10个demo示例学会CSS3 radial-gradient径向渐变

这篇文章发布于 2017年11月23日，星期四，00:23，归类于 CSS相关。 阅读 23383 次, 今日 24 次 15 条评论

一、语法细节记不住怎么办?

实际开发的时候，当要使用 `radial-gradient` 径向渐变的时候，脑中会有大概的语法，但是细节却记不住，于是想快捷找个案例看看具体怎么用，然后直接套一下。通常一番搜索，会发现虽然是个简单需求，但是正好满足这个需求的页面却不好找，乱糟糟的。

下次遇到这种场景，直接来本站搜“径向渐变”，或者直接搜索“渐变”，就有专门展示 `radial-gradient` 径向渐变基本语法使用案例的文章。文章共展示了10例常见使用案例，相信一定可以覆盖你的使用场景的。

首先，假设我们使用同一段HTML作为示意：

```
<div class="radial-gradient"></div>
```

然后…

二、示例1：最基础最简单使用

CSS如下：

```
.radial-gradient {
    width: 400px; height: 200px;
    background: radial-gradient(yellow, red);
}
```

最终效果如下图：

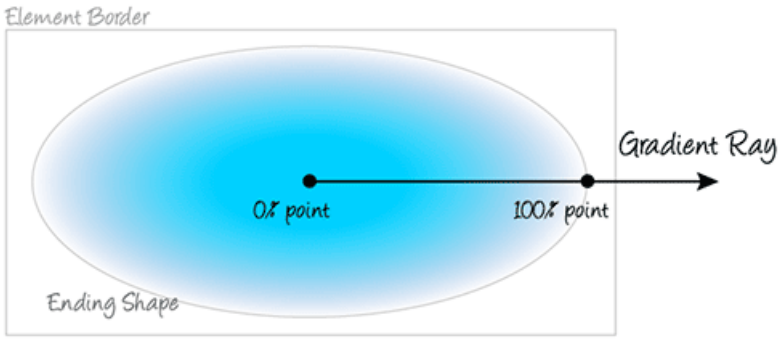
若要亲自确认效果，您可以狠狠地点击这里：radial-gradient径向渐变最基本语法效果demo

可见，对于径向渐变，在不指定渐变类型以及位置的情况下，其渐变距离和位置是由容器的尺寸决定的。

例如本例之中，容器的宽高比是2:1，最终渐变呈现出来的形状也是一个2:1的椭圆形，并且渐变颜色自动

终止与容器的边缘。

原理示意如下：



如果我们想指定渐变的形状是一个圆形，形状不受外部容器尺寸影响，可以使用`circle`关键字。

三、示例2：简单的圆形渐变

CSS代码如下：

```
.radial-gradient {
    width: 400px; height: 200px;
    background: radial-gradient(circle, yellow, red);
}
```

最终效果如下图：

若要亲自确认效果，您可以狠狠地点击这里：radial-gradient径向渐变简单圆形渐变效果demo

其渐变范围（渐变结束线）示意如下图，会发现既不是按照宽度来的，也不是按照高度来的，是按照最远边角距离作为渐变结束线的：

要证明上面结论比较简单，我们可以计算下对角线一半长度，为：Math.sqrt(200*200 + 400*400) ≈ 223.6，于是，我们设置：

```
.radial-gradient {
    width: 400px; height: 200px;
    background: radial-gradient(223.6px circle, yellow, red);
}
```

会发现，最终效果和上面的效果截图几乎就是一模一样的。而`200px`的半径：

```
.radial-gradient {
```

```
    width: 400px; height: 200px;
    background: radial-gradient(200px circle, yellow, red);
}
```

取色就会发现边缘颜色差异较大，说明默认不是按照最短边来渲染的。

四、示例3：指定渐变起始点位置

起始点位置可以通过`at`来指定，例如：

```
.radial-gradient {
    width: 400px; height: 200px;
    background: radial-gradient(circle at 50px 50px, yellow, red);
}
```

效果如下图所示：

若要亲自确认效果，您可以狠狠地点击这里：径向渐变at确定渐变起始点demo

如果希望渐变是和容器保持一致比例的椭圆，则`circle`可以缺省，也就是直接`radial-gradient(at 50px 50px, yellow, red)`也是可以的。

`50px 50px`我们也可以使用百分比值代替，例如：

```
.radial-gradient {
    width: 400px; height: 200px;
    background: radial-gradient(circle at 12.5% 25%, yellow, red);
}
```

