

Hell is other browsers - *Sartre*

JavaScript Events

Peter-Paul Koch (ppk)

<http://quirksmode.org>

<http://twitter.com/ppk>

Voices that Matter, 28 April 2009

Event	IE 5.5	IE 6	IE 7	IE8b1	FF 2	FF 3b5	Saf 3.0 Win	Saf 3.1 Win	Opera 9.26	Opera 9.5b	Konqueror 3.5.7
<p>blur</p> <p>When an element loses the focus.</p> <ul style="list-style-type: none"> 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. 	yes	yes	yes	yes	too many	almost	incomplete	almost	incomplete	incomplete	incomplete
<p>change</p> <p>When a form field value changes.</p> <ul style="list-style-type: none"> IE has a serious bug in its handling of this event on checkboxes and radios. 	buggy	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
<p>click</p> <p>When a mousedown and mouseup event occur on the same element OR an element is activated by the keyboard.</p>	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
Event	IE 5.5	IE 6	IE 7	IE8b1	FF 2	FF 3b5	Saf 3.0 Win	Saf 3.1 Win	Opera 9.26	Opera 9.5b	Konqueror 3.5.7
<p>contextmenu</p> <p>When the user right-clicks to get the context menu.</p> <p>Preventing the default (i.e. preventing the context menu from appearing) is the whole point of this event.</p>	yes	yes	minimal	yes	yes	buggy	yes	yes	no	no	no

Event	IE 5.5	IE 6	IE 7	IE8b1	FF 2	FF 3b5	Saf 3.0 Win	Saf 3.1 Win	Opera 9.26	Opera 9.5b	Konqueror 3.5.7
<u>blur</u>		yes			too many	almost	incomplete	almost	incomplete		incomplete

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.

<u>change</u>	buggy				yes		yes		yes		yes
---------------	-------	--	--	--	-----	--	-----	--	-----	--	-----

When a form field value changes.

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

<http://quirksmode.org/dom/events/>

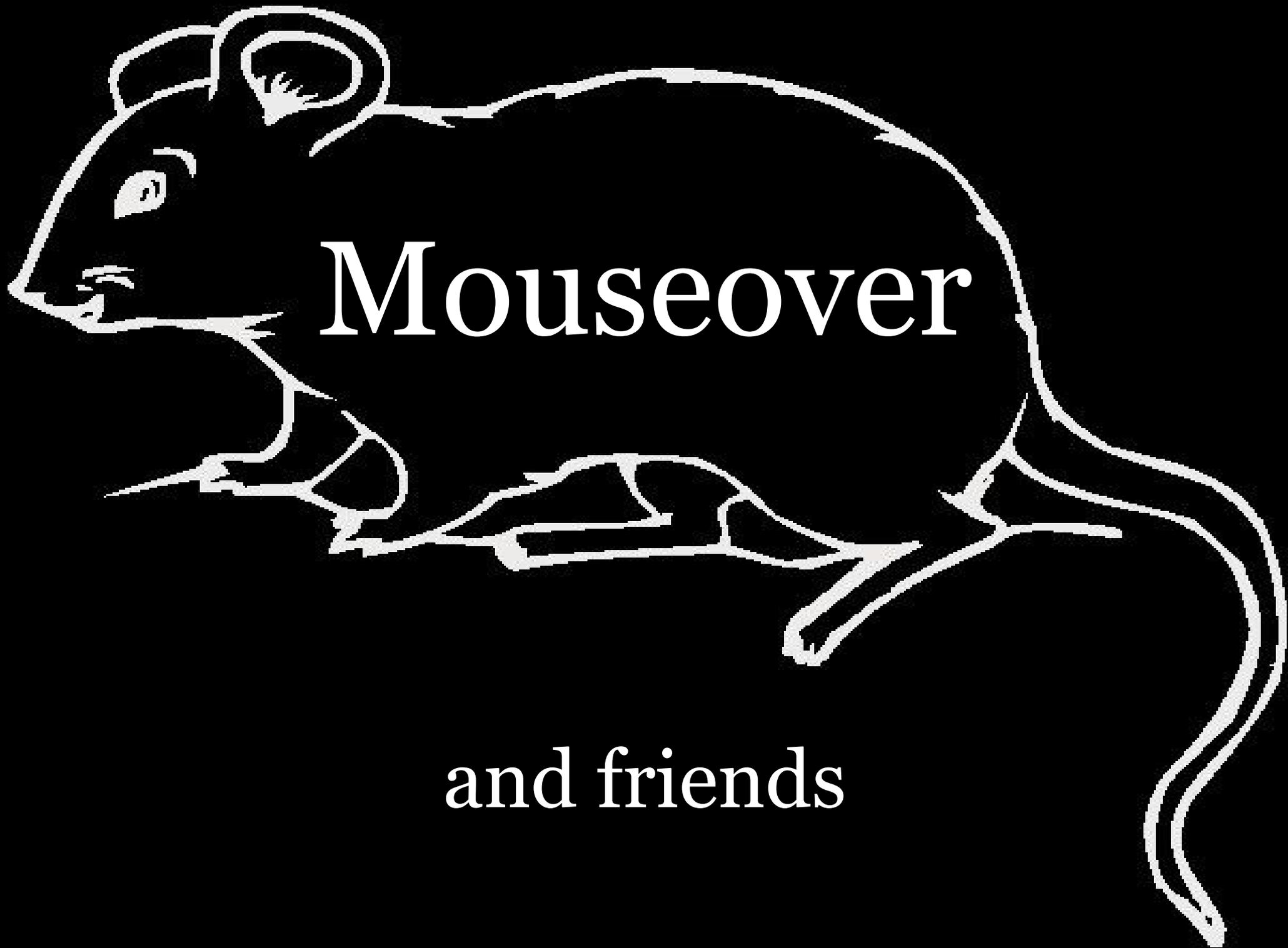
<u>click</u>	yes				yes		yes		yes		yes
--------------	-----	--	--	--	-----	--	-----	--	-----	--	-----

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

Event	IE 5.5	IE 6	IE 7	IE8b1	FF 2	FF 3b5	Saf 3.0 Win	Saf 3.1 Win	Opera 9.26	Opera 9.5b	Konqueror 3.5.7
<u>contextmenu</u>		yes		minimal	yes	buggy	yes		no		no

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.



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.

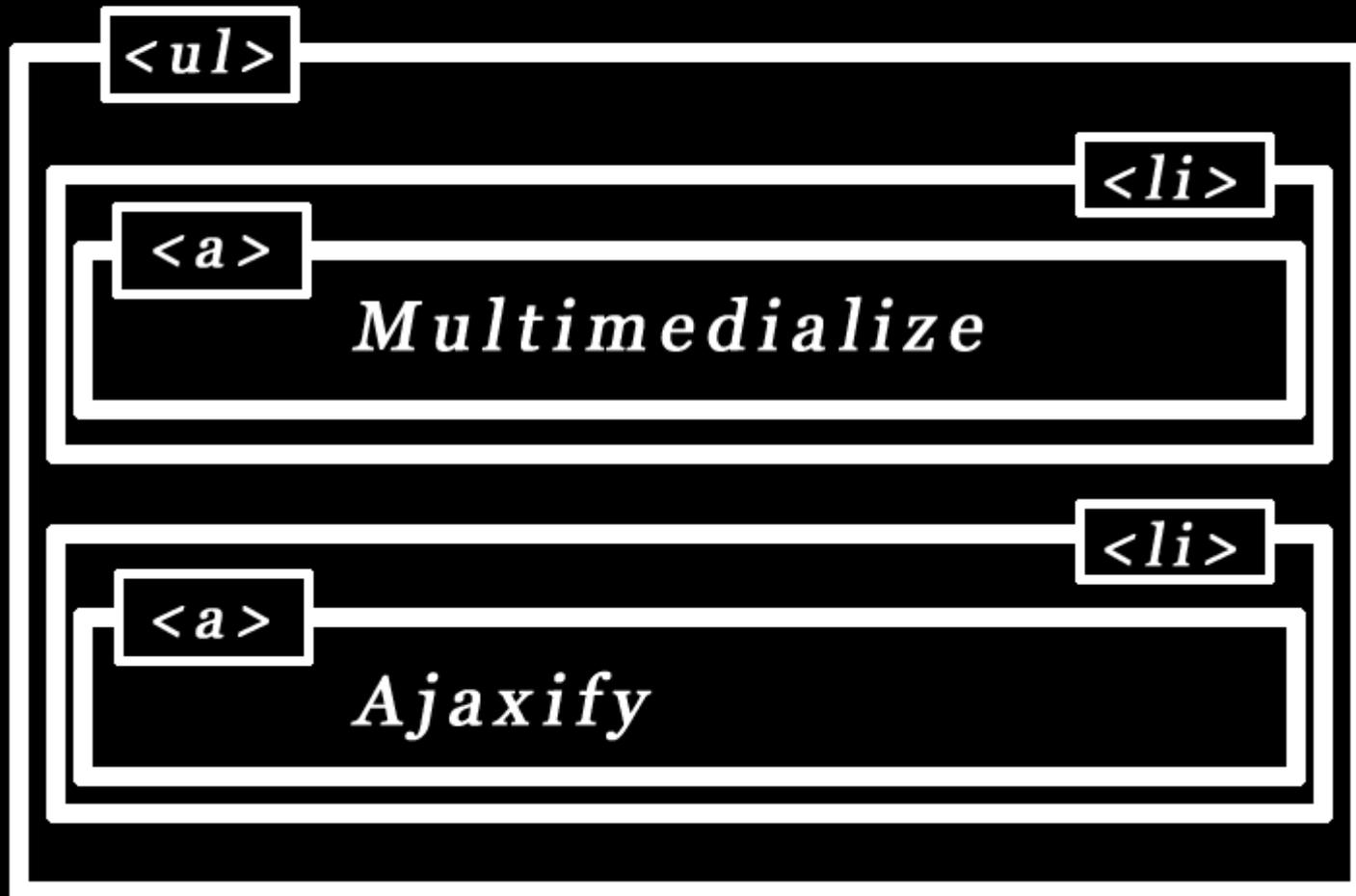
Perfect support

The logo for the World Wide Web Consortium (W3C), consisting of the letters 'W3C' in a blue, sans-serif font.

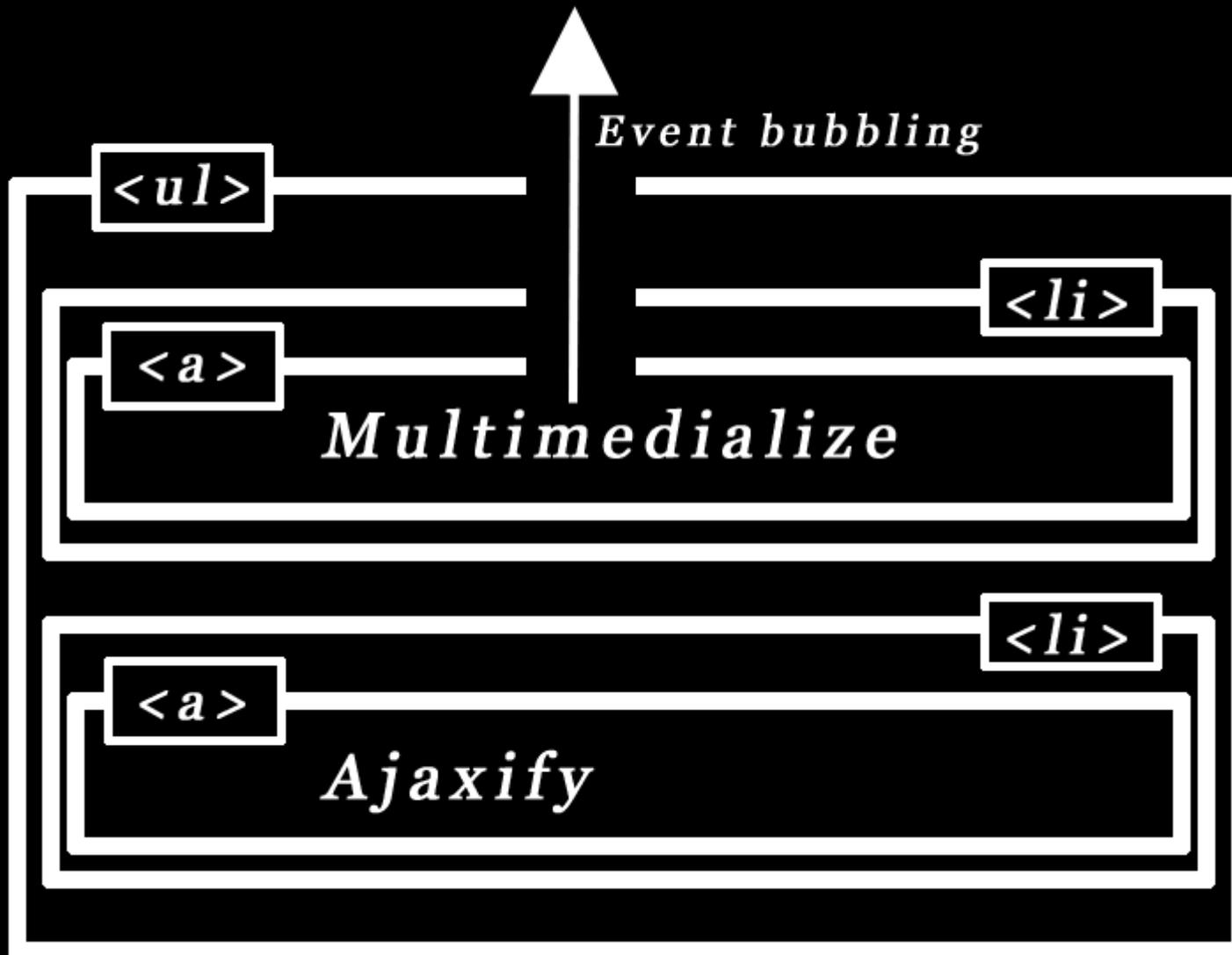
Dropdown menu <sigh />

```
<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 />



Dropdown menu <sigh />



Dropdown menu <sigh />

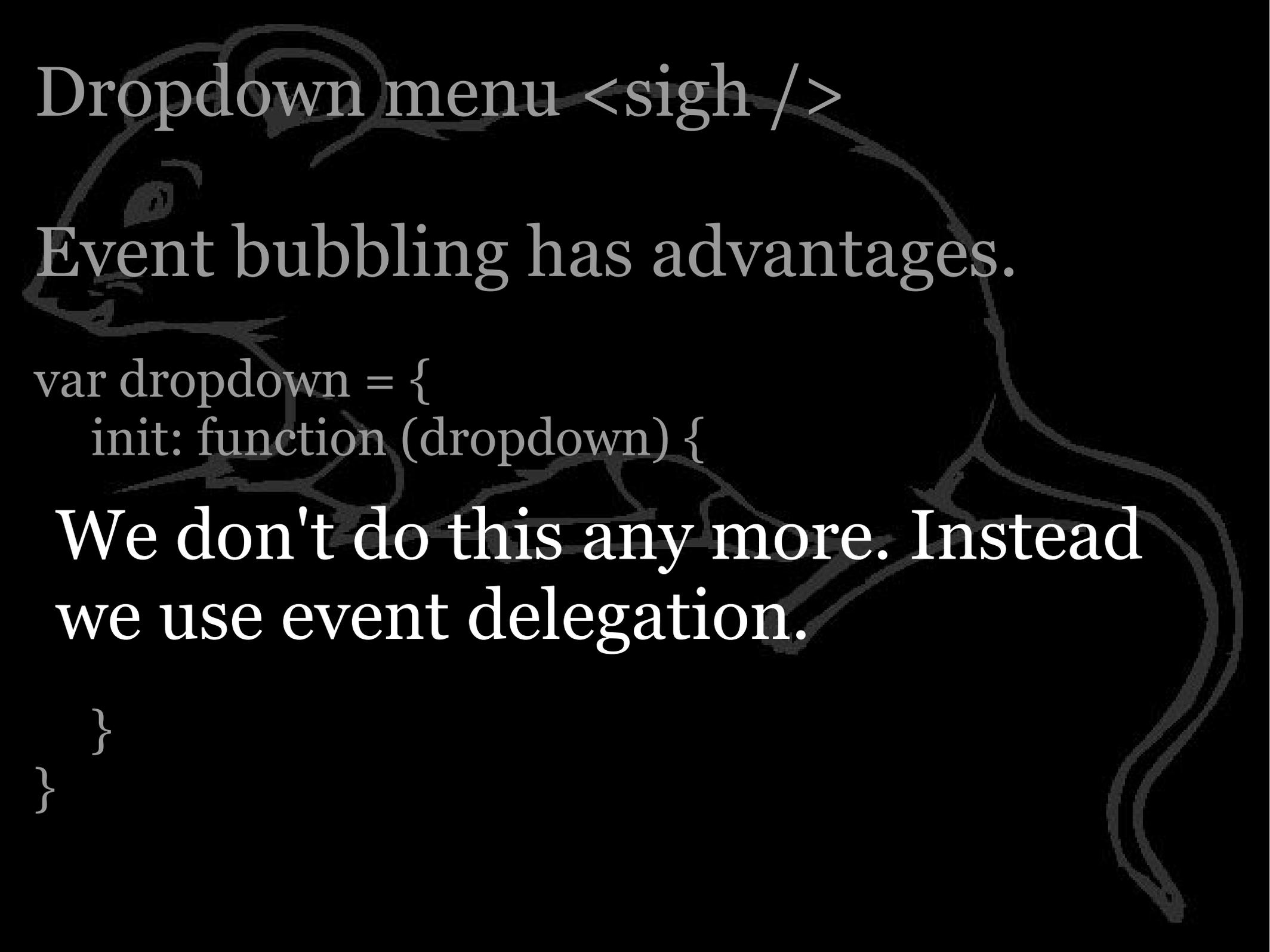
Event bubbling has advantages.

