JavaScript – Events Handling

Events and Event Handling

- An *event* is a notification that something specific has occurred, either with the browser or an action of the browser user
- An *event handler* is a script/code that is implicitly executed in response to the appearance of an event
- The process of connecting an event handler to an event is called *registration*
- Don't use `document.write` in an event handler because the output may go on top of the display

Click a button

- Method1: write code in HTML

```
<form id = "form1">
  <input type = "button"  id = "myButton"
       onclick = "alert('you clicked this button');" />
</form>
```

- Method2: assign handler in HTML

```
<form id = "form1">
  <input type = "button"  id = "myButton"
       onclick = "myButtonHandler()" />
</form>
```

Click a button

- Method3: assign handler in JavaScript
  - HTML

```
<form id = "form1">
  <input type = "button"  id = "myButton"/>
</form>
```

  - JavaScript

```
function myButtonHandler(){
                alert("You clicked the button!!");
}

document.getElementById("myButton").onclick = myButtonHandler;
```

Events and Attributes

Event	Tag Attribute
blur	onblur
change	onchange
click	onclick
dblclick	ondblclick
focus	onfocus
keydown	onkeydown
keypress	onkeypress
keyup	onkeyup
load	onload
mousedown	onmousedown
mousemove	onmousemove
mouseout	onmouseout
mouseover	onmouseover
mouseup	onmouseup
reset	onreset
select	onselect
submit	onsubmit
unload	onunload

(Some) Event attributes and their tags

Attribute	Tag	Description
onblur	<a>	The link loses the input focus
	<button>	The button loses the input focus
	<input>	The input element loses the input focus
	<textarea>	The text area loses the input focus
	<select>	The selection element loses the input focus
onchange	<input>	The input element is changed and loses the input focus
	<textarea>	The text area is changed and loses the input focus
	<select>	The selection element is changed and loses the input focus
onclick	<a>	The user clicks on the link
	<input>	The input element is clicked
ondblclick	Most elements	The user double clicks the left mouse button
onfocus	<a>	The link acquires the input focus
	<input>	The input element receives the input focus
	<textarea>	A text area receives the input focus
	<select>	A selection element receives the input focus

Event attributes and their tags (continued)

Attribute	Tag	Description
onkeydown	<body>, form elements	A key is pressed down
onkeypress	<body>, form elements	A key is pressed down and released
onkeyup	<body>, form elements	A key is released
onload	<body>	The document is finished loading
onmousedown	Most elements	The user clicks the left mouse button
onmousemove	Most elements	The user moves the mouse cursor within the element
onmouseout	Most elements	The mouse cursor is moved away from being over the element
onmouseover	Most elements	The mouse cursor is moved over the element
onmouseup	Most elements	The left mouse button is unclicked
onreset	<form>	The reset button is clicked
onselect	<input>	The mouse cursor is moved over the element
	<textarea>	The text area is selected within the text area
onsubmit	<form>	The *Submit* button is pressed
onunload	<body>	The user exits the document

Events and Event Handling (continued)

- The same attribute can appear in several different tags

  e.g., The `onclick` attribute can be in `<a>` and `<input>`

- *A HTML element gets focus in three ways:*
  1. When the user puts the mouse cursor over it and presses the left button
  2. When the user tabs to the element
  3. By executing the `focus` method
- Reference: all the events for HTML tags: http://www.w3schools.com/tags/tag_a.asp

The `load` event at the `body` element

- the `load` event - triggered when the loading of a document is **completed**

```
<head>
    <title> load.html </title>
    <script type = "text/javascript"  src = "load.js" >
    </script>
  </head>
<body onload = "loadGreeting();">
   This is my page<p/>
  </body>
```

load.js

```
// load.js
//   An example to illustrate the load event

// The onload event handler

function loadGreeting () {
  alert("WELCOME!!!");
}
```

Handling Events from Radio buttons

- Radio buttons - use the `onclick` property, treat different values for different choices

radio_click.html

```
<body>
  <form id = "myForm"  action = "">
      <label> <input type = "radio“
                     name = "planeButton"
                     value = "152"
                     onclick = "planeChoice(152)" />
                     Model 152 </label>
        …

  <script type = "text/javascript"  src = "radio_click.js" />
</body>
```

radio_click.js

```
function planeChoice (plane) {
  switch (plane) {
    case 152:
      alert("A small two-place airplane for flight
  training");
      break;
    …
    default:
      alert("Error in JS function planeChoice");
      break;
  }
}
```

radio_click2.html

```
<form id = "myForm"  action = "">
    <label> <input type = "radio"
                   name = "planeButton"
                   value = "152" />
        Model 152 </label>
    <br />
    <label> <input type = "radio"
                   name = "planeButton"
                   value = "172" />
        Model 172 (Skyhawk) </label>
    <br />
...
    <script type = "text/javascript" src = "radio_click2r.js">
    </script>
```

radio_click2r.js

```
// radio_click2r.js
//   The event registering code for radio_click2

var dom = document.getElementById("myForm");
dom.elements[0].onclick = planeChoice;
dom.elements[1].onclick = planeChoice;
```

radio_click2.js

```
function planeChoice () {
  var dom = document.getElementById("myForm");
  for (var index = 0; index < dom.planeButton.length; index++) {
    if (dom.planeButton[index].checked) {
      var plane = dom.planeButton[index].value;
      break;
    }
  }
  switch (plane) {
    case "152":
      alert("A small two-place airplane for flight training");
      break;
    case "172":
      alert("The smaller of two four-place airplanes");
      break;
      // ...
  }}
```

Assign handlers to element properties

- The disadvantage of specifying handlers by assigning them to element properties is that there is no way to use parameters

- The advantages of specifying handlers by assigning them to event properties are:
  1. It is good to keep HTML and JavaScript separate
  2. Can use parameters

Handling Events from Textbox and Password Elements

- The focus event
  - See forced blur()

Input validation

- Things that must be done:
  1. Detect the error and produce an `alert` message (or any other sort of notification)
  2. Put the element in focus (the `focus` function)
  3. Select the element, if there is a value (the `select` function)
  4. The handler returns `false`, if not valid

Exercise 1

- Have a form with a:
  - text field: to enter a series of numbers separated by ;
  - Button(s): to select the arithmetic operation (+ *)
  - a paragraph to display the result
    - Result in black
  - Another paragraph to display errors, if any
    - Error in red
- User inputs a series of numbers (separated by ;) and selects an operation, then presses the button to do the calculation.
- Display the result, or
- If there is an error, display it

Exercise 2

- Have a form with a:
  - text field: to enter a series of numbers separated by ;
  - **combo box: to select the arithmetic operation (+ *)**
  - **Button: to initiate the calculation**
  - a paragraph to display the result
  - Another paragraph to display errors, if any
- User inputs a series of numbers (separated by ;) and selects an operation, then presses the button to do the calculation.
- Display the result, or
- If there is an error, display it

Exercise 3

- Have a form with a:
  - text field: to enter a series of numbers separated by ;
  - combo box: to select the arithmetic operation (+ *)
  - Button: to initiate the calculation
    - **Button should be disabled if input is invalid**
  - a paragraph to display the result
  - Another paragraph to display errors, if any
- User inputs a series of numbers (separated by ;) and selects an operation, then presses the button to do the calculation.
- Display the result, or
- If there is an error, display it

Exercise 4

- Develop a calculator
  - Minimize the number of handlers (JS functions) as much as you can

Useful links

- HTML/XHTML standard event model (http://www.w3schools.com/TAGS/ref_eventattributes.asp)
- JavaScript object reference (http://www.w3schools.com/jsref/default.asp)