称。

3.使用过渡属性

表 13-01 支持过渡的属性

属性	说明
颜色属性	通过红、绿、蓝和透明度组合过渡（每个数值处理），如 background-color、border-color、color 等 CSS 样式属性
具有长度值（length）、百分比（%）的属性	真实的数字，如 word-spacing、width、top、right、bottom、left、line-height 等
离散步骤（整个数字）	在真实的数字空间、以及使用 floor()转换为整数时发生，如 outline-offect、z-index 等
number	真实的（浮点型）数值，如 zoom、opacity、font-weight 等
变形系列属性	如 rotate()、rotate3d()、scale()、scale3d()、skew()、translate()、translate3d()等
rectangle	通过 x、y、width 和 height（转换为数值）变换，如 crop 属性
visibility	离散步骤，在 0~1 范围内，0 表示隐藏，1 表示完全显示、如 visibility 属性
阴影	作用于 color、x、y 和 blur 属性，如 text-shadow 属性
渐变	通过每次停止时的位置和颜色进行变化

3.使用过渡属性

- transision-duration属性主要用来设置元素从一个属性过渡到另一个属性所需的时间，即从旧属性过渡的新属性所消耗的时间，其语法如下所示。

```
transition-duration: time;
```

  - time是数值，单位为秒（s）或者毫秒（ms），可以作用于所有的元素，包括:before和:after伪元素。
- transition-duration属性的默认值为0，也就是说元素的属性变换是即时的。

现场演示：

- 案例13-13：transision-duration与transition-porperty属性效果展示

3.使用过渡属性

- 过渡函数有两种，分别是transition-timing-function()和step()。
  - transition-timing-function属性指定元素过渡过程中的“缓动函数”。此属性可指定元素的过渡速度以及过渡期间的操作进展情况，可以将某个值指定为预定义函数、阶梯函数或者三次贝塞尔曲线，其语法如下所示。

```
transition-timing-function: functionname
```

  - 其值的类型有两种：单一类型的过渡函数和三次贝塞尔曲线。

3.使用过渡属性

- 过渡函数有两种，分别是transition-timing-function()和step()。
  - 单一的过渡函数取值效果如下所示。
    - linear：元素样式从初始状态过渡到终止状态是恒速。
    - ease：默认值，元素样式从初始状态过渡到终止状态时速度由快到慢，逐渐变慢。
    - ease-in：元素样式从初始状态到终止状态时速度越来越快，呈现一种加速状态。这种效果称之为渐显效果。
    - ease-out：元素样式从初始状态到终止状态时，速度越来越慢，呈现一种减速状态。这种效果称为渐隐效果。
    - ease-in-out：元素样式从初始状态到终止状态时，先加速再减速。这种效果称为渐显渐隐效果。

3.使用过渡属性

- 过渡函数有两种，分别是transition-timing-function()和step()。
  - 三次贝塞尔曲线有着多个精确控制点，可以精确的控制函数的过渡过程，其语法如下所示。

```
cubic-bezier(P0,P1,P2,P3)
```

    - 需要注意的是，三次贝塞尔曲线中每个点值只允许0~1的值。
  - step()函数是过渡中的另外一个函数，可以将整个操作领域划成同样的大小间隔，每个间隔都是相等的，该函数还指定发生在开始或者结束的时间间隔是否另外输出，可采用百分比的形式（如输出百分比为0%表示输入变化的初始点）。step()方法非常独特，允许在固定的间隔播放动画，可以用来制作逐帧动画，其语法如下所示。

```
step(n,strat|end)
```

3.使用过渡属性

- 过渡函数有两种，分别是transition-timing-function()和step()。
  - step()方法主要包括两个参数。
    - 第一个参数是一个数值n，主要用来指定step()方法间隔的数量，此值必须是一个大于0的正整数。
    - 第二个参数是可选的，是start或end，如果第二个参数忽略，则默认为end值。

现场演示：

- 案例13-14：transition-timing-function属性简单函数效果展示
- 案例13-15：使用cubic-bezier实现过渡效果

3.使用过渡属性

- transition-delay用来指定一个动画开始执行的时间，也就是说当前元素属性值多长时间后开始执行过渡效果，其语法如下所示。

```
transition-delay: time;
```

