奇思妙想CSS之百变背景

身为一个前端开发者，背景是开发中的常客。大到整个网站的主题背景，小到一个按钮的背景。CSS 的 `background` 属性基本上每天开发都会遇到，绝大多数情况下我们都只会使用到了纯色背景或者图片背景。如果你想让你开发的内容看起来更加生动有趣，通过本文让你用纯CSS也可以开发出炫酷的背景。

开始之前

在开始之前，先请你回答下面的问题，如果你能全部回答正确，说明你对 `background` 属性掌握的还不错哦！

1.径向渐变默认形状是什么？

A：原型　B：椭圆形

2. `background` 属性的值为多个时，哪个值的图层在最顶部？

A：第一个值　B：最后一个值

3. `background: green, linear-gradient(red, pink);` 效果是什么？

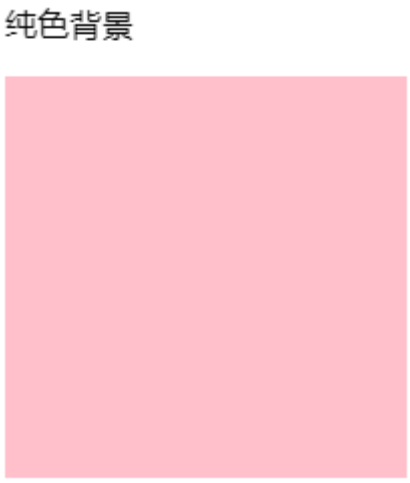
A：绿色背景　B：红粉渐变背景　C：没有背景

4. 当 `background` 属性有多个值时，如何指定每层背景的大小？

基础背景

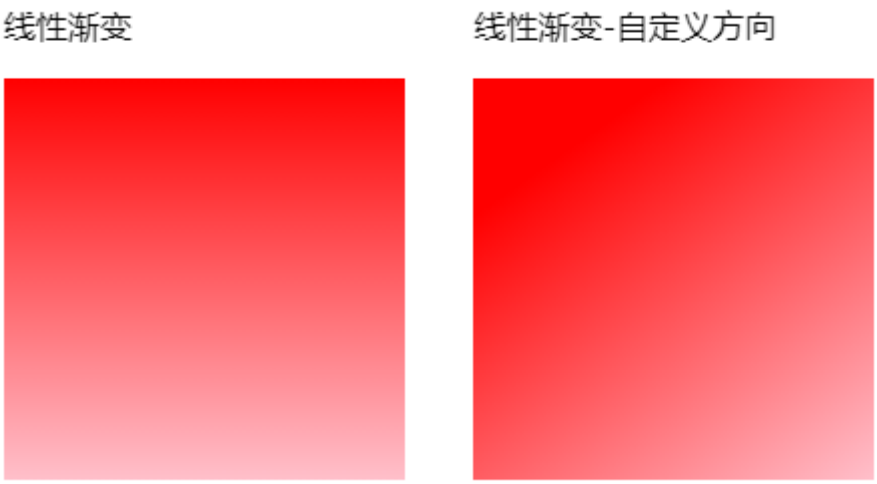
首先还是先回顾一下基础背景有哪些，最简单的就是 **纯色背景**：

```
background: pink;
```



线性渐变，当然你还可以自定义方向：

```
.linear {
    background: linear-gradient(red, pink);
}
.linear1 {
    background: linear-gradient(145deg, red 20%, pink);
}
```



径向渐变

```
background: radial-gradient(red, pink);
```

角向渐变

```
background: conic-gradient(red, pink);
```

角向渐变

基础背景扩展

纯色背景就没什么可说的了，只能改变颜色。

线性背景

对于**线性背景**，我们可以设置多种颜色，还可以指定每种颜色的绘制位置：

```
background: linear-gradient(-50deg, #F7A37B, #F7A37B, #FFDEA8, #FFDEA8, #D0E4B0,
#D0E4B0, #7CC5D0, #7CC5D0, #00A2E1, #00A2E1, #0085C8, #0085C8);
```

多色线性渐变

```
background: linear-gradient(-50deg, #F7A37B, #F7A37B, #FFDEA8, #FFDEA8 20%,
#D0E4B0, #D0E4B0,  #00A2E1 20%, #00A2E1, #0085C8, #0085C8);
```

现在看起来每种颜色之间都是有一定的过渡效果的，通过控制每种颜色绘制的位置，可以消除掉过渡效果：

```
background: linear-gradient(-50deg,#F7A37B,#F7A37B 10%,#FFDEA8 10%,#FFDEA8
20%,#D0E4B0 20%,#D0E4B0 30%,#7CC5D0 30%,#7CC5D0 40%,#00A2E1 40%,#00A2E1
50%,#0085C8 50%,#0085C8 60%);
```

但是蓝色占了一大块，并不美观，一组颜色通过使用重复背景可以达到令人满意的效果。

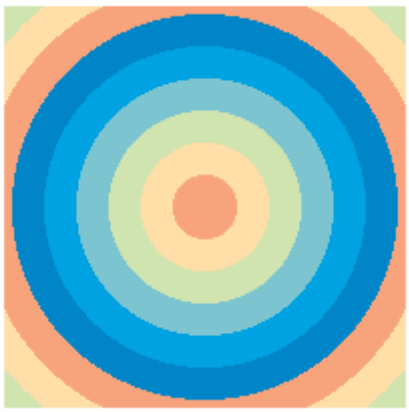
```
background: repeating-linear-gradient(-50deg,#F7A37B,#F7A37B 1em,#FFDEA8
1em,#FFDEA8 2em,#D0E4B0 2em,#D0E4B0 3em,#7CC5D0 3em,#7CC5D0 4em,#00A2E1
4em,#00A2E1 5em,#0085C8 5em,#0085C8 6em);
```

