

# 基于CSS color属性的静态UI组件重构策略

这篇文章发布于 2016年11月18日, 星期五, 01:25, 归类于 [CSS相关](#)。阅读 16298 次, 今日 8 次 [17 条评论](#)

by zhangxinxu from <http://www.zhangxinxu.com/wordpress/?p=5775>

本文可全文转载, 但需得到原作者书面许可, 同时保留原作者和出处, 摘要引流则随意。

## 一、传统静态UI组件实现的隐隐痛点

我们都知道, 一个网站, 只要设计师稍微有点专业, 其站点的一些基础颜色都是贯穿始终的。

主色, 链接色, 警示颜色, 以及各种状态颜色等等, 都是一脉相承的, 如果贵站的文字的红色和按钮的红色是不一样的红色, 文字的绿色和按钮的绿色不是一个绿色, 文字的蓝色和按钮的蓝色不是一个蓝色, 我只能深表遗憾。

传统的实现, 包括现在移动端几乎所有的实现套路都是下面这样的, 无论是标签还是按钮, 都是先设定一个基础类名, 写下基本样式, 然后不同的颜色重新命名一个状态类名, 覆盖默认的边框色或者背景色或者文字颜色, 是什么颜色就写什么颜色。

我们不妨看看微信开源的weui中按钮的实现: <https://weui.io/#button>



HTML部分:

```
<a href="#" class="weui_btn weui_btn_primary">按钮</a>
<a href="#" class="weui_btn weui_btn_warn">确认</a>
<a href="#" class="weui_btn weui_btn_default">按钮</a>
```

对应的CSS实现如下：

```
.weui-btn {
    position: relative;
    display: block;
    margin-left: auto; margin-right: auto;
    padding-left: 14px; padding-right: 14px;
    box-sizing: border-box;
    font-size: 18px;
    text-align: center;
    text-decoration: none;
    color: #FFFFFFF;
    line-height: 2.55555556;
    border-radius: 5px;
    overflow: hidden;
}
.weui-btn_primary {
    background-color: #1aad19;
}
.weui-btn_default {
    color: #000000;
    background-color: #F8F8F8;
}
.weui-btn_warn {
    background-color: #E64340;
}
/* 下面是2个线框按钮 */
.weui-btn_plain-default {
    color: #353535;
    border: 1px solid #353535;
}
.weui-btn_plain-primary {
    color: #1aad19;
    border: 1px solid #1aad19;
}
```

基本上，我们都是这么实现的，用起来也还行，但是，不知道大家在写类似上面代码的时候，有没有一种冗余和啰嗦的感觉？——就颜色不一样而已，但是，却要重新命名一个类名，然后重新写颜色；只要再多一个状态，就要再写一批。好麻烦！

为什么隐隐会有上面这样的感觉呢？

我来带大家剖析下。

### 1. 按钮状态

本质上，按钮就3个状态，常用态，默认态和警示态，但是，从类名的数量上来看，却要5个，而且命名上又长有啰嗦，真的很难忍受，如果再和weui中按钮禁用样式混在一起，足有这么长，看我的手臂，全部张开都不够啊！例如下面这个绿色小按钮的禁用态表示：

```
<a href="#" class="weui-btn weui-btn_mini weui-btn_primary weui-btn_disabled">按钮</a>
```

实际上，我们可以把状态单独抽象出来，如：`default`，`primary`，`warn`，`mini`，`disabled`，与基础按钮样式相互结合，生成UI效果，这样，在HTML中使用类名的时候会清爽很多，而且记忆的成本也降低了。

其核心思想和技术细节参见我前不久刚写的文章：[“基于active.checked等状态类名的web前端交互开发”](#)。

如果上面的按钮套用这种策略，则代码应该类似下面：

HTML部分：

```
<a href="#" class="weui_btn primary">按钮</a>
<a href="#" class="weui_btn warn">确认</a>
<a href="#" class="weui_btn default">按钮</a>
```

对应的CSS实现如下：

```
.weui-btn {
    ...
}
.weui-btn.primary {
    background-color: #1aad19;
}
.weui-btn.default {
    color: #000000;
    background-color: #F8F8F8;
}
.weui-btn.warn {
    background-color: #E64340;
}
/* 下面是2个线框按钮 */
.weui-btn_plain.default {
    color: #353535;
    border: 1px solid #353535;
}
.weui-btn_plain.primary {
    color: #1aad19;
    border: 1px solid #1aad19;
}
```

