JavaScript Events

Peter-Paul Koch (ppk)
http://quirksmode.org
http://twitter.com/ppk
Yahoo!, 23 April 2009

Hell is other browsers - Sartre
<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><strong>blur</strong></td>
<td>yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>too many</td>
<td>almost</td>
<td>incomplete</td>
<td>incomplete</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>Firefox 2 fires too many events in a variety of circumstances.</td>
<td></td>
<td>Safari and Opera don't support these events on links and/or form fields in all circumstances.</td>
<td></td>
<td>Firefox 3 fires too many events when blurring the window.</td>
</tr>
<tr>
<td><strong>change</strong></td>
<td>buggy</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td></td>
<td></td>
<td></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>IE has a serious bug in its handling of this event on checkboxes and radios.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>click</strong></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>
<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><strong>contextmenu</strong></td>
<td>yes</td>
<td>minimal</td>
<td>yes</td>
<td>buggy</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</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>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>
</tr>
</tbody>
</table>
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.
Today's program:

- the key events
- the change event
- delegating the focus event
- first results of mobile event tests
The key events
keydown
When a key is depressed.
Repeats.
keypress
keyup
keydown
  When a key is depressed.
  Repeats.
keypress
  When a character key is depressed.
  Repeats.
keyup
keydown
When a key is depressed.
Repeats.

keypress
When a character key is depressed.
Repeats.

keyup
When a key is released.
keydown and keypress
keydown only
Originally this theory was created by Microsoft.

Safari has copied it.

It's the only theory; Firefox and Opera just fire some random events.
keydown
When a key is depressed.
Repeats.

keypress
When a character key is depressed.
Repeats.
Which key did my user press?

Two properties: 
keyCode and charCode

Two bits of data:  
- the key code  
- the character code
Which key did my user press?

Obviously, having one property contain one bit of data and the other property the other would be far too simple.
Which key did my user press?

keyCode

- onkeydown: key code
- onkeypress: character code
Which key did my user press?

charCode

- onkeydown: 0
- onkeypress: character code
Which key did my user press?

If you need the key:

```javascript
el.onkeydown = function (e) {
    e = e || window.event;
    var realKey = e.keyCode;
}
```
Which key did my user press?

If you need the key:

el.onkeydown = function (e) {
  e = e || window.event;
  var realKey = e.keyCode;
}

Which key did my user press?

If you need the character:

```javascript
el.onkeypress = function (e) {
  e = e || window.event;
  var char = e.keyCode || e.charCode;
}
```
Which key did my user press?

If you need the character:

```javascript
el.onkeypress = function (e) {
  e = e || window.event;
  var char = e.keyCode || e.charCode;
}
```
How can I prevent the default action?

e1.onkeydown = function (e) {
  e = e || window.event;
  var key = e.keyCode;
  if (key is incorrect) {
    // cancel default action
  }
}
How can I prevent the default action?

el.onkeydown = function (e) {
  e = e || window.event;
  var key = e.keyCode;
  if (key is incorrect) {
    // cancel default action
  }
}
<table>
<thead>
<tr>
<th>Registration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traditional</td>
</tr>
<tr>
<td>addEventListener (capture)</td>
</tr>
<tr>
<td>addEventListener (bubble)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>The following events are registered:</td>
</tr>
<tr>
<td>change</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>The events are registered on the following elements:</td>
</tr>
<tr>
<td>change</td>
</tr>
<tr>
<td>form</td>
</tr>
<tr>
<td>text</td>
</tr>
<tr>
<td>checkbox</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Miscellaneous</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prevent default action</td>
</tr>
<tr>
<td>Cancel bubble</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Event properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show event properties</td>
</tr>
</tbody>
</table>
The change event fires when the value of a form field is changed.

This could be a very useful event; after all it fires only when the user actually changes something instead of whenever he focuses on a form field.
- text fields
- select boxes
- checkboxes and radios
- text fields
- select boxes
- checkboxes and radios

No change event. The value hasn't been modified.
- text fields
- select boxes
- checkboxes and radios

Focus

Blur

Change event. The value has been modified.
- text fields
- select boxes
- checkboxes and radios

Mouse:

Click on select
- text fields
- select boxes
- checkboxes and radios

Mouse:

Click on new option

CHANGE
- text fields
- select boxes
- checkboxes and radios

Keyboard:

Focus on select
- text fields
- select boxes
- checkboxes and radios

Keyboard:

focus
arrow

Arrow keys to move to other option
CHANGE
This is a BUG!