重复线性渐变

径向背景

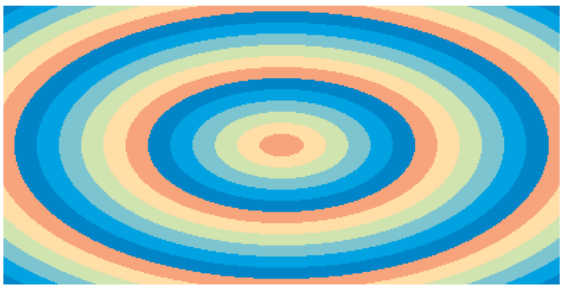
重复背景在径向中同样耐看

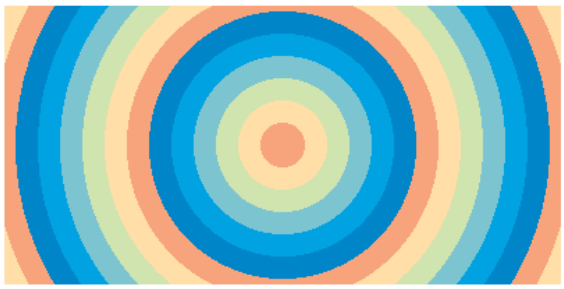
```
background: repeating-radial-gradient(#F7A37B,#F7A37B 1em,#FFDEA8 1em,#FFDEA8
2em,#D0E4B0 2em,#D0E4B0 3em,#7CC5D0 3em,#7CC5D0 4em,#00A2E1 4em,#00A2E1
5em,#0085C8 5em,#0085C8 6em);
```



当宽高不同时，会呈现出椭圆形，如果你想让图案为原型，可以指定 `repeating-radial-gradient` 的第一参数为 `circle` 。

```
background: repeating-radial-gradient(circle,#F7A37B,#F7A37B 1em,#FFDEA8
1em,#FFDEA8 2em,#D0E4B0 2em,#D0E4B0 3em,#7CC5D0 3em,#7CC5D0 4em,#00A2E1
4em,#00A2E1 5em,#0085C8 5em,#0085C8 6em);
```



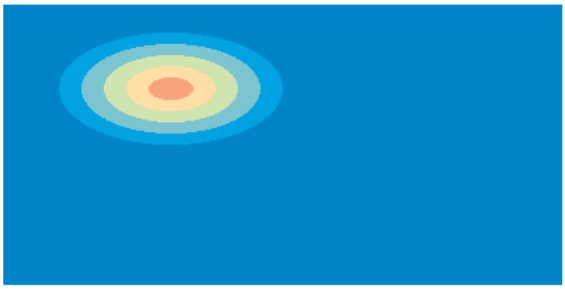
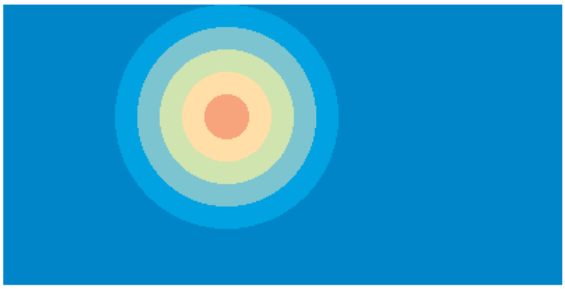


调整第一个参数，你可以得到不同的图案

第一个参数除了可以指定园的类型外，还可以定义渐变大小以及位置：

```
background: repeating-radial-gradient(100% 100% at 30% 30%,#F7A37B,#F7A37B
1em,#FFDEA8 1em,#FFDEA8 2em,#D0E4B0 2em,#D0E4B0 3em,#7CC5D0 3em,#7CC5D0
4em,#00A2E1 4em,#00A2E1 5em,#0085C8 5em,#0085C8 6em);
```

```
background: repeating-radial-gradient(100% 200% at 40% 40%,#F7A37B,#F7A37B
1em,#FFDEA8 1em,#FFDEA8 2em,#D0E4B0 2em,#D0E4B0 3em,#7CC5D0 3em,#7CC5D0
4em,#00A2E1 4em,#00A2E1 5em,#0085C8 5em,#0085C8 6em);
```

定义形式为：`size` at `positon`

- `position`：定义圆心位置，默认值为 `center`，可以设定为 `top`、`bottom`、`left`、`right`，也可以使用坐标表示
- `size`：定义渐变大小，如 `100% 200%`，它表示的意思是横向保持，纵向在原来的基础上扩展一倍。因此上面的第二张图本来是椭圆形，现在呈现为圆形。

关键知识点

`size` 不要误解为是缩放比例，上面的 `100% 200%` 与 `5% 10%` 的效果一样的，并不是相对于原来的大小进行缩放，大小在指定颜色的时候进行设定。

你如果设置为 `500% 500%`，其效果与未设置该值是相同的，其表现应该是：在较大的值的方向上（第一个值大时为横向，第二个值大时为纵向），放大**较大的值/较小的值** 倍。

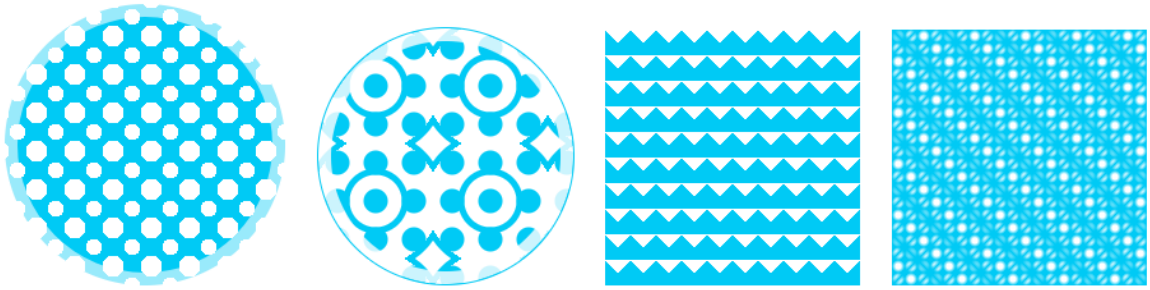
角向渐变

把div圆角设为50%，首尾颜色相同，一个色盘就出来了

```
background: conic-gradient(red, orange, yellow, green, teal, blue, purple, red);
border-radius: 50%;
```