```
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.

```
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.

```
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
anyway.

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

So why not handle it at that level?

Saves a lot of event handlers.

Dropdown menu <sigh />

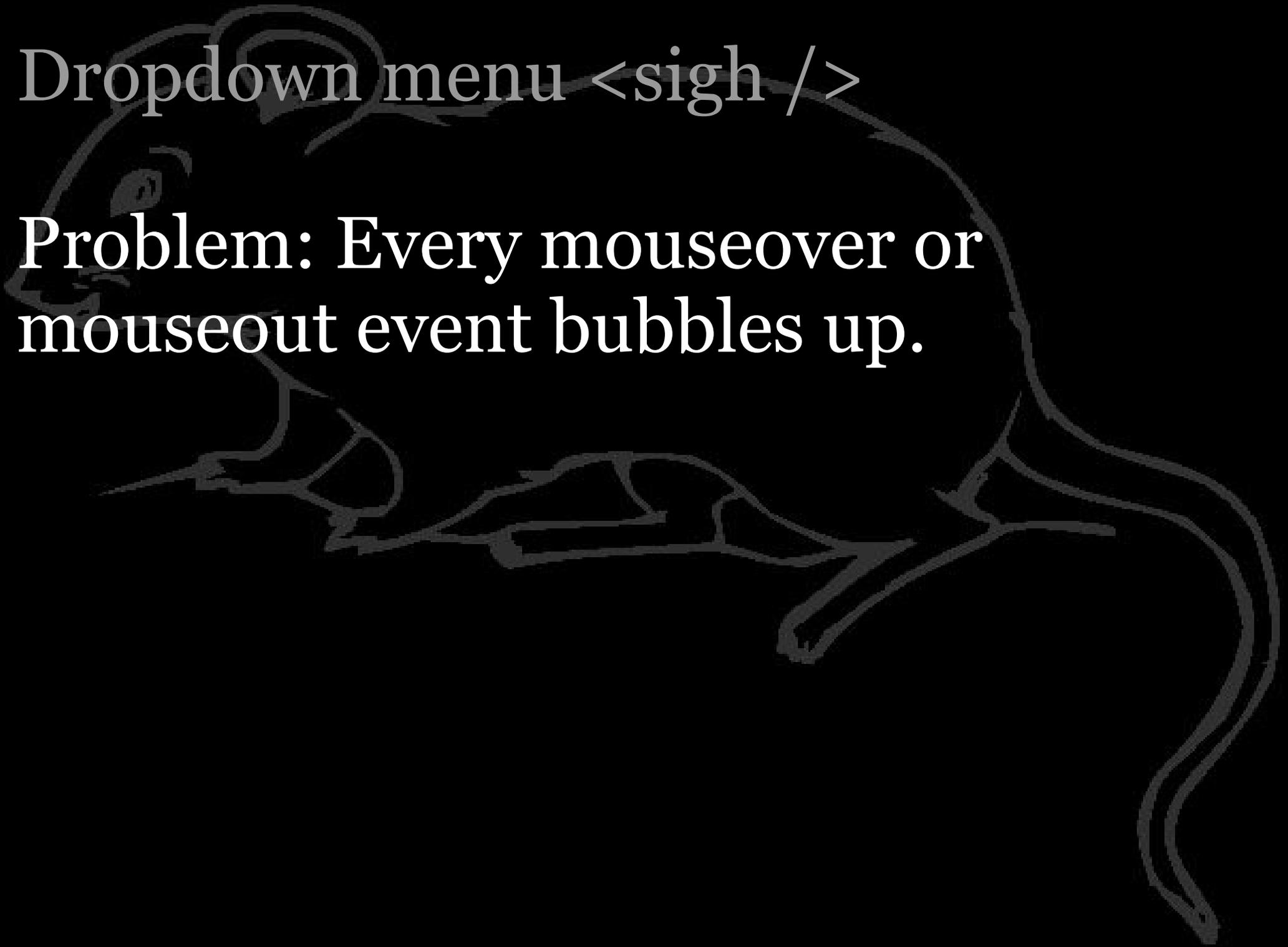
```
var dropdown = {  
  init: function (dropdown) {  
    dropdown.onmouseover = mouseOver;  
    dropdown.onmouseout = mouseOut;  
  }  
}
```

Works in all browsers.

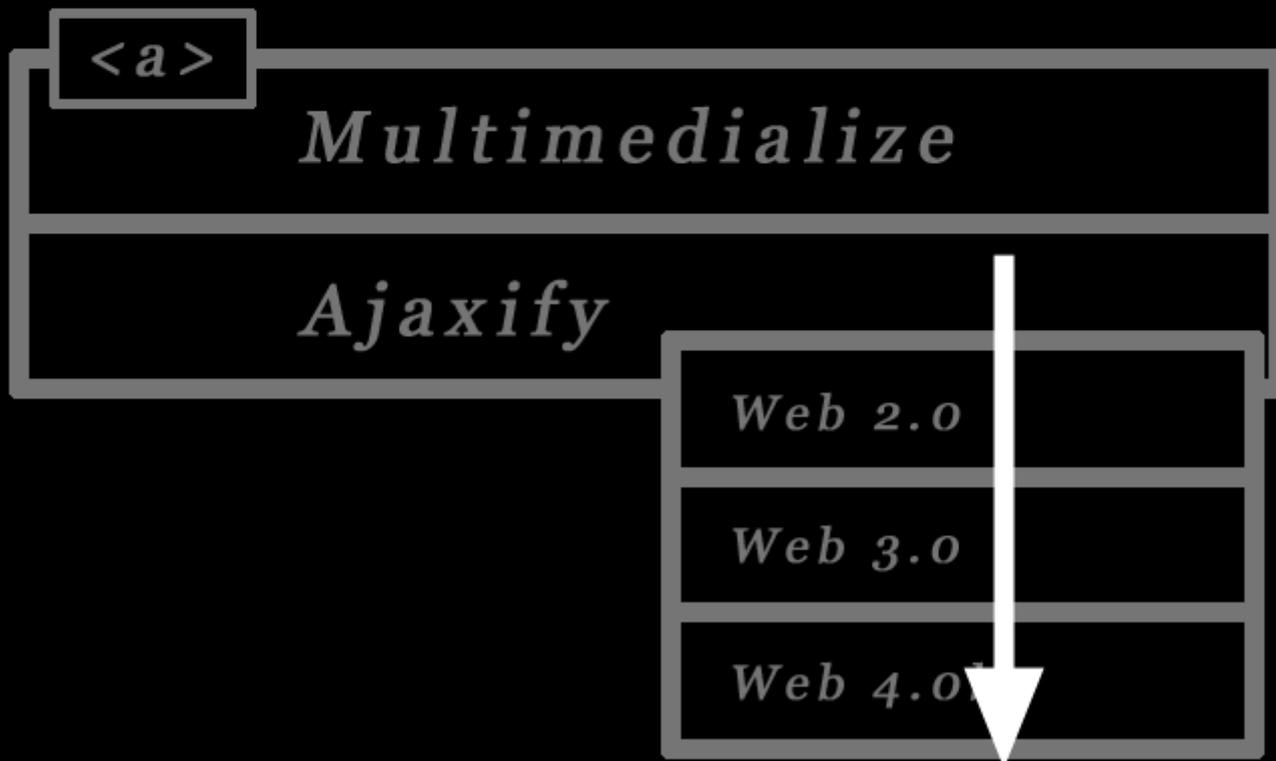
The logo for the World Wide Web Consortium (W3C), consisting of the letters 'W3C' in a blue, sans-serif font.

Dropdown menu <sigh />

Problem: Every mouseover or mouseout event bubbles up.



Dropdown menu <sigh />



Dropdown menu <sigh />

a.mouseover

a.mouseout and a.mouseover

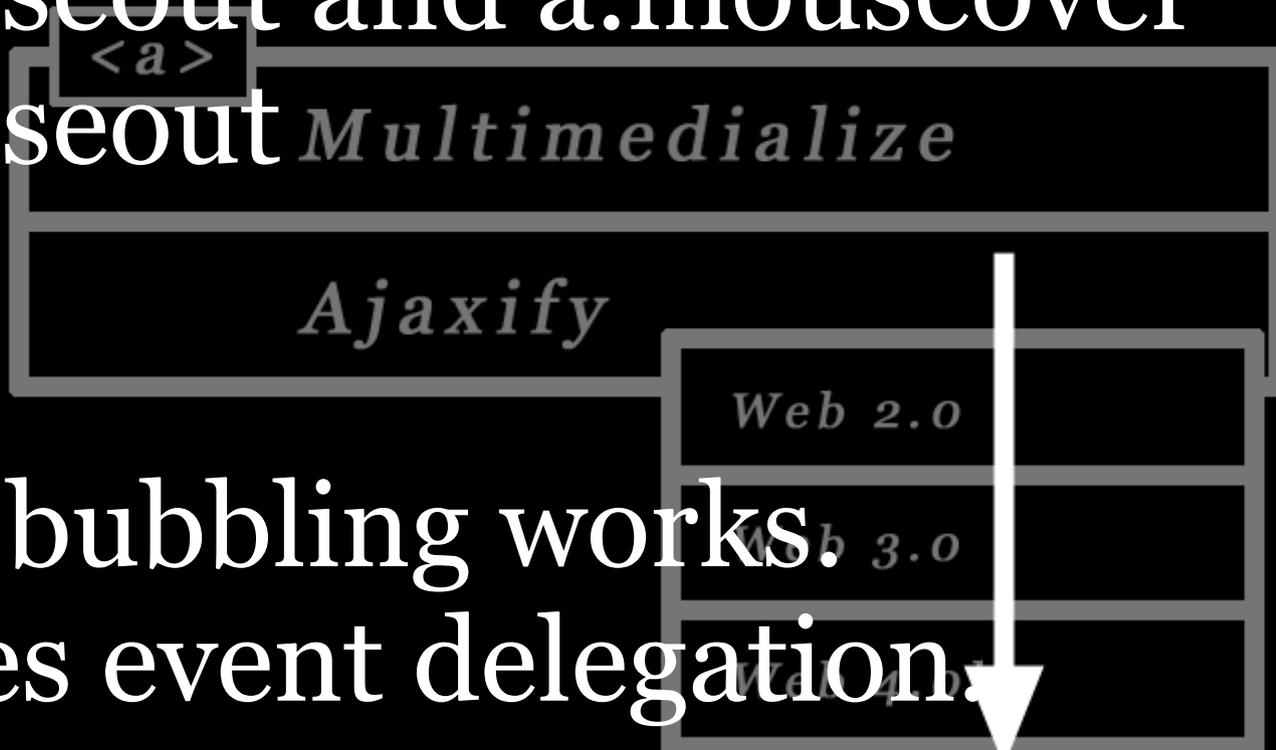
a.mouseout and a.mouseover

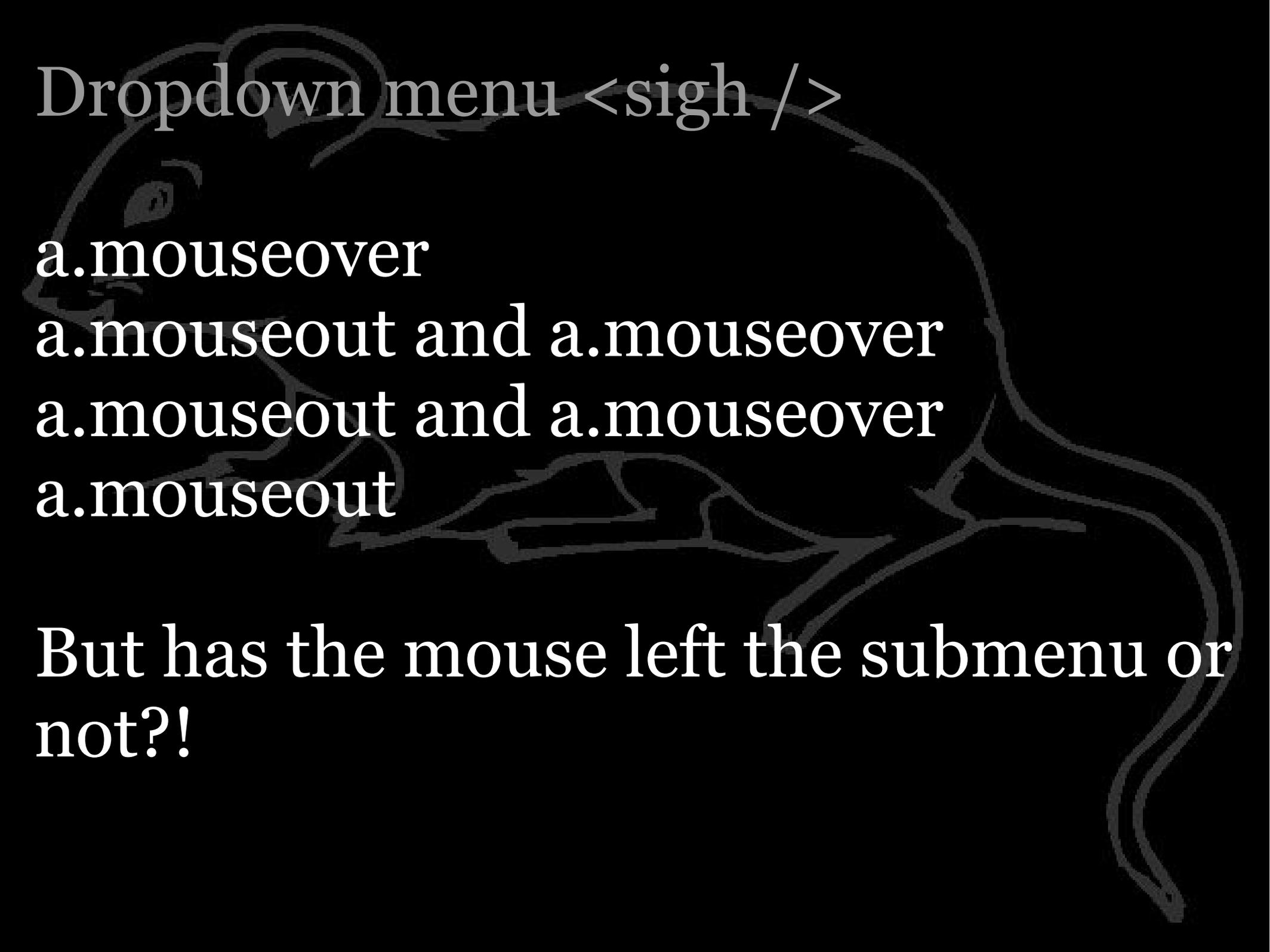
a.mouseout *Multimedialize*

Fun!

Event bubbling works.

As does event delegation.





Dropdown menu <sigh />

a.mouseover

a.mouseout and a.mouseover

a.mouseout and a.mouseover

a.mouseout

But has the mouse left the submenu or not?!

Dropdown menu <sigh />

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

Dropdown menu <sigh />

```
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 10 minutes

Dropdown menu <sigh />

```
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?

