Six Revisions

Skip site navigation

- Home
- All Articles
- Tutorials
- Freebies
- About
- Contact
- Subscribe: RSS Feed
- Follow on Twitter

Tweet 988 Like 93

10 Random CSS Tricks You Might Want to Know About

Oct 22 2010 by Catalin Rosu | 88 Comments | Stumble Bookmark

CSS is the fundamental way of styling our web pages. Its deceptively easy syntax allows us to do many things to affect the visual layer of our work. And especially with CSS3, the language has gotten even more powerful.

There are many useful CSS techniques and tricks out there for you to take advantage of. This is a collection of a few useful CSS snippets that you might want to keep in your toolkit.

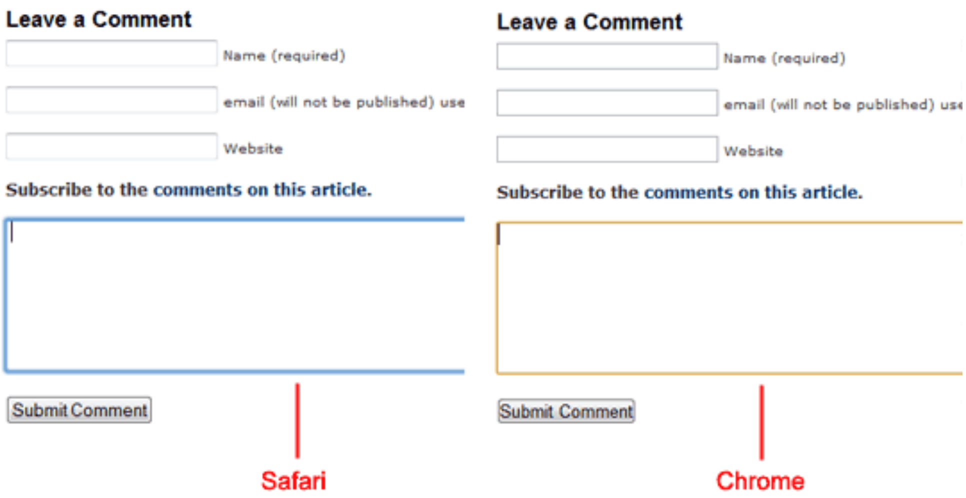
1. Set body font-size to 62.5% for Easier em Conversion

If you would like to use relative units (`em`) for your font sizes, declaring 62.5% for the `font-size` property of the `body` will make it easier to convert `px` to `em`. By doing it this way, converting to `em` is a matter of dividing the `px` value by 10 (e.g. 24px = 2.4em).

```
body {
  font-size: 62.5%; /* font-size 1em = 10px */
}
p {
  font-size: 1.2em; /* 1.2em = 12px */
}
```

2. Remove outline for WebKit Browsers

When you focus (`:focus`) on an `input` element, perhaps you have noticed that Safari adds a blue border around it (and Chrome, a yellow one).


Leave a Comment
Name (required)
email (will not be published) use
Website
Subscribe to the comments on this article.
Submit Comment
Safari
Leave a Comment
Name (required)
email (will not be published) use
Website
Subscribe to the comments on this article.
Submit Comment
Chrome

If you would like to remove this border outline, you can use the following style rule (this removes the outline from text fields):

```
input[type="text"]:focus {
  outline: none;
}
```

Please note that `outline` is used for accessibility purposes so that it is easier to see what input field is active. This is beneficial for those with motor impairments who cannot use a point-and-click device (such as a mouse), and thus must rely on alternative means of navigating a web page, such as the Tab key. The outline is also useful for able-bodied users who use keyboard shortcuts to get to web form input fields (it's easier for them to see which input is currently active). Therefore, rather than completely taking out the outline, consider styling your input fields such that it indicates that it is the focused element.

3. Use CSS transform for Interesting Hover Effects

For progressive enhancement, you could use the `transform` property that is supported by many browsers that have

CSS3 support.

Here's a trick for enlarging `a` elements on hover by 110%.

```
a {
  -moz-transform: scale(1.1);
  -webkit-transform: scale(1.1);
  -o-transform: scale(1.1);
}
```

BUTTON
BUTTON
Normal
On Hover

4. Target IE6 and IE7 Browsers without Conditional Comments

Need to target IE browsers? Here is a quick hack that doesn't require conditional comments (note that your CSS will therefore not pass auto-validation, which is fine if you are aware of why it doesn't).

The code below will change the `background-color` of `div`s depending on what browser the user is viewing the web page under. Since `*` cascades down to IE7 and below, we use _ after that declaration so that IE6 (and below) has a different background color from IE7.

```
div {
  background-color: #999; /* all browsers */
  *background-color: #ccc; /* add a * before the property - IE7 and below */
  _background-color: #000; /* add a _ before the property - IE6 and below */
}
```

5. Support Transparency/Opacity in All Major Browsers

This example gives the `div` element a 70% opacity. We need to use proprietary CSS to get it to work on Internet Explorer (which will invalidate our code under W3C standards).

```
div {
/* standards-compliant browsers */
opacity:0.7;

/* The following is ignored by standards-based browsers */
-ms-filter: "progid:DXImageTransform.Microsoft.Alpha(Opacity=70)";  /* IE8 */
filter: alpha(opacity=70); /* IE 5-7  */
}
```

6. Use !important to Override Normal CSS Precedence Rules

In CSS, when two properties apply to the same element, the one that is farther down the stylesheet is the one that will take effect. However, by using `!important` at the end of a property value, this can be changed.

