Web standards and IE8

Peter-Paul Koch (ppk)
http://twitter.com/ppk
http://quirksmode.org
MS Dev Days, 29 mei 2009
<table>
<thead>
<tr>
<th>Event</th>
<th>IE 5.5</th>
<th>IE 6</th>
<th>IE 7</th>
<th>IE8b1</th>
<th>FF 2</th>
<th>FF 3b5</th>
<th>Saf 3.0 Win</th>
<th>Saf 3.1 Win</th>
<th>Opera 9.26</th>
<th>Opera 9.5b</th>
<th>Konqueror 3.5.7</th>
</tr>
</thead>
<tbody>
<tr>
<td>blur</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>almost</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>yes</strong></td>
<td>too many</td>
<td>almost</td>
<td>incomplete</td>
<td><strong>incomplete</strong></td>
<td><strong>incomplete</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>When an element loses the focus.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Firefox 2 fires too many events in a variety of circumstances.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Firefox 3 fires too many events when blurring the window.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Safari and Opera don’t support these events on links and/or form fields in all circumstances.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Konqueror doesn’t support these events on the browser window.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>change</td>
<td>buggy</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>When a form field value changes.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• IE has a serious bug in its handling of this event on checkboxes and radios.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>click</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>When amousedown and mouseup event occur on the same element OR an element is activated by the keyboard.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>contextmenu</td>
<td>yes</td>
<td>minimal</td>
<td>yes</td>
<td>buggy</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Preventing the default (i.e. preventing the context menu from appearing) is the whole point of this event.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Event</td>
<td>IE 5.5</td>
<td>IE 6</td>
<td>IE 7</td>
<td>IE8b1</td>
<td>FF 2</td>
<td>FF 3b5</td>
<td>Saf 3.0 Win</td>
<td>Saf 3.1 Win</td>
<td>Opera 9.26</td>
<td>Opera 9.5b</td>
<td>Konqueror 3.5.7</td>
</tr>
<tr>
<td>---------</td>
<td>--------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
<td>------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
<td>----------------</td>
</tr>
<tr>
<td>blur</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>too many</td>
<td>almost</td>
<td>incomplete</td>
<td>almost</td>
<td>incomplete</td>
<td></td>
<td></td>
</tr>
<tr>
<td>When an element loses the focus.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>change</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>buggy</td>
<td>yes</td>
<td></td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td>When a form field value changes.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>click</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td>When a mousedown and mouseup event occur on the same element OR an element is activated by the keyboard.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>contextmenu</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>yes</td>
<td></td>
<td>minimal</td>
<td></td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>When the user right-clicks to get the context menu.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Firefox 2 fires too many events in a variety of circumstances.
- Firefox 3 fires too many events when blurring the window.
- Safari and Opera don't support these events on links and/or form fields in all circumstances.
- Konqueror doesn't support these events on the browser window.

- It has a serious bug in its handling of this event on checkboxes and radios.

Preventing the default (i.e. preventing the context menu from appearing) is the whole point of this event.
Web standards

W3C specifications for the three Web languages:

- HTML (structure; what does it mean?)
- CSS (presentation; how does it look?)
- JavaScript (behaviour; what does it do?)
Web standards

<h1>Hello, <span>ppk</span></h1>

HTML says: this is a page header (doesn't really have a meaning)
Web standards

<h1>Hello, <span>ppk</span></h1>

We want the text to be red (CSS) and we want the user name to appear when the user mouses over the header (JavaScript)
Web standards

<h1 style="color: red" onmouseover="appear()">Hello, <span>ppk</span></h1>
Web standards

This is the wrong way to apply CSS and JavaScript.
Web standards

Hello, <span>ppk</span></h1>

If we need a similar header on another page we have to copy all this code.
Web standards

Hello, <span>ppk</span>

and if we want to change the colour to blue, we have to go through all pages in the site and change it.
Web standards

Hello, <span>ppk</span>

Instead we use separation of concerns:

Separate files for our HTML, CSS, and JavaScript.
Web standards

<h1>Hello, <span>ppk</span></h1>

CSS:
```css
h1 {
    color: red
}
```

Ease of maintenance 1: Change the colour
Web standards

Hello, <span>ppk</span>

CSS:

```css
h1 {
  color: red blue;
}
```
Web standards

Hello, <span>ppk</span>

JavaScript:

```javascript
$('h1').onmouseover = appear;
```

Ease of maintenance 2: It's useful to add a focus event to every mouseover event.
Web standards