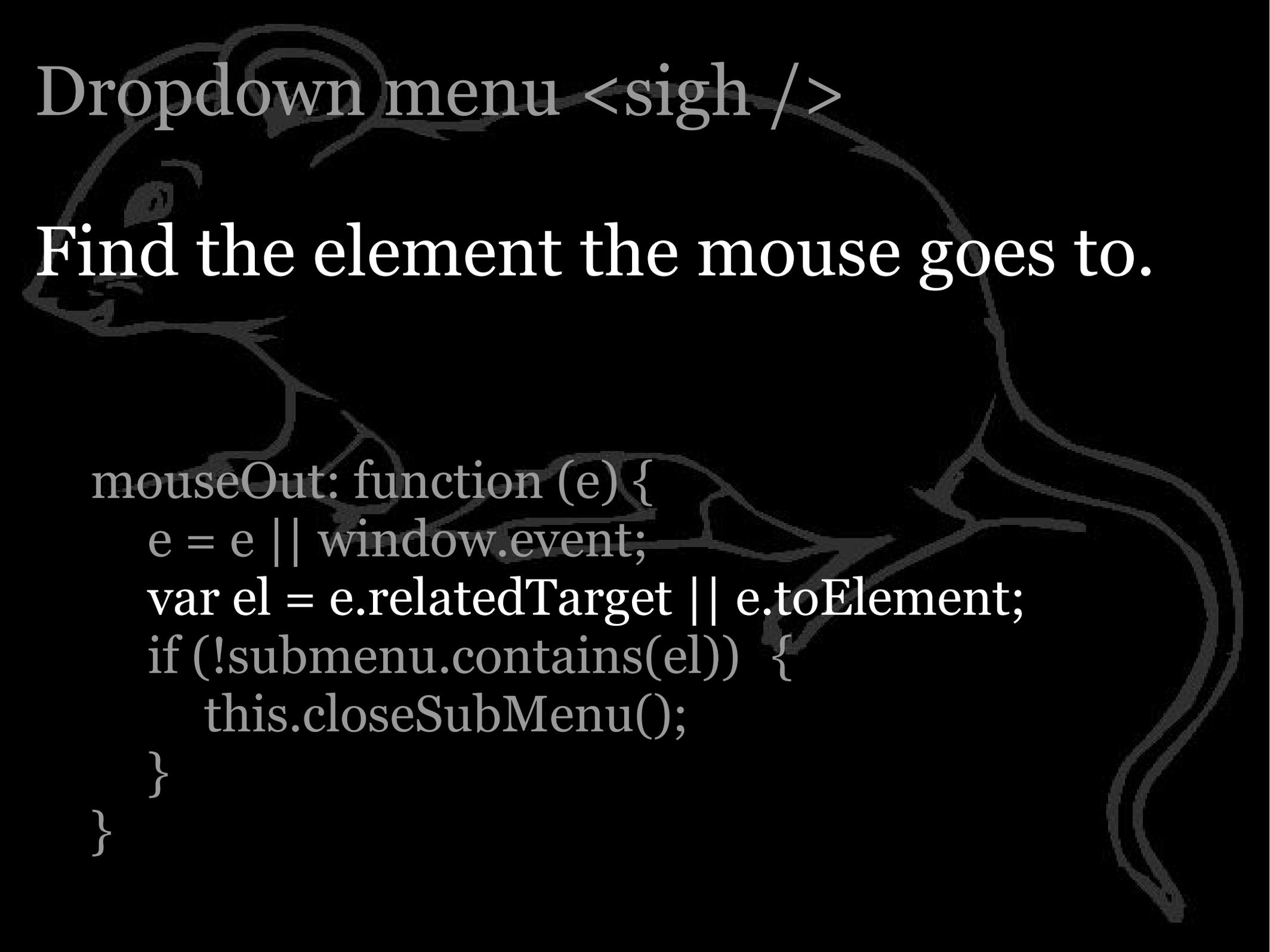
onmouseout, find out which element the mouse goes *to*.

If that element is *not* a part of the submenu, fold the submenu.

Dropdown menu <sigh />

How do we do this?

```
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 />

Find the element the mouse goes to.

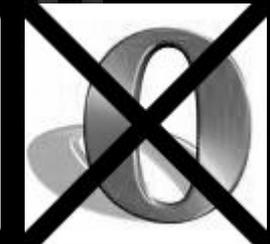
```
mouseout: function (e) {  
  e = e || window.event;  
  var el = e.relatedTarget || e.toElement;  
  if (!submenu.contains(el)) {  
    this.closeSubMenu();  
  }  
}
```



Dropdown menu <sigh />

Find the element the mouse goes to.

```
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.

```
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.

```
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();  
  }  
}
```

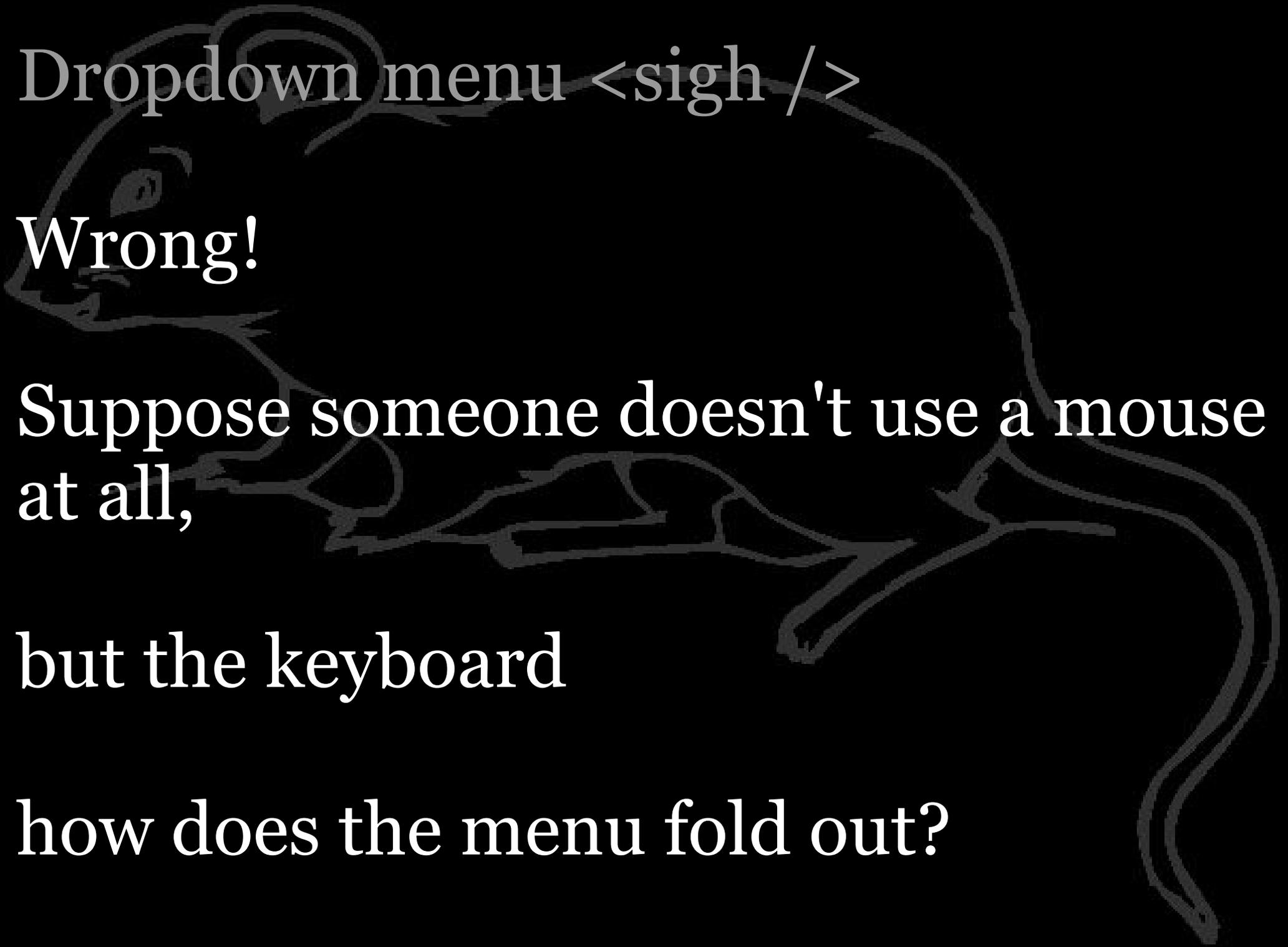
Dropdown menu <sigh />

Wrong!

Suppose someone doesn't use a mouse
at all,

but the keyboard

how does the menu fold out?

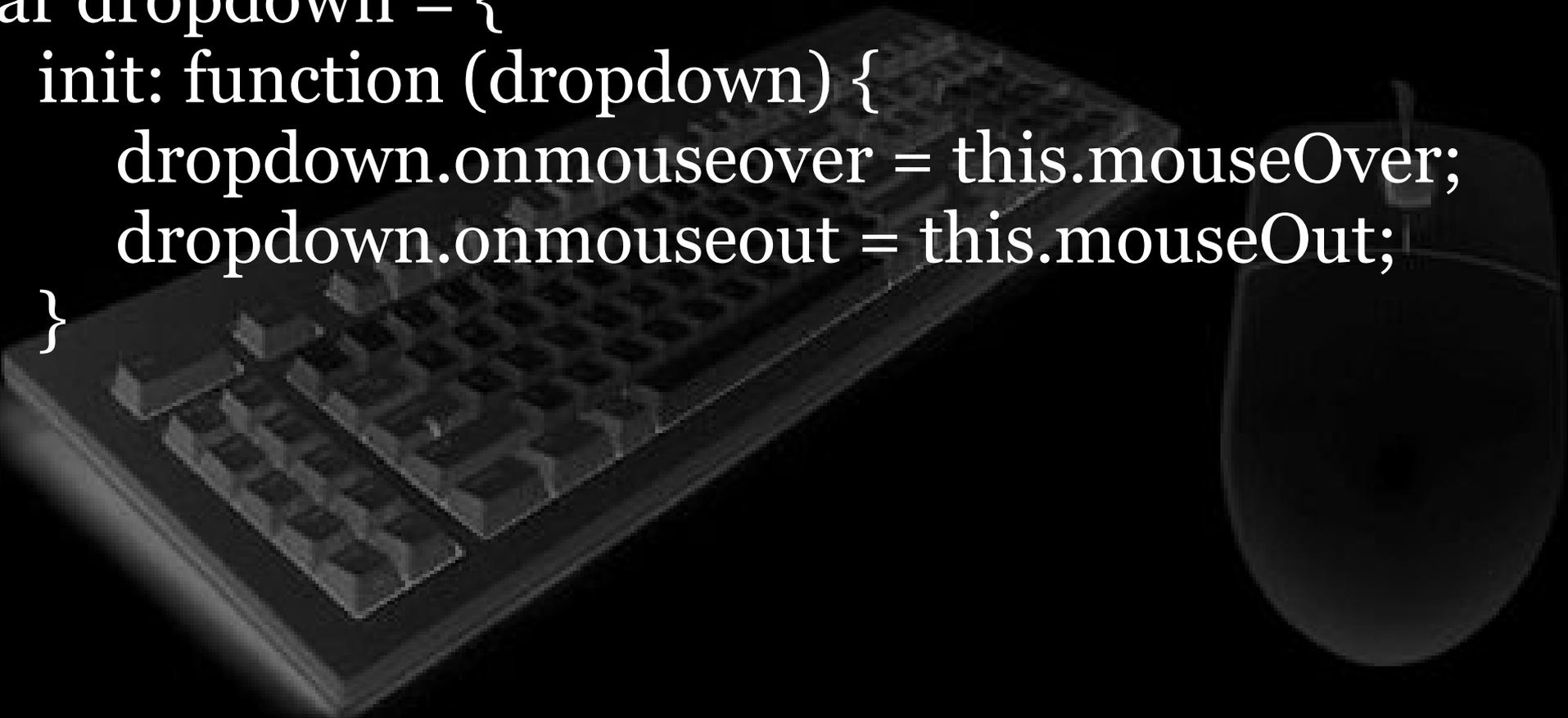


A computer keyboard and mouse are shown on a black background. The keyboard is on the left, and the mouse is on the right. The text "Device independence" is overlaid in the center in a white serif font.

Device
independence

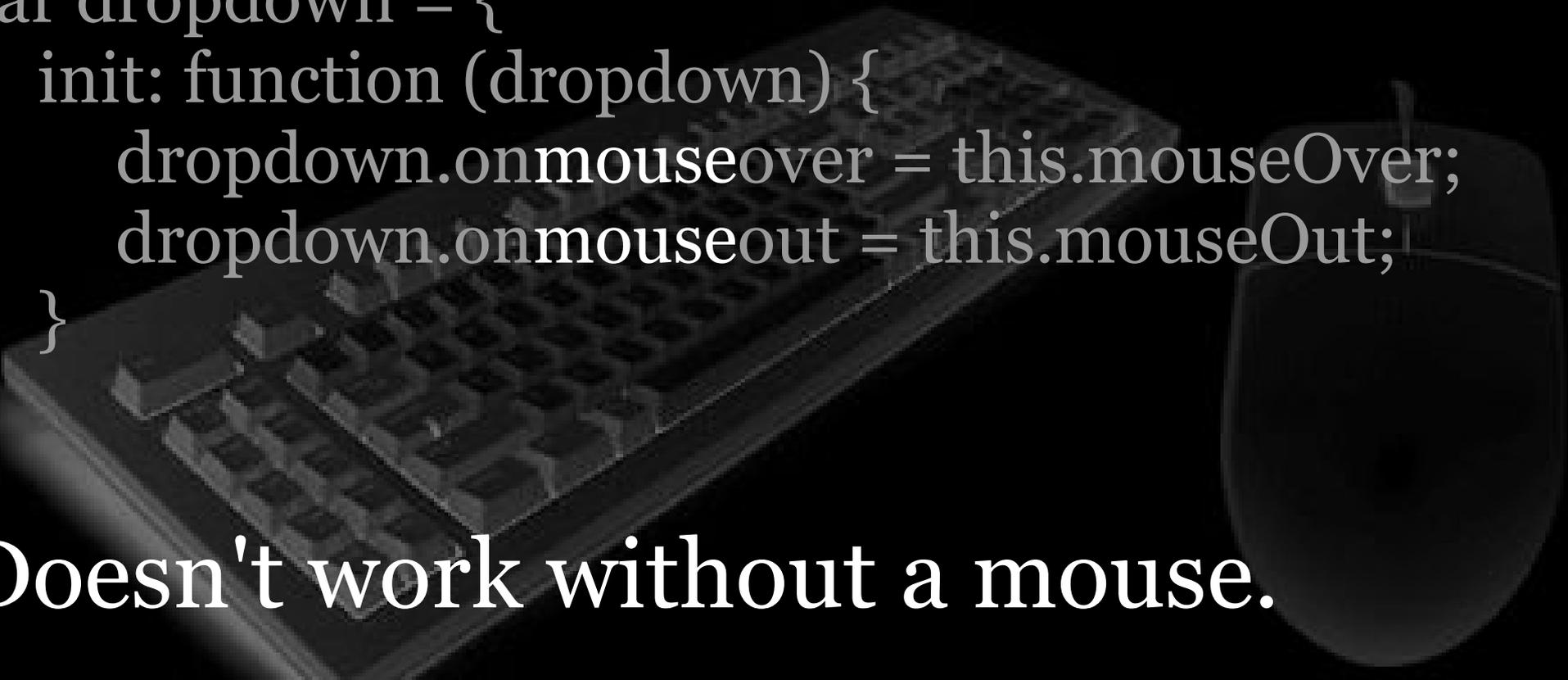
Dropdown menu < sigh />

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



Dropdown menu <sigh />

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



Doesn't work without a mouse.

Dropdown menu <sigh />

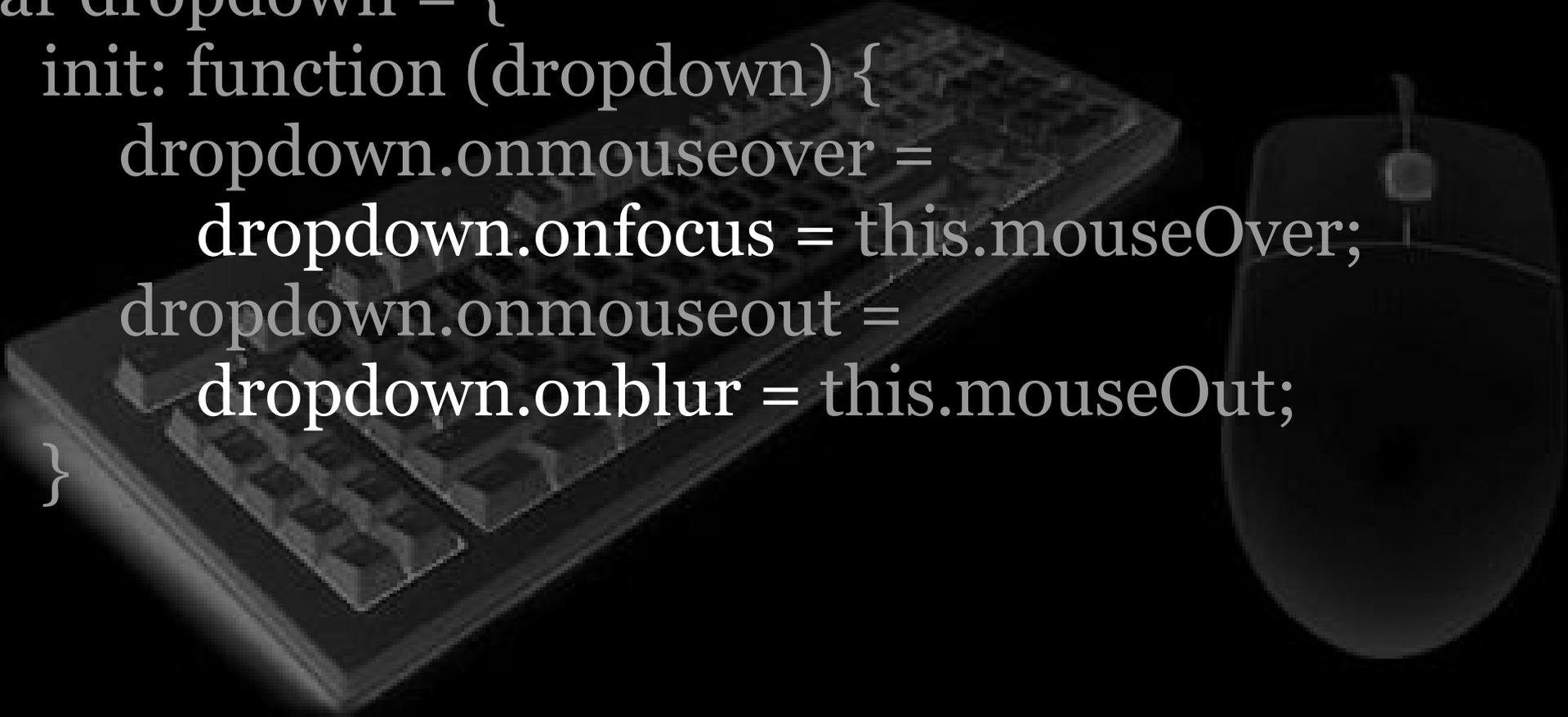
```
var dropdown = {  
  init: function (dropdown) {  
    dropdown.onmouseover = this.mouseOver;  
    dropdown.onmouseout = this.mouseOut;  
  }  
}
```



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

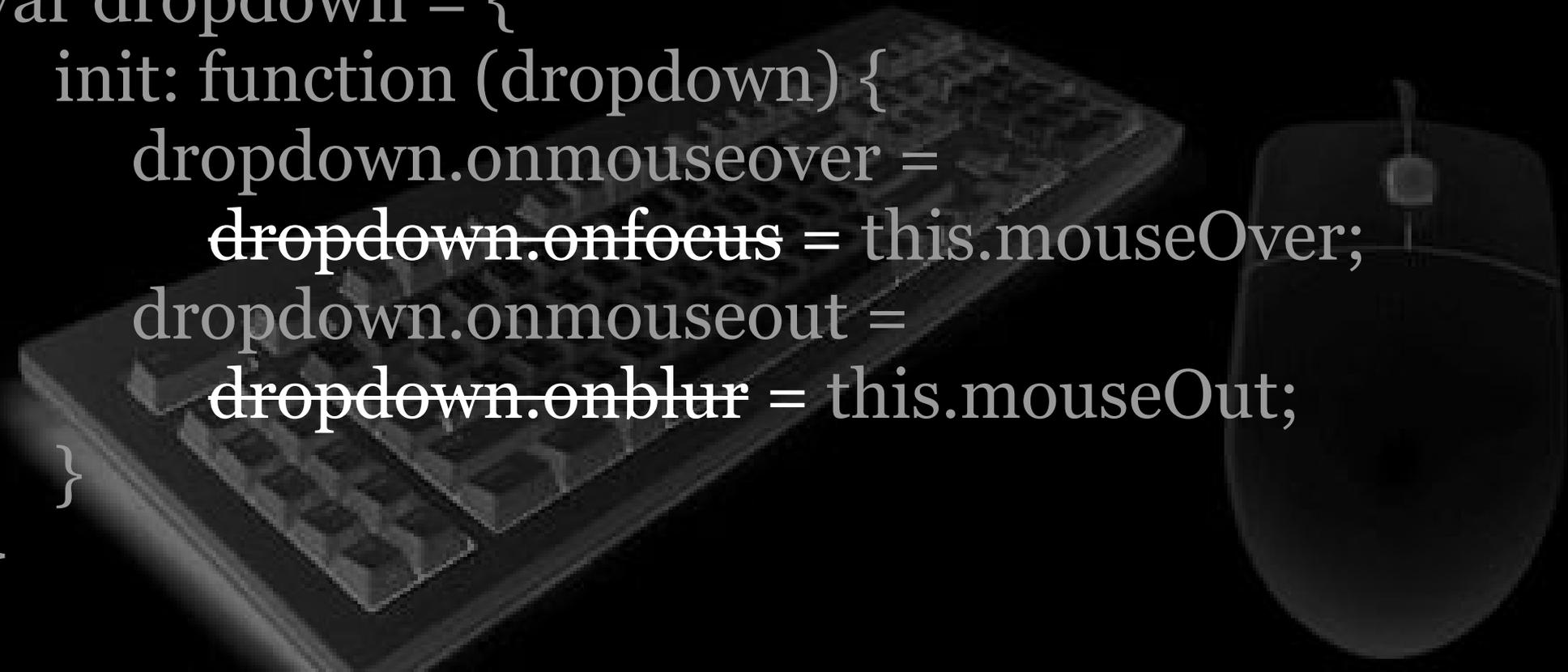
Dropdown menu <sigh />

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



Dropdown menu <sigh />

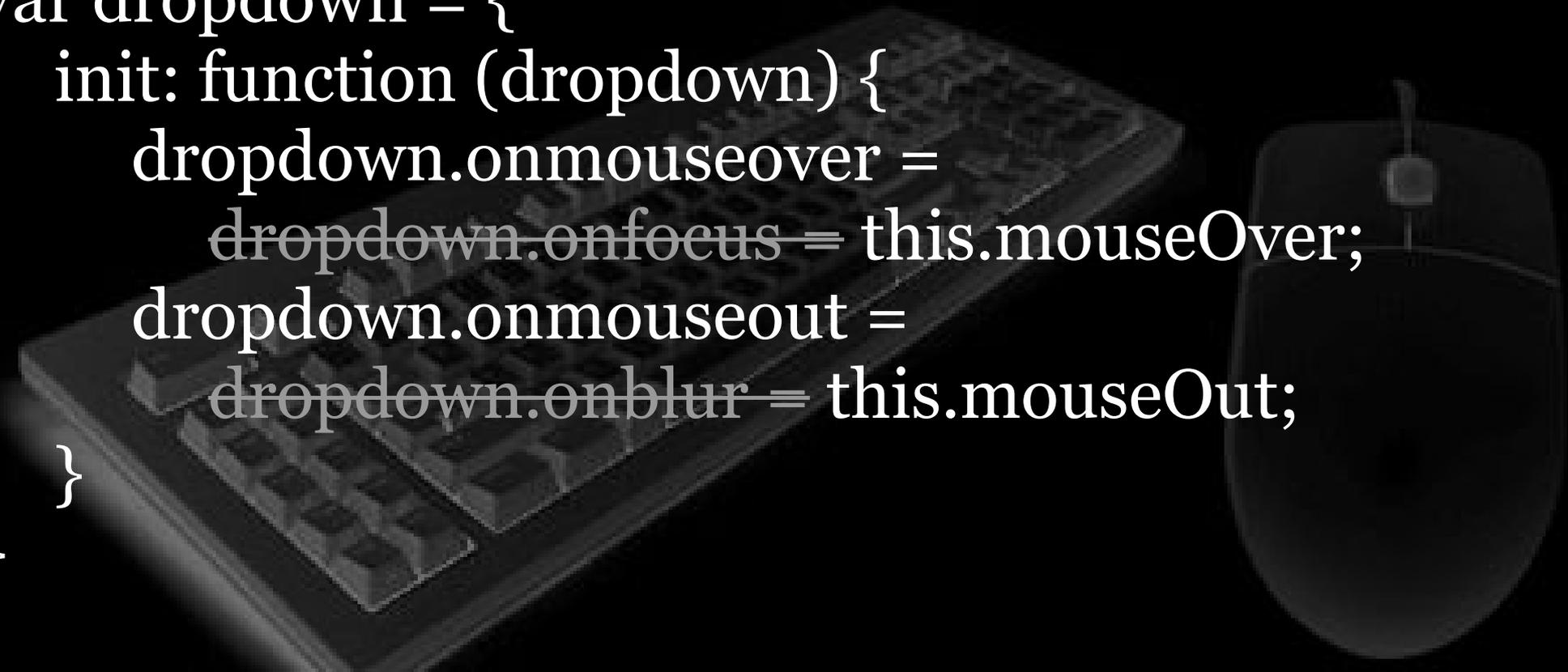
```
var dropdown = {  
  init: function (dropdown) {  
    dropdown.onmouseover =  
    dropdown.onfocus = this.mouseOver;  
    dropdown.onmouseout =  
    dropdown.onblur = this.mouseOut;  
  }  
}
```



Doesn't work.

Dropdown menu <sigh />

