<< WebKit GSoC Students Announced

CSS Reflections >>

Home
Surfin' Safari Blog
Planet WebKit
Project Goals
Keeping in Touch
Trac
Contributors Meeting

Working with the Code
Installing Developer Tools
Getting the Code
Building WebKit
Running WebKit
Debugging WebKit
Contributing Code
Commit and Review Policy
Security Policy

Documentation
Wiki
Projects
Code Style Guidelines
Technical Articles
Web Inspector
Web Developer Resources
Demos

Testing
Regression Testing
Leak Hunting
Writing New Tests
Getting a Crash Log

Bugs
Reporting Bugs
Bug Report Guidelines
Bug Prioritization
Test Case Reduction
Bug Life Cycle

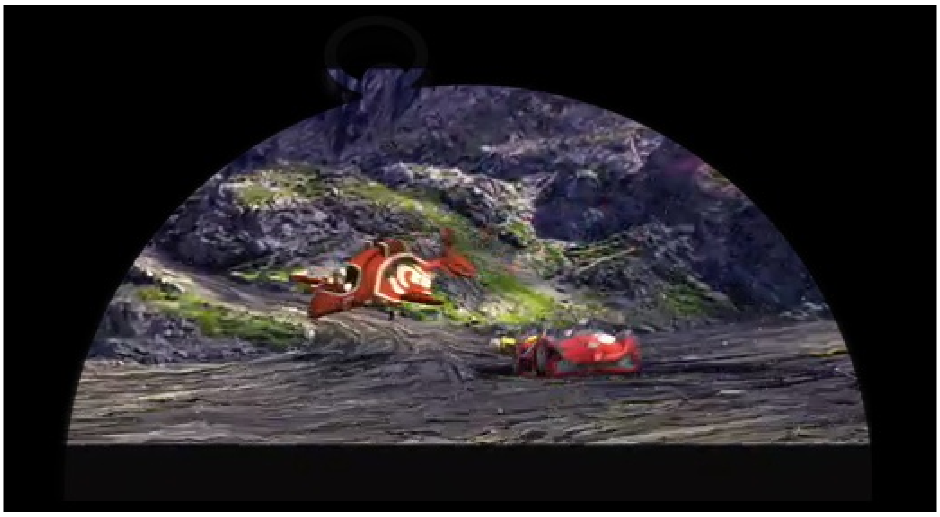
Archives
June 2010
May 2010
April 2010
March 2010
February 2010
January 2010
December 2009
November 2009
October 2009
September 2009
July 2009
June 2009
May 2009
April 2009
March 2009
February 2009
December 2008
October 2008
September 2008
June 2008
May 2008
April 2008
March 2008
February 2008

CSS Masks

Posted by **Dave Hyatt** on Thursday, April 24th, 2008 at 11:53 am

WebKit now supports alpha masks in CSS. Masks allow you to overlay the content of a box with a pattern that can be used to knock out portions of that box in the final display. In other words, you can clip to complex shapes based off the alpha of an image.

Here is a snapshot of the Speed Racer movie trailer clipped to the Safari compass icon. This is in fact a mask applied to the `<video>` element.



We have introduced new properties to provide Web designers with a lot of control over these masks and how they are applied. The new properties are analogous to the background and border-image properties that already exist.

-webkit-mask (background)
-webkit-mask-attachment (background-attachment)
-webkit-mask-clip (background-clip)
-webkit-mask-origin (background-origin)
-webkit-mask-image (background-image)
-webkit-mask-repeat (background-repeat)
-webkit-mask-composite (background-composite)
-webkit-mask-box-image (border-image)

Use of a mask results in a stacking context being created (similar to how opacity and transforms work). The mask will therefore overlay the child and all of its descendants, and at a minimum will knock out everything outside the border box of the object.

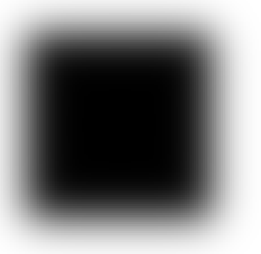
From a Web designer's perspective, think of the mask as being constructed the way the backgrounds and border-image of a box are constructed. Multiple backgrounds are stacking on top of one another (possibly overlapping) with their own tiling and clipping rules. A border-image can then potentially be stretched or tiled over the backgrounds.