组合背景

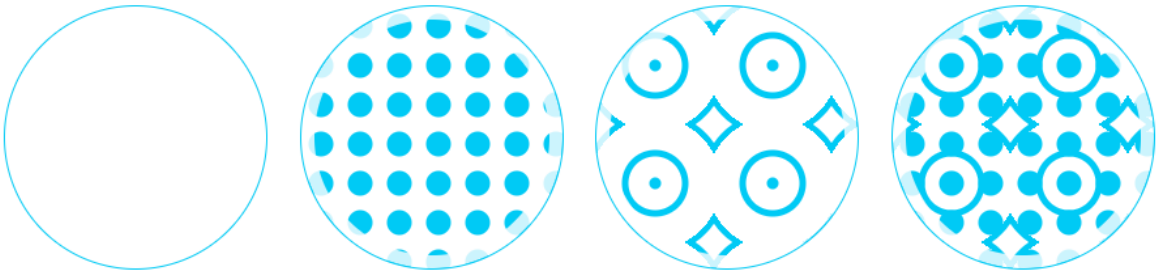
`background` 属性可以同时设置多个背景，多个背景相互叠加可以组合出奇妙图案。



制作这种类型的背景，有两点需要注意：

- 通过 `background-size` 属性设置每个背景的大小；
- 巧妙运用透明色 `transparent`

拿第二个背景来作为例子讲解，它的过程如下：



第一步：边框

这里可以使用背景来设置，使用 `border` 和 `box-shadow` 的方式更加简单

```
border-radius: 50%;
border: 1px solid #00caf5;
box-shadow: 0 0 0 10px rgba(255,255,255,0.8) inset;
```

第二步：小球

通过 `background-size` 设置背景宽高都为30px，然后通过 `radial-gradient` 绘制小球。其中蓝色是从圆心到半径为9px的地方，大于10px的地方为透明色。

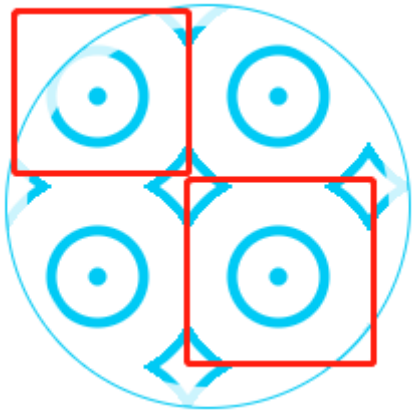
```
background: radial-gradient(#00caf5 9px,transparent 10px);
background-size: 30px 30px;
```

第三步：圆环

同样还是使用镜像渐变，与画小球的原理相同，圆心可以做是上一步的小球，在外面多了圆环，这里需要让圆环过程无限的重复，需要使用 `repeating-radial-gradient`。

```
background: repeating-radial-gradient(#00caf5 0,#00caf5 4px,transparent
5px,transparent 20px,#00caf5 21px,#00caf5 25px,transparent 26px, transparent
50px);
background-size: 90px 90px;
```

中间的四角星是由于背景的宽高只有90px，超出部分隐藏，这样每个重复背景的四个角衔接时就形成了看到的样子。



第四步：合成

最后把上面的几个背景进行合并，多个 `background` 和 `background-size` 的值以 `,` 分隔，`background` 与 `background-size` 的值是一一对应的，若 `background` 值的数量多于 `background-size`，那么多余的部分会从 `background-size` 的第一个开始轮询对应。

点 这里 看效果

关键知识点：

1. 当 `background` 存在多个值时，前面的值优先级更高，也就是说前面的背景会覆盖后面的背景。

来看个例子：

```
background: radial-gradient(#00caf5 9px,transparent 50px), pink;
```

```
background: linear-gradient(pink, pink), radial-gradient(#00caf5 9px,transparent 10px);
```

如果第一个值需要是纯色背景的情况下，不能使用颜色直接表示，否则 `background` 属性设置值无效，我的方法是使用线性渐变代替。

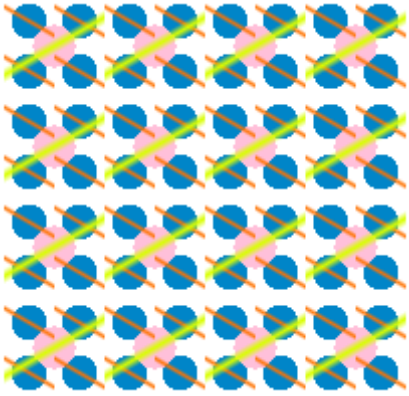
我后面也想了一下原因，由于前面的优先级更高，如果第一个值为纯色背景的话，后面的值效果也无法展现，那么与直接只设置一个纯色值为背景的效果是一样。

```
background: linear-gradient(pink, pink), radial-gradient(#00caf5 9px,transparent 10px);
/*等效于*/
background: pink;
```

2. 可以通过 `background-size` 设置每层背景的大小，如果现在有4层背景，但是 `background-size` 只设置了两个值，那么第3个和第4个背景应该是多大?

```
background:linear-gradient(-30deg, transparent, transparent 45%, rgb(217, 255, 0), transparent 55%, transparent 100%),
    linear-gradient(30deg, transparent, transparent 45%, rgb(255, 115, 0), transparent 55%, transparent 100%),
    radial-gradient(rgb(255, 192, 218),rgb(255, 192, 218) 30%, transparent 20%),
    radial-gradient(#0085C8 , #0085C8 50%,transparent 50%);
    background-size: 50px 50px, 25px 25px;
```

直接看效果：



可以看出，第三个背景的大小为 `50px 50px` ，第4个背景的大小为 `25px 25px`，可以得出第 `n` 个背景的大小为 `background-size` 的第 `n%background-size值的个数` 个值。

混合背景

说到混合背景，不得不引出另外一个css属性 `mix-blend-mode` 混合模式。混合模式最常见于 photoshop 中，是 PS 中十分强大的功能之一。在 CSS 中，我们可以利用混合模式将多个图层混合得到一个新的效果。