<h1>Hello, <span>ppk</span></h1>

$('<h1>').onmouseover = appear;
Web standards

Hello, <span>ppk</span></h1>

$('h1').onmouseover = $('h1').onfocus = appear;

Change one file, apply throughout your site.
Connect them

Hooks

CSS

HTML

JavaScript
Connect them

- id

```javascript
#id {
document.getElementById();
```
Connect them

- id

document.getElementById('special').onmouseover = appear;
Connect them

- id

```javascript
var el = document.getElementById('special');
if (el) {
  el.onmouseover = appear;
}

“Is this element available?”
```
Connect them

- id
- class

.getElementsByClassName();
or a library function
Connect them

- id
- class

```javascript
var els = document.getElementsByClassName('special')
if (els.length) {
    // go through all elements and do something
}
```
Connect them

- id
- class

Use the same hook for presentation and behavior; for CSS and JavaScript.
Connect them

<ol class="dropdown">

Now what would this <ol> be?

Surprise: it's a dropdown menu
Connect them

<ol class="dropdown">

The class is a hook for both layers.

ol.dropdown {
    // presentation layer
}
Connect them

The class is a hook for both layers.

```javascript
var dropdowns = $('dropdown');
if (dropdowns.length) {
    // initialize behavior layer
}
```
BE CAREFUL
THIS MACHINE HAS NO BRAIN
USE YOUR OWN
Some common myths about JavaScript
JavaScript is always available

Nonsense!
JavaScript is always available

- Primitive cell phones don't support it (sufficiently)
- Speech browsers' support may be spotty
- Company networks may filter out `<script>` tags
JavaScript is always available.

Make sure that the content and navigation of the site can be used without JavaScript.
JavaScript is always available

Make sure that the content and navigation of the site can be used without JavaScript.

The page will remain accessible in all circumstances.
JavaScript is always available

Make sure that the content and navigation of the site can be used without JavaScript.

You can use JavaScript for nice extras, though.
JavaScript is always available

However...

Without JavaScript the page will become less user-friendly.

Can't be helped.
JavaScript is always available

However...

Without JavaScript the page will become less user-friendly.

After all, the purpose of JavaScript is to add interactivity to a page.
Everybody uses a mouse

Nonsense!
Device independence
Take a dropdown menu:

```javascript
var dropdown = {
  setEventHandlers: function (obj) {
    obj.onmouseover = this.over;
    obj.onmouseout = this.out;
  },
  over: function () {
    // code
  },
  // etc
}
```
It doesn't work without a mouse.

```javascript
var dropdown = {
    setEventHandlers: function (obj) {
        obj.onmouseover = this.over;
        obj.onmouseout = this.out;
    },
    over: function () {
        // code
    },
    // etc
};
```
var dropdown = {
  setEventHandlers: function(obj) {
    obj.onmouseover = this.over;
    obj.onmouseout = this.out;
  },
  over: function () {
    // code
  },
  // etc
};

We need events that are fired when the user “enters” or “leaves” a link by using the keyboard.

focus and blur
var dropdown = {
  setEventHandlers: function (obj) {
    obj.onmouseover = obj.onfocus = this.over;
    obj.onmouseout = obj.onblur = this.out;
  },
  over: function () {
    // code
  },
  // etc
}
Restriction: the object must be able to gain the keyboard focus.

- links
- form fields
Restriction: the object must be able to gain the keyboard focus.

- links
- form fields
- elements with tabindex
And what about click?

We're in luck: the click event fires also when the user activates an element by the keyboard. 

click should be called activate.
Want to know more about web development techniques?

fronteers.nl

The organisation of web developers in Holland.
(I'm chairman)
IE 6

Released October 2001

- Decent CSS
- Excellent JavaScript

This allowed Microsoft to win the browser war.
IE 6

Unfortunately, Microsoft disbanded the IE team after the war had been won.

It assumed it would never have to create another browser.
The browser market back then

IE
NN4
Opera
Meanwhile, competition on the browser market has returned.

- Mozilla -> Firefox
- Safari (Apple)
- Opera
- more recently Google Chrome
The current browser market

IE
Chrome
FF
Safari
Opera
IE 6

The problem is that all other browsers supported JavaScript and especially CSS better than IE.