Let's take, for example, this set of style rules:

```
h1 { color: green; }
h1 { color: red; }
```

The `h1` element will have a red color with the CSS above.

If we wanted to change the style rule's priority without changing the order of the property declaration, then we simply do the following:

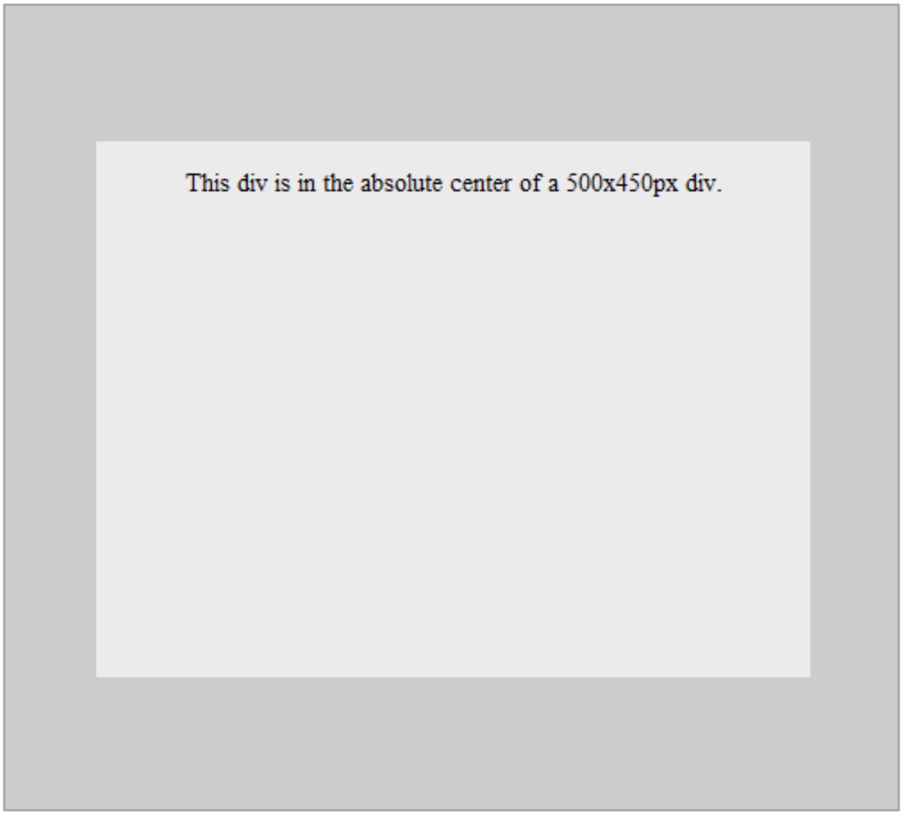
```
h1 { color: green !important; }
h1 { color: red; }
```

Now, the `<h1>` element will be green.

7. Centering a Fixed-Sized Element

Here is one way to center a fixed-width/fixed-height `div` at the center of its container. This could be adapted to centering text, images, etc. within their containers. Essentially, we do a bit of arithmetic to get the fixed-sized element centered using absolute positioning and margins. Note that the parent container must have a `position: relative` property for this to work.

```
div {
  position: absolute;
  top: 50%;
  left: 50%;
  width: 400px;
  height: 300px;
  margin-top: -150px; /* 1/2 of your element height*/
  margin-left: -200px; /* 1/2 of your element width */
}
```



This div is in the absolute center of a 500x450px div.

8. Easy Web Fonts with Google Font API

Web fonts allow you to step outside of the normal web-safe fonts by taking advantage of CSS's @font-face rule. However, right now, browsers aren't uniform in its implementation of `@font-face`. More specifically, web browsers differ in the types of font files they support (hopefully this will change with the WOFF standards). Additionally, you must be careful with the fonts you use since some of them might not be licensed for web use.

To sidestep the issues with `@font-face`, the Google Font API is here to the rescue.

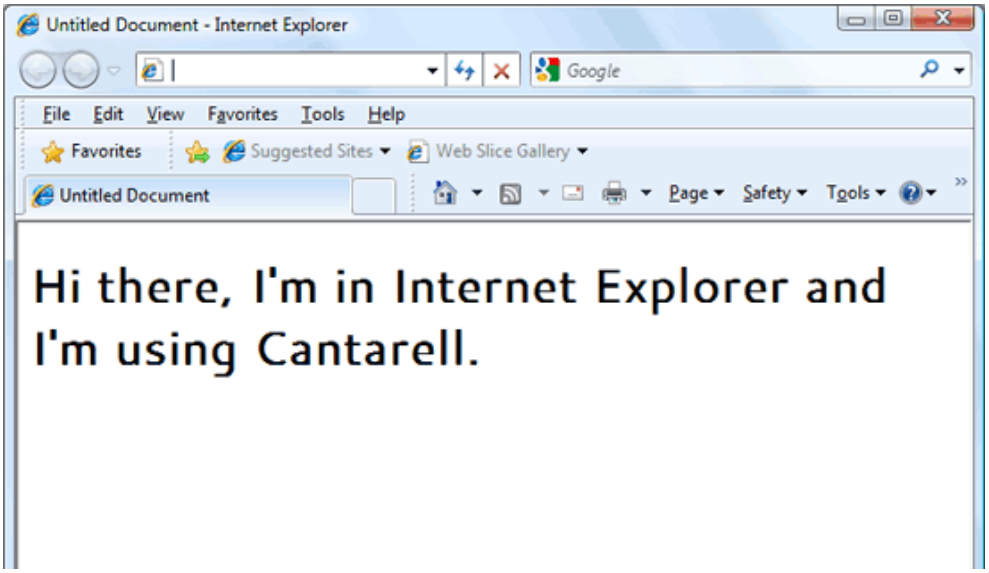
Here is an example of using the Cantarell font on `<h1>` elements that takes advantage of Google Fonts API.