将两个线性渐变背景重叠起来，看一下会得到什么：

在我们意料之中，因为背景是不透明的，前面的背景会覆盖后面的背景。

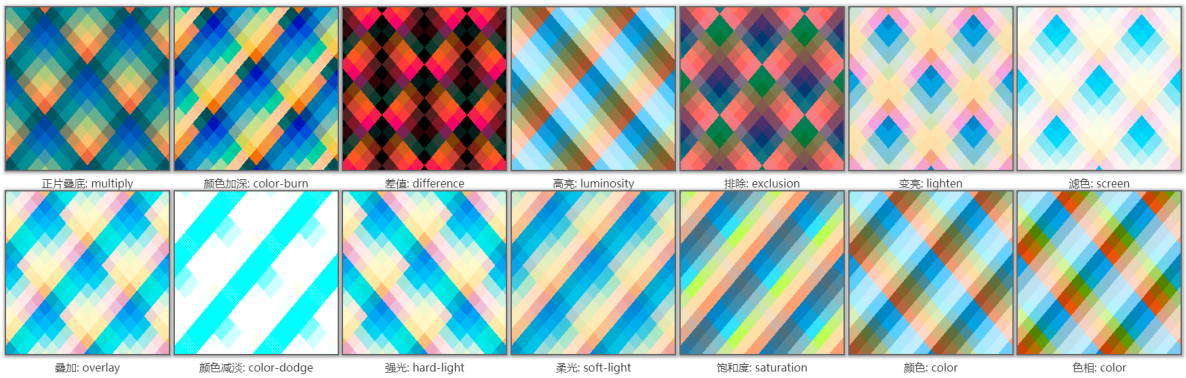
现在为它加上`mix-blend-mode: multiply`属性看看呢：

此时两个背景重叠后会产生一些奇妙的效果，前后背景重叠部分的颜色叠加到一起了，完整代码如下：

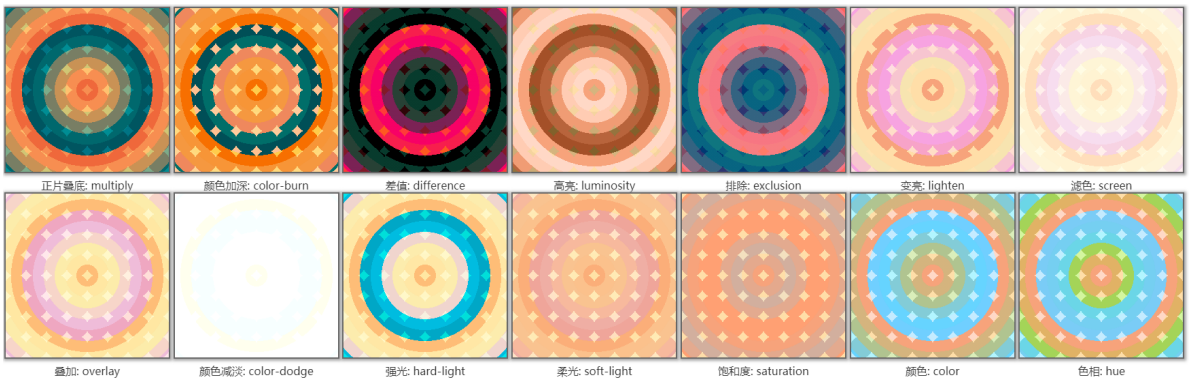
```
 div {
      display: inline-block;
      position: relative;
      width: 240px;
      height: 240px;
      border: 2px solid #666;
      box-shadow: 1px 1px 4px 2px #aaa;
      margin: auto;
      color: #333;
      text-align: center;
      font-size: 16px;
      line-height: 520px;
    }
    div::after {
      content: "";
      position: absolute;
      top: 0;
      left: 0;
      bottom: 0;
      right: 0;
      background:
        repeating-linear-gradient(#F7A37B,#F7A37B 1em,#FFDEA8 1em,#FFDEA8
2em,#D0E4B0 2em,#D0E4B0 3em,#7CC5D0 3em,#7CC5D0 4em,#00A2E1 4em,#00A2E1
5em,#0085C8 5em,#0085C8 6em);
      animation: move 5s infinite linear;
      mix-blend-mode: multiply;
    }

    div::before {
      content: "";
      position: absolute;
      top: 0;
      left: 0;
      bottom: 0;
      right: 0;
      background:
        repeating-linear-gradient(#F7A37B,#F7A37B 1em,#FFDEA8 1em,#FFDEA8
2em,#D0E4B0 2em,#D0E4B0 3em,#7CC5D0 3em,#7CC5D0 4em,#00A2E1 4em,#00A2E1
5em,#0085C8 5em,#0085C8 6em);
      background-size: 30px 30px;
    }
```

