JavaScript Events

Peter-Paul Koch (ppk)
http://quirksmode.org
http://twitter.com/ppk
Voices that Matter, 28 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></td>
<td></td>
<td></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>- 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>
<td></td>
</tr>
<tr>
<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>
<td></td>
</tr>
<tr>
<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>
<td></td>
</tr>
<tr>
<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>
<td></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></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<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>
<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>no</td>
<td>no</td>
<td>no</td>
<td></td>
<td></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>
<tr>
<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>
<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>almost</td>
<td>incomplete</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.

<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>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>
</tbody>
</table>

- 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>click</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>
</tbody>
</table>

<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></td>
<td></td>
<td>no</td>
<td>no</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.

http://quirksmode.org/dom/events/
Mouseover
and friends
The mouseover event fires when the user's mouse enters an element.

The mouseout event fires when the user's mouse leaves an element.

Perfect support
Dropdown menu <sigh />

<ul>
  <li><a href=”#”>Multimedialize</a>
    <ul>
      <li><a href=”#”>Sound</a></li>
      <li><a href=”#”>Java applets</a></li>
    </ul>
  </li>
  <li><a href=”#”>Ajaxify</a>
    <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>
  </li>
</ul>
Dropdown menu <sigh />

- Multimedialize
- Ajaxify
Dropdown menu `<sigh />`

```
<ul>
    <li>
        <a>
            Multimedialize
        </a>
    </li>
    <li>
        <a>
            Ajaxify
        </a>
    </li>
</ul>
```

Event bubbling
Event bubbling has advantages.

```javascript
var dropdown = {
    init: function (dropdown) {
        var x = dropdown.getElementsByTagName('a');
        for (var i=0;i<x.length;i++) {
            x[i].onmouseover = mouseOver;
            x[i].onmouseout = mouseOut;
        }
    }
}
```
Dropdown menu <sigh />

Event bubbling has advantages.

```javascript
var dropdown = {
    init: function (dropdown) {
        var x = dropdown.getElementsByTagName('a');
        for (var i=0; i<x.length; i++) {
            x[i].onmouseover = mouseOver;
            x[i].onmouseout = mouseOut;
        }
    }
}
```
Dropdown menu <sigh />

Event bubbling has advantages.

```javascript
var dropdown = {
  init: function (dropdown) {
    We don't do this any more. Instead we use event delegation.
  }
}
```
Dropdown menu <sigh />

The event bubbles up to the `<ul>` anyway. So why not handle it at that level?

Saves a lot of event handlers.
Dropdown menu <sigh />

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

Works in all browsers.
Problem: Every mouseover or mouseout event bubbles up.
Dropdown menu <sigh />

Fun!

Event bubbling works.

As does event delegation.
Dropdown menu <sigh />

amouseover
amouseout and amouseover
amouseout and amouseover
amouseout

But has the mouse left the submenu or not?!
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
    },
    mouseOut: function (e) {
        if (this.mouseout is important) {
            this.closeSubSubMenu();
        }
    }
};
Dropdown menu <sigh />

```javascript
var dropdown = {
  init: function (dropdown) {
    dropdown.onmouseover = this.mouseOver;
    dropdown.onmouseout = this.mouseOut;
  },
  mouseOut: function (e) {
    if (this mouseout is important) {
      this.closeSubSubMenu();
    }
  }
}
```

Development time: about 10 minutes
Dropdown menu <sigh />

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

  mouseOut: function (e) {
    if (this mouseout is important) {
      this.closeSubMenu();
    }
  }
}
```

Development time: about 2 days
Dropdown menu <sigh />

How do we do this?

onmouseout, find out which element the mouse goes to.

If that element is not a part of the submenu, fold the submenu.
Dropdown menu <sigh />

How do we do this?

```javascript
mouseOut: function (e) {
    e = e || window.event;
    var el = e.relatedTarget || e.toElement;
    if (!submenu.contains(el)) {
        this.closeSubMenu();
    }
}
```
Find the element the mouse goes to.

mouseOut: function (e) {
  e = e || window.event;
  var el = e.relatedTarget || e.toElement;
  if (!submenu.contains(el)) {
    this.closeSubMenu();
  }
}
Dropdown menu <sigh />

Find the element the mouse goes to.

```javascript
mouseOut: function (e) {
    e = e || window.event;
    var el = e.relatedTarget || e.toElement;
    if (!submenu.contains(el)) {
        this.closeSubMenu();
    }
}
```
Dropdown menu <sigh />

Find the element the mouse goes to.

mouseOut: function (e) {
    e = e || window.event;
    var el = e.relatedTarget || e.toElement;
    if (!submenu.contains(el)) {
        this.closeSubMenu();
    }
}
Dropdown menu <sigh />

See whether that element is contained by the submenu.

```javascript
mouseOut: function (e) {
  e = e || window.event;
  var el = e.relatedTarget || e.toElement;
  if (!submenu.contains(el)) {
    this.closeSubMenu();
  }
}
```
Dropdown menu <sigh />

See whether that element is contained by the submenu.