The final resultant image built from putting all of the mask images together, tiling them, stretching them, etc., then becomes the mask used to clip the content. Alpha values of 0 in the mask mean that nothing should be drawn. Alpha values of 1 mean the content should display normally. Anything in between ends up partially transparent.

One key difference between `mask-box-image` and its border-image counterpart is that it doesn't attempt to fit itself to border widths. It will just render the corners unscaled and tile/stretch itself into the border box without regard for the border itself. This lets you easily use nine-piece-image effects as masks on elements without borders (often image or video elements).

Here are two sample images. The source image that we want to mask and then the image that we will use as the mask.

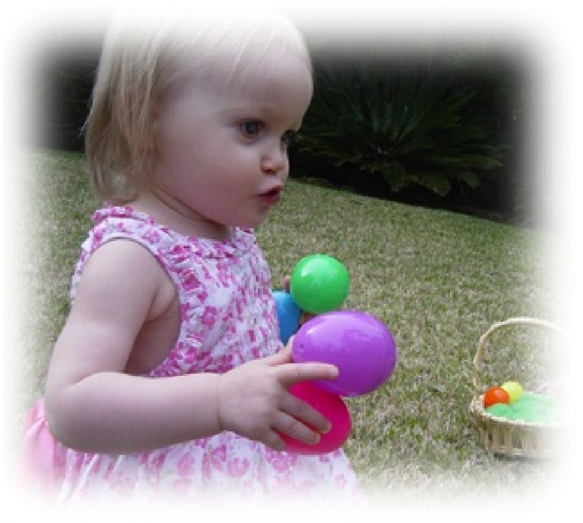
February 2008
January 2008
December 2007
November 2007
October 2007
September 2007
August 2007
July 2007
June 2007
May 2007
April 2007
March 2007
February 2007
January 2007
December 2006
November 2006
October 2006
September 2006
August 2006
June 2006
May 2006
April 2006
March 2006
February 2006
January 2006
December 2005
November 2005
October 2005
September 2005
August 2005
July 2005
June 2005



We can place a mask on the image element simply by doing this:

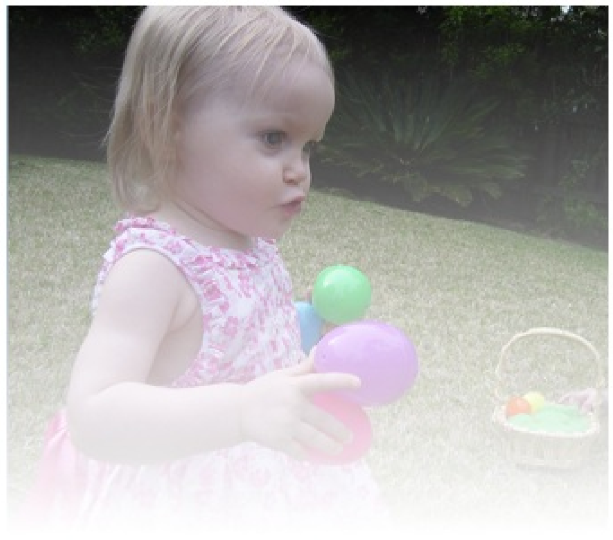
```
<img src="kate.png" style="-webkit-mask-box-image: url(mask.png) 75
stretch;">
```

The result looks like this:



CSS gradients can be used as the mask image source as well. In the following example a linear gradient is used to achieve a fade effect. For example:

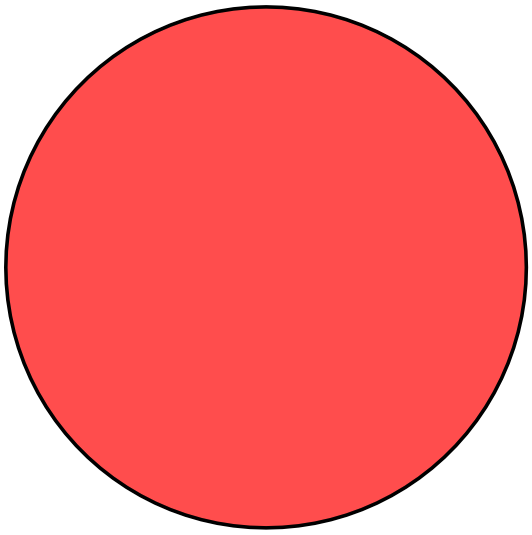
```
<img src="kate.png" style="-webkit-mask-image: -webkit-gradient(linear, left
top, left bottom, from(rgba(0,0,0,1)), to(rgba(0,0,0,0)))">
```