效果是一模一样的。

五、示例4：指定渐变终止点位置

`radial-gradient`径向渐变支持4个关键字可以指定渐变终止点位置，见下表：

关键字	描述
`closest-side`	渐变中心距离容器最近的边作为终止位置。
`closest-corner`	渐变中心距离容器最近的角作为终止位置。
`farthest-side`	渐变中心距离容器最远的边作为终止位置。
`farthest-corner`	渐变中心距离容器最远的角作为终止位置。

根据上面的示意效果可以看出默认值是`farthest-corner`。

现在，我们对CSS进行调整，如下：

```
.radial-gradient {
    width: 400px; height: 200px;
```

```
    background: radial-gradient(closest-side circle at 50px 50px, yellow, red);
}
```

也就是圆形渐变终止于最短边，结果效果如下图：

若要亲自确认效果，您可以狠狠地点击这里：径向渐变终止于最短边demo

六、示例5：指定渐变颜色断点

如果指定多个颜色，但未指定具体断点的位置，则这些颜色会均匀分配0%~100%的渐变区域，例如，下面4色渐变：

```
.radial-gradient {
    width: 200px; height: 200px;
    border: 1px solid silver;
    background: radial-gradient(closest-side circle, yellow, orange, red, white);
}
```

结果如下图所示：

但从肉眼视觉我们是看不出是不是均匀分配渐变区域，但是我们可以通过其他方式验证我们的观点，如下指定颜色断点位置的CSS：

```
.radial-gradient2 {
    width: 200px; height: 200px;
    border: 1px solid gray;
    background: radial-gradient(closest-side circle, yellow, orange 33.33%, red 66.666%, w
hite);
}
```

由于渐变颜色默认第一个颜色位置是0%，最后一个颜色位置是100%，因此上面`yellow`和`white`的断点位置可以缺省。

然后发现效果和上面的是一致的：

我做了个更能直观体验一致性的demo页面，您可以狠狠地点击这里：radial-gradient多个颜色均匀分配渐变区域demo

按下第一个圈圈渐变，会让后面的圈圈瞬间覆盖在上面。如果两个渐变颜色有差异，此时肉眼可以感觉到明显变化；但是实际操作下来，就好像后面渐变突然消失一般，这就说明上下两个渐变实际上效果是一致的。

七、示例6：椭圆类型的径向渐变

对于圆形界面，我们只需要指定一个半径就可以了，但是对于椭圆类型的径向渐变，我们需要同时指定横轴和纵轴的长度，例如：

```
.radial-gradient {
    width: 200px; height: 200px;
```

```
    background: radial-gradient(50px 100px ellipse, transparent 40px, yellow 41px, red);
}
```

`50px 100px ellipse`中第一个数值`50px`表示横轴半径，`100px`表示纵轴半径，于是这段语句表示绘制一个长度`100px`高度`200px`的椭圆。

效果如下图：

若要亲自确认效果，您可以狠狠地点击这里：radial-gradient径向渐变椭圆渐变demo

需要注意的是，在上面示意CSS代码中，透明到黄色分界那里有一个`1px`的偏差过渡，也就是`transparent 40px, yellow 41px`中`yellow`是`41px`，而不是设置的`40px`，原因在于在Chrome下，如果颜色直接`0`偏差过渡，会有比较严重的锯齿，类似下图这样（背景色设为`#333`）：

通过有1像素或者半像素的过渡缓冲可以有效避免锯齿的出现。

八、示例7：可累加的径向渐变背景图

我们可以把多个径向渐变累加在一起实现某些效果，例如下面CSS：

```
.radial-gradient {
    width: 200px; height: 200px;
    background: radial-gradient(50px 100px ellipse, transparent 40px, yellow 41px, red 49p
x, transparent 50px),
                radial-gradient(30px circle, red, red 29px, transparent 30px);
}
```

实现了“邪王真眼”效果：


若要亲自感受效果，您可以狠狠地点击这里：radial-gradient多个径向渐变累加demo

九、示例8：渐变背景的尺寸控制

配合`background-size`的尺寸控制和背景重复特性，我们可以实现渐变式的复杂纹理效果：

```
.radial-gradient {
    width: 200px; height: 200px;
    background: radial-gradient(5px 10px ellipse, transparent 4px, yellow 5px, red 6px, tr
ansparent 7px),
                radial-gradient(3px circle, red, red 3px, transparent 4px);
    background-size: 25px;
}
```