If you want to use the Cantarell font from Google Font API, first reference the remote stylesheet inside your `<head>` tags as such:

```
<link href="http://fonts.googleapis.com/css?family=Cantarell" rel="stylesheet" type="text/css">
```

To use the font in `h1` elements, simply use the `font-family` CSS property.

```
h1 {
  font-family: 'Cantarell', Arial, serif;  /* Use a font stack, just in case. */
}
```

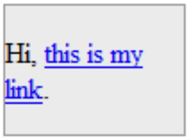


Hi there, I'm in Internet Explorer and I'm using Cantarell.

9. Prevent Line-Wrapping of Text Elements

Sometimes, you don't want your text to wrap to the next line if it happens to reach the end of the width of its containing element.

Here is how a normal anchor text works when it reaches the end of its parent element's width:

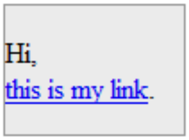


Hi, this is my link.

Notice that the link wraps at a whitespace in the text. What if we always want our links to be on the same line all the time (i.e. to prevent wrapping)? Simple. We just use the `white-space` property.

```
a { white-space: nowrap; }
```

Now, our links won't wrap.



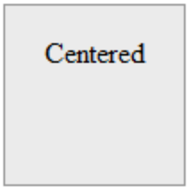
Hi,
this is my link.

10. Vertically Align Text

We can use a variety of methods for horizontally aligning text (such as `text-align: center` or `margin: 0 auto`) but it's slightly trickier to vertically align text.

However, for single-line text, we can use the `line-height` property. By setting the `line-height` property of the text element to the same height of its container, it will become vertically centered.

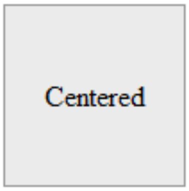
Here is a `p` element that is horizontally-centered inside a 100x100px `div` using `text-align: center`:



Centered

As you can see, `text-align` doesn't center it vertically. To fix that, we can set `line-height` to the same height as the containing `div` (100px).

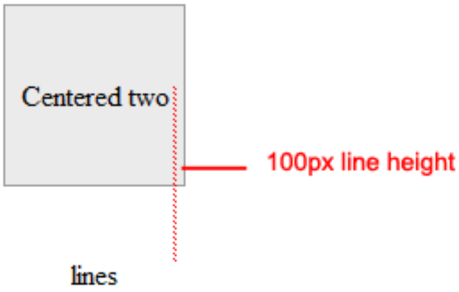
```
div { width:100px; height:100px; }
div p { line-height:100px; }
```



Centered

Note that this assumes the `p` element has no margin or padding. If you have margin or padding at the top or bottom of the `p` element, you need to compensate for them accordingly or just simply set `padding` and `margin` to 0 to make life easier.

Also, this trick becomes troublesome when there is more than one line of text (i.e. when the text wraps) because there will be a space between the text lines that is equivalent to the `line-height` value.



Centered two
lines

It's your turn to share. ***Share your own favorite CSS tricks and techniques in the comments.***

Related Content

- 12 Common CSS Mistakes Web Developers Make
- 10 Interesting CSS3 Experiments and Demos
- 3 Advanced CSS3 Techniques You Should Learn
- *Related categories*: CSS and Web Design

About the Author

Catalin Rosu, a.k.a. Red, is a web designer and developer with over 5 years experience in the industry. You can find articles or tutorials on CSS tips and tricks by visiting his blog at red-team-design.com. You can connect with him on Twitter as @catalinred.

88 Comments

Young

October 22nd, 2010

Well done! #4 and #5 I've found quite useful over the years. #1 is awesome cuz I'm pretty px-loyal but have wanted to use ems. I wish the centering tricks weren't so…inflexible. I sometimes miss the deprecated attributes.

Scott Munn

October 22nd, 2010

Great article. Learned a few tricks I didn't know, especially regarding the trick for vertical centering using line-height.

nipsy

October 22nd, 2010

This is perfect for me! Finally CSS tips and rules I not only need but can understand. You just joined my book mark folder of VIP coding help. Thank You!

Vitor Neves

October 22nd, 2010

Hi Catalin,

About the trick nr 7. Centering a Fixed-Sized Element.
It also could became a problem depending on the resolution.
I had use this trick until 2 months ago when i discover, when we are seeing the website on a small monitor and height is bigger than the resolution, it cuts on the top as you can see here http://www.arquitecturaperitraco.com if you resize your browser, you will see the image being cut.