Arrow keys to move to other option

CHANGE
- text fields
- select boxes
- checkboxes and radios

Keyboard:

focus  arrow

Arrow keys to move to other option
- text fields
- select boxes
- checkboxes and radios

Keyboard:

focus  arrow  blur

Blur select box.

CHANGE
- text fields
- select boxes
- checkboxes and radios

click

CHANGE when the checked property changes.
- text fields
- select boxes
- checkboxes and radios

click

...
- text fields
- select boxes
- checkboxes and radios

click  blur

**CHANGE** when the element loses the focus.
- text fields
- select boxes
- checkboxes and radios

**This is a BUG!**

**CHANGE** when the element loses the focus.
Event delegation
Event delegation

<ul>
  <li><a href="#">Multimedialize</a></li>
  <ul>
    <li><a href="#">Sound</a></li>
    <li><a href="#">Java applets</a></li>
  </ul>
  <li><a href="#">Ajaxify</a></li>
  <ul>
    <li><a href="#">Web 2.0</a></li>
    <li><a href="#">Web 3.0</a></li>
    <li><a href="#">Web 4.0b</a></li>
  </ul>
</ul>
Event delegation

Event bubbling

Multimedialize

Ajaxify
Event delegation

The event bubbles up to the `<ul>` anyway.

So why not handle it at that level?

Saves a lot of event handlers.
Event delegation

```javascript
var dropdown = {
  init: function (dropdown) {
    dropdown.onmouseover = mouseOver;
    dropdown.onmouseout = mouseOut;
  }
}
```

Works in all browsers.
Event delegation

But suppose someone doesn't use a mouse at all,

but the keyboard

how does the menu fold out?
Device independence
Event delegation

```javascript
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
    }
}
```
Event delegation

```javascript
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
    }
}
```

Doesn't work without a mouse.
Event delegation

```javascript
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
    }
};
```

We need events that tell us whether the user enters or leaves a link. focus and blur
Event delegation

```javascript
var dropdown = {
  init: function (dropdown) {
    dropdown.onmouseover = dropdown.onfocus = this.mouseOver;
    dropdown.onmouseout = dropdown.onblur = this.mouseOut;
  }
}
```
Event delegation

```javascript
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = dropdown.onmouseout = this.mouseOver;
        dropdown.onfocus = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
    }
}

Doesn't work.
```
Event delegation

```javascript
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = dropdown.onfocus = this.mouseOver;
        dropdown.onmouseout = dropdown.onblur = this.mouseOut;
    }
}

Focus and blur don't bubble.
```
To bubble or not to bubble

Two kinds of events:
1) Mouse and key events
2) Interface events
To bubble or not to bubble

Two kinds of events:
1) Mouse and key events
2) Interface events

Fire when the user initiates a device-specific action.
mouseover, mouseout, click, keydown, keypress
To bubble or not to bubble

Two kinds of events:
1) Mouse and key events
2) Interface events

In general they bubble.
To bubble or not to bubble

Two kinds of events:
1) Mouse and key events
2) Interface events

Fire when a certain event takes place, regardless of how it was initialised. load, change, submit, focus, blur
To bubble or not to bubble

Two kinds of events: 
1) Mouse and key events
2) Interface events

In general they don't bubble.
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
        var x = dropdown.getElementsByTagName('a');
        for (var i=0; i<x.length; i++) {
            x[i].onfocus = this.mouseOver;
            x[i].onblur = this.mouseOut;
        }
    }
}
Event delegation

So we're stuck with setting a focus and blur event on every single link.

Or are we ... ?
Event delegation

Event capturing to the rescue.

Event capturing is the opposite of event bubbling, and it is supported in all W3C-compliant browsers.
Event bubbling

```javascript
addEventListener('click', fn, false)
```
Event capturing

`addEventListener('click', fn, true)`
Event delegation

The point is that if you capture a focus event, event handlers on the target element's ancestors are executed.
Event bubbling
Focus on a:
a.onfocus executed

Event bubbling

Multimedialize

Ajaxify
Event capturing
Focus on a: ul.onfocus, li.onfocus and a.onfocus executed

Event capturing
Multimedialize

Ajaxify
Event capturing and bubbling

Why this difference?

I have no idea.