当然，对于weui这种开源的项目而言，上面基于状态类名策略不一定合适，因为和其他项目混杂，容易有冲突风险。

## 2. 各种颜色

例如 `primary` 状态，明明色值都是一模一样的 `#1aad19`，但是上下却出现了3次，如果使用Sass, Less变量还好，要是传统的CSS写法，以后换起颜色来，怕是要一个一个替换了。

每一种状态都要写一批颜色，要是遇到状态多的场景（如下面这张项目设计稿截图）：



卧槽，那样式就要茫茫多了，一行行全部边框色文字颜色，边框色文字颜色，不过凡事都要两面看，至少这样子代码看上去很壮观。

每一种颜色状态都要写一遍 `color` 属性和 `border` 属性，这种做苦力的感觉没人会喜欢的，明明是重

复的东西，难道就不能前端工程化吗？

实际上，以上两个隐隐的痛点可以一次性全部搞定，就是采用本文所要介绍的“基于color属性的UI组件重构策略”。

## 二、CSS color属性驱动的静态UI组件实现策略

这种实现策略具体如下：

### 1. 提取专门的颜色类名

例如，类似下面这样：

```
.dark { color: #33373d; }
.gray { color: #969ba3; }
.blue { color: #4284ed; }
.green { color: #7ed321; }
.orange { color: #f0643a; }
.yellow { color: #f0c53a; }
.purple { color: #a091ff; }
.red { color: #ed424b; }
.white { color: #fff; }
```

然后，建议放在所有公用样式的最底部。

### 2. 静态UI组件所有动态颜色全部走原生变量

例如，`border` 边框色默认就是 `color` 属性的颜色，因此，写 `border` 时候，颜色值可以直接缺省，直接：

```
.btn-normal { border: 1px solid; }
```

对于背景色，我们可以走CSS的 `currentColor` 变量，关于 `currentColor` 变量可以参见我之前的文章：[“currentColor-CSS3超高校级好用CSS变量”](#)。

```
.btn-normal { background-color: currentColor; }
```

### 3. 颜色类名既扮演状态类名角色又扮演颜色控制角色

HTML直接变成类似下面这样：

```
<a href="" class="btn-normal red">红色按钮</a>
<a href="" class="btn-normal blue">蓝色按钮</a>
<a href="" class="btn-normal green">绿色按钮</a>
```

### 4. 大功告成！

我们看一个具体例子加深理解，关于实色按钮的实现。同样的，我们有一个基础的类名和基本样式：

```
.btn-normal {
  font-size: 14px;
  line-height: 30px;
  display: inline-block;
  padding: 0 16px;
  text-align: center;
  border-radius: 2px;
  background-color: currentColor;
}
```

注意，和传统实现不一样，我们这里直接指定了背景色，但是是以 `currentColor` 变量的形式，也就是背景色和我们的文字颜色保持一致。

什么？背景色和我们的文字颜色！那岂不是按钮文字颜色和背景色混在一起，看个毛啊！

没错，按钮的文字颜色确实不能和背景色一样，但是，由于通常按钮上的文字都只有一行，于是，注意，本文最精华部分来了——考虑到按钮上的文字都是白色，因此我们可以这样处理：

```
.btn-normal::first-line {  
  color: #fff;  
}
```

利用 `::first-line` 伪元素，于是，`.btn-normal` 标签上的颜色实际上是设置给 `background-color` 的，而真正按钮呈现的颜色已经被 `::first-line` 伪元素牢牢设置好了，完全就不用担心文字颜色和背景色混在一起的情况了。

于是乎，配合基础颜色类名，各种颜色按钮全部都出来了。

要实现设计师设计的绿色和红色按钮，直接HTML：

```
<a href="#" class="btn-normal green">绿色按钮</a>  
<a href="#" class="btn-normal red">红色按钮</a>
```

后来，设计师突然发现这两者颜色按钮不够用，又设计了一款同尺寸的蓝色按钮，如果是传统实现，那必须要分别给实色按钮和线框按钮重新起个命名，而且要在CSS代码中继续添加相关的背景色和边框样式代码，你说烦不烦啊！如果是基于 `color` 属性实现，我去，工作量不要太轻松，直接HTML加个 `blue` 类名就可以了：

```
<a href="#" class="btn-normal blue">蓝色按钮</a>
```