于是就可以看到一大波写轮眼：

通常我们使用径向渐变构建一些简单实用图形的时候，`background-size`往往是关键属性，例如，我们要实现个水波效果，可以这样：

```
.radial-gradient {
    width: 200px; height: 100px;
    background: red;
    position: relative;
}
.radial-gradient:after {
    content: '';
    position: absolute;
    height: 10px;
    left:0 ; right: 0;
    bottom: -10px;
    background: radial-gradient(20px 15px ellipse at top, red 10px, transparent 11px);
    background-size: 20px 10px;
}
```

然后就有下图所示的效果：

实时效果体验可以狠狠地点击这里：径向渐变尺寸控制下的波纹效果demo

十、示例9：可重复的径向渐变

如果想要实现可重复的径向渐变，亦可以使用原生的`repeating-radial-gradient()`方法，特别适合实现从中心扩散的光波效果。

然而，相比重复线性渐变`repeating-linear-gradient()`方法，`repeating-radial-gradient()`的实际应用场景实际上很有限。因为，实际开发的时候，我们很少用到从中心扩散的光波效果。

除了一些本身就是波纹类型效果，如水波，声波或者唱片纹理：

如下CSS（取自https://codepen.io/thebabydino/embed/HjJlL）：

```
.radial-gradient {
    position: relative;
    width: 262px; height: 262px;
    border-radius: 50%;
    background: linear-gradient(30deg, transparent 40%, rgba(42, 41, 40, .85) 40%) no-repe
at 100% 0, linear-gradient(60deg, rgba(42, 41, 40, .85) 60%, transparent 60%) no-repeat 0
100%, repeating-radial-gradient(#2a2928, #2a2928 4px, #ada9a0 5px, #2a2928 6px);
    background-size: 50% 100%, 100% 50%, 100% 100%;
}
.radial-gradient:after {
    position: absolute;
    top: 50%; left: 50%;
    margin: -35px;
    border: solid 1px #d9a388;
    width: 68px; height: 68px;
    border-radius: 50%;
    box-shadow: 0 0 0 4px #da5b33, inset 0 0 0 27px #da5b33;
    background: #b5ac9a;
    content: '';
}
```

效果如下图：

效果体验，可以狠狠地点击这里：可重复径向渐变下的胶片效果

十一、示例10：作为border-image的径向渐变

径向渐变不仅可以作为 `background` 的背景图，还可以作为 `border-image` 的边框图使用。

例如：

```
.radial-gradient {
    width: 100px; height: 100px;
    border: 50px solid;
    border-image: radial-gradient(circle, transparent 50px, yellow 51px, red) 50 stretch;
}
```

效果：

天然镂空效果，只可惜 `border-image` 是无法和 `border-radius` 同时生效的，否则， `border-image` 可就要牛炸天了。

效果体验，可以狠狠地点击这里：radial-gradient径向渐变与border-image呈现demo

十二、结束语

我真能写！

> 《CSS世界》签名版独家发售，包邮，可指定寄语，点击显示购买码

（本篇完） // 想要打赏？点击这里。有话要说？点击这里。

« 3种纯CSS实现中间镂空的12色彩虹渐变圆环方法

照片位图转SVG矢量图片JS工具primitive.js等简介 »

猜你喜欢

- CSS3动画那么强，requestAnimationFrame还有毛线用？
- CSS3 border-image详解、应用及jQuery插件
- 实现兼容性的CSS粗虚线边框(dashed)效果
- 3种纯CSS实现中间镂空的12色彩虹渐变圆环方法
- 小tips: 纯CSS实现蜡烛、火焰以及熄灭后烟雾效果
- PIE使IE支持CSS3圆角盒阴影与渐变渲染
- CSS渐变图片背景下高度亦自适应按钮
- 拾人牙慧 - CSS3实现Opera浏览器的logo
- CSS页面重构“鑫三无准则”之“无图片”准则

- 使用SVG实现gradient背景渐变
- 写给自己看的display: grid布局教程

分享到： 0

标签： border-image, css3, radial-gradient, 径向渐变, 渐变, 渐变背景

发表评论（目前15条评论）

名称 (必须)

邮件地址(不会被公开) (必须)

网站

提交评论

1. **deepblue说道：**
2018年09月28日 11:40

九、示例8：渐变背景的尺寸控制
水波效果
{ background: radial-gradient(20px 15px ellipse at top, red 10px, transparent 11px);}
其中的 20px,15px表示的是椭圆ellipse的 横向半径和纵向半径吧?
第二个参数15px随意改变，椭圆纵向 伸缩，没问题。
但是，第一个参数20px，改变数值，它的横向半径并无变化，反而是纵向会伸缩?