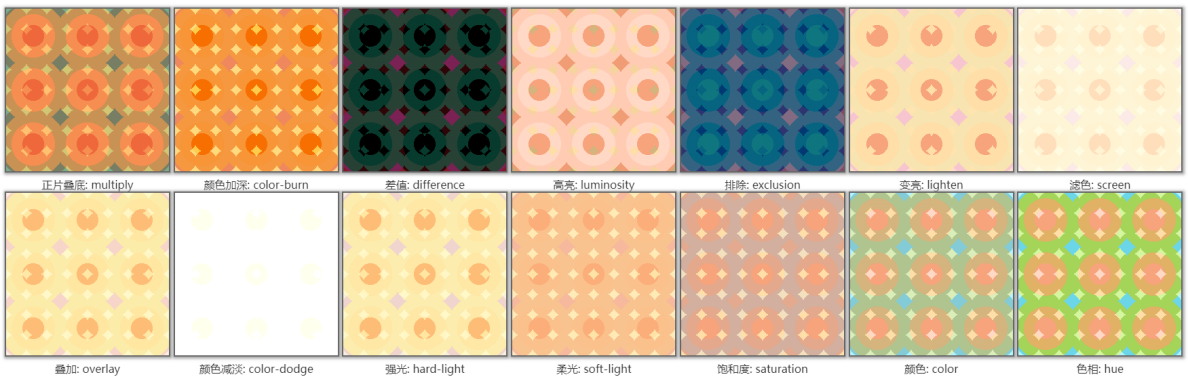
当然，你可使用不同的混合模式来做出一些不同的效果：



也可以使用径向渐变：



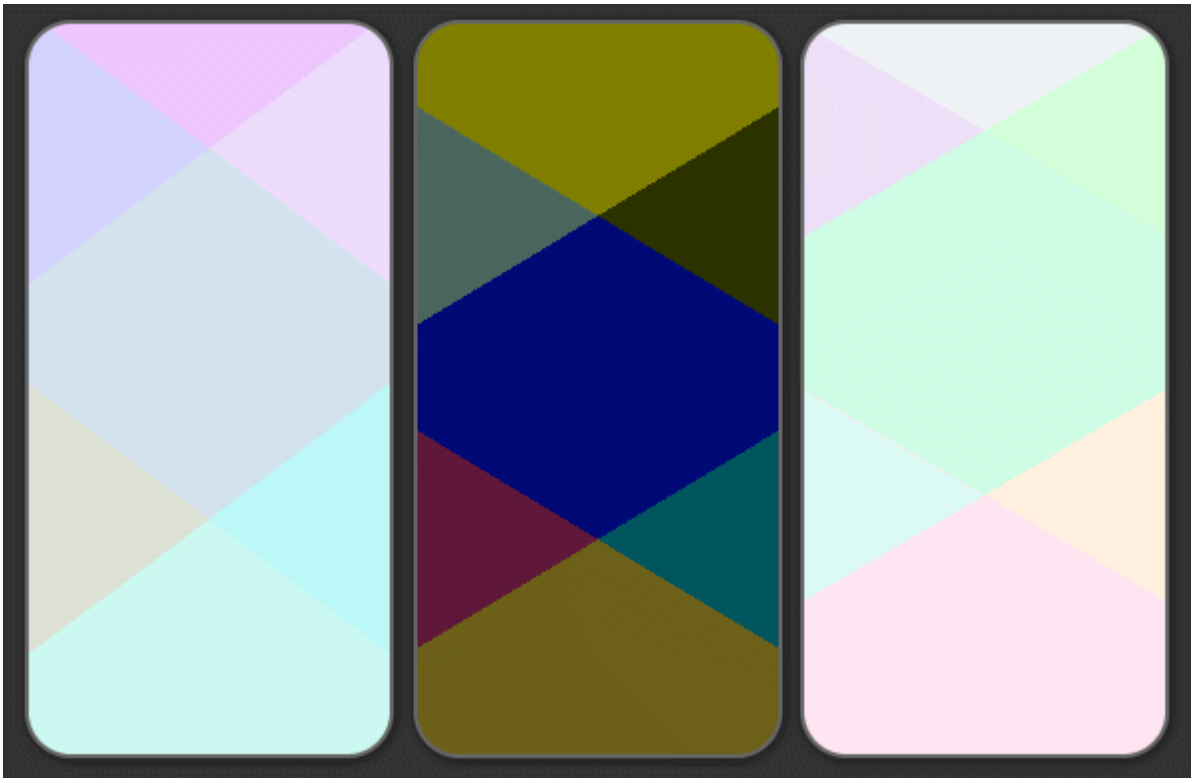
你还可以使用各种组合方式，通过改变颜色、大小、渐变方式、混合模式等可以得到各种各样的背景。

