

Dismiss

Join GitHub today

GitHub is home to over 20 million developers working together to host and review code, manage projects, and build software together.

Sign up

谈谈一些有趣的CSS题目（20）-- 巧妙地制作背景色渐变动画！ #10

New issue

Open chokcoco opened this issue on 21 Mar 2017 · 8 comments



chokcoco commented on 21 Mar 2017 • edited ▾

Owner

有的时候，嗯，应该说某些特定场合，我们可能需要下面这样的动画效果，渐变 + animation：



假设我们渐变的写法如下：

```
div {
  background: linear-gradient(90deg, #ffc700 0%, #e91e1e 100%);
}
```

按照常规想法，配合 animation，我们首先会想到在 animation 的步骤中通过改变颜色实现颜色渐变动画，那么我们的 CSS 代码可能是：

```
div {
  background: linear-gradient(90deg, #ffc700 0%, #e91e1e 100%);
  animation: gradientChange 2s infinite;
}

@keyframes gradientChange {
  100% {
    background: linear-gradient(90deg, #e91e1e 0%, #6f27b0 100%);
  }
}
```

上面我们用到了三种颜色：

- #ffc700 黄色
- #e91e1e 红色
- #6f27b0 紫色

最后，并没有我们预期的结果，而是这样的：



[CodePen Demo](#)



我们预期的过渡动画，变成了逐帧动画。

Assignees

No one assigned

Labels

动效  
奇技淫巧



Projects

None yet

Milestone

No milestone

5 participants



也就是说，线性渐变是不支持动画 `animation` 的，那单纯的由一个颜色，变化到另外一个颜色呢？像下面这样：

```
div {
  background: #ffc700;
  animation: gradientChange 3s infinite alternate;
}

@keyframes gradientChange {
  100% {
    background: #e91e1e;
  }
}
```

发现，单纯的单色值是可以发生渐变的：

[CodePen-Demo](#)

So

总结一下，线性渐变（径向渐变）是不支持 `animation` 的，单色的 `background` 是支持的。

查找了下文档，在 `background` 附近区域截图如下：



哪些 CSS 属性可以动画?，上面的截图是不完整的支持 CSS 动画的属性，完整的可以戳左边。

对于 `background` 相关的，文档里写的是支持 `background` 但是没有细说不支持 `background: linear-gradient()/radial-gradient()` 。

那么是否我们想要的背景色渐变动画就无法实现了呢？下面我们就发散下思维看看有没有其他方式可以达到我们的目标。

## 通过 background-position 模拟渐变动画

上面 [哪些CSS 属性可以动画](#)的截图中，列出了与 `background` 相关还有 `background-position`，也就是 `background-position` 是支持动画的，通过改变 `background-position` 的方式，可以实现渐变动画：

```
div {
  background: linear-gradient(90deg, #ffc700 0%, #e91e1e 50%, #6f27b0 100%);
  background-size: 200% 100%;
  background-position: 0 0;
  animation: bgposition 2s infinite linear alternate;
}

@keyframes bgposition {
  0% {
    background-position: 0 0;
  }
  100% {
    background-position: 100% 0;
  }
}
```

这里我们还配合了 `background-size`。首先了解下：

- `background-position`：指定图片的初始位置。这个初始位置是相对于以 `background-origin` 定义的背景位置图层来说的。
- `background-size`：设置背景图片大小。当取值为百分比时，表示指定背景图片相对背景区的百分比大小。当设置两个参数时，第一个值指定图片的宽度，第二个值指定图片的高度。

通过 `background-size: 200% 100%` 将图片的宽度设置为两倍背景区的宽度，再通过改变 `background-position` 的 `x` 轴初始位置来移动图片，由于背景图设置的大小是背景区的两倍，所以 `background-position` 的移动是由 `0 0 -> 100% 0`。

## 通过 background-size 模拟渐变动画

既然 `background-position` 可以，那么另一个 `background-size` 当然也是不遑多让。与上面的方法类似，只是这次 `background-position` 辅助 `background-size`，CSS 代码如下：

```
div {
  background: linear-gradient(90deg, #ffc700 0%, #e91e1e 33%, #6f27b0 66%, #00ff88 100%);
  background-position: 100% 0;
  animation: bgSize 5s infinite ease-in-out alternate;
}

@keyframes bgSize {
  0% {
    background-size: 300% 100%;
  }
  100% {
    background-size: 100% 100%;
  }
}
```