  - transition-delay取值为time，它可以是正整数、负整数和0，非零的时候必须将单位设置为秒（s）或者毫秒（ms）。
    - 正整数：元素的过渡动作不会被立即触发，当过了设定的时间值之后才触发。
    - 负整数：元素的过渡动作会从该时间点开始显示，之前动作被截断。
    - 0：默认值，元素的过渡动作会立即触发，没有任何反应。

现场演示：

- 案例13-16：transition-delay属性效果展示

3.使用过渡属性

- 伪元素触发
  - 动画的触发可以使用伪元素触发，如鼠标指向时触发（:hover）、用户点击某个元素时触发（:active）、元素获得焦点状态时触发（:focus）以及元素被选中时触发（:checked）。
- 媒体查询触发
  - 媒体查询触发即通过@media属性触发，能够根据某些元素的更改应用不同的元素样式，同时也可以用来触发动画。
- JavaScript触发
  - 如果可以基于CSS的状态更改触发CSS动画，自然的可以使用JavaScript做到这一点。
  - 最简单的的方法是通过切换元素的类名称进行CSS动画的触发。

现场演示：

- 案例13-17：通过不同的方法触发过渡动画

3.使用过渡属性

- 简介
  - 导航是网站中的重要组成部分，通过导航用户可以了解网站的内容层次结构，并访问网站的各个部分，并且美观恰当的导航可以使页面更快的抓住用户的注意力，提高用户粘度。

现场演示：

- 案例13-18：制作动态导航

4.使用动画属性

- 在CSS3中，除了可以使用transition功能实现的动画效果外，还可以通过animation功能实现更为复杂的动画效果。
- animation功能与transition功能基本相同，都是通过改变元素属性值来实现动画效果的，其区别在于：
  - 使用transition功能时只能通过指定属性的开始值与结束值，然后在这两个属性值之间进行平滑过渡的方式实现动画效果，因此不能实现较为复杂的动画效果。
  - 使用animation功能时，通过定义多个关键帧以及定义每个关键帧中元素的属性值来实现更为复杂的动画效果，其语法如下所示：

```
animation: name duration timing-function delay iteration-count direction;
```

现场演示：

- 案例13-19：animation属性效果展示

4.使用动画属性

- animation动画创建的原理是，通过设置关键帧的方式将一套CSS样式逐渐变化为另一套样式。
  - 在动画过程中，要能够多次改变CSS样式，既可以使用百分比来规定改变发生的时间。
  - 同时也可以通过关键词“from”和“to”来实现，“from”和“to”等价于0%和100%。0%是动画的开始时间，100%是动画的结束时间。
  - animation定义关键帧的语法如下所示。

```
@keyframes animationname {keyframes-selector {css-styles;}}
```

4.使用动画属性

- animation动画创建的原理
  - @keyframes有自身的语法规则：
    - 其命名由@keyframes开头，后面紧跟着是“动画名称”加上一对花括号“{...}”，括号中是不同时间段的样式规则，类似于CSS的写法。
  - 一个@keyframes中的样式规则是由多个不同的百分比构成的，如0%~100%，可以在这个规则中创建更多的百分比，分别给每个百分比中需要动画效果的元素加上不同的属性，从而让元素达到一种不断变化的效果。
  - 如果使用百分比设置关键帧，则其中的%不能省略，如果没有加上，将没有任何效果，因为@keyframes的单位只接受百分比值。

4.使用动画属性

- animation动画创建的原理
  - 使用from、to的方式设置关键帧的代码如下所示。

```
@keyframes mymove
{
from {top:0px;}
to {top:200px;}
}
@-moz-keyframes mymove /* Firefox */
{
from {top:0px;}
to {top:200px;}
}
```

4.使用动画属性

- animation动画创建的原理
  - 使用百分比的方式设置关键帧的代码如下所示。

```
@keyframes mymove
{
0%    {top:0px;}
25%   {top:200px;}
50%   {top:100px;}
75%   {top:200px;}
100% {top:0px;}
}
@-moz-keyframes mymove /* Firefox */
{
0%    {top:0px;}
25%   {top:200px;}
50%   {top:100px;}
75%   {top:200px;}
100% {top:0px;}
}
```