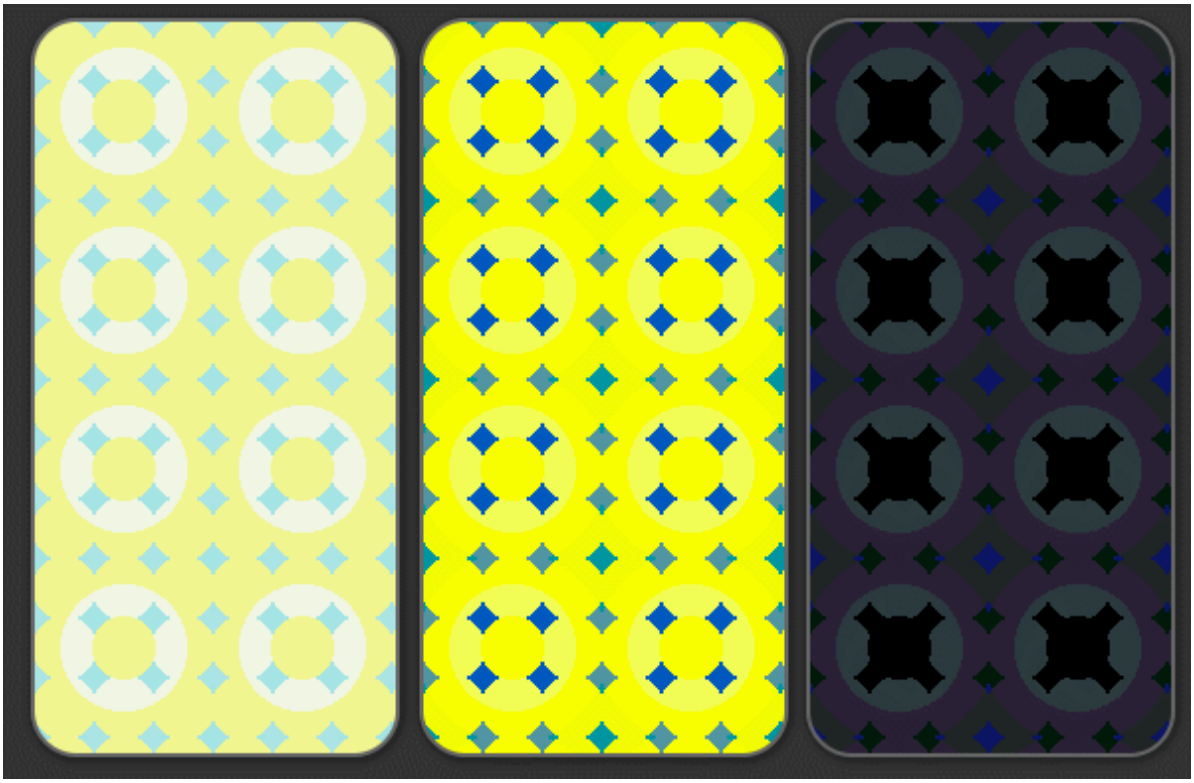


使用 `mix-blend-mode` 还可以做出一些炫酷的效果，比如说 `故障艺术` ，感兴趣的可以看下这篇文章 **奇思妙想CSS之“故障艺术”**

随机背景

先看效果，每次刷新后的背景都是随机生成的 预览1 预览2



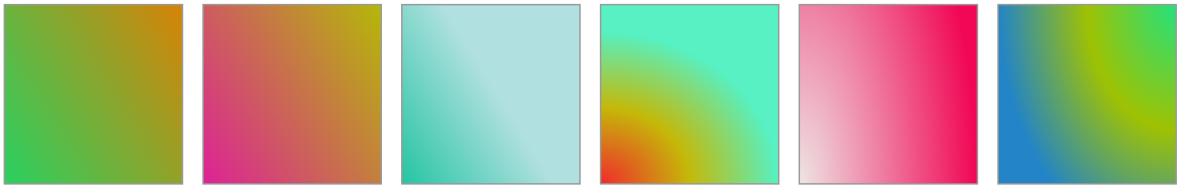


要实现这样的背景，这就需要借助一个强大的CSS图案WEB组件 css-doodle，后面会专门写一篇针对css-doodle的讲解。

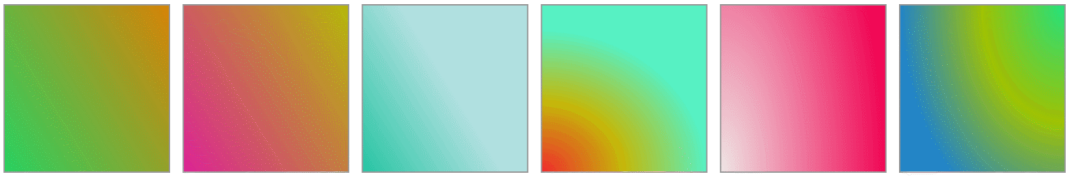
动态背景

上面所有的背景都是静态的，我们现在要让背景动起来。

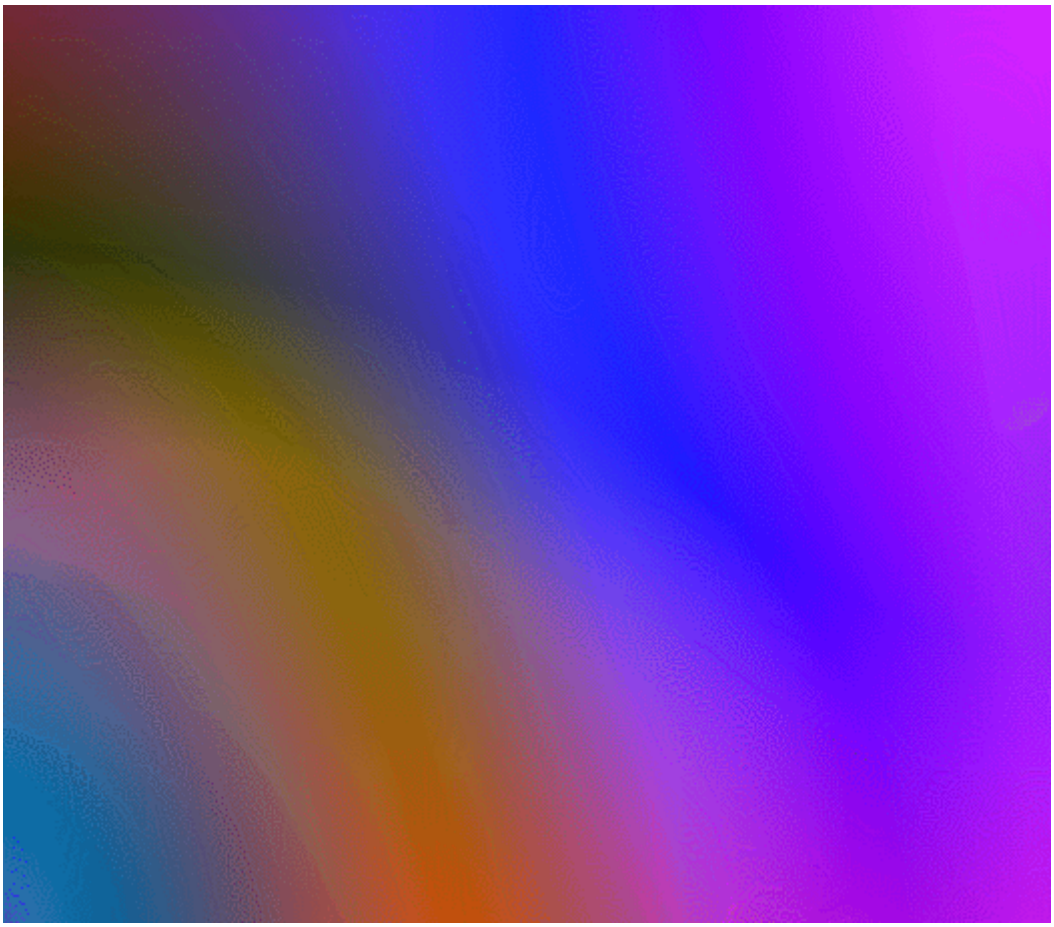
这次我们多创建几个不同类型的渐变背景



分别使用不同的混合模式将他们叠加起来



效果看起来是不是很棒，还没完，现在让背景动起来

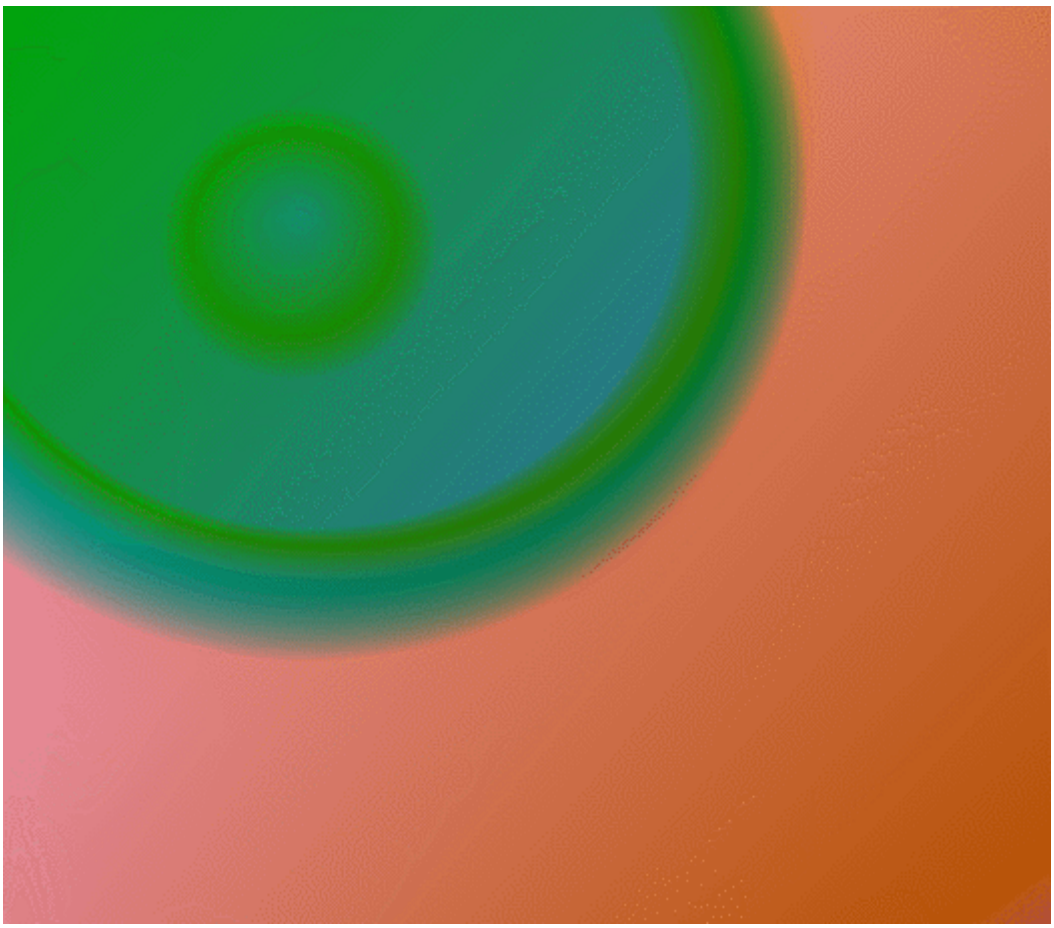


对每个背景都使用 `hue-rotate` 滤镜，从而改变每层背景的色相值，使用混合模式相互叠加后，就可以看到杂乱无序的眩光效果了。