Since i discover this problem i start using jQuery to get it right, without this problem.

Regards,
Vitor

Nate

October 22nd, 2010

So #7 seems kinda of crazy. margin:0 auto; is all you need to center a fixed sized element.

RussellUresti

October 22nd, 2010

These are good, basic tricks that most front-end developers will need to use pretty often. Though, you forgot one on number 4, and that’s targeting IE8 as well. IE8 USUALLY renders just like the other browsers, but it can often behave like IE7 in terms of collapsing margins. So, to adjust your example, this is what I would show:

```
div {
background-color: #999; /* all browsers */
background-color: #aaa\9; /* add a \9 after the value – IE8 and below */
*background-color: #ccc; /* add a * before the property – IE7 and below */
_background-color: #000; /* add a _ before the property – IE6 and below */
}
```

Brian Cody

October 22nd, 2010

Cross browser Gradients:

```
background: #999; /* for non-css3 browsers */

filter: progid:DXImageTransform.Microsoft.gradient(startColorstr=’#cccccc’, endColorstr=’#000000′); /* for IE */
background: -webkit-gradient(linear, left top, left bottom, from(#ccc), to(#000)); /* for webkit browsers */
background: -moz-linear-gradient(top, #ccc, #000); /* for firefox 3.6+ */
```

See http://www.webdesignerwall.com/tutorials/cross-browser-css-gradient/ for more details

using jquery

October 22nd, 2010

I have never understood the point of em’s for fonts. if you need to know what the font size is in pixels, why not use pixels?

Mayoorathen

October 22nd, 2010

Bookmarked it.. Really useful.

Thanks.

James Bull

October 22nd, 2010

#6 is a handy trick, especially when dealing with IE6. As IE6 ignores the “!important” and will use the last value set for the same property.

In that specific case IE6 would display the h1 color as red.

Simon Willans

October 22nd, 2010

For the 62.5% = 10px – that’ll only be correct if the default browser font size is set to 16px right?

Ranjit

October 22nd, 2010

Interesting.. Thanks for sharing

Red

October 22nd, 2010

Hi and thanks for your comments!

@Vitor, regarding your good remark… indeed this is a side effect of this centering method.

Although you also must keep in mind that this solution applies best for centering blocks that are smaller than your browser viewport.

The best example for that would be a centered login panel.

Lauhakari [mikko]

October 22nd, 2010

Trick #1, well this doesn't work if someone, like myself, has changed the default text-size in your browser!

So instead I'd set the font-size:10px; on the body element, or as part of the reset. As then you'll know for sure what the start size is…

Dave

October 22nd, 2010

With the #7 alternative "margin: 0 auto;" you also need to declare a width on the element you are centering (inline css is only used for example):

Dave

October 22nd, 2010

Edit to prior comment (html was removed):

```
div {
width:15em;
margin:0 auto;
}
```

helium

October 22nd, 2010

Haven't Yet really played around with css transforms.

Mikesh Vulco

October 22nd, 2010

very nice post… thx

Red

October 22nd, 2010

@Nate it's about vertical alignment and not horizontal

@RussellUresti thanks for adding the targeting example for IE8. I didn't added that because very rarely developers need to target IE8 specifically.

@Brian Cody thanks for your good addition!

@Simon Willans you are right. And as I know the default html font size it's usually 16px for all web-browsers.

@Lauhakari [mikko]
If you want to be 100% sure about your default font size you can always add "html{font-size:16px}" in your CSS file.

Amber Weinberg

October 22nd, 2010

Using browser hacks in stylesheets is NOT the way to make your sites work in IE. It invalidates the CSS and is a backwards practice from where we're trying to move development to today.

Marco Monteiro

October 22nd, 2010

wow, there was some really good stuff in here. =) specially the thing about the white space, i never use it and is really useful.

Jacob Gube

October 22nd, 2010

@Amber Weinberg: I want to put in my two cents on that. This being a site that is for modern web developers/web designers with a strong emphasis on standards-based web development, I agree with the spirit of what you're saying.

But, this site is also about usefulness and pragmatism, and being grounded in reality instead of thinking about perfect scenarios and theoreticals.

If you've ever worked for big multinationals, governments, acedemia, and anything in the mainstream — the reality is that many of us can't code for Google Chrome or for Firefox. Many of us are under strict resource constraints and thus must resort to these things. This is the sad reality for many of us — we don't live in a perfect world.

Validation shouldn't be used as a reason not to do something. It is a set of standards and best practices, but at the end of the day, we gotta get paid and we gotta produce and we gotta output.

Now there are plenty of ways not to use hacks and proprietary prefixes for IE, and still come out with a great design that looks awesome in IE6. I've, in the past, spent 30-40% of dev time to avoid hacks and conditional comments in a site design and still support IE6.

But think about that. I've been a programmer (C, C++, VBscript) and web developer (PHP, CSS, JS) since I was in my teens (hobbyist at the start). I run a dev blog, and I've written a book on a very technical subject (MooTools). It's still takes me that long just to avoid breaking standards AND support IE6 at the same time.