```
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



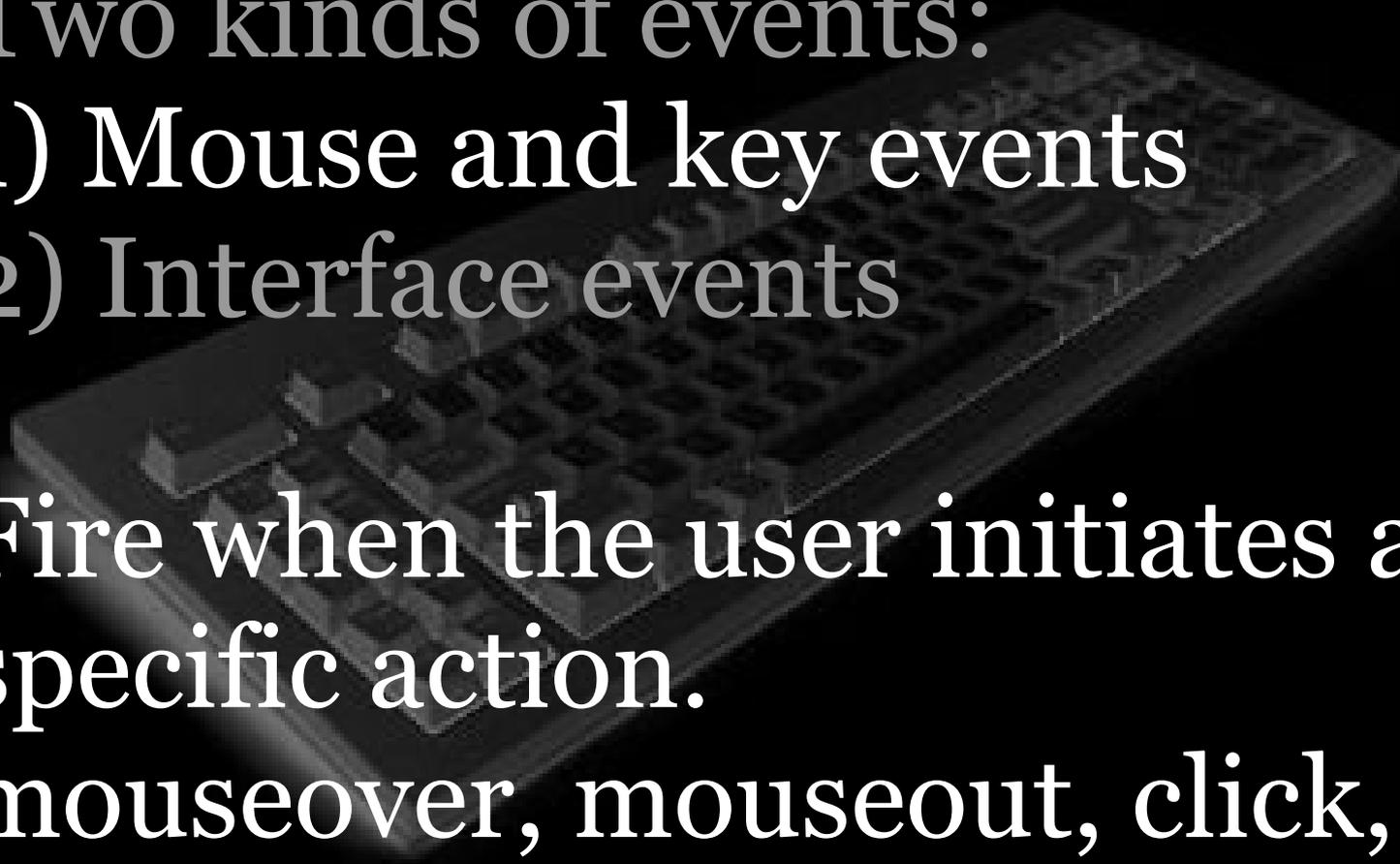
To bubble or not to bubble

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

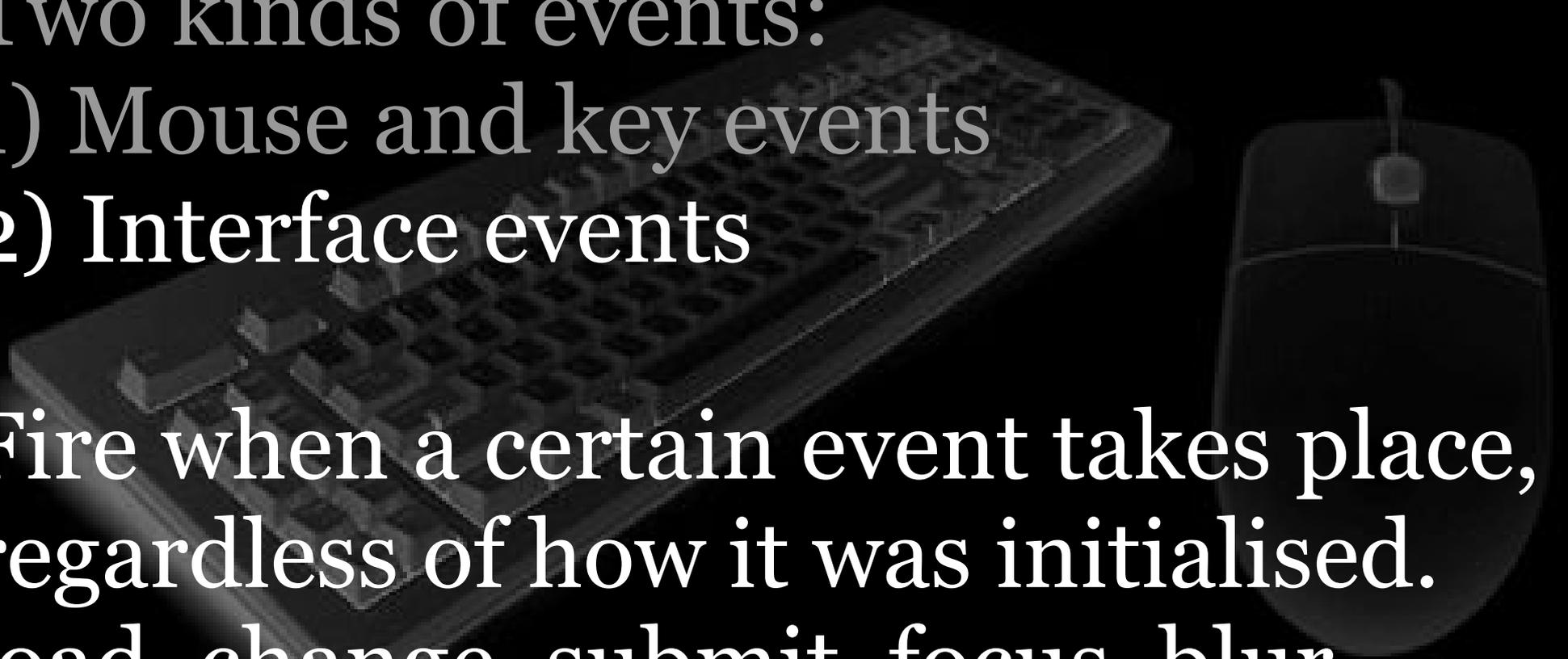


To bubble or not to bubble

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:

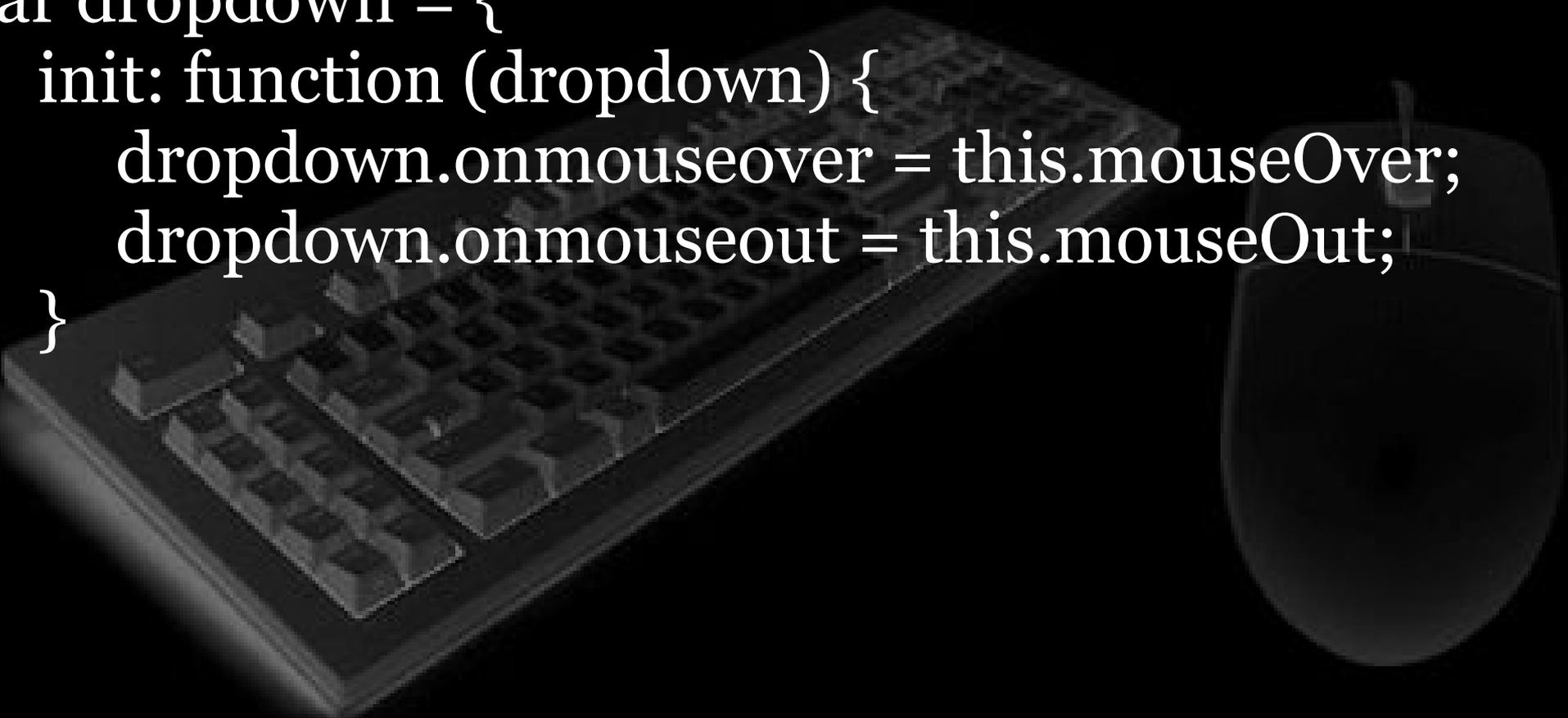
- 1) Mouse and key events
- 2) Interface events

Generally don't bubble



Dropdown menu < sigh />

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



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;  
    }  
  }  
}
```

Dropdown menu <sigh />

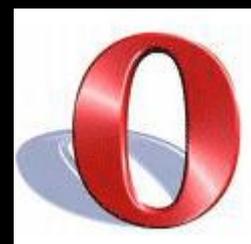
```
var dropdown = {
  init: function (dropdown) {
    dropdown.onmouseover = this.mouseOver;
    dropdown.onmouseout = this.mouseOut;
    var x = dropdown.getElementsByTagName('li');
    for (var i=0;i<x.length;i++) {
      x[i].onfocus = this.mouseOver;
      x[i].onblur = this.mouseOut;
    }
  }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

Doesn't work.

Dropdown menu <sigh />

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

- links
- form fields



Dropdown menu <sigh />

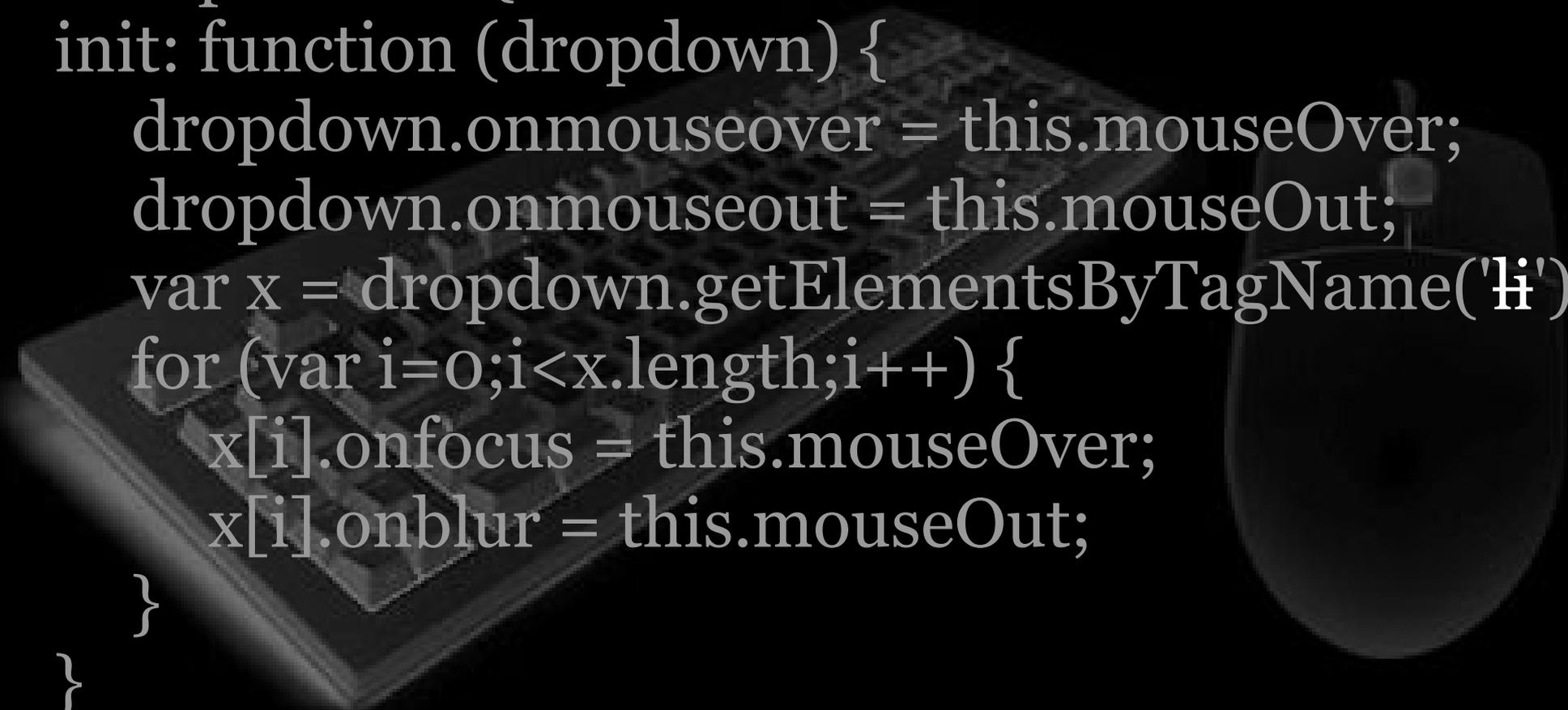
The HTML elements must be able to receive the keyboard focus.

- links
- form fields
- elements with tabindex



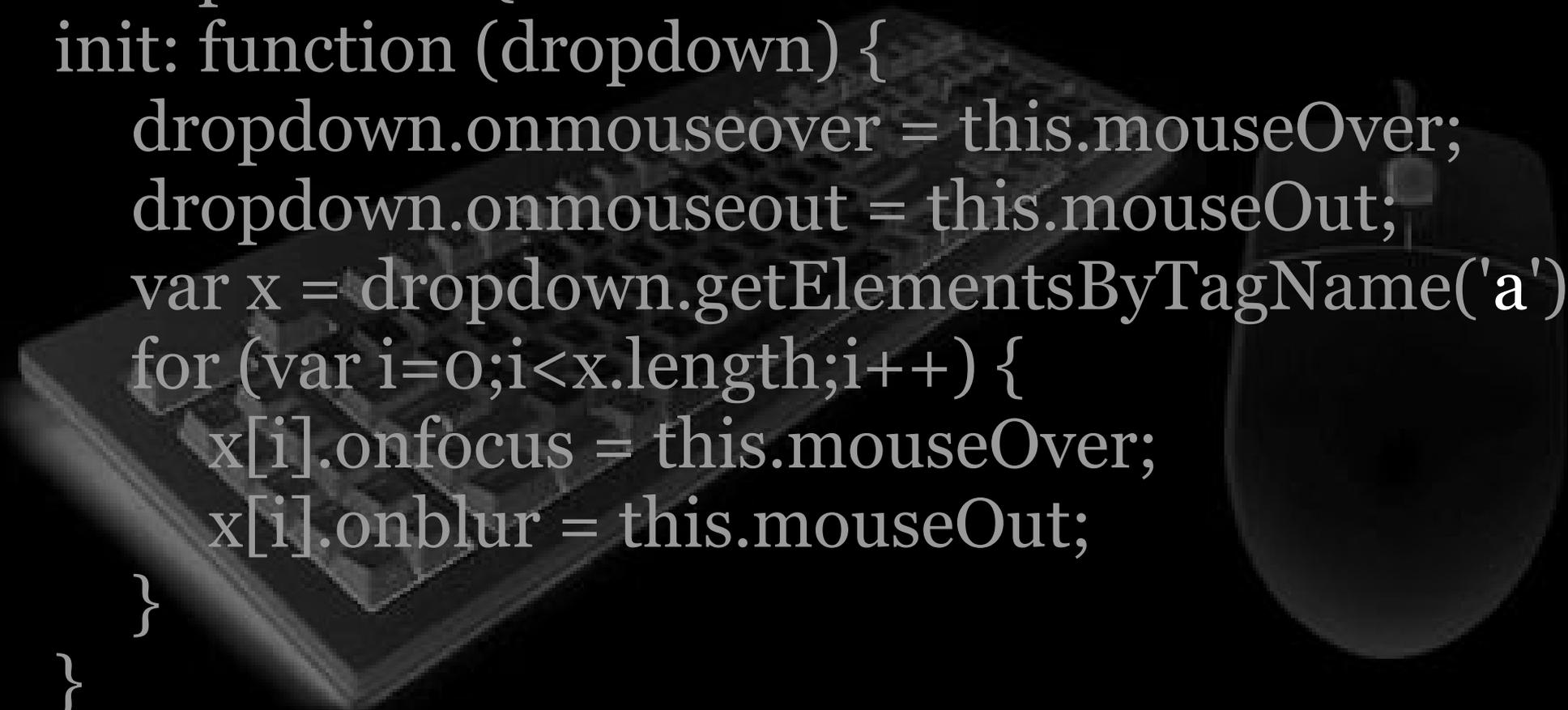
Dropdown menu <sigh />

```
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;
    }
  }
}
```



Dropdown menu <sigh />

```
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;  
    }  
  }  
}
```

A faint, grayscale background image of a computer keyboard and mouse is visible behind the text. The keyboard is on the left and the mouse is on the right, both appearing as semi-transparent overlays.