```
@keyframes move {
    0% {
        filter: unset;
    }

    100% {
        filter: hue-rotate(360deg);
    }
}
```

当然，我们同样可以借助 `css-doodle` 来随机生成眩光，，每次点击后的效果都不一样。[在线体验]



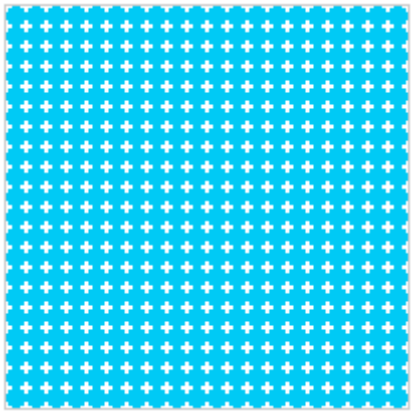
使用mask

顾名思义，mask 译为遮罩。在 CSS 中，mask 属性允许使用者通过遮罩或者裁切特定区域的图片的方式来隐藏一个元素的部分或者全部可见区域。

`mask` 提供一种基于像素级别的，可以控制元素透明度的能力，类似于 `png24` 位或 `png32` 位中 `alpha` 透明通道的效果。

现在有这样一个图案：

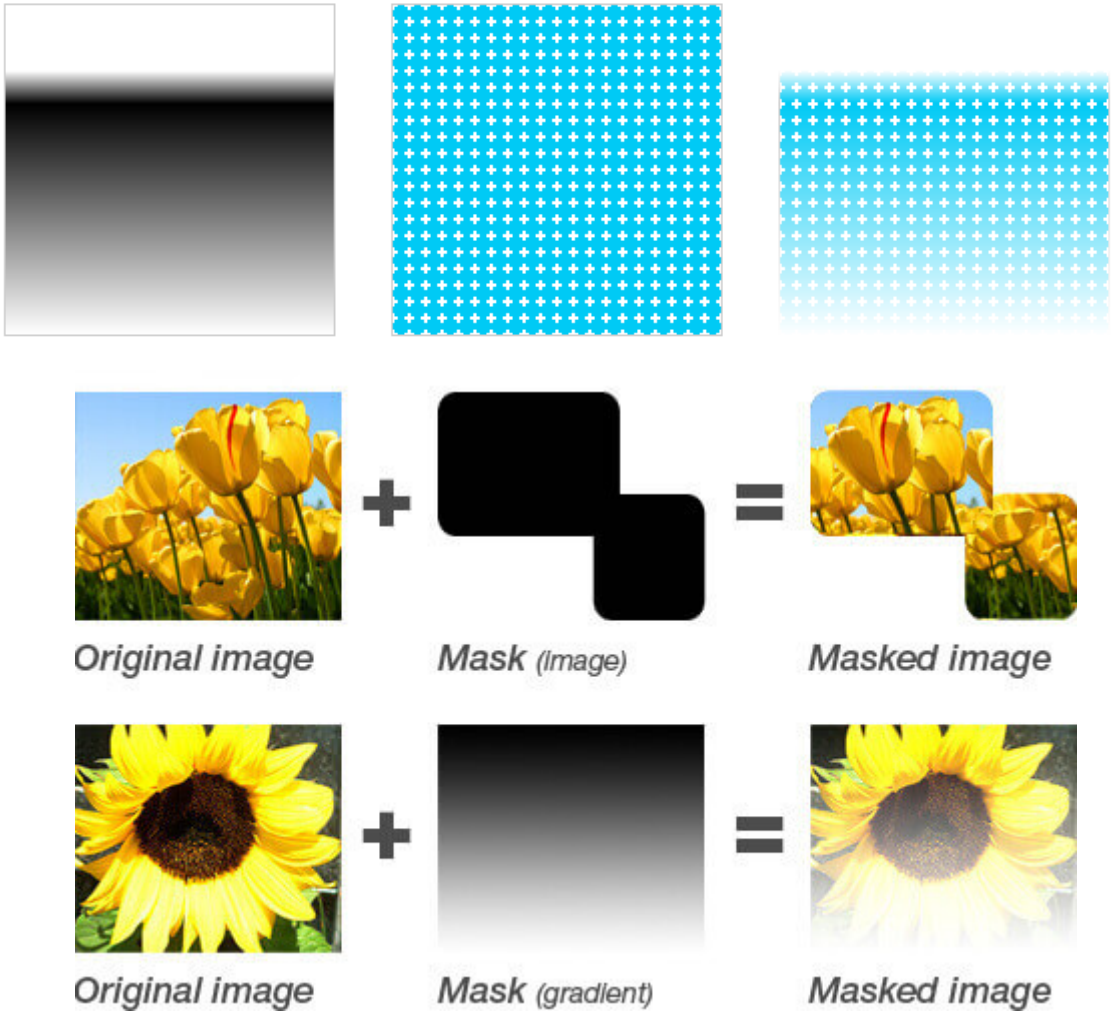
```
background: radial-gradient(circle,#00caf5 5px,transparent 0) 0 0;
background-size: 10px 10px;
```