As long as you know why it's breaking the given specs, don't let auto-validation stop you.

I value developers who understand the specs enough to know what is important and what isn't, and who understand what it will mean to deviate from those set practices.

Those that blindly follow — or worse, use perfect W3C validation as a badge of honor — isn't being realistic with the situation that most of us are under and is a very narrow way of looking at our industry as a whole. And quite frankly, this situation is as bad as a developer who doesn't know how to write code under best practices.

Most of us (well, less so me nowadays) don't get the luxury of working with clients that have an audience of mostly Chrome, Safari, Firefox in a way that they can neglect IE. We all (again — less for me, nowadays) don't get to make a living talking about ideal situations (writing books, running a web development blog).

Again, my $0.02.

Red

October 22nd, 2010

@Dave please note it's about vertical alignment for a block element

@Amber thanks for you comment, I agree with you regarding the fact that a CSS file won't pass the W3C validation if hacks are used.

This way, to fix well-known IE bugs, there are three solutions:

1. the use of conditional comments – now the main CSS file will contain no more hacks and it will validate

2. the above example of targeting browsers

3. just ignore the old browsers

tehk

October 22nd, 2010

I love the hover zoom buttons! I just started using that after reading your article…. BRILLIANT!

6-2 website design

October 22nd, 2010

@Jacob Gube: Amen and well put. It is a bit of a tightrope that we walk as website designers, often having to balance out the need for standards compliance and interoperability in favor of sites which get finished and actually work in the real (digital) world.

I've spent countless hours swearing at IE for doing something that it shouldn't, only to discover that the answer lie in some hack or IE-specific way of doing things, leading to invalidated code but ho-hum; as you rightly put it, we live in an imperfect world, and we're just trying to do our best with it.

@Amber Weinberg: Unforunately browser hacks ARE the way to make your site work in IE because, as we well know, not all versions of IE properly support valid HTML/XHTML and in saying this I point the finger of blame agressively at IE6 and cite the difficulties it has with the CSS Box Model.

I think the following will provide some illuminating reading for you : http://www.456bereastreet.com/archive/200612/internet_explorer_and_the_css_box_model/

Anyhoo, handy tips and hacks, love the post!

Mary

October 22nd, 2010

@Jacob I am sorry Jacob, but this is one of the worst statements I read in a while concerning practices in web design and development. I completely agree with Amber on that topic. There is a reason why we try to standardize things and validation is essential, and anybody coming from the field of programming will understand why. Your "excuses" for using hacks is just so lousy because you can always do one thing: build in a conditional statement and use a stylesheet for IE6, IE7 and the rest of the browsers. Offering qualitative web design and development services should include doing things right and NOT using invalid workarounds because of nonsense time-saving excuses…

Rachell

October 22nd, 2010

I almost passed this article by thinking "What weird-ass CSS are they going to talk about now?" and then was floored! Almost all these tricks you mention are things I've been struggling with! I'm properly humbled and now bow down to you, Oh Lord of CSS! ;)

Amber Weinberg

October 22nd, 2010

@Jacob I'm a web developer and I work with many large companies, so I know what they need on a day to day basis. All of my sites have to work in IE as well. I don't approve of hacks, or even conditional stylesheets (of course for IE6 somethings you HAVE to use coditional sheets) but certainly never for IE7 or above. All IEs after IE6 follow web standards pretty well, so if you're having a layout issue, it's more likely you've coded it wrong somehow and it needs to be done.

Also, I've never talked about "ideal" situations on my site, I've always discussed what's going on now and how to deal with things.

It should never take longer than a few seconds to validate your sites if you're coding correctly…

Jacob Gube

October 22nd, 2010

@Amber Weinberg: I'm not directing the comment to you as a web developer, but it does come off that way (and so maybe I shouldn't use "you", I have a terrible habit of speaking to a general audience using "you". I also have a habit of saying "we" as in "us" even if it may or may not include you or I). I know you are a web developer, I know you are a GREAT web developer. But rather, my comment was more towards the notion of such a very straight and strict no-exception thinking of how we should do things. Because with this viewpoint, the person should also be opposed to progressive enhancement and using vendor prefixes for CSS3 or HTML5 until W3C places them in final recommendation (REC).

I think a "this is the way things should be" doesn't really move us forward, innovation-wise. It keeps the status quo, more than anything.

Edit: And just to clarify, my direct comment to you (Amber) ended in the first paragraph. Afterwards, it was more of a general comment to everyone based off your statement. Re-reading my first comment, it's reading "you" as in "Amber Weinberg" rather than "you" as in "people who are reading this comment."

Peter

October 22nd, 2010

I agree with Jacob,

Not because I'm lazy, but sometimes you inherit jobs that require you to do a lot of patch work. If something is built the way it is, you can't always start fresh.

Red

October 22nd, 2010

Regarding this, I can say I agree both with W3C validation and using CSS IE hacks as long the web developer is aware of its consequences.

SmakDaddy

October 22nd, 2010

I also agree 100% with Jacob. The IE family of browsers certainly do not follow the standards very well (geez, just see #5 above!!) at all and in a commercial environment, developing for organisations who's entire internal staff are forced to use IE6, it takes an additional 30% of development time to make sites work correctly. By default we don't support IE6 anymore, but if a client wants it we factor it in to their estimate.