Dropdown menu <sigh />

```
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.

<http://yuiblog.com/blog/2009/04/27/video-ppk-jsevents/>

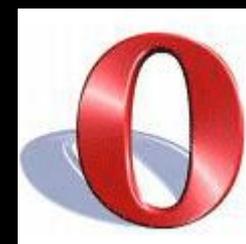
A computer keyboard and mouse are shown on a black background. The keyboard is on the left, and the mouse is on the right. The text "More device independence" is overlaid in the center in a white serif font.

More device
independence

And what about click?

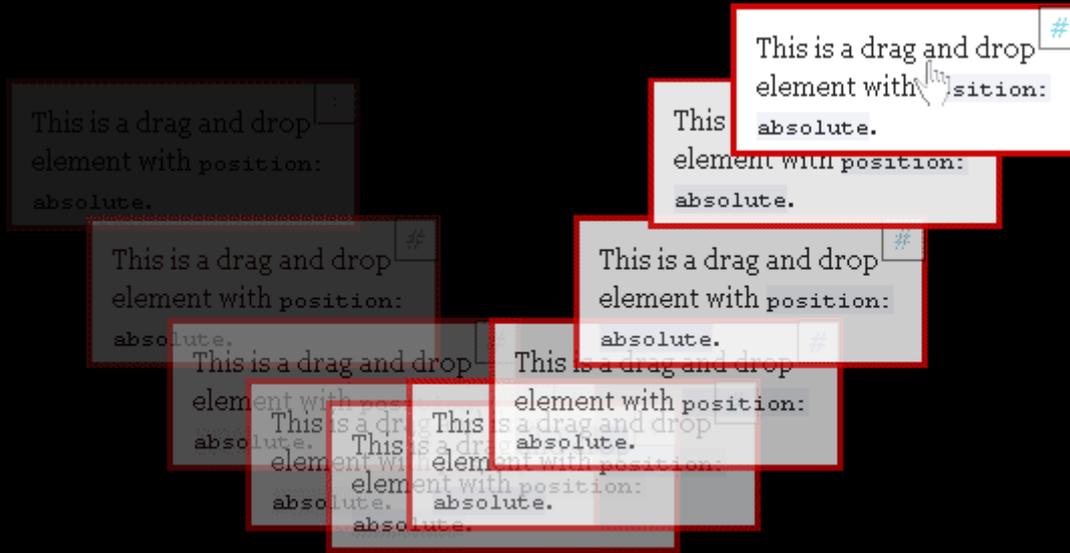
We're in luck: click also fires when the user activates an element by keyboard.

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

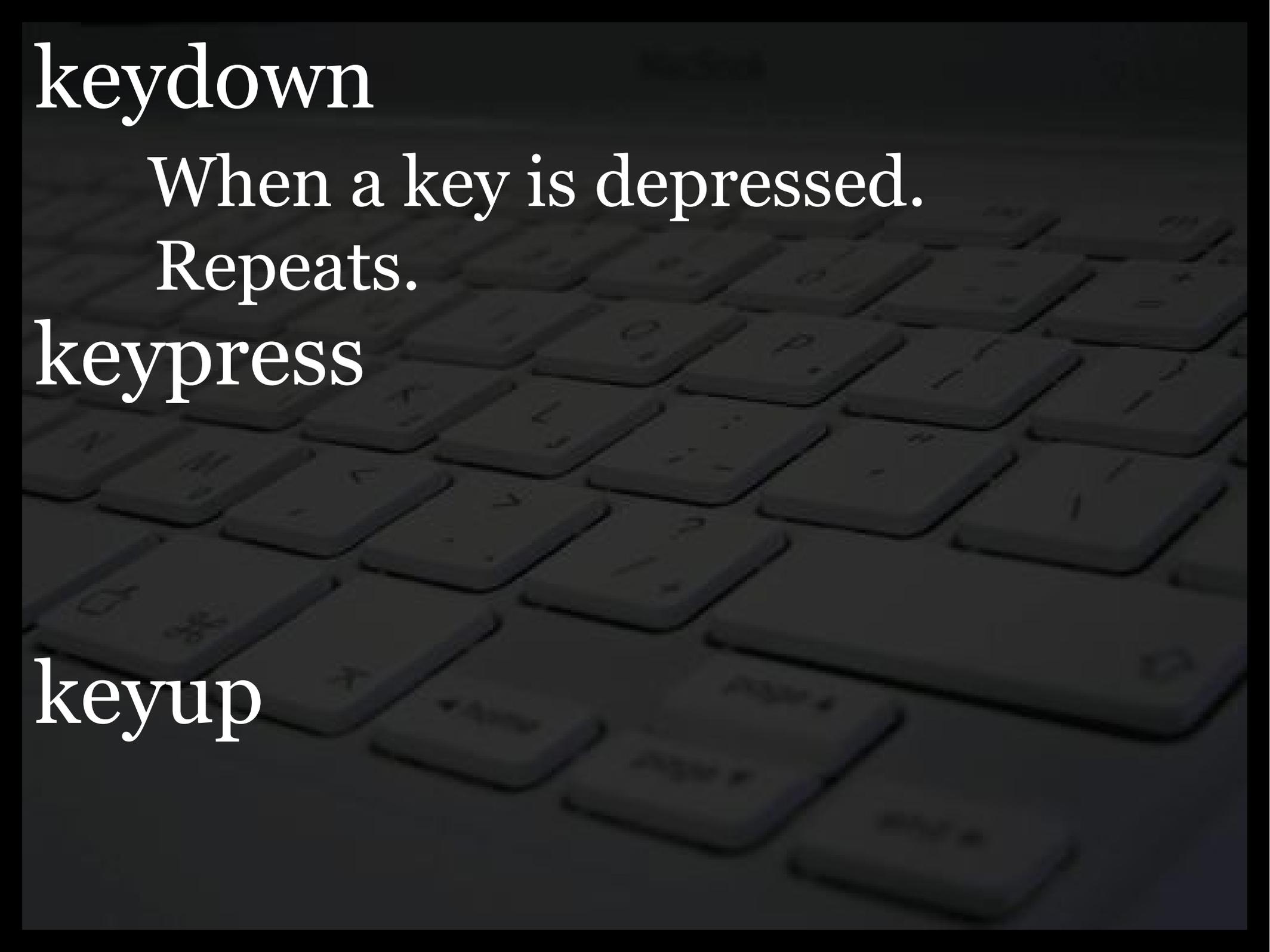
How do we make this keyboard
accessible?

By allowing the user to use the arrow
keys.

Key events.



The key events



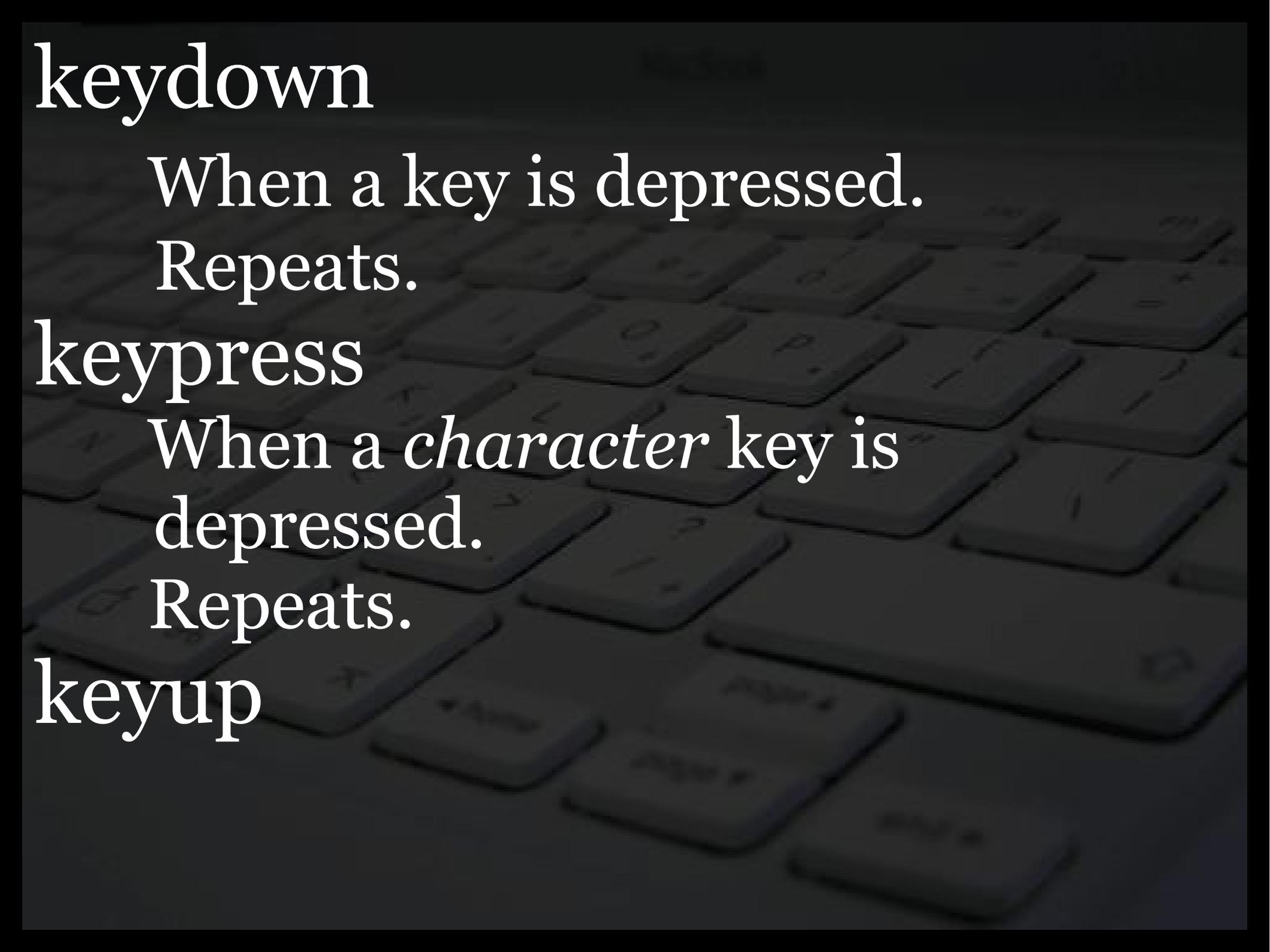
keydown

When a key is depressed.

Repeats.

keypress

keyup



keydown

When a key is depressed.

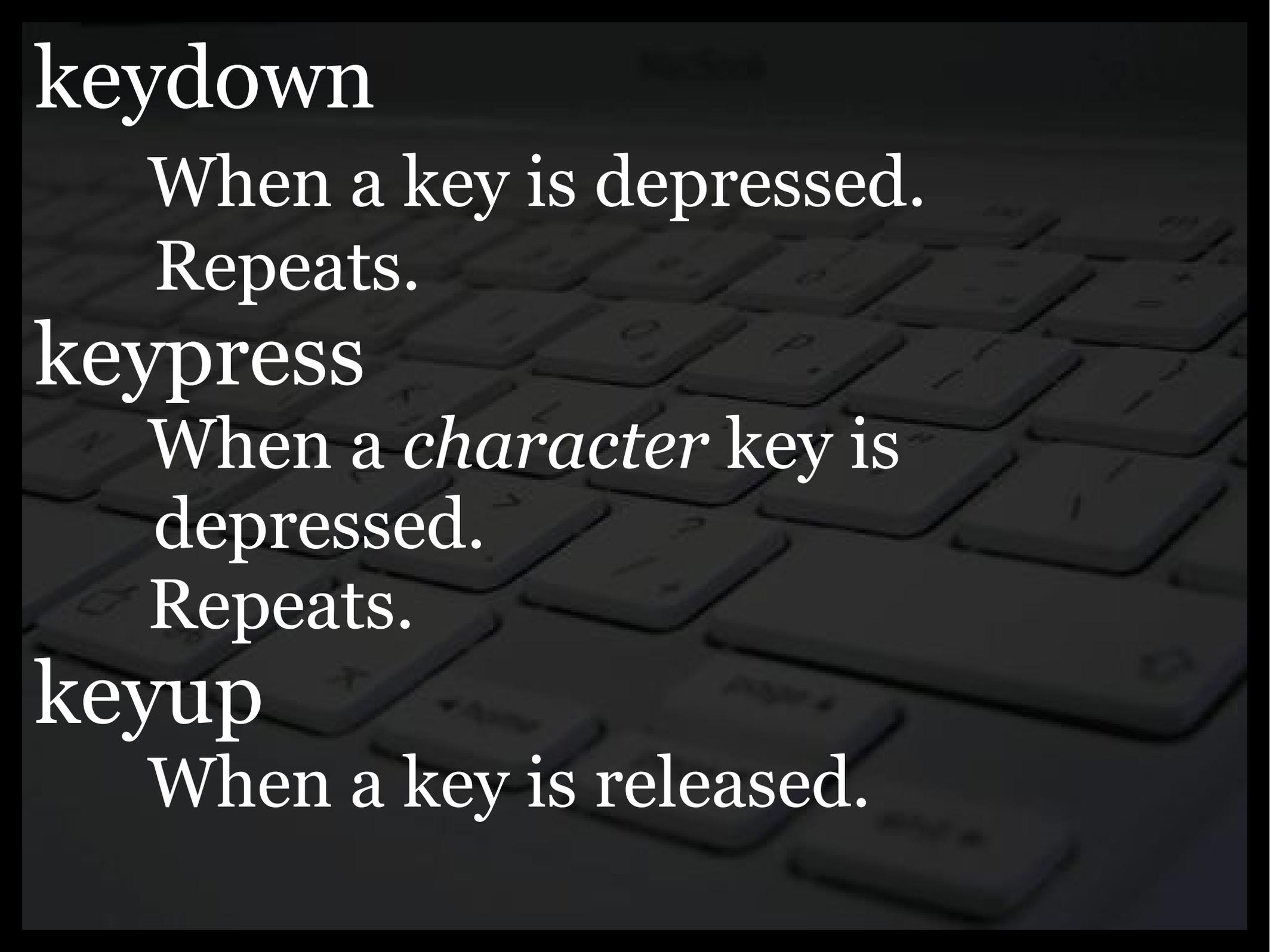
Repeats.

keypress

When a *character* key is depressed.

Repeats.

keyup



keydown

When a key is depressed.

Repeats.

keypress

When a *character* key is depressed.

Repeats.

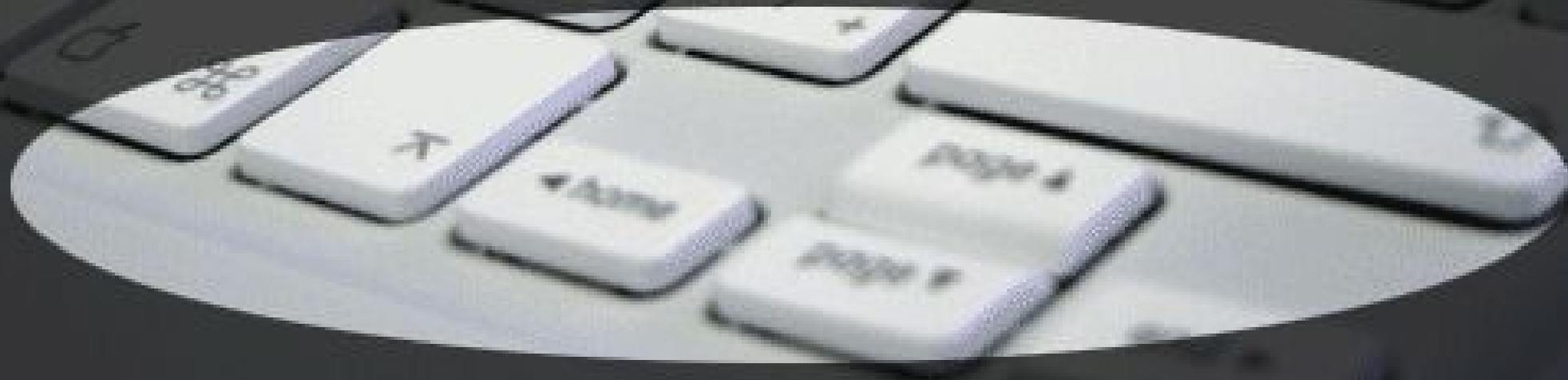
keyup

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

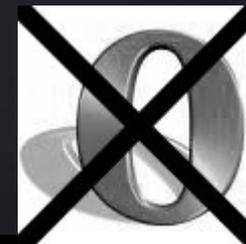
When a key is depressed.

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;  
}
```

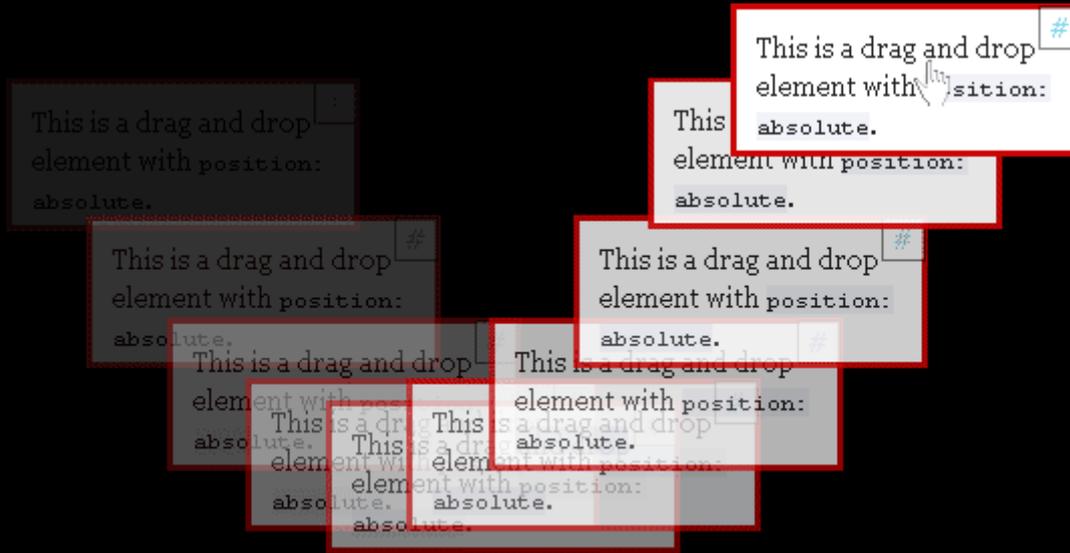
Which key did my user press?

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



Separate concepts

Back to the drag-and-drop



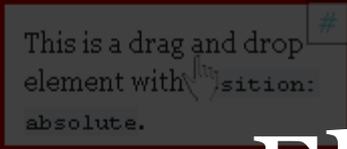
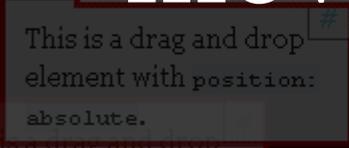
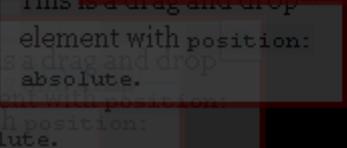
Separate concepts

Drag-and-drop

We need the keydown event, because arrow keys are special keys.

Separate concepts

Drag-and-drop

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

Separate concepts

Drag-and-drop

~~obj.onmousemove =~~
~~obj.onkeydown =~~ ~~moveElement;~~

Doesn't work.

Separate concepts

Drag-and-drop

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

MouseEvent expects mouse coordinates.

The layer moves to these coordinates.

Separate concepts

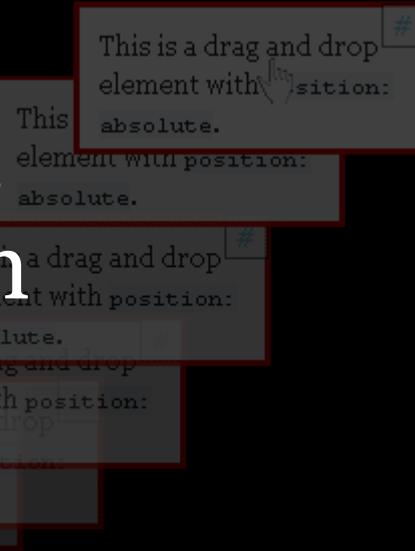
Drag-and-drop

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

The key events expect a keystroke.

```
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.