```javascript
mouseOut: function (e) {
  e = e || window.event;
  var el = e.relatedTarget || e.toElement;
  if (!submenu.contains(el)) {
    this.closeSubMenu();
  }
}
```
Dropdown menu <sigh />

That's it, right?

<grin type="evil" />

mouseOut: function (e) {
    e = e || window.event;
    var el = e.relatedTarget || e.toElement;
    if (!submenu.contains(el)) {
        this.closeSubMenu();
    }
}
Wrong!

Suppose someone doesn't use a mouse at all,

but the keyboard

how does the menu fold out?
Device independence
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
    }
};
Dropdown menu <sigh />

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

Doesn't work without a mouse.
Dropdown menu <sigh />

```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
Dropdown menu <sigh />

```javascript
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = dropdown.onfocus = this.mouseOver;
        dropdown.onmouseout = dropdown.onblur = this.mouseOut;
    }
}
```
Dropdown menu <sigh />

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

Doesn't work.
```
Dropdown menu <sigh />

```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

Generally don't bubble
Dropdown menu <sigh />

```javascript
var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
    }
}
```
var dropdown = {
  init: function (dropdown) {
    dropdown.onmouseover = this.mouseOver;
    dropdown.onmouseout = this.mouseOut;
    var x = dropdown.getElementsByTagName('li');
    for (var i=0; i<x.length; i++) {
      x[i].onfocus = this.mouseOver;
      x[i].onblur = this.mouseOut;
    }
  }
};
Dropdown menu <sigh />

```javascript
var dropdown = {
  init: function (dropdown) {
    dropdown.onmouseover = this.mouseOver;
    dropdown.onmouseout = this.mouseOut;
    var x = dropdown.getElementsByTagName('li');
    for (var i=0;i<x.length;i++) {
      x[i].onfocus = this.mouseOver;
      x[i].onblur = this.mouseOut;
    }
  }
}

Doesn't work.
```
Dropdown menu <sigh />

The HTML elements must be able to receive the keyboard focus.

- links
- form fields
The HTML elements must be able to receive the keyboard focus.

- links
- form fields
- elements with tabindex
Dropdown menu <sigh />

var dropdown = {
    init: function (dropdown) {
        dropdown.onmouseover = this.mouseOver;
        dropdown.onmouseout = this.mouseOut;
        var x = dropdown.getElementsByTagName('li');
        for (var i=0;i<x.length;i++) {
            x[i].onfocus = this.mouseOver;
            x[i].onblur = this.mouseOut;
        }
    }
}
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;
        }
    }
}
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 ... ?

In my recent Yahoo! presentation I give an outline of the solution.

And what about click?

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

Restriction: the element must be able to receive the keyboard focus
Separate concepts

Drag-and-drop uses the mousemove event
Separate concepts

Drag-and-drop uses the mousemove event

and if there's one thing that's impossible to emulate with the keyboard

it's moving the mouse
Separate concepts

Drag-and-drop uses the mousemove event

How do we make this keyboard accessible?

By allowing the user to use the arrow keys.

Key events.
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?

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

Which key did my user press?

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

Back to the drag-and-drop
We need the keydown event, because arrow keys are special keys.
Separate concepts

Drag-and-drop

```javascript
obj.onmousemove = moveElement;
obj.onkeydown = moveElement;
```
Separate concepts

Drag-and-drop

```javascript
obj.onmousemove = moveElement;
obj.onkeydown = moveElement;
```

Doesn't work.
Separate concepts

Drag-and-drop

```javascript
obj.onmousemove = moveElement;
obj.onkeydown = moveElement;
```

mousemove expects mouse coordinates.
The layer moves to these coordinates.
Separate concepts

Drag-and-drop

\begin{verbatim}
obj.onmousemove = moveElement;
obj.onkeydown = moveElement;
\end{verbatim}

The key events expect a keystroke.
```html
obj.onkeydown = function (e) {
  e = e || window.event;
  var key = e.keyCode;
  switch (key) {
    case 37: // left
    case 38: // up
    case 39: // right
    case 40: // down
      return false;
    default:
      return true;
  }
}
```
Separate concepts

But what does “user hits right arrow once” mean?

10px?

50px?

“Move to next receptor element?”

Something else that fits your interface?
Separate concepts

Drag-and-drop

We have to program for two totally different situations. We need separate scripts.

```javascript
obj.onmousemove = moveByMouseMove;
obj.onkeydown = moveByByKeys;
```
Separate concepts

Drag-and-drop

Yes, that's more work.

But if you do it right you've got a generic drag and drop module you can use anywhere.
Separate concepts

Drag-and-drop

Besides, I created a first draft for you.

http://quirksmode.org/js/dragdrop.html
Registration
- Traditional
- addEventListener (capture)
- addEventListener (bubble)

Events
The following events are registered:
- change

Elements
The events are registered on the following elements:
- window
- #document
- form
- text
- checkbox

Miscellaneous
- Prevent default action
- Cancel bubble

Event properties
- Show event properties
Registration
- Traditional
- addEventListener (capture)
- addEventListener (bubble)

Events
The following events are registered:
- change

Elements
The events are registered on the following elements:
- form
- text
- checkbox

Miscellaneous
- Prevent default action
- Cancel bubble

Event properties
- Show event properties
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
- text fields
- select boxes
- checkboxes and radios

This is a BUG!

Arrow keys to move to other option

CHANGE
- text fields
- select boxes
- checkboxes and radios

Keyboard:

Focus: Option 1
Arrow: Option 2

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

Keyboard:

**Option 1**

**Option 2**

**Option 2**

Focus

Arrow

Blur

Blur select box.

CHANGE

W3C
- text fields
- select boxes
- checkboxes and radios

Click

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

The following events are registered:
- change

Elements
- The events are registered on the following elements:
  - window
  - #document
  - form
  - text
  - checkbox

Miscellaneous
- Prevent default action
- Cancel bubble
- text fields
- select boxes
- checkboxes and radios

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.
<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></td>
<td></td>
<td><strong>too many</strong></td>
<td><strong>almost</strong></td>
<td>incomplete</td>
<td>incomplete</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>buggy</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</td>
<td>yes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></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>click</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td>yes</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></td>
<td>buggy</td>
<td>yes</td>
<td></td>
<td></td>
<td></td>
<td>no</td>
<td>no</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