Masks respect the border-radius curves of an object. We can add a border-radius to the previous example, and have both the gradient fade effect with rounded corners.

```
<img src="kate.png" style="-webkit-border-radius: 10px; -webkit-mask-image:
-webkit-gradient(linear, left top, left bottom, from(rgba(0,0,0,1)),
to(rgba(0,0,0,0)))">
```

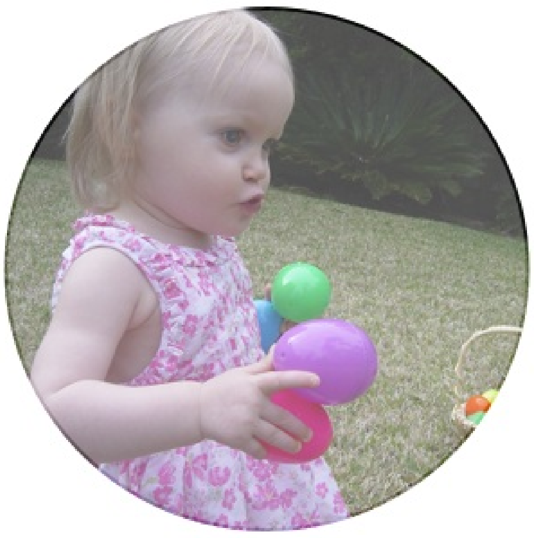
SVG images can be used as masks. For example, a partially transparent circle:



can be applied as a mask like so:

```
<img src="kate.png" style="-webkit-mask-image: url(circle.svg)">
```

The end result is shown below:



All of the power of the multiple background and border-image syntax is at your disposal when building masks!

As usual leave feedback in the comments and file bugs on any problems you notice at http://bugs.webkit.org/.

33 Responses to “CSS Masks”

jeffr Says:
April 24th, 2008 at 12:07 pm

Hyatt, you are an amazing man.

kL Says:
April 24th, 2008 at 12:48 pm

CSS2 specification mentions about extending clip: property in the future. Why use mask then, w hen clip w as supposed to be name for this feature?

marcoskirsch Says:
April 24th, 2008 at 1:07 pm

This is very cool. It seems that all this w ill be Safari/WebKit specific though (so useful for Dashboard Widgets, iPhone w eb-apps, etc. but not for regular plain old w ebsites).
Are there any efforts going on to make these things standard so they w ork accross different brow sers? I'm not completely familiar w ith how these things are w orked out.

memil Says:
April 24th, 2008 at 1:11 pm

w ow , if you make any of these new css extensions standard I don’t know w hat cool new w eb services it w ill lead to… maybe it’ll be the start of w eb 3.0 😉

Dave Hyatt Says:
April 24th, 2008 at 1:11 pm

Yes, these w ill be proposed to CSS.

cparnot Says:
April 24th, 2008 at 1:33 pm

You guys rock. I still think the css animations is the best idea of all time, but this stuff is also pretty amazing. Thanks!!!

MysteryQuest Says:
April 24th, 2008 at 1:41 pm

Could be very useful in some situations. Thank you for including this Hyatt.

How ever I w onder if you w ant to propose this feature for the final CSS3 spec or propose it for CSS4/future-CSS since a lot of end-users are w aiting very long already for CSS3 to get a final spec and such proposals could delay the current targeted recommendation date.

thebigreason Says:
April 24th, 2008 at 2:09 pm

When you look up no brainer in the dictionary, it says, “see -w ebkit-mask: url(…);”

Simple. Brilliant.

mbovett Says:
April 24th, 2008 at 2:36 pm

So w hen does Apple typically implement this stuff into Safari?

guimers8 Says:
April 24th, 2008 at 2:40 pm

Oh my god! Is that true?

Synchro Says:
April 24th, 2008 at 3:00 pm

@mbovett

Right now – go download a nightly and it will be there. This is what we like about WebKit! As far as its appearance in mainstream Safari goes, probably not for several months at least.

I think it's extremely cool that WebKit is in a situation where it's reasonable for key developers to be adding nifty tricks like this rather than wrestling with bugs in some clunky code base.