Web developers started to get fed up with IE6.
Microsoft decided to re-enter the browser game, re-formed the IE team and went to work. Nonetheless, they are a few years behind the competition now.
IE 6

Released October 2001
IE 7

Released October 2006

Five years between 6 and 7. That's the root cause of the current problems.
IE 7

Web developers were moderately enthusiastic.

IE7 solved many IE6 problems but not all of them.

The real gain was that IE was being improved again.
IE 7

Corporate clients of Microsoft were less enthusiastic.
Many Intranet applications are written for IE6 only and don't work in IE7.

(The error was creating these apps for only one browser.)
IE 7

Thus, clients either upgraded to IE7 and ran into serious problems or didn't upgrade at all. This is the backward compatibility problem.
IE 6 -> 7 adoption

Thus, IE 6 users were relatively sluggish in picking up IE7 partly for backward compatibility reason partly because many IT departments are slow in rolling out new software
IE 6 -> 7 adoption

IE6
IE7

2005 2006 2007 2008
IE 7

So for the future, the IE team had to
- increase standard compliance
- remain backward compatible

The problem is: these two goals are incompatible.
IE 8

Released March 2009

Standard compliance has been improved dramatically. (full CSS 2)

It is also backward compatible with IE7.
IE 8

The compatibility mode switch

Web authors can declare whether they want their sites to be shown in the new IE8 mode or the old IE7 mode. (IE6 mode is not available.)
IE 8

The compatibility mode switch

<meta http-equiv="X-UA-Compatible" content="IE=8" />
IE 8

The compatibility mode switch

<meta http-equiv="X-UA-Compatible" content="IE=8" />
IE 8

The compatibility mode switch

<meta http-equiv="X-UA-Compatible" content="IE=7" />
IE 8

The compatibility mode switch

\texttt{\textless{}meta http-equiv="X-UA-Compatible" content="IE=8" /\}}

If undefined: IE 8
Thus, web developers can use the enhanced standard compliance of IE8 or opt to keep their site frozen in IE7 mode.
IE 7 -> 8 adoption

With the backward compatibility problem partly solved, IE 8 will be adopted a bit faster than IE 7 but it will gain market share from IE 7, and not from IE 6.
IE 7 -> 8 adoption

- IE6
- IE7
- IE8
The IE landscape

Thus, web developers will have to cater to IE6, IE7, and IE8 and the first one to disappear will be IE7, so IE6 may have a long, long afterlife.
The IE landscape

However, optimising a normal website for IE6 costs about 20% to 30% extra time (and money).

I expect that web developers are actually going to charge this money, and that some clients will decide not to bother with IE6.
The IE landscape

Thus, IE6 will degrade to an Intranet browser for companies that haven't upgraded their Intranet in years while the companies' employees will use another browser for their daily surfing.
The IE landscape

And that other browser will not be IE because you can't install one IE version next to another (well, you can, but the average sysadmin won't like it)
IE 9?

I assume the IE team will continue to work on their browser, and I assume a future IE 9 will support still more CSS and JavaScript.
IE 9?

The compatibility switch will also remain available.

If you've locked your site in IE 8 mode, IE 9 will execute it in IE 8 mode, too.
IE 9?

The compatibility switch will also remain available.

If you do not use the switch at all, IE 9 will execute your site in IE 9 mode.
IE 9?

IE will continue to become better.

The only question is whether they'll overtake the other browsers.
When an element loses the focus.

- Firefox 2 fires too many events in a variety of circumstances.
- Firefox 3 fires too many events when blurring the window.
- Safari and Opera don’t support these events on links and/or form fields in all circumstances.
- Konqueror doesn’t support these events on the browser window.

When a form field value changes.

- IE has a serious bug in its handling of this event on checkboxes and radios.

When a mousedown and mouseup event occur on the same element OR an element is activated by the keyboard.

Preventing the default (i.e. preventing the context menu from appearing) is the whole point of this event.
Questions?

Ask away.

Or ask me on Twitter
http://twitter.com/ppk
or on my site
http://quirksmode.org

- Firefox 2 fires too many events in a variety of circumstances.
- Firefox 3 fires too many events when blurring the window.
- Safari and Opera don’t support these events on links and/or form fields in all circumstances.
- Opera doesn’t support these events on the browser window.

Preventing the default (i.e. preventing the context menu from appearing) is the whole point of this event.