没错，结束了，CSS文件都不需要打开了，实际上，一个基础按钮样式写好，其实等同于所有颜色的按钮全部都写好了。

这孰优孰劣明眼人一看就看得出来了。

而且，最最关键的是，这稍微有点分量的项目中的静态UI组件可不止按钮这一种啊！

各种颜色文字本质也是UI组件，然后，各种等级标签，荣誉标签，特殊按钮，还有模拟控件啊，等等。

如果所有的这些静态UI组件全部都采用基于 `color` 属性实现的策略，那这些颜色就成为了真正意义上的贯穿整个项目的颜色变量了，这日后的维护成本是大大的降低，尤其在没有使用Sass，Less，[Stylus](#)这些预编译工具的情况下。

并且，CSS书写的工作量以及CSS代码量那都是明显下降啊，并且HTML层面代码更加精简直白，超好记忆。要知道，HTML代码后期往往可能就是开发人员维护了，除非你文档写得好，否则开发人员怎么知道按钮换个颜色要怎么办？但是你走颜色关键字的独立类名，完全不会CSS的开发人员他也能轻松维护啊。

总而言之，各种爽各种舒畅！

关于基于 `color` 属性实现各类静态UI组件，我特意整理了一个完成的大demo，均是源自真实的项目，您可以狠狠地点击这里：[基于CSS color属性的静态UI组件开发demo](#)

几个颜色类名搞定了下面30多个不同颜色不同风格的静态UI控件，可以说少了上百行CSS代码都不为过

:



最后，再提一句，本策略能够实现的重点技巧就是利用 `::first-line` 伪元素的文字控色技术。

### 三、技术发展与思维转换

技术的发展往往会带来相应的思维方式上的转变，这样才能相辅相成，发挥新技术的潜力，如果还是使用以前的思维模式，怎么说呢，有着劳斯莱斯加着92#汽油在跑的感觉。

weui中按钮的CSS书写就是典型的传统实现策略，扎实而稳固，传统最佳实践的遗留产物。

因为毕竟PC是先发展的，由于技术的限制，我们的思维也被限制了。

例如，由于IE6浏览器对 `.a.b` 这种级联类名样式支持有严重bug，同时考虑到其效率在选择器中排很后，因此，很长一段时间里，我们对于按钮的样式的覆盖策略都不是基于状态，而是基于完整的按钮类名+状态类名的这种方式，因为可以很好地支持IE6，且几乎不可能发生冲突。

放到显示世界，大部分的项目都是自成体系，不会和其他项目直接参杂在一起，因此，可以适当降低考虑冲突的风险；第二，很多项目写页面的就一个人，由于不要担心会遇到黄油手同事，所以，只要自己严格按照准则来书写，则简单的代码和快速的书写带来的收益会更高。

再例如，由于 `currentColor` 变量IE9浏览器才支持，所以，长期的PC项目开发并没有让重构同学意识到我们现在再做移动端项目的时候，可以直接通过一个 `color` 属性，改变按钮或者图标或者标签的颜色而样式不乱。如果这个世界先出现移动端，再出现PC端，我想，我们静态UI组件的书写策略可能就会如本文所言，基于 `color` 属性的系统和贯穿书写，而非来一枪打一炮的这种游击策略。

换句话说，虽然CSS3技术带来了许多很棒的东西，但是，我们的思维方式却似乎还停留在老PC时代，或许是因为关注的仅仅是CSS3表面的那点特性表现的缘故吧。

### 四、基于CSS color重构UI组件适用项目和场景

没有哪一种策略是一方通行的，本文的 `color` 策略虽然精妙且好处多多，但同样有其局限性。

首先是兼容性限制，技术关键点之一 `currentColor` IE9+浏览器才支持，因此，目前适用于移动端开发，以及一些不需要管低版本IE浏览器的项目。虽然浏览器的进步，相信不需要多久，PC项目也能看到这种策略的慢慢普及。

其次，不适用于开源项目。开源意味着会和其他众多使用者的项目混在一起，我们自己团队中开发，自然会有规范约束，但是一旦鱼龙混杂在一起，则，类似 `.red` 这样的短命名类名很容易被其他样式中 `.red` 类名给冲掉。因此，从这一点来看，weui的实现策略实际上是最合适的，使用本文的 `color` 属性策略反而会容易被提issues，但是，细节上可以再完善，例如：

```
.weui-btn_plain-default {
  color: #353535;
  border: 1px solid #353535;
}
.weui-btn_plain-primary {
  color: #1aad19;
  border: 1px solid #1aad19;
}
```

