**CSS3 clip-path 用法详解**

[CSS3](http://yunkus.com/category/fewd/css3/)  2016-06-09  2017-06-06  朝夕熊  0

**前言**

css3 的出现无疑是我们这些作为前端人的福音。因为它的出现让我们的在前端的世界变得更炫，多美，更多姿！接下来要给大家介绍的这个属性clip-path 就是一个可以让我们的页面更多姿多彩的 css3 属性。

**clip-path 作用**

正如前言中所说到的，可以让我们的页面变得多姿多彩，何为多姿，何为多彩呢？

**多姿：**可以让一个元素显示出不同的形式（如：正方形，圆形，长方形，椭员形，多边形……），这个你应该没想到吧？但CSS3 clip-path属性确确实实为我们做到了。

**多彩：**多彩其实就是在多姿的基本上，稍微加点东西就是了。例如把图片作为裁剪的对象，那么你就可以得到一个多姿多彩的效果的，没有你做不到，只有你想不到。

上面只是我个人对这个clip-path属性的看法，但我们还是有必要来看一下对clip-path 这个属性的作用的一个比较严肃的解释：

The clip-path CSS property prevents a portion of an element from getting displayed by defining a clipping region to be displayed i.e, only a specific region of the element is displayed. The clipping region is a path specified as a URL referencing an inline or external SVG, or shape method such as circle(). The clip-path property replaces the now deprecated clip property.  
来源：MDN

大概的意思就是：clip-path 属性可以通过定义裁剪区域来决定目标元素哪个区域可见，哪个区域不可见，也就是只有在闭合路径内的部分才显示，区域以外的部分就不显示。可以通过url 引入内嵌的或者外置的SVG，又或者是直接用自带的形状为clip-path 属性指定路径。 Clip-path 属性是clip属性的替代者，也可以说是click 属性的升级版。

**基本语法**

1. /\* Keyword values \*/
2. clip-path: none;
3. clip-path: fill-box;
4. clip-path: stroke-box;
5. clip-path: view-box;
6. /\* Image values \*/
7. clip-path: url(resources.svg#c1);
8. clip-path: linear-gradient(blue, transparent);
9. /\* Geometry values \*/
10. clip-path: inset(100px 50px);
11. clip-path: circle(50px at 0 100px);
12. clip-path: polygon(50% 0%, 100% 50%, 50% 100%, 0% 50%);
13. /\* Global values \*/
14. clip-path: inherit;
15. clip-path: initial;
16. clip-path: unset;

**浏览器支持**

浏览器支持情况

注意，在现代浏览器中需要使用 -webkit- 前缀。

**实现原理**

Clip-path 实现裁剪的原理也非常简单，clip-path 属性为我们提供了一系列的坐标来创建路径也就是我们常常说的X，Y。当创建完一个闭合的路径时，位于路径内部的区域就可见，而路径外的区域就不可见。这样就实现了裁剪效果。我们可以用这个属性来创建各种图形，如圆形，多边形，三角形，椭圆形等等，没有你做不到，就怕你没想到。

**例子**

**三角与多边形**

**1.简单的小三角**

1. <!DOCTYPE html>
2. <html lang="en">
3. <head>
4. <meta charset="UTF-8">
5. <title>clip-path 例子-小三角</title>
6. <style>
7. .path {
8. background: #33363b;
9. width: 20px;height: 20px;
10. -webkit-clip-path: polygon(0px 10px, 20px 0px, 20px 20px);
11. /\*-webkit-clip-path: polygon(0 50%, 100% 0, 100% 100%);\*/
12. }
13. </style>
14. </head>
15. <body>
16. <div class="path"></div>
17. </body>
18. </html>

上面的代码我们只需要关注 -webkit-clip-path: polygon(0px 10px, 20px 0px, 20px 20px); 就可以了，下面我们就来看看这个属性值的用法以及里面的几个参数都是什么意思吧！

前面的-webkit-clip-path: 应该就不用说了，polygon(); 方法就是传说中的创建多边形的函数。而里面的参数（以上面的代码为例）就是3个点的坐标，0px 10px、 20px 0px、20px 20px ，在polygon 方法中坐标之间用逗号隔开。也就是这三个点就组成了一个闭合的三角形。

css 样式中注释的那行代码 -webkit-clip-path: polygon(0 50%, 100% 0, 100% 100%); 效果其实是跟第一行代码 -webkit-clip-path: polygon(0px 10px, 20px 0px, 20px 20px); 是一样的，只是写法不一样，一个用px，一相用百分比。在这里也给大家分析下百分比的用法。

不管是以像素作为单位，还是百分比作为单位，都是代码坐坐标点。坐标一：(0,50%)，坐标二：(100%,0)，坐标三：(100%,100%)，估计你对这里不感兴趣，你更想知道为什么要这么写，为什么这样写能实现这个小三角。并且跟上面的以像素作为单位的效果一样。

首先这里的百分比是以父元素为参考的，于是乎，就像本例子中的父元素的宽、高都为20px，所以坐标一：(0,50%) = （0,10），坐标二：(100%,0) = （20,0），坐标三：(100%,100%) = （20,20） 。这下子你应该明白了百分比与像素之间的关系了吧！

**2.三角裁剪图片**

你甚至可以用这个三角形来裁剪一张图片，这样的用法应该会更有意思，更实在，也有机会用到我们的项目当中去，不过这可能还需要点时间，不过没关系，我们还是得对未来有所期待，请看代码（下面的所有例子都基于这个HTML结构）