Maybe it's time to have bubbling and capturing act the same. The current situation doesn't make sense (though it's useful).
Event delegation

```javascript
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
    }
};
```
Event delegation

```javascript
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
        dropdown.addEventListener('focus',this.mouseOver true);
        dropdown.addEventListener('blur',this.mouseOut true);
    }
}
```
Event delegation

```javascript
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
        if (dropdown.addEventListener) {
            dropdown.addEventListener('focus', this.mouseOver, true);
            dropdown.addEventListener('blur', this.mouseOut, true);
        }
    }
};
```
Event delegation

And what about IE?

It doesn't support addEventListener, but fortunately it supports the focusin and focusout events which are like focus and blur, except that they bubble.
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
        if (dropdown.addEventListener) {
            dropdown.addEventListener
                ('focus', this.mouseOver, true);
            dropdown.addEventListener
                ('blur', this.mouseOut, true);
        }
    }
};
Event delegation

var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
        if (dropdown.addEventListener) {
            dropdown.addEventListener('focus',this.mouseOver,true);
            dropdown.addEventListener('blur',this.mouseOut,true);
        }
        dropdown.onfocusin = this.mouseOver;
        dropdown.onfocusout = this.mouseOut
    }
}
Event delegation

dropdown.onmouseover = this.mouseOver;
dropdown.onmouseout = this.mouseOut;
if (dropdown.addEventListener) {
    dropdown.addEventListener('focus', this.mouseOver, true);
    dropdown.addEventListener('blur', this.mouseOut, true);
}
dropdown.onfocusin = this.mouseOver;
dropdown.onfocusout = this.mouseOut;
Device independence
Mobile phones

The Mobile Web is finally coming, and thanks to Vodafone I'm able to give you some information, although the subject remains a tricky one.
Mobile phones – input modes

On mobile phones there are three input modes:
- Touch
- Cursor (or rather, pseudo-cursor)
- Four-way navigation ("arrow" keys)
Opera Mini 4.2 on Nokia E71

Pseudo-cursor input mode

Does the browser handle the sortable table?

Opera has some problem handling the re-sort, but this is slow to appear, and in and out of existence a tweaking is clearly in order. Oddly, the Opera 8.00 on just fine; no problems.
Opera Mobile 8.00 on Motorola V3xx

Four-way navigation
NetFront on Sony Ericsson K770i

Four-way navigation, but which link do you follow when you click?
Mobile phones – events

In such an environment, what does “mouseover” mean?

And what about mouseout, mousemove, mousedown, mouseup?

And click?
Mobile phones – events

I set up a test in which I “click” on a `<div>` element and see which events take place.

First some good news:
S60 WebKit on Nokia E71
Input: cursor

The same as desktop browsers
Opera Mobile 9.5 on HTC Touch Diamond
Input: touch

This is the same as the iPhone does.
Mobile phones – events

So Nokia cursor phones behave exactly as desktop computers, while the latest Opera behaves exactly as the iPhone.
Mobile phones – events

iPhone/Opera model:

As soon as the user touches an element mouseover,mousemove, mousedown, mouseup, and click fire, and the :hover styles are applied.

When user touches another element, mouseout fires and :hover styles are removed
Mobile phones – events

Now for some bad news.
Blackberry Storm
Input: touch

No mouseover, -out, or -move
NetFront on Samsung F700
Input: touch

Where's the click?
Mobile phones – events

These are only 4 of the 19 browsers I tested, and there are hundreds of browsers on mobile phones.
Mobile phones – events

General rules for events on mobile phones 1/3:
- use click (and let's politely but firmly give the finger to browsers that do not support it)
- forget about the mouse events

In fact, I think the time has come to retire the mouse events on all devices (with a few exceptions)
Mobile phones – events

General rules for events on mobile phones 2/3:
- use the resize AND the orientationchange event

orientationchange is supported only by iPhone and Blackberry
resize is supported by Opera and all WebKits
NetFront doesn't support either – too bad
Mobile phones – events

General rules for events on mobile phones 3/3:
- use key events only for setting general access keys;
and NOT for reading out user input in a form field

You should read out the field's value instead.
Event compatibility for desktop:

- 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 on form fields in all circumstances.
  Konqueror doesn’t support these events on the browser window.

Mobile compatibility (work in progress)

http://quirksmode.org/m/

Preventing the default (i.e. preventing the context menu from appearing) is the whole point of this event.
Thank you for your attention

<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>yes</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>
</tbody>
</table>

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.

<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>change</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

When a form field value changes:

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

<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>ch</td>
<td>yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

<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>contextmenu</td>
<td>yes</td>
<td>minimal</td>
<td>yes</td>
<td>buggy</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

When the user right-clicks to get the context menu.

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