```
obj.onmousemove = moveByMouse;  
obj.onkeydown = moveByKeys;
```

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:

- document
- form
- text
- checkbox

Miscellaneous

- Prevent default action
- Cancel bubble

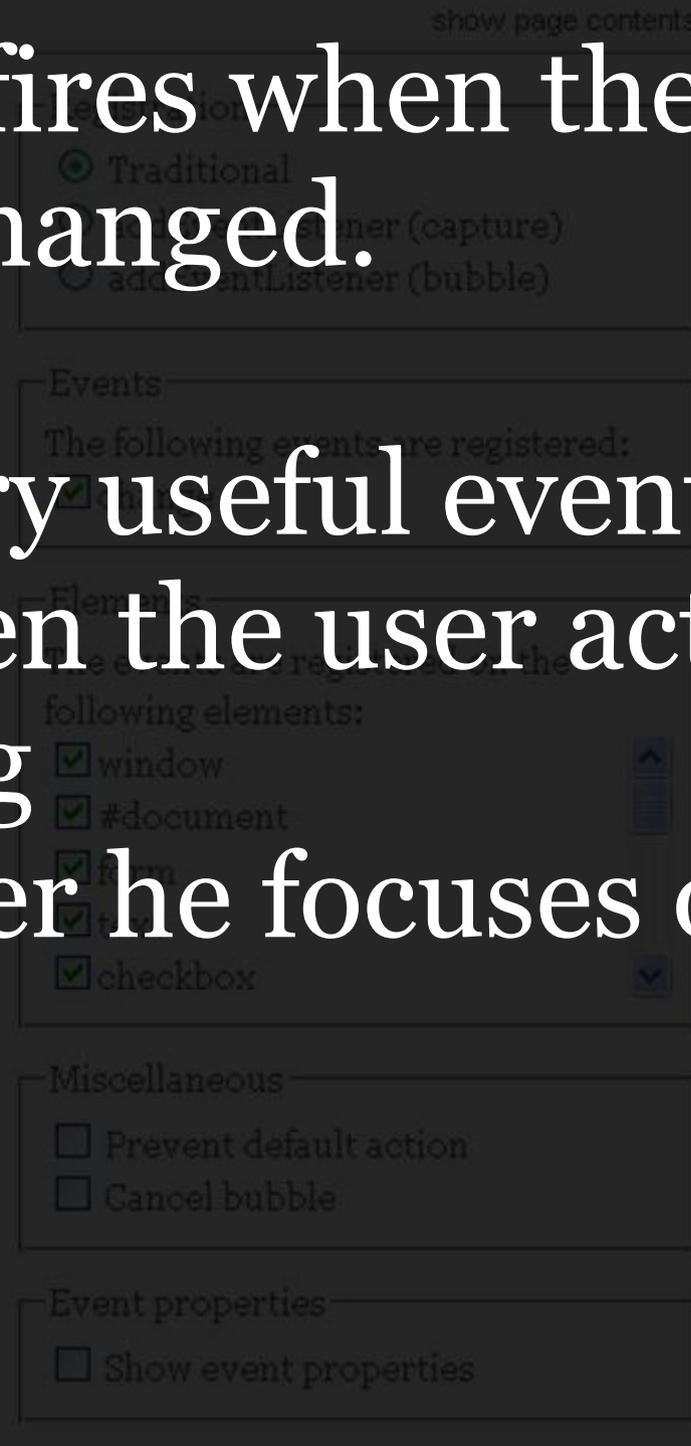
Event properties

- Show event properties

change

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

show page contents

Registration

- Traditional
- addEventListener (capture)
- addEventListener (bubble)

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

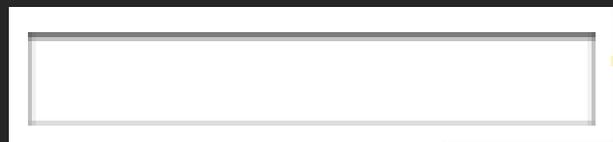
Event properties

- Show event properties

- text fields
- select boxes
- checkboxes and radios



focus



blur

No change event. The value hasn't been modified.

show page contents

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
- Do not scroll

Event properties

- Show event properties

- text fields
- select boxes
- checkboxes and radios

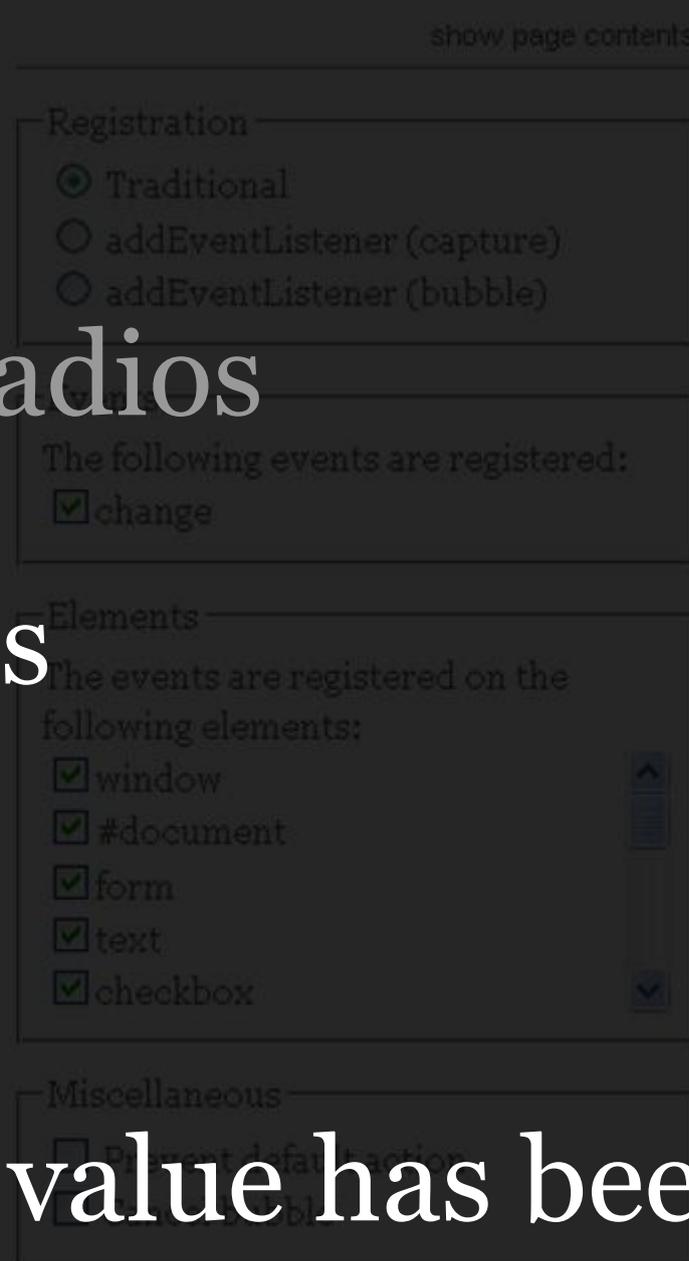


focus



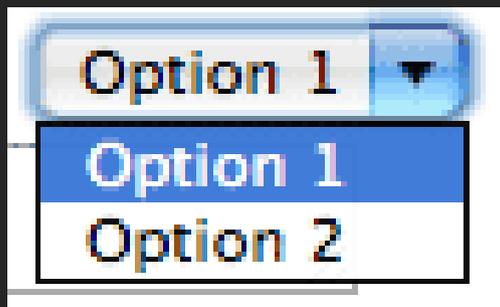
blur

Change event. The value has been modified.



- text fields
- select boxes
- checkboxes and radios

Mouse:



show page contents

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

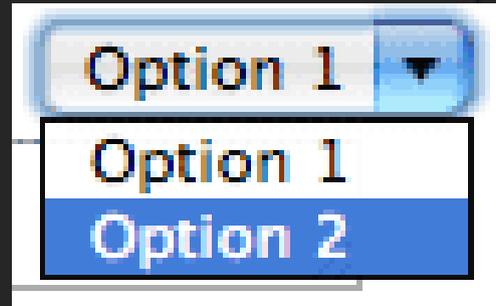
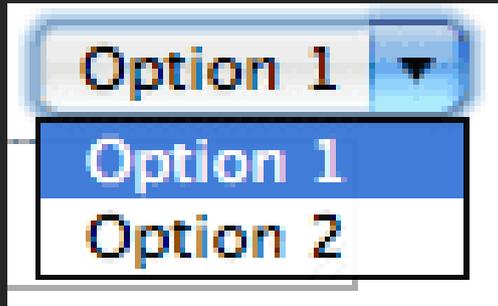
- Prevent default action
- Cancel bubble

Event properties

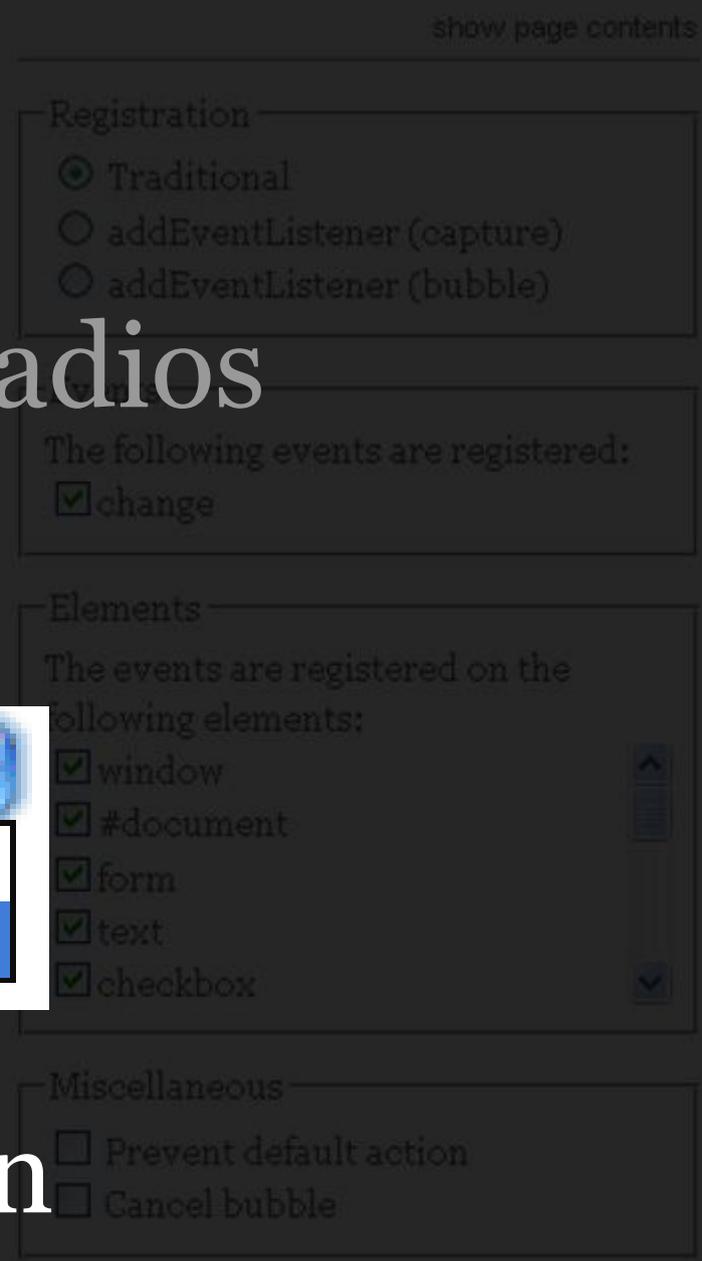
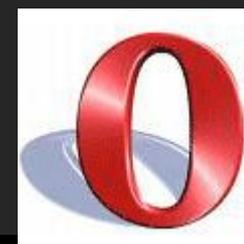
- Show event properties

- text fields
- select boxes
- checkboxes and radios

Mouse:



Click on new option
CHANGE



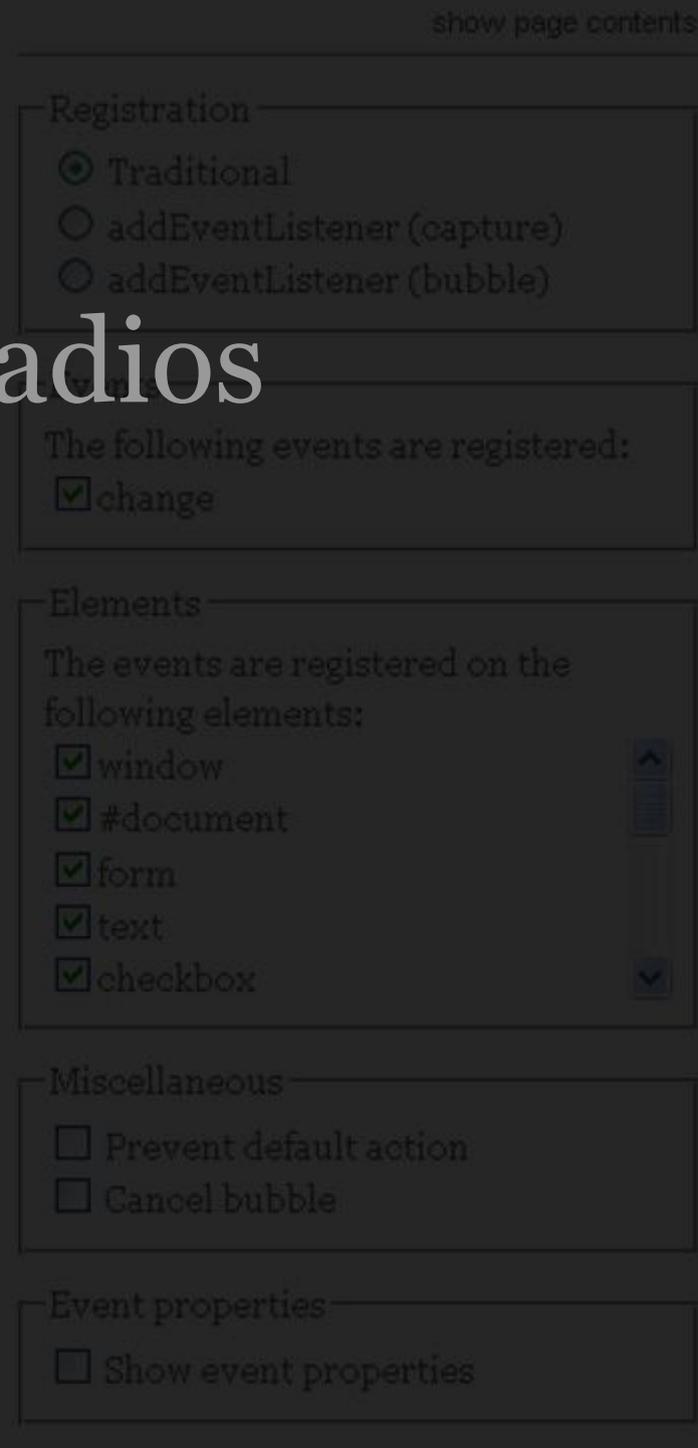
- text fields
- select boxes
- checkboxes and radios

Keyboard:



focus

Focus on select



- text fields
- select boxes
- checkboxes and radios

Keyboard:

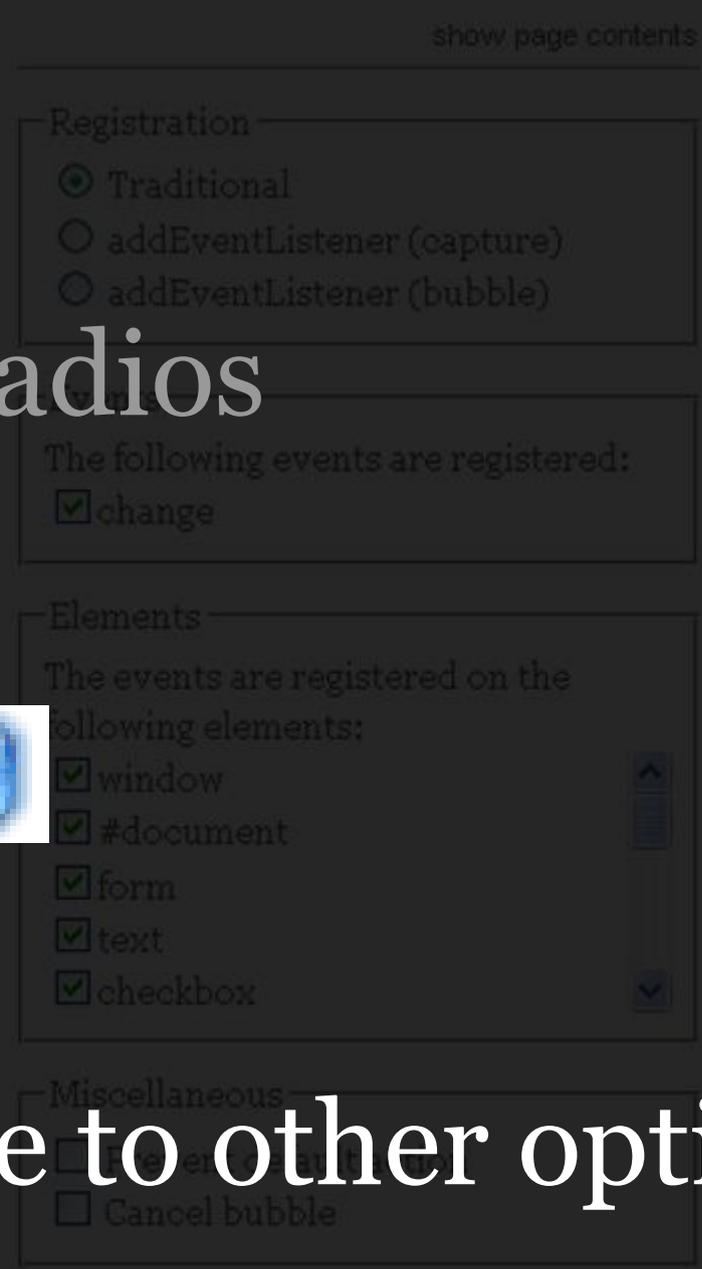
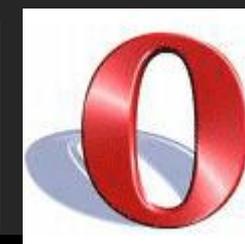


focus



arrow

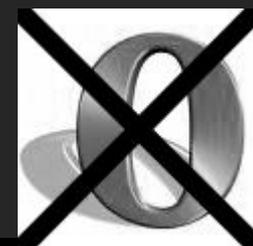
Arrow keys to move to other option
CHANGE



- text fields
- select boxes
- checkboxes and radios

This is a
BUG!

Arrow keys to move to other option
CHANGE



show page contents

Registration

- Traditional
- addEventListener (capture)
- addEventListener (bubble)

The following events are registered:

- change

Elements

The following events are registered on the following elements:

- window
- #document
- form
- text
- checkbox

Miscellaneous

- Prevent default
- Cancel bubble

Properties

Event properties

- text fields
- select boxes
- checkboxes and radios

Keyboard:

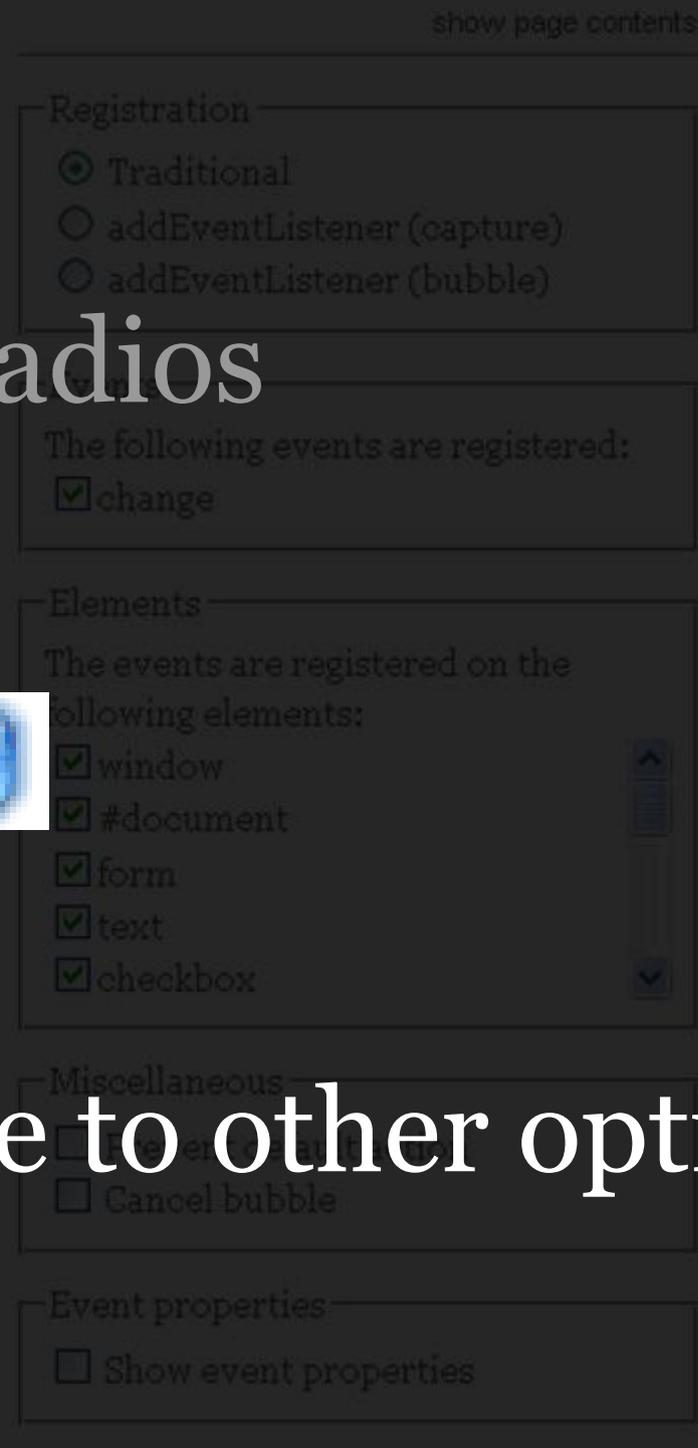


focus



arrow

Arrow keys to move to other option