有这样个遮罩，这里先使用背景代替：

```
background: linear-gradient(transparent 20%, rgb(29, 26, 26) 30%, transparent);
```

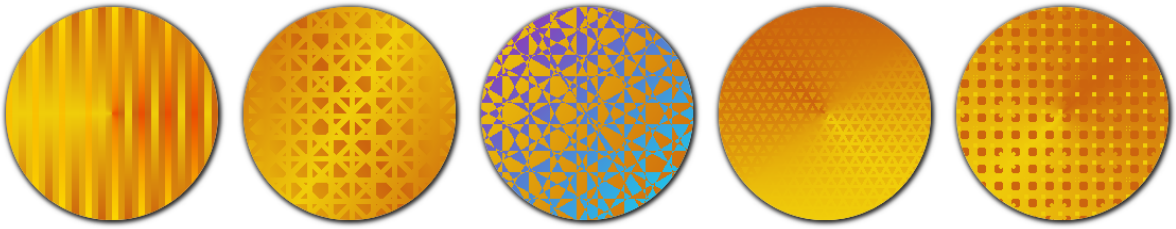
现在将他们重叠起来，定义这样的规则：遮罩白色部分为不透明，黑色部分为透明，就会得到这样的效果



这个遮罩就是 `mask`，完整代码如下：

```
background: radial-gradient(circle,#00caf5 5px,transparent 0) 0 0;
background-size: 10px 10px;
mask: linear-gradient(transparent 20%, #000 30%, transparent);
```

利用 `mask` 可以做出一些完全想象不到的图案，及时你看到源代码也无法想象它最终呈现的样子。



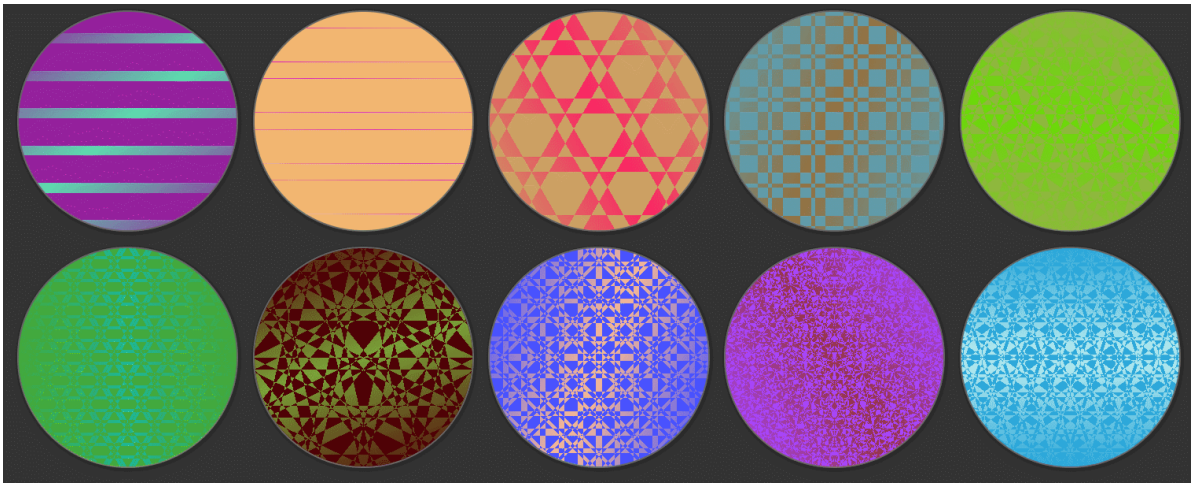
但是很遗憾的是，`mask` 的兼容性不是很好，目前只有 Firefox 和 Edge 浏览器支持。

	Chrome	Edge	Firefox	Internet Explorer	Opera	Safari	WebView Android	Chrome Android	Firefox Android	Opera Android	iOS Safari	Samsung Internet
mask	1 ★	12	2 ★	No	15 ★	3.2 ★	2 ★	18 ★	4 ★	14 ★	3.2 ★	1.0 ★

Full support　Partial support

No support　★ See implementation notes.

同样，借助 css-doodle 可以做出各式各样的万花筒效果：预览



使用 box-shadow

除了使用 `background` 属性外，我们还可以使用**阴影**来制作背景。

我们用 `div` 来制作一个按钮，首先还是使用 `background` 来制作背景：

```
display: inline-block;
background: seagreen;
color: #fff;
padding: 0 10px;
line-height: 2em;
border-radius: 3px;
```

为了使按钮赋有立体感，现在为它加上阴影和高光：

```
box-shadow: 2px 2px 3px #555;
background: linear-gradient(#fff, transparent 5px), linear-gradient(to right,
#fff, transparent 5px), radial-gradient(seagreen 0%,seagreen 100%);
```

上面所有的效果都可以使用 `box-shadow` 来实现：

```
box-shadow: 2px 2px 3px #555, 2px 2px 3px #fff inset, 100px 100px seagreen
inset;
```

可以看出，这种实现方式简单很多，只需要一行代码，这里巧妙的运用了内阴影来充当背景。


```
box-shadow: 2px 2px 3px #555, 2px 0px 10px 5px red inset, 100px 100px yellow
inset;
```

```
box-shadow: 2px 2px 3px #555, -5px -5px 10px 1px #fff inset, 0 0 13px 1px rgb(0,
255, 170) inset, 100px 100px yellow inset;
```

与 `background` 属性有一个共同点，当 `box-shaodow` 存在多个值时，前面的值所在图层更高。

总结

通过本文的学习后，我们认知不再局限于纯色背景或者渐变背景，利用 `mix-blend-mode` 、 `box-shadow` 、 `filter` 、 `mask` 可以创作出一些意想之外的效果。

虽然说某些属性目前的兼容性还不是很好，不过随着CSS3的日益壮大，通过 `background` 来替换一些图片资源得到的效果更佳。来动动你的手指尝试下吧。

background-attachment：决定背景图像的位置是在视口内固定，或者随着包含它的区块滚动。

background-clip：设置元素的背景（背景图片或颜色）是否延伸到边框、内边距盒子、内容盒子下面

background-origin：规定了指定背景图片 `background-image` 属性的原点位置的背景相对区域.