可以直接写成：

```
.weui-btn_plain-default {
  color: #353535;
  border: 1px solid;
}
.weui-btn_plain-primary {
  color: #1aad19;
  border: 1px solid;
}
```

继而，`border:1px solid` 可以放在基础样式中，因此，CSS进一步缩减为：

```
.weui-btn_plain-default {
  color: #353535;
}
.weui-btn_plain-primary {
  color: #1aad19;
}
```

看到没，本质上就是个 `dark` 和 `green` 两个颜色类名。

补充于翌日

评论有人提到，如果日后按钮要从蓝色变成绿色，现在的命名会很麻烦，样式和语义不合，虽然我写页面这么多年，没遇到过类似的变更，但是，难保其他项目不会，因此，对于按钮的命名，还是基于状态比较保险，或者使用比较隐晦的命名，例如 `primary`，`default` 等，但是，样式这块还是使用颜色控制，两者并不冲突。

例如(Sass)：

```
.primary {
  @extend .green;
}
```

或者：

```
.green,
.primary {
  color: ...;
}
```

但，对于很多个颜色种类的标签，显然，还是基于颜色处理最佳，否则，你光想命名就要头疼好一会啊，最后，往往是 `*-1`，`*-2`，`*-3` ...收场，鬼知道 `1`，`2`，`3` 对应的是什么标签。

评论还有人询问 `hover` 态和 `active` 态该如何处理，其实demo页面的按钮那里有示意，可以使用 `box-shadow` 内阴影，或者使用 `background-image` 渐变，如下CSS：

```
.btn-normal:active {
  background-image: linear-gradient(to top, rgba(0, 0, 0, .05), rgba(0, 0, 0, .05));
}
```

所有按钮统一变深处理，如果要变亮可以试试淡淡的白色透明覆盖。

## 五、没有结语总觉得怪怪地

可是人困眼乏，想不到该吐槽什么东西，早年嘛前一篇文章已经拜过了。

那随便放点什么文字吧：

BBC上说，70%婴儿夜间哭闹都是故意假装的，目的是吸引大人来一起玩；而90%男性家长会假装未醒，好让孩子母亲起来去照顾。

没错，我就是那 `90%` ！

《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](#)。



« 基于HTML5 drag/drop模块拖动插入排序删除完整实例

解决文字和text-decoration:underline下划线重叠问题 »

### 猜你喜欢

- CSS背景色镂空技术实际应用及进阶
- CSS镂空图片transition过渡初加载背景色块问题解决
- 面向设计的半封装web组件开发(概要版)
- 基于原生HTML的UI组件开发
- 以20像素为基准的CSS网页布局实践分享
- CSS1-CSS3 <color>颜色知识知多少？
- CSS3图标图形生成技术个人攻略



- 秀几种CSS背景渐变图片transition过渡效果技巧
- AMCSS (CSS属性模式) 开发简介
- SVG图标颜色文字般继承与填充
- 常见的CSS图形绘制合集

分享到: 1

标签: background-color, border, color, currentColor, UI, 按钮, 组件

## 发表评论 (目前17条评论)

<input type="text"/>	名称 (必须)
<input type="text"/>	邮件地址(不会被公开) (必须)
<input type="text"/>	网站
<div></div>	
<div>提交评论</div>	

### 1. wdf说道:

2017年02月16日 10:37

两行文字可以这样。但是多套了一层标签。(伪类前面还有个空格)

```
.btn_normal :first-child{color: #fff;}
```

灰色按钮

蓝色按钮

白色按钮

一直看张老师的文章，崇拜！！

[回复](#)



### 2. Kevin说道:

2016年12月13日 22:18

::first-line是可以解决文本颜色问题，但是如果按钮上的文字不止一行呢？

[回复](#)

张鑫旭说道:

2016年12月13日 22:19

则此方法不适用

[回复](#)



### 3. 最爱的是不二说道:

2016年11月22日 14:47

好想艾特嫂子看最后一句话



[回复](#)

4. **ziven27**说道:

2016年11月21日 11:59

张老师，这里如果只有currentColor不兼容低版本浏览器的话，直接同时设定

```
.dark { background-color: #33373d; color: #33373d; }
```