Malena

October 22nd, 2010

Useful tips..although #4 makes me wanna vomit. I think one must always try to make things work without resorting to hacks as much as possible. Like Red has mentioned earlier.

Young

October 22nd, 2010

I'm partial with Jacob's opinion on this. I've had some things work on IE8 and not on IE7 – I wish I remembered exactly what it was – but to my knowledge and forum experts' there was no other way to do it except with browser hacks or…pretty hacky javascript. The argument really comes down to how much time you have, not how well you know CSS. I don't really care if I can't declare a website is validated unless that was something my client considered important. To me the ultimate purpose of standardization is so the code nor the website look objectively stupid. In that Machiavellian way browser hacks get the job done in next to no time. Yeah it's a backwards practice – like keeping kosher – but W3C validations are also guidelines. I think Red's way of looking at it is the best for all of us.

Evan

October 22nd, 2010

Nice information…I haven't ever really grasped how CSS is used properly…so this is really blowing my mind, thanks for posting…I'll share it to on FB and Twitter for ya…keep cranking it out man!

Ahmad

October 22nd, 2010

Pretty simple love em

Luis Herrero

October 22nd, 2010

Here’s my two tricks:

1.- At the reset css document:
div { position: relative; }
This will give you the ability to position:absolute anything inside a div.

2.- When you need a container get the height of the floating child nodes:
overflow:auto
No need to add an element with some clear-both class.

Excellent article, btw.

Regards

paul

October 22nd, 2010

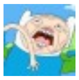
I agree with Amber that we shouldn’t use hacks. But I don’t agree that stylesheets need to validate. Don’t you use CSS3?

Using tools such as modernizr, it’s possible to develop standards compliant and semantic HTML and CSS, and give any browser the best experience possible for its level of support.

I suggest reading Andy Clarke’s Hardboiled Web Design, it’s been a revelation to me!

http://hardboiledwebdesign.com/

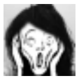
using Microsoft proprietary filters is really not a good ides by the way.



Kayin

October 22nd, 2010

Curious as to why use a Body font size of 62.5% over one of 10px ? Wouldn’t the % be based on the user’s browser font size ?



ardianzzz

October 23rd, 2010

OMG, i love the point 1! Thank you! :)

james

October 23rd, 2010

I read the headline and told myself "another worthless tricks" – on the contrary! Item 4 was of value to me.

To throw some fire into this discussion board, I have never followed W3C guidelines and I never will. If a visitor comes to my site on IE 6, this is what they have coming…

http://www.jailface.com/ie6.html

VagrantRadio

October 23rd, 2010

In the Remove outline for WebKit Browsers example you show the focus on a textarea but give an Attribute selector for just text inputs. For the textarea you should use the Type selector instead – textarea:focus {outline:none;}

Alternatively, you can just use the Universal selector *:focus {outline:none;}. It will also remove the webkit input outline, the dotted outline around links and buttons but it will hinder accessibility for screen readers so use carefully.

Alex

October 23rd, 2010

Using Conditional Comments are hacks too, they are invalid HTML.

Just because the W3C validator ignores them because they are in a comment, does not mean that they are valid. They are not part of the spec.

The real truth is that hacks are in fact NOT necessary, but you will have to total control over your HTML and CSS and often rely on additional mark-up and hooks to make your site truly cross browser compatible.

Tom Hughes

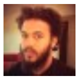
October 23rd, 2010

I can’t believe we’re actually having a discussion about IE hacks in 2010. Conditional comments are the accepted best practice and neither the author nor Jacob have provided a compelling reason not to use them. If you read the comments above you can see that several people don’t understand what conditional comments are, if you want to teach people css, you might start there.

Kannan Sanjeevan

October 23rd, 2010

Great list .. Being a CSS fan it always feels great to learn about CSS .. Thanks :)



Thomas Davis

October 23rd, 2010

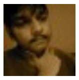
“I think a “this is the way things should be” doesn’t really move us forward, innovation-wise. It keeps the status quo, more than anything.”

You win

Leichim

October 23rd, 2010

Very nice tricks! I especially like the trick using important!, never heard of that before.



Satish Gandham

October 23rd, 2010

Are u sure about the first one, I think 1em is not always equal to 10px.

if body{font-size:18px}
then 1em=18px;

Correct me if im wrong.

Dreb

October 23rd, 2010

I'm with Jacob. It's not about following the standard coding (of course we do follow them) but how should we deal with incompatibility of those license applications and open source. IE have different conventions since it's license so there are some syntax that won't work even if its fine with open sources browsers (okay, enough of this. i know you pretty much knew what i'm talking about). So i guess it's only part of innovating idea out of web developer's creative minds. "Hack" may be bad in some form but surely useful when use in good way.

Gabriel

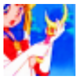
October 23rd, 2010

There are better alternatives for CSS hacks and !important rules.

CSS hacks are based on syntax errors, and that can never be a good idea.
The !important rule should only be used when the property you want to override is declared under a very-high-I-can't-beat specificity selector.

I would recommend the following:
1 – Write a good article about CSS Selectors Specificity, I would be glad to help if you want;
2 – Use the "conditional commented body tag declaration" as in html5boilerplate.com, and setting higher specificity selectors to avoid hacking, and therefor, getting you CSS valid, clean and correct.

