Changing SVG Colors with ColdFusion

Posted At : October 12, 2015 12:00 PM | Posted By : Steve
Related Categories: ColdFusion, SVG

I love SVG. I remember when I first heard about it (in 2002) and I just loved the idea. Images made of text! I remember being excited about generating images with dynamic data from my server-side code. I didn't have a problem that I could solve with it yet, but that didn't stop me from immediately buying two books on SVG (" **Teach Yourself SVG**" and "**Designing SVG Web Graphics**").

It would, of course, be years until SVGs had enough browser support to be useful. Recently, I finally had a problem to solve that required me to dynamically serve up an SVG image (like I had dreamed of doing over a decade earlier). One of the systems that we work on is an **MSOC system** (one code base serving up multiple sites – a topic that **Steve "Cutter" Blades** has a good **blog series about**).

In this particular system, the designs of the different sites look pretty dramatically different. There is no commonality between the either in terms of HTML structure or in terms of color scheme. The first sounds more challenging, but it is the second issue that is in play this time.

Historically, we have avoided any images in the content that is common to the sites that would need to integrate with the look of the site. When we started working on this site (in 2004, I think), that would have presented a serious problem as we couldn't have an image which looked good with any web site design. The request came up again recently, however, and we could handle it.

We got some icons that happened to be a dark blue color and went well with the design we were working on at the time. But we also knew that the images would be used in other sites where the dark blue color could potentially clash. Fortunately, SVG is now an option.

This allowed us to build SVG images that were written by ColdFusion to serve up the image in any color we want. The goal was to be able to call as SVG image like this:

```
<img src="myimage.svg.cfm?color=FF0000">
```

We considered JavaScript, but that would require embedding the SVG image in the document rather than calling it as an image element on the page. This would make for messier code and remove the ability for the browser to cache the image. By passing in the color value, the browser caches the image for each color but can still load up the same image with different colors.

How to do it

First, convert the images to .svg image. We used Adobe Illustrator. There are other tools that can handle this, but I was able to get .ai files for the images and I happen to have an older version of Illustrator running on a computer that can handle this pretty well. Ilustrator does a good job of converting images to SVG, but sometimes the positioning get off. If you run into this, make sure to group the image and then copy it into a new image in Ilustrator that is the right size and save that image as SVG.

Once you have the .svg images tested and working (make sure to set up the mime-type for SVG on the web server), convert them to ".svg.cfm" files. The ".svg" in the file name isn't really needed, but makes it so that you can easily see what the file is by just looking at its name.

Then it is time to manually edit the file a bit. First, set up the so that the file will still work as an SVG image:

In the SVG documents I got, the XML declaration was:

```
<?xml version="1.0" encoding="iso-8859-1"?>
```

I changed it to:

```
<cfcontent type="image/svg+xml; charset=utf-8" reset="true"><?xml version="1.0" encoding="iso-8859-1"?>
```

That is enough to get the SVG working again even though it is a .cfm file. To set it up to use a dynamic color, add a URL variable above the line (because the reset="true" in the will redact any white space (or anything else) that happens before it.

```
<cfparam name="URL.color" default="">
<cfif ReFindNoCase("^[0-9a-fA-F]{3,6}$",URL.color)>
   <cfset URL.color = "###URL.color#">
<cfelseif NOT ReFindNoCase("^##[0-9a-fA-F]{3,6}$",URL.color)>
   <cfset URL.color = "##22527D">
</cfif>
```

Basically, this makes sure that URL.color is a valid HTML color code and if not it uses the original blue color that I was provided. You could have changed up the code to allow for named colors, but I didn't need them so I wanted to keep my code short and secure and avoid that complexity.

You could use a Session or Application variable for the color, but a URL variable is simple and flexible. That allows the image to be cached in the browser based on the color and still viewed with two different colors on the same page (should the need to do so ever arise).

Now that the color variable is created, convert the rest of the code to use the dynamic color.

Wrap everything after the XML declaration in a and replaced all instances of that color with "#URL.color# and then escape any remaining pound signs (replacing "#" with "##" to avoid syntax errors). Alternately, include just around each "#URL.color#".

So, for example this:

```
<g>
   <rect x="1.501" y="1.501" style="fill:white;" width="45" height="45"/>
   <path style="fill: #22527D;" d="M48.001,48.001h-48v-48h48V48.001z M3.001,45.001h42v-42h-42V45.001z"/>
</g>
```

Becomes this:

```
<g>
   <rect x="1.501" y="1.501" style="fill:white;" width="45" height="45"/>
   <path style="fill:#URL.color#;" d="M48.001,48.001h-48v-48h48V48.001z M3.001,45.001h42v-42h-42V45.001z"/>
</g>
```

No this code shows the original dark blue:

```
<img src="myimage.svg.cfm">
```

But this shows a dark green:

```
<img src="myimage.svg.cfm?color=008000">
```

The code that calls the image can then use a variable to set the color. Maybe you would have that stored in session scope, for example.

```
<img src="myimage.svg.cfm?color=#Session.Color#">
```

Good luck and enjoy!