不是就都兼容了？

基础颜色不正是拿来覆盖的吗？

感觉于情于理都应该放到前面呢。

对于这种容易和别人冲突的的附属类名，

我习惯添加一个前缀。

```
像这样._dark { background-color: #33373d; color: #33373d; }
```

并且我约定，带通用前缀的代码是不能单独使用的，

它们都必须配合父亲，或者兄弟元素使用。

这样应该避免一部分和别人冲突的问题。

老实说：::first-line真的是奇技淫巧。佩服！

这是我根据我的想法处理的版本：

<http://jsbin.com/toheni/edit?html,css,output>

[回复](#)



张 鑫旭说道:

2016年11月21日 20:00

赞的！

[回复](#)



5. **dont**说道:

2016年11月21日 09:56

文章思想好棒。

==于是嫂子看了结语没

[回复](#)



6. **风海流**说道:

2016年11月18日 21:01

IE9+ 使用 currentColor 是没问题的。如果真的要兼容 IE8，可以通过伪元素模拟背景 + border 的颜色来实现。

```
.btn-normal {overflow: hidden; position: relative}
```

```
.btn-normal:before {content: ' '; position: absolute; left: 0; right: 0; top: 0; bottom: 0; z-index: -1; border: 100px solid}
```

[回复](#)



7. **Thestral**说道:

2016年11月18日 15:22

啊 那请问hover 与active应该怎么处理呢？通过透明度来改变嘛？

[回复](#)

张 鑫旭说道:

2016年11月18日 16:03

background-image 或者 box-shadow，demo有示意

[回复](#)



8.

AJ说道:

2016年11月18日 11:37

好像不太符合语义化规范，如果要把网站的所有蓝色按钮改称绿色，那是不是要把所有的blue类都改了呢?

回复

张鑫旭说道:

2016年11月18日 14:50

后面一句话很有道理，虽然概率很小，但确实有这个风险，我补充下。对于语义化规范，“规范”二字可以去掉，类名的命名方式如何并没有所谓的规范。

回复
9.

sapjax说道:

2016年11月18日 11:11

semantic-ui 就是这种命名方式

回复
10.

风满楼说道:

2016年11月18日 10:16

结语真棒！

回复

hijude说道:

2016年11月18日 13:23

结语真的很棒！

回复
11.

meepo说道:

2016年11月18日 09:48

细细读完，受用良多。

我在实际项目中，UE对border颜色设置略显随意，和文字color有时一样，有时不一样。

觉得以后可以沟通好，如果能统一，离“工程化”就又近一步了。

回复
12.

mg说道:

2016年11月18日 03:09

学习了，回头思考一下 换肤功能

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

#### 今年热议

- » [《CSS世界》女主角诚寻靠谱一起奋斗之人](#) <sup>(76)</sup>
- » [不借助Echarts等图形框架原生JS快速实现折线图效果](#) <sup>(64)</sup>
- » [看, for..in和for..of在那里吵架!](#) <sup>(60)</sup>
- » [是时候好好安利下LuLu UI框架了!](#) <sup>(47)</sup>
- » [原来浏览器原生支持JS Base64编码解码](#) <sup>(35)</sup>
- » [妙法攻略: 渐变虚框及边框滚动动画的纯CSS实现](#) <sup>(33)</sup>
- » [炫酷H5中序列图片视频化播放的高性能实现](#) <sup>(31)</sup>
- » [CSS scroll-behavior和JS scrollIntoView让页面滚动平滑](#) <sup>(30)</sup>
- » [windows系统下批量删除OS X系统.DS\\_Store文件](#) <sup>(26)</sup>
- » [写给自己看的display: flex布局教程](#) <sup>(26)</sup>

#### 猜你喜欢

- [CSS背景色镂空技术实际应用及进阶](#)
- [CSS镂空图片transition过渡初加载背景色块问题解决](#)
- [面向设计的半封装web组件开发\(概要版\)](#)
- [基于原生HTML的UI组件开发](#)
- [以20像素为基准的CSS网页布局实践分享](#)
- [CSS1-CSS3 <color>颜色知识知多少?](#)
- [CSS3图标图形生成技术个人攻略](#)
- [秀几种CSS背景渐变图片transion过渡效果技巧](#)
- [AMCSS \(CSS属性模式\) 开发简介](#)
- [SVG图标颜色文字般继承与填充](#)
- [常见的CSS图形绘制合集](#)