Now all we need is for script.aculo.us to add support for the CSS animations as automatic replacements for all that javascript.

hawkman Says:
April 24th, 2008 at 3:50 pm

Hmm. The gradient doesn't work for me in the latest nightly, maybe I'm being dumb? This is awesome stuff, though.

DeanEdwards Says:
April 24th, 2008 at 5:04 pm

Unbelievably awesome.

I'm now watching Surfin' Safari for the future of web development.

Mark Rowe Says:
April 24th, 2008 at 5:05 pm

All of the bits are not yet in the latest Windows nightly due to build problems earlier today. Now that the Windows build is fixed you can expect the shiny new features to show up in a nightly later this evening.

Ryan Cannon Says:
April 24th, 2008 at 5:15 pm

These innovations to CSS are exactly what the Web needs. Is there, to date, cumulative documentation of Webkit-only CSS additions?

Scripto Says:
April 24th, 2008 at 9:46 pm

@Ryan: There's CSS Masks, CSS Gradients, CSS Canvas drawing, CSS Transforms and CSS Transitions. There are also proposals for CSS Variables and CSS Animations which to my knowledge aren't actually implemented in WebKit yet.

@hyatt or any other WebKit developer: While all these things are great, I mean, really great, at this time it's just marginally useful to webdevelopers as no other vendor implements any of this. Have you heard any other vendor (informally) express intent to implement any of these great additions?

Pingback from Ajaxian » WebKit keeps going with CSS Masks:
April 24th, 2008 at 10:23 pm

[...] we had CSS animations, and we are now going with CSS Masks which run across images and <video> elements: WebKit now supports alpha masks in CSS. Masks [...]

Dave Jeffery Says:
April 25th, 2008 at 12:47 am

Can the mask property be applied over text?

Pingback from Something Witty Goes Here:
April 25th, 2008 at 5:44 pm

[...] 2008-04-24 – CSS Masks [...]

thinsoldier Says:
April 26th, 2008 at 11:17 am

Wow! It's like you read my mind (or my many posts elsewhere about css masks)… all the way down to using svg paths like clipping masks!!! This is the happiest day of 2008 so far!

Next we need the ability to wrap text inside the shape of an svg path and text along an svg path (inside and out).

aloneinkyoto Says:
April 27th, 2008 at 6:11 am

Can the mask be animated, by for example animating an SVG mask using Javascript?

Dave Hyatt Says:
April 27th, 2008 at 9:39 am

@Dave Jeffery, Yes the mask can be applied over arbitrary HTML content, so yes you can put it over text.

@aloneinkyoto, Yes the mask can be animated, although animated SVGs don't actually work yet when used as images. You can use an animated GIF though (or swap the frames of a PNG by hand).

jdaggett Says:
April 27th, 2008 at 5:40 pm

Maybe using SVG CSS properties would be a more general solution here?

http://groups.google.com/group/mozilla.dev.tech.layout/browse_thread/thread/9c7eac547c33607b/21cc772aac633d27?hl=en&lnk=st

hawkman Says:
April 28th, 2008 at 3:15 am

Ok, got the gradient example working (brackets are wrong).

One problem seems to be that if -webkit-box-shadow is specified, it won't render if you're using masks… Is this by design?

Another question is about consistency of -webkit-border-radius application. Images *don't* respect it, but -webkit-box-shadow (on an image) and -webkit-mask-image *do* respect it. Any chance this could be harmonised, one way or the other?

Maciej Stachowiak Says:
April 28th, 2008 at 7:47 am

@jdaggett

In general, you can think of these as shortcuts to avoid the need for an external SVG file to achieve simple presentational effects like gradients and image masks in CSS. Masks are a bit of a special case, because SVG has confusingly weird semantics for "mask". Most masking operations use either only a grayscale color channel or only the color channel. SVG's masks convert RGB color channels to luminosity (using a different coefficient for each color channel) and then multiply by the alpha to get the final alpha mask. This is confusing and usually not what you want.

I do think allowing SVG styling properties on any element is a neat idea, but I don't think it is a substitute for making the simple cases easy.

Dave Hyatt Says:
April 28th, 2008 at 11:57 am

@hawkman,

Right now the current values of background-clip are border, padding and content. Therefore at a minimum the mask is going to clip out everything outside the border (since it takes the same values). There really needs to be a new value for mask-clip that lets you specify that nothing should be clipped out initially. That's basically the problem you're running into with the shadow vanishing once you apply a mask.