Keep up the good work!



Jpop

October 23rd, 2010

Be careful when using line-height to vertically-align text like that. If you have a 100px line-height and you want two lines of text vertically centered, it's not going to work right :)

Vertical-align:middle works sometimes!

Red

October 23rd, 2010

Thank you all for your comments and I am glad you found this article useful!

Gabriele Maidecchi

October 23rd, 2010

Articles like these make me feel like a rookie. Very very nice tips.
Thanks for sharing! :)

Jacob Gube

October 23rd, 2010

@Gabriel: Shoot me an email (contact information found here) if you'd like to contribute a CSS selectors specificity guide.

@Jpop: It talks about that issue (with illustrations, even)…

Amber Weinberg

October 23rd, 2010

I can promise you CSS/HTML hacks are NOT the way to the future of development…nor to the best clients (Several of my big name clients will refuse to hire you if you use them). Like I said before, if you're having a layout issue, it's more likely that you've coded it wrong somehow and it needs to be redone.

Also, CSS3 isn't considered a hack, so invalidating CSS due to CSS3 isn't an issue, it's browser hacks that is.

But I'll end the argument there.

Haze

October 23rd, 2010

Thanks. Great tips. I love the ie tip!

Teresa

October 23rd, 2010

Very helpful tips. Thanks for sharing!

Clint Davis

October 24th, 2010

OOOOOOOH nice! I'll especially be trying out #8

Scott

October 24th, 2010

Great collection, very useful.

andre.roesti

October 24th, 2010

Number #1 is incorrect. It might be right for some Browsers, but what if a browser defines an other default font-size (e.g. via userstylesheet)? There's an amazing german Article by Thomas Scholz about that myth: http://toscho.de/2009/mythos-font-size-62dot5-prozent/

Webranes

October 24th, 2010

There are a lot of gems in this post! They're all pretty useful. Bookmarked this awesome page. Thanks for sharing this to us.

keith

October 24th, 2010

"Conditional Comments – not valid, but accepted"
"CSS Hacks – not valid, not accepted, but they keep all your CSS in one place"

I think the beauty of CSS Hacks is that they keep all those conditional codes in one place. One CSS file to update, so when you've inherited (or passed on) the code, it's not a mystery to why IE6 looks different.

Conditional Comments – they cut down on your CSS file size (negligible, with GZIP enabled), increase on HTTP requests, and make maintenance a pain when you inherit it.

So, which is better, validation for the sake of validation (even though every browser has some specific prefixes that are not valid…), additional HTTP requests and invalid HTML, or invalid CSS with a slightly larger file size?

Or, WHAT IF:
Someone writes a parser (ant, php, whatever) that takes your single CSS file with invalid CSS that then parses it for production to create the specific IE conditional styles?

Just a thought/pondering.

KC

October 25th, 2010

nice article … thanks number 10 i ever missed this.. thanks a lot i found it now

Red

October 25th, 2010

@andre.roesti regarding number 1 statement:

The default size for 'medium' text in ALL modern browsers is **16px**.

Of course this happens only if the user hasn't deliberately modified his default font settings.

Hassan Raza

October 25th, 2010

I didn't use CSS IE Hacks ever but i was taking its CSS validation issue as i take invalidated CSS3 in my code. So i think i code well so that i need include conditional CSS very rare. Still i want to say thanks to Amber Weinberg and Jacob Gube for their helpful arguments.

Succo Design

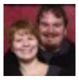
October 25th, 2010

Very useful! thanks ;)

Des

October 25th, 2010

WC3 is a croc. I have seen sites which say we are validated blah blah, then if you test them in older browsers they stuff up!! At the end of the day its better to have a site that doesnt validate, but works correctly in the browsers than to have a site that spouts its so ggooood but doesnt actually render properly in the browsers.



Amos Vryhof

October 25th, 2010

For the browser hacks vs. Conditionals, I just want to add my 2 cents. I’ve been using the HTML5 Boilerplate for many pages/sites I have been building lately, and like their approach to handling IE Specific stuff. They use conditionals to present the BODY tag with browser specific classes, so on the different IE versions, you have different classes. Then all you have to do is create rules for these specific classes

div.ie6 { filter:blah-blah-blah; }

it takes a little more typing to target more than one IE version, but it avoids the need for targeting hacks in a real slick way.

Satya Prakash

October 25th, 2010

Check Google font Viewer: http://www.satya-weblog.com/2010/06/google-font-api-font-viewer-2.html

Sebastian

October 25th, 2010

in #6 I´d rather use the whole path to override your css. example: div#container div#contact h1 {color:green}.
it could also work if you id your body in the section you want the h1 to be green.

Mike

October 25th, 2010

Great tricks! 1, 7 and 10 are especially useful. Thanks for sharing.

Joshua

October 25th, 2010

Great tricks! I like number 2 the most.
I also have a great trick: Style links depending on url or file formats.
For example:
a[href$=".pdf"] {
color: green;
}
See for another small example my post about it (it gives two other examples).
http://joshualuckers.nl/2010/08/07/style-links-depending-on-url-or-file-format/

WTN

October 26th, 2010

I really don't recommend using CSS cross browser hacks like that.