#### CodePen--Demo--position-size 实现渐变动画

通过改变 `background-size` 的第一个值，我将背景图的大小由 3 倍背景区大小向 1 倍背景区大小过渡，在背景图变换的过程中，就有了一种动画的效果。

而至于为什么要配合 `background-position: 100% 0`。是由于如果不设置 `background-position`，默认情况下的值为 `0% 0%`，会导致动画最左侧的颜色不变，像下面这样，不大自然：



## 通过 transform 模拟渐变动画

上面两种方式虽然都可以实现，但是总感觉不够自由，或者随机性不够大。

不仅如此，上述两种方式，由于使用了 `background-position` 和 `background-size`，并且在渐变中改变这两个属性，导致页面不断地进行大量的重绘（repaint），对页面性能消耗非常严重，所以我们还可以试试 `transform` 的方法：

下面这种方式，使用伪元素配合 `transform` 进行渐变动画，通过元素的伪元素 `before` 或者 `after`，在元素内部画出一个大背景，再通过 `transform` 对伪元素进行变换：

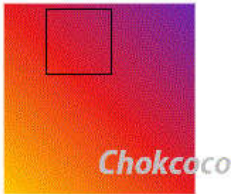
```
div {
  position: relative;
  overflow: hidden;
  width: 100px;
  height: 100px;
  margin: 100px auto;
  border: 2px solid #000;

  &::before {
    content: "";
    position: absolute;
    top: -100%;
    left: -100%;
    bottom: -100%;
    right: -100%;
    background: linear-gradient(45deg, #ffc700 0%, #e91e1e 50%, #6f27b0 100%);
    background-size: 100% 100%;
    animation: bgposition 5s infinite linear alternate;
    z-index: -1;
  }
}

@keyframes bgposition {
  0% {
    transform: translate(30%, 30%);
  }
  25% {
    transform: translate(30%, -30%);
  }
  50% {
    transform: translate(-30%, -30%);
  }
  75% {
    transform: translate(-30%, 30%);
  }
  100% {
    transform: translate(30%, 30%);
  }
}
```

```
}  
}
```

实现原理如下图所示：



[CodePen--Demo--伪元素配合transform实现背景渐变](#)

上面列出来的只是部分方法，理论而言，伪元素配合能够产生位移或者形变的属性都可以完成上面的效果。我们甚至可以运用不同的缓动函数或者借鉴蝉原则，制作出随机性十分强的效果。

当然，本文罗列出来的都是纯 CSS 方法，使用 SVG 或者 Canvas 同样可以制作出来，而且性能更佳。感兴趣的读者可以自行往下研究。

### 运用背景色渐变动画

背景色渐变动画具体可以运用在什么地方呢，稍微举个例子。

#### 背景色渐变过渡实现按钮的明暗变化



[CodePen -- Demo -- 背景色渐变过渡实现按钮的明暗变化](#)

除此之外，在背景板凸显文字，让一些静态底图动起来吸引眼球等地方都有用武之地。

6

1

1

chokcoco changed the title from 背景色渐变动画！？ to 谈谈一些有趣的CSS题目（20）-- 背景色渐变动画！？ on 21 Mar 2017

chokcoco changed the title from 谈谈一些有趣的CSS题目（20）-- 背景色渐变动画！？ to 谈谈一些有趣的CSS题目（20）-- 巧妙地制作背景色渐变动画！ on 23 Mar 2017



kosl90 commented on 24 Mar 2017

linear-gradient 严格来说是 background-image 属性的值，MDN 上的 animation type 为 discrete，咋一看有点懵，从 spec 上来看就比较清晰了：Animatable 是 no



shellphon commented on 4 Aug 2017

由于使用了 background-position 和 background-size，并且在渐变中改变这两个属性，导致页面不断地进行大量的重绘（repaint），对页面性能消耗非常严重，所以我们还可以试试 transform 的方法：

拜读到这一句有个小疑问，请教一下，transform translate不会引起重绘么？翻了一会没找到相关信息



chokcoco commented on 4 Aug 2017

Owner