As for images not respecting border-radius, they shouldn't if overflow is visible, since the portions of the image that stick out of the border box are overflowing, and so clipping that in the visible case would be wrong. However, clipping the foreground content to the border radius when overflow is hidden is (I believe) a reasonable thing to do. We have a bug filed on that.

tomliv Says:
April 29th, 2008 at 5:11 am

OK. What am I doing wrong? I pulled down the original image here and copied the code for the gradient and nothing happens.

tomliv Says:

April 29th, 2008 at 5:19 am

I guess READING is in order before posting . Got teh gradient working now … parentheses are incorrect in code example.

So… this is frickin' COOL!

Ryan Cannon Says:

May 1st, 2008 at 1:20 am

@scripto: thanks, but that only gets me halfway there.

What I'm looking for is a matrix of all -webkit CSS additions and in what version of *shipping* products they appear. This is more of an Apple & friends thing than a Webkit thing, but suddenly we have Safari, MobileSafari, OmniWeb, Adobe Air, and (soon) Android, all with different versions of WebKit. We need to get all of this information down.

Mark Rowe Says:

May 1st, 2008 at 12:15 pm

@Ryan Cannon: Does http://developer.apple.com/documentation/AppleApplications/Reference/SafariCSSRef/Introduction.html provide the information that you're after? If you feel that it could do with more information, please take advantage of the feedback links at the bottom of each page on developer.apple.com.

hawkman Says:

May 7th, 2008 at 10:54 pm

@ Dave Hyatt

That makes sense. Thanks for the explanation.

Joel@MMCC Says:

June 18th, 2008 at 8:12 pm

Where would one go to propose new ideas for CSS extensions?

I would like to see "-webkit-border-radius-style:" — this could take parameters such as "convex" (default), "concave", "flat" (or perhaps "beveled" or "angled" for the same effect), and "cutout" (as well as standard CSS keywords such as "inherit").

"convex", the default, means a convex rounded corner, rounded away from the center of the block node.

"concave" would be rounded towards the center. Imagine the corners of an old-fashioned photographic print.

"flat" / "beveled" / "angled" isn't rounded at all, but instead renders a straight diagonal. If the X and Y size of the radius are the same (as is the case if only one radius is specified instead of two), then the angle would be 45°. But just as elliptical rounded corners can be done in CSS3 Borders and Webkit (but not Gecko, let alone Presto [Opera] nor Trident [WinIE] at present) by specifying two separate radius values per corner (one for X and one for Y), using this would allow any desired angle to be specified.

"cutout" (or perhaps "inset") would be a square (or rectangular) inset.

Also, if X and Y radii are specified separately (on a per-corner basis), then either (but not both) should also be allowed to be negative. If negative, then the corner would be reversed in that dimension (in effect, the corner radius ellipse would be outside the box instead of inside). For example, if the X radius of the bottom two corners of a box were negative but its Y were positive, while both X and Y were positive for the top two corners, and the default "convex" style were used then the resulting box would be shaped like a file-folder tab. Instant tabbed navigation bar, without graphic files of any kind (not even SVG)!

```
a.navtab {
height: 18px;
border: 1px outset #color;
border-bottom: 0;
-webkit-border-top-left-radius: 9px;
-webkit-border-top-right-radius: 9px;
-webkit-border-bottom-left-radius: -4px 4px;
-webkit-border-bottom-right-radius: -4px 4px;
}
```

Using just these two enhancements, a very wide variety of box shapes could be done without using graphics. Want an octagon (perhaps for a "STOP" sign)?

```
div.stopsign {
background-color: #900;
-w ebkit-outline: solid medium w hite;
border: solid medium #900;
color: w hite;
height: 100px;
w idth: 100px;
-w ebkit-border-radius: 25px;
-w ebkit-border-radius-style: flat;
}
```

Want a diamond? A hexagon? Not much different: just change the radii!

That old-fashioned photographic print I referred to earlier?

```
img.borderedPhoto {
border: solid thick w hite;
-w ebkit-border-radius: 10px;
-w ebkit-border-radius-style: concave;
}
```

And that's not even getting into the really cool effects that could be done w ith independent per-corner border radius styles!

My other idea is even more pow erful than this.

Mark Rowe Says:

June 18th, 2008 at 9:26 pm

@Joel: That sounds like a topic that could be suggested on the W3Cs w w w -style mailing list.