回复

2. **deepblue说道：**
2018年09月28日 09:37

示例7中
radial-gradient(30px circle, red, red 29px, transparent 30px);不明白为什么这么写?

radial-gradient(30px , red 29px, transparent 30px);完全效果一样啊！

只给一个数值，会自动默认= 正圆 吧?
{red, red 29px,}第一个 red,是什么意思呢?

新人，求教

回复

3. **deepblue说道：**
2018年09月28日 09:22

示例8中{background-size: 25px;}需要写成{background-size: 25px 25px;}才能实现你的图效果

回复

4. 盖大楼说道：
2017年12月27日 16:30

攒 大佬666

回复

5. 墨轩mo说道：
2017年12月13日 11:08

挺赞的，给了我不少启发想法，感谢大大~

回复

6. pan说道：
2017年12月6日 20:02

太不直观了,直接用工具, CSS3本来就是抄PS的

回复

> lyww说道：
> 2017年12月14日 10:03
>
> 能用代码搞定的，坚决不用图片
>
> 回复

7. fate说道：
2017年11月28日 18:49

确实牛。。。受益良多

回复

8. 请教说道：
2017年11月28日 12:55

求问一个刚毕业自学前端的大学生该何去何从

回复

9. 幾米说道：
2017年11月25日 09:33

border-image无法和border-radius同时生效
在外面套多一层，还是能实现一些不错的效果

回复

10. luble说道：
2017年11月24日 13:36

总结的真详细，厉害啦！

回复

11. 菱薇说道：
2017年11月24日 10:26

赞赞赞

回复

12. 张琦说道：
2017年11月23日 23:04

大神，第三部分radial-gradient径向渐变示例2，半径计算应该是Math.sqrt(200*200 + 100*100)，不应该是Math.sqrt(200*200 + 400*400)。

回复

> **cray说道：**
> 2018年09月4日 14:39
>
> /2
>
> 回复

13. 刘东奇说道：
2017年11月23日 09:44

真厉害

回复

最新文章

- 常见的CSS图形绘制合集
- 粉丝群第1期CSS小测点评与答疑
- 分享三个纯CSS实现26个英文字母的案例
- 小tips: 纯CSS实现打字动画效果
- CSS/CSS3 box-decoration-break属性简介
- CSS :placeholder-shown伪类实现Material Design占位符交互效果
- 从天猫某活动视频不必要的3次请求说起
- CSS vector-effect与SVG stroke描边缩放
- CSS ::backdrop伪元素是干嘛用的?
- 周知：CSS -webkit-伪元素选择器不再导致整行无效

今日热门

- 常见的CSS图形绘制合集 (190)
- 未来必热：SVG Sprite技术介绍 (119)
- 粉丝群第1期CSS小测点评与答疑 (115)
- HTML5终极备忘大全（图片版+文字版） (93)
- 让所有浏览器支持HTML5 video视频标签 (86)
- Selectivizr-让IE6~8支持CSS3伪类和属性选择器 (82)
- CSS3下的147个颜色名称及对应颜色值 (79)
- 视区相关单位vw, vh..简介以及可实际应用场景 (76)
- 写给自己看的display: flex布局教程 (76)
- 小tips: 纯CSS实现打字动画效果 (76)

今年热议

- 《CSS世界》女主角诚寻靠谱一起奋斗之人 (76)
- 不借助Echarts等图形框架原生JS快速实现折线图效果 (64)

» 看，for..in和for..of在那里吵架！ (60)
» 是时候好好安利下LuLu UI框架了！ (47)
» 原来浏览器原生支持JS Base64编码解码 (35)
» 妙法攻略：渐变虚框及边框滚动动画的纯CSS实现 (33)
» 炫酷H5中序列图片视频化播放的高性能实现 (31)
» CSS scroll-behavior和JS scrollIntoView让页面滚动平滑 (30)
» windows系统下批量删除OS X系统.DS_Store文件 (26)
» 写给自己看的display: flex布局教程 (26)

猜你喜欢

- CSS3动画那么强，requestAnimationFrame还有毛线用？
- CSS3 border-image详解、应用及jQuery插件
- 实现兼容性的CSS粗虚线边框(dashed)效果
- 3种纯CSS实现中间镂空的12色彩虹渐变圆环方法
- 小tips: 纯CSS实现蜡烛、火焰以及熄灭后烟雾效果
- PIE使IE支持CSS3圆角盒阴影与渐变渲染
- CSS渐变图片背景下高度亦自适应按钮
- 拾人牙慧 - CSS3实现Opera浏览器的logo
- CSS页面重构“鑫三无准则”之“无图片”准则
- 使用SVG实现gradient背景渐变
- 写给自己看的display: grid布局教程