- text fields
- select boxes
- checkboxes and radios

Keyboard:



focus

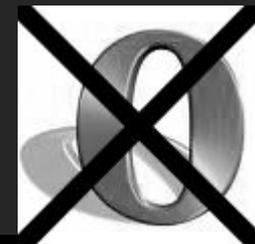


arrow



blur

Blur select box.
CHANGE



show page contents

Registration

- Traditional
- addEventListener (capture)
- addEventListener (bubble)

The following events are registered:

- change

Elements

The events are registered on the

- form
- text
- checkbox

Miscellaneous

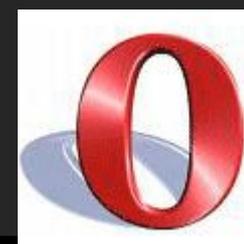
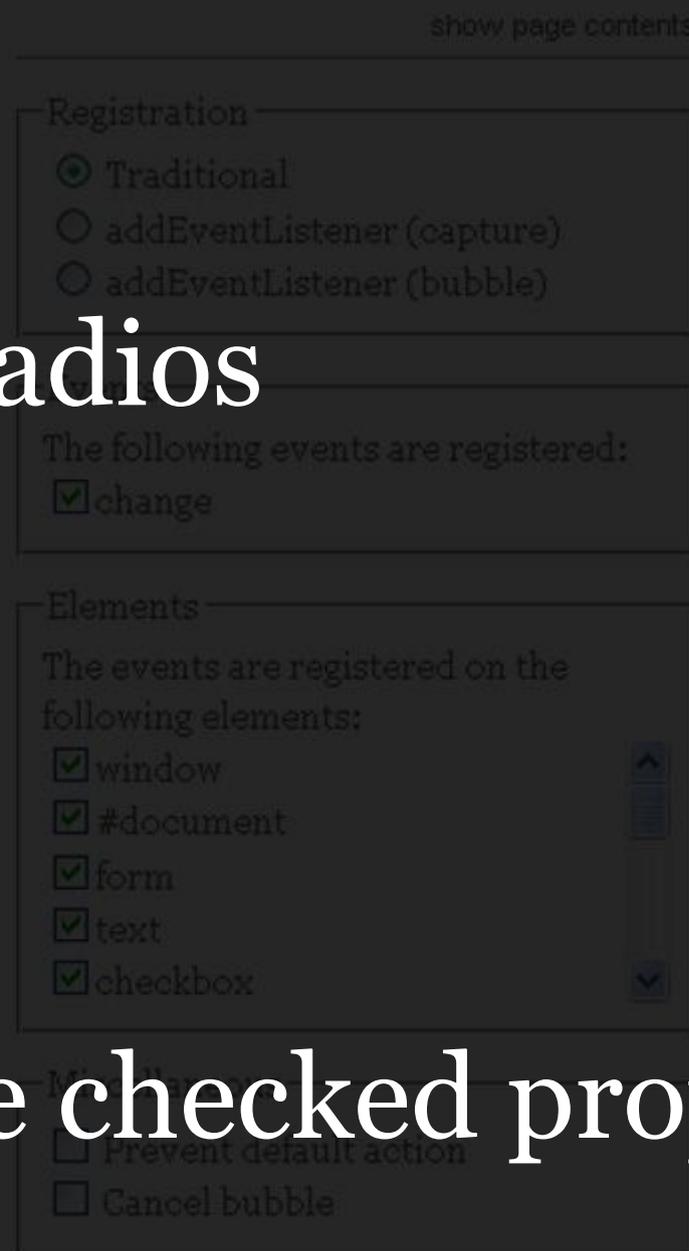
- Prevent default action
- Cancel bubble

- text fields
- select boxes
- checkboxes and radios



click

CHANGE when the checked property changes.

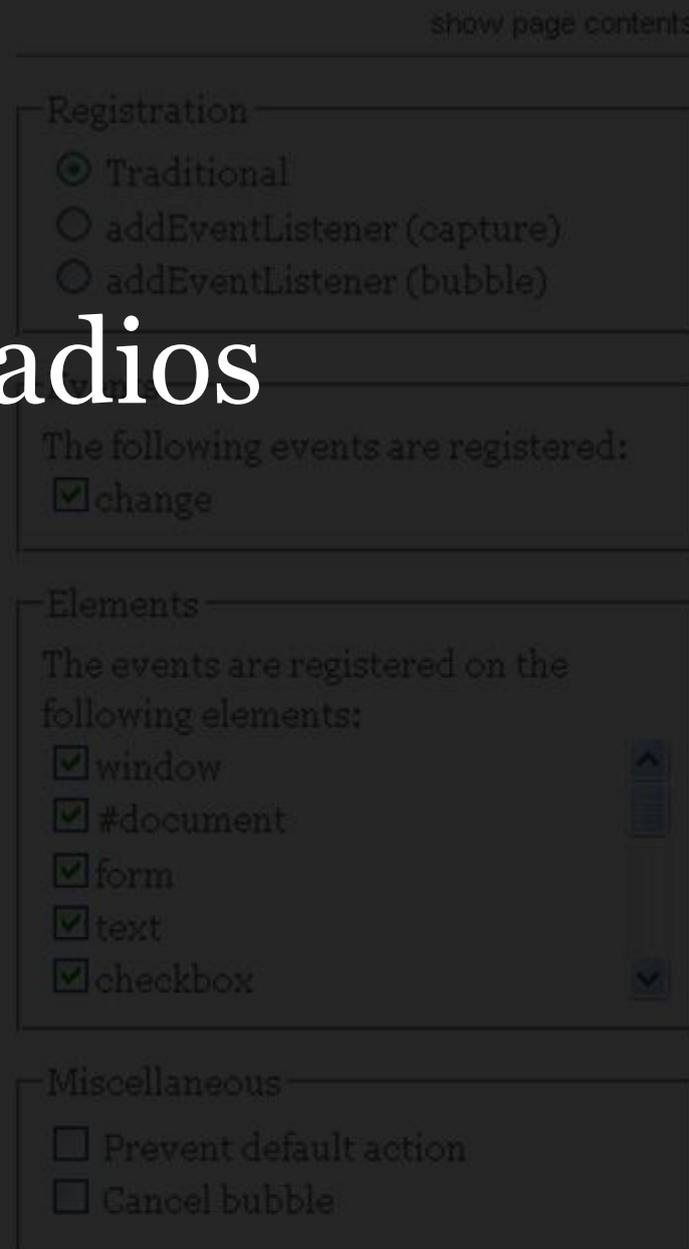


- text fields
- select boxes
- checkboxes and radios

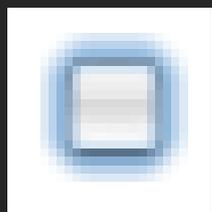


click

...



- text fields
- select boxes
- checkboxes and radios



click



blur

CHANGE when the element loses the focus.

show page contents

Registration

- Traditional
- addEventListener (capture)
- addEventListener (bubble)

The following events are registered:

- change

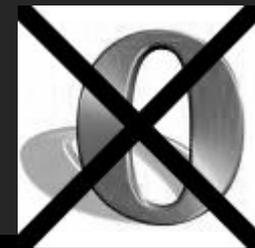
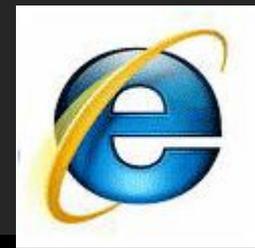
Elements

The events are registered on the following elements:

- window
- #document
- form
- text
- checkbox

Prevent default action

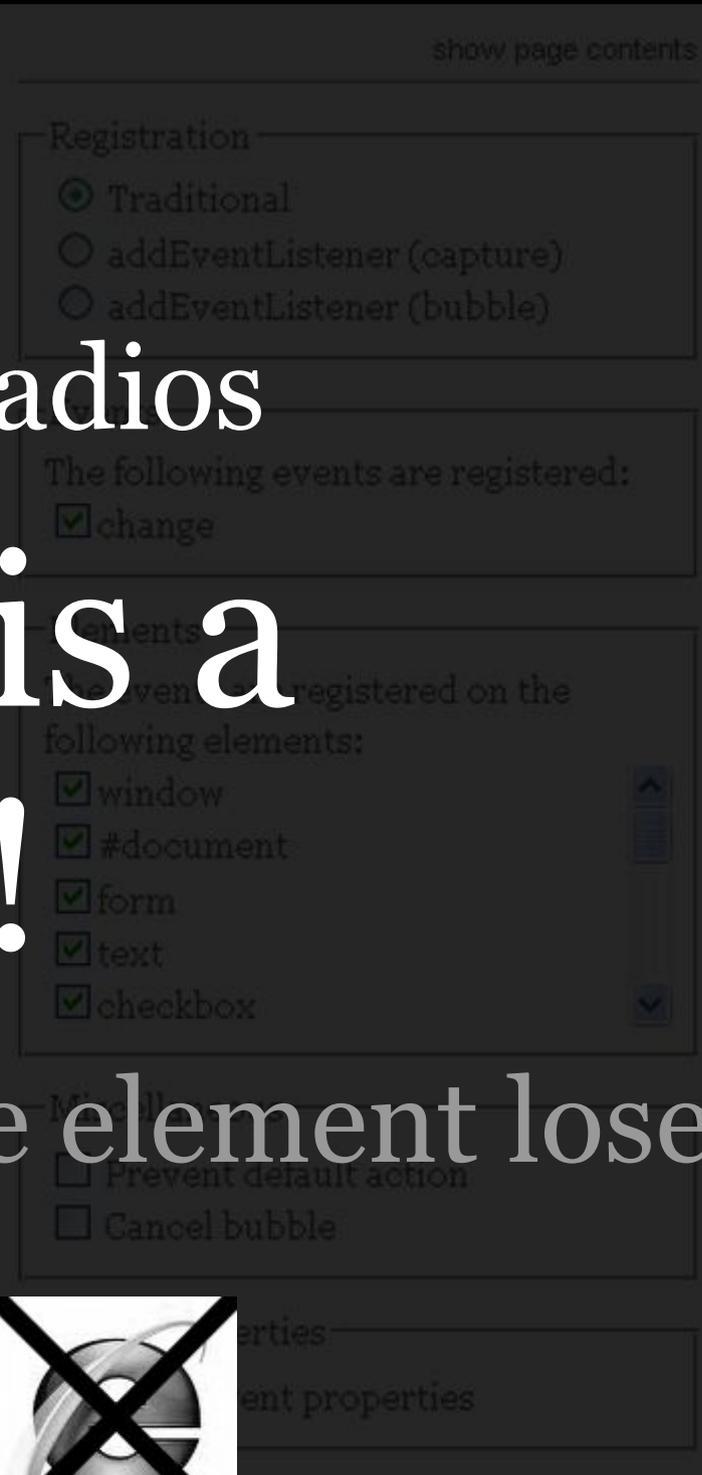
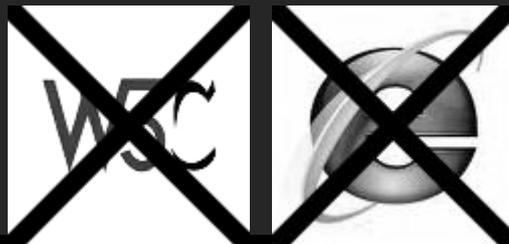
Cancel bubble



- text fields
- select boxes
- checkboxes and radios

This is a
BUG!

CHANGE when the element loses the focus.



Event	IE 5.5	IE 6	IE 7	IE8b1	FF 2	FF 3b5	Saf 3.0 Win	Saf 3.1 Win	Opera 9.26	Opera 9.5b	Konqueror 3.5.7
<u>blur</u>	yes				too many	almost	incomplete	almost	incomplete		incomplete

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.

<u>change</u>	buggy				yes		yes		yes		yes
---------------	-------	--	--	--	-----	--	-----	--	-----	--	-----

When a form field value changes.

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

<http://quirksmode.org/dom/events/>

<u>click</u>	yes				yes		yes		yes		yes
--------------	-----	--	--	--	-----	--	-----	--	-----	--	-----

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

Event	IE 5.5	IE 6	IE 7	IE8b1	FF 2	FF 3b5	Saf 3.0 Win	Saf 3.1 Win	Opera 9.26	Opera 9.5b	Konqueror 3.5.7
<u>contextmenu</u>		yes		minimal	yes	buggy	yes		no		no

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?

- 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.

Ask away.

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

Or ask me on Twitter

<http://twitter.com/ppk>

or on my site

<http://quirksmode.org>