Use something like the following and then you can keep your hacks away from your main (and valid) css code.

<!–[if lt IE 7]> <![endif]–>

Regarding em sizes. I've stopped using them in favour for the much easier to control/use pixel sizes. em's are always relative to it's container and personally I end up in a mess. With pixel sizes everything feels much cleaner and easier to maintain.

Avoid !important as much as possible. I wish this wasn't even allowed.

I personally hate using remote javascript libraries like Google type, jquery etc.. Keep them local, they'll load faster and it won't break if something changes on the host after a few years.

momo

October 26th, 2010

nice, but you don't need google to embed fonts with @font-face

Graham

October 26th, 2010

Why not just set the font-size to 10px, instead of 62.5%, which assumes the default is 16px? — All your ems would be wrong if that were ever to change. Setting the default font-size to an absolute value like 10px prevents that.

ritika

October 27th, 2010

Very nice article. The point about the vertical alignment using the line-height is something that I didn't know before and is going to be very useful

Dan

October 27th, 2010

Great post. For "7. Centering a Fixed-Sized Element", is there a way to do this based on percentage width? For example, if you want the box to always be 60% wide by 60% tall on the screen?

Red

October 29th, 2010

@Dan you can check this http://sixrevisions.com/css/semantic-css3-lightboxes/ to see how this centering method works with percents.

@ritika Thanks, much appreciated.

@Graham This is just a method, you can set the default as you want but also must be aware that if you'll set a px default value you'll miss font scalability in case the user change its default font size.

pkachhia

October 30th, 2010

Thanks, very nice article, You saved my lots of hours.

Ochko

October 30th, 2010

Thanks, very useful, especially the trick with Google Font API, but *margin: 0 auto;* is perfect for positioning. Bookmarked!

jimmysingh

November 2nd, 2010

@post 8 (Vertically Align Text)
We can make text center of the box by using this method display:table-cell and vertical-align:middle

```
.class {
width:100px;
height:100px;
background-color:#999999;
color:#000000;
border:1px solid red;
text-align:center; /*to make text horizontally center*/
display:table-cell; /*this is the trick*/
vertical-align:middle; /*to make text vertically center*/
}
```

Red

November 3rd, 2010

@pkachhia, @Ochko thanks for your appreciations!

@jimmysingh Nice one, but you must be aware that’s not a cross-browser solution.

jimmysingh

November 10th, 2010

@RED thanks for ur comment and u can check this in IE also

Haamrialune

November 16th, 2010

Perfect! I got rid of that yellow frame on my input fields:) So cool.

Thanks a lot!

Catherine

December 17th, 2010

For the vertically centered text, simply put this rule instead:

div p:first-line { line-height:100px; }

It should work (haven’t tested)

Adrian

January 12th, 2011

great tricks!

mnu

February 7th, 2011

Not Bad

Dinesh babu

March 7th, 2011

Nice article :-)

Leave a Comment

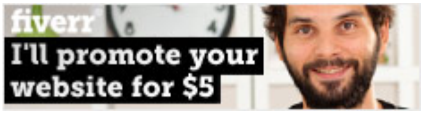
Name (required)

email (will not be published) used for Gravatars (required)

Website

Subscribe to the comments on this article.

Submit Comment



Search

- Search

Topics

- AJAX
- CSS
- Design Showcase / Inspiration
- Flash
- Freebies
- Graphic Design
- JavaScript
- Mobile
- Photoshop
- Project Management
- Resources
- Tools
- Tutorials
- Usability / Accessibility
- User Interface
- Web Applications
- Web Design
- Web Development
- Web Standards
- WordPress

Recent

- Giveaway: 5 Packs of 20 Responsive Web Templates from DX Themes
- 10 Free High Quality Paint and Paper Textures
- Announcement: Winners of Monitive Lifetime Subscriptions
- Free Icon Font: Pictonic Cube
- Announcement: Winners of Fluid UI Lifetime Subscriptions

Partners

- maxCDN™ CONTENT DELIVERY NETWORK

Friends

- 1stwebdesigner
- Addictive Fonts
- AddToDesign
- App Sheriff
- Blog.SpoonGraphics
- BrushLovers
- Burbia
- Chris Wallace
- CoolHomepages
- CSS Globe
- Design Bump
- DesignOra
- DesignM.ag
- Desizn Tech
- fudgegraphics
- Function
- GraphicsFuel
- iBrandStudio
- InstantShift
- LaptopLogic.com
- Marcofolio.net
- MyInkBlog
- Naldz Graphics
- NETTUTS
- N.Design Studio
- Noupe
- Onextrapixel
- ProductiveDreams
- psdfan.com
- PSDVIBE
- Queness
- [Re]Encoded.com
- Smashing Apps
- Smashing Magazine
- Stylegala
- Speckyboy Design Magazine
- Stylized Web
- Technology.am
- TheBestDesigns.com
- Vandelay Design
- Walyou
- Web Designer Help
- Webdesigner Depot

- Web Design Ledger
- Wordpress Themes Shock
- WPBeginner

Become a Facebook Fan of Six Revisions. **Advertise** - Contact - RSS Feed Six Revisions mobile version by Mobify.

87981 readers
BY FEEDBURNER