4.使用动画属性

- animation-name属性
  - 该属性主要用来调动动画，其调用的动画是通过@keyframes关键帧定义好的动画，其语法如下所示。

```
animation-name: keyframename|none;
```

  - animation-name用来定义一个动画的名称，其主要有两个值。
    - keyframename：是由@keyframes创建的动画名称，也就是说此处的keyframename需要和@keyframes中的animationname一致，如果不一致将不会实现任何动画效果。
    - none：默认值，当值为none时，将没有任何动画效果，其可以用于覆盖任何动画。

4.使用动画属性

- aimation-duration属性
  - 该属性主要用来设置CSS3动画播放时间，其用法和transition-duration类似，其语法如下所示。

```
animation-duration: time;
```

- animation-timing-function属性
  - 该属性用来设置动画播放的方式，其语法如下所示。

```
animation-timing-function: value;
```

4.使用动画属性

- animation-delay属性
  - 该属性用来定义动画开始播放的时间，其语法如下所示。

```
animation-delay: time;
```

- animation-iteration-count属性
  - 该属性用来定义动画播放的次数，其语法如下所示。

```
animation-iteration-count: n|infinite;
```

  - 此属性主要用于定义动画播放多少次，其值通常为整数，但也可以使用带小数的数字。
  - 此属性默认值为1，这意味着动画从开始到结束只播放一次。如果取值为infinite，动画将会无限次的播放。

4.使用动画属性

- animation-direction属性
  - 该属性主要用来设置动画播放的方向，其语法如下所示。

```
animation-direction: normal|alternate;
```

  - animation-direction是用来指定元素动画的播放方向的，其主要有两个值，分别为：normal和alternate。
    - normal：默认值，设置为normal时，动画的每次循环都是向前播放。
    - alternate：动画播放为偶数次是向前播放，为奇数次则是反向播放。

4.使用动画属性

- animation-play-state属性
  - 该属性用来控制元素动画的播放状态，其语法如下所示。

```
animation-play-state: paused|running;
```

  - 该属性主要有两个值：running和paused。
    - running属性：默认值，主要作用类似于音乐播放器。
    - paused属性：将正在播放的动画停止下来。
    - 通过running将暂停的动画重新播放，这里的重新播放不一定是从元素动画的头部开始播放，也可能是从暂停的那个位置开始播放。如果停止了动画的播放，元素的样式将会回到最初始的设置状态。

4.使用动画属性

- animation-fill-mode属性
  - 定义动画在开始之前和结束之后发生的操作，其语法如下所示。

```
animation-fill-mode : none | forwards | backwards | both;
```

  - 该属性主要有四个值：none、forwards、backwards和both。
    - none：默认值，表示动画将按预期进行和结束，在动画完成最后一帧时，动画会反转到初始帧处。
    - forwards：动画在结束后继续应用最后关键帧的位置。
    - backwards：向元素应用动画样式时迅速应用动画的初始。
    - both：元素动画同时具有forwards和backwards效果。

4.使用动画属性

- 使用@keyframes声明动画

```
@keyframes mymove
{
0%    {top:0px; left:0px; background:red;}
25%   {top:0px; left:100px; background:blue;}
50%   {top:100px; left:100px; background:yellow;}
75%   {top:100px; left:0px; background:green;}
100% {top:0px; left:0px; background:red;}
}
```

  - 通过@keyframes声明一个名字为“mymove”的动画，它的动画经历了从0%到100%的变化，其中还有着25%、50%、75%三个过程。

4.使用动画属性

- 调用@keyframes声明的动画
  - 调用@keyframes声明的动画，在CSS中有两种方式：animation和transition。
    - animation类似于transition属性，都是随着时间改变元素的属性值。
  - 两个属性主要区别是如下：
    - transition需要一个触发事件。
    - animation在不需要任何触发事件的情况下也可以显式的随着时间变化来改变元素的CSS属性值，从而达到一种动画的效果。

4.使用动画属性

- 页面加载动画是Web前端开发中常用的一个动画，在以前的开发中，Web前端开发者常使用GIF图片制作页面加载动画，现在通过CSS3的animation属性可更加简单方便的实现页面加载动画。

现场演示：

- 案例13-20：实现页面加载动画

5.案例：引人入胜的动态照片墙

现场演示：

- 案例13-21：引人入胜的动态照片墙

Thanks.