1. <!DOCTYPE html>
2. <html lang="en">
3. <head>
4. <meta charset="UTF-8">
5. <title>clip-path 例子-三角裁剪图片</title>
6. <style>
7. .path {
8. /\*-webkit-clip-path: polygon(0px 10px, 20px 0px, 20px 20px);\*/
9. -webkit-clip-path: polygon(50% 0, 0 100% ,100% 100%);
10. }
11. </style>
12. </head>
13. <body>
14. <img class="path" src="http://yunkus.com/wp-content/themes/zxx/images/T900-500.png" alt="">
15. </body>
16. </html>



**3.对话窗裁剪图片**

1. .path {
2. -webkit-clip-path: polygon(0% 0%,30% 0%,30% 40%,23% 40%,25% 50%,15% 40%,0% 40%);
3. }



**4.星形裁剪图片**

1. .path {
2. -webkit-clip-path: polygon(50% 0%, 65% 33%, 100% 33%, 80% 60%, 90% 100%, 50% 75%, 10% 100%,20% 60%,0 33%,35% 33%);
3. }



**圆形**

我们只需要对上面的 css 样式和 HTML 代码稍作修改就可以了。

1. .path {
2. -webkit-clip-path: circle(30% at 50% 50%);
3. /\*-webkit-clip-path: circle(80px at 150px 83px);\*/
4. }



circle() 中三个值分别由半径，一个圆心坐标点(x,y)组成，定义了圆的半径后，我们可以用 at 关键字来定义圆心坐标就像上面那样。同样的，不仅可以用百分比，也可以用像素来表示。

**椭圆**

椭圆的实现代码跟圆的差不多，并没有什么改动，只是换了个方法而已。下面就是改动了的css样式代码

1. .path {
2. -webkit-clip-path: ellipse(20% 20% at 50% 45%);
3. }



椭圆样式里把width删掉了，其实删不删都没关系，只是用来控制下图片大小以达到所需的显示大小而已。

而更有意思的是如果我们把里面CSS样式的一个值稍作修改，那么另一个图形就出现在我们的面前了。

1. /\*修改后的代码\*/
2. -webkit-clip-path: ellipse(100% 20% at 50% 45%);



**叶子**

1. .path {
2. -webkit-clip-path: inset(35% 35% 35% 35% round 0 70% 0 70%);
3. }

  
上面的这行代码 -webkit-clip-path: inset(25% 0 25% 0 round 0 25% 0 25%); 是什么意思呢？

我们来看看 inset();它的基本语法如下：

1. inset(<top> <right> <bottom> <left> round <top-radius> <right-radius> <bottom-radius> <left-radius>)

如果看完语法还是不怎么明白，也不用担心，我慢慢给你分析，让你看完这个方法后绝对没有任何负担。我举个例子，你看到inset里面的top、right、bottom、left，你首先会想到什么？是不是跟我们常常用到的绝对定位类似呀，都有四个值。没错，其实inset前面四个参数就类似于绝对定位中用到的top、right、bottom、left。也就是裁剪的路径距离对象元素四个边的距离。inset后面四个参数就相当于 border-radius。这下子你应该知道了这个inset()要怎么用了吧！

上面的那个例子中的代码还可以简写成：

1. -webkit-clip-path: inset(35% round 0 70%);

**动画应用**

除了上面介绍到的裁剪效果外，我们还可以通过 clip-path 来实现变形动画，值得兴奋的是，这种动画类似于变形金刚那样，很有动感，看起来非常的动人。

**1.简单的变形动画**

1. .path {
2. -webkit-clip-path: polygon(50% 0%, 65% 0%, 100% 0%, 100% 60%, 100% 100%, 50% 100%, 0% 100%,0% 50%,0 33%,0% 0%);
3. transition: all 0.5s;
4. }
5. .path:hover{
6. -webkit-clip-path: polygon(50% 0%, 65% 33%, 100% 33%, 80% 60%, 90% 100%, 50% 75%, 10% 100%,20% 60%,0 33%,35% 33%);
7. }

为了简单明了，上面的代码就不给transition 加前缀了，如果要用到实际项目中的话，可以按需添加。

**2.炫丽转身**

你要做的只需要在上面的 :hover 样式中加上 transform:rotate(360deg); 完整的 CSS 样式代码如下：

1. .path {
2. -webkit-clip-path: polygon(50% 0%, 65% 0%, 100% 0%, 100% 60%, 100% 100%, 50% 100%, 0% 100%,0% 50%,0 33%,0% 0%);
3. transition: all 0.5s;
4. }
5. .path:hover{
6. transform:rotate(360deg);
7. -webkit-clip-path: polygon(50% 0%, 65% 33%, 100% 33%, 80% 60%, 90% 100%, 50% 75%, 10% 100%,20% 60%,0 33%,35% 33%);
8. }

注意：要想实现这样一种效果有一个前提，就是变化前和变化后组成裁剪路径的点的数量不变。

css3 给了我们一种可能，那就是没有你做不到的，只有你想不到的，你可以使用上面的这些属性加上自己的一些独到想法，那么一个独一无二，炫丽多姿的动画就出现了。只是到目前止，我们还稍微欠了一股东风：浏览器支持。虽然是这样，但我们还是可以在移动端小试牛刀的，至于要不要一炫到底，那就得看看你的项目是否允许了，祝你好远！

**相关资料**

火狐MDN：<https://developer.mozilla.org/en-US/docs/Web/CSS/clip-path/>

w3c：<https://www.w3.org/TR/css-masking-1/#the-clip-path/>