

FORMS & Event Handlers - Lecture 6 by Robert Berdan

Of all the things JavaScript can do its ability to process forms is perhaps its most useful feature. Forms may be used to process credit card transactions, enter passwords, perform calculations, collect information in surveys.

Information submitted with a form is usually processed using a server side CGI (Common Gateway interface) script that is usually written in Perl. Forms can also be sent using the mailto: command. JavaScript is often used to validate or encrypt the information typed into a form.

The basic form tags are:

`<Form></Form>` is used to create forms

`<input>` tags are used to create buttons, text boxes, check boxes, radio buttons etc.

`<Form name="myform" method="Post" Action="mailto: or /cgi-bin/program.pl">`

`<input type= text size="26" name="email">`

`<input type="button" onClick="functionName(this.form)">`

`<input type="submit" value="submit" Name="submit">`

`<input type='reset" value="reset" Name="reset">`

`</Form>`

The `onClick` event handler usually calls a function located within the head tags.

this keyword refers to the current object and is used to pass information from the event handler to the function.

The `this` keyword by itself refers to the current object which is usually a button in a form, by using "this.form" it instructs the information within the form to be passed to the function.

NOTE: remember to always provide a name for the form and each element (object) you use. The name reference allows you to reference objects in your JavaScript program .

Names of objects within the form should follow standard variable/function naming i.e. no special characters except underscore, no leading numbers, no spaces etc.

`<Form name="myform">` This name property is useful if you have several forms on the same page.

EVENT HANDLERS

Forms use a variety of Event handlers - events are actions the user performs while visiting your web site. There are currently 21 different event handlers in JavaScript 1.2. See you text page 8, Table 1.1 for list of 12. Event handlers usually exist in pairs according to their function. .

The basic syntax is:

`<TAG onEvent="event_handler()">`

E.g.

onLoad and onUnload
onMouseover and onMouseout
onError and onAbort
onChange and onSelect
onSubmit and onReset
onClick
onBlur and onFocus
onError and onAbort

onBlur - user left the object, when an object loses focus, it is no longer selected
onFocus - user made the object active
onChange - user changed the object
onSelect - user selected object e.g. user highlights contents of text box by clicking & dragging
onClick - user clicks on button and triggers a function or other operation
onError - triggered when an image cannot be loaded
onSubmit - generated prior to submit associated with the form object
onReset - generated prior to resetting the form

E.g. onSubmit - gives user another opportunity to change their choice

```
<Form onSubmit = "return(confirm('okay to submit?'))">
```

user can respond by pressing confirm OK or cancel

Buttons respond to onClick(), onBlur() and onFocus events.

Text boxes and text areas respond to onBlur(), onChange(), onFocus(), and onSelect() events.

onmouseover and onmouseout events are generated whenever the user moves the mouse cursor into or out of any form element, image or link.

E.g. onmouseover Event within a form used to provide information about the button.

```
<Form>  
<input type="button" value="Clear" Name="clearbutton" onmouseover="alert('The clear button  
clears all entered values');" >  
</Form>
```

Which Event Handlers Can Be Used?

Object Event Handlers Available
Selection list onBlur, onChange, onFocus
Text Element onBlur, onChange, onFocus, onSelect
Textarea element onBlur, onChange, onFocus, onSelect
Button element onClick
Checkbox onClick
Radio button onClick
Hypertext link onClick, onmouseover, onmouseout
Image map onmouseover, onmouseout
Reset button onClick

Submit button onClick
Document onLoad,onUnload,onError
Window onLoad,onUnload,onBlur,onFocus
Framesets onBlur, onFocus
Form onSubmit, onReset
Image onLoad,onError,onAbort

Note: not all objects within a form are capable or responding to all events.

The form Object

form Properties Description

form.action The URL that information is to be sent to, usually the address of a CGI script

form.elements[] An array of form elements

form.elements.length The length of the elements array

form.encoding String specifying how form data is to be encoded for transmission

form.method The method by which form data is to be submitted. Only GET and POST may be used.

form.target The name of the window in which the results of the form submission are to be displayed

form Methods

form.reset() Resets the form to its default value

form.submit() Causes the contents of the form to be submitted to the server

Exercises with Forms

Attached are a number of form exercises starting from simple to more complex. By the time you finish these examples you should be able to complete the assignment.

List of script exercises.

1. Create a text box and pass the information input into the text box to an alert box. This script demonstrates the use of this.form to pass information to the function SaySomething.
2. This modification of the first script shows you how to add a second input box so the output includes information typed into both text boxes.
3. This script includes two buttons that produce different output messages.
4. This script takes values input into the top text box and outputs them to the lower message box.
5. Radio buttons allow you to pick on selection from a group, in this case the selection will take you a different web page. To test this script you will have to create 3 web pages, page1.html, page2.html and page3.html.
6. This script uses a for loop to output to an alert box which button was selected.
7. Optional - this script does the same as #5 but does so without using a function within the head tags.

8. Optional - this script does the same as #5, but does not include a button

9 . This script checks to see if an answer typed into a text box is correct, if it is the value is submitted, if not the script resets the value in the textbox back to nothing or null.

10. This script checks if the answer to a question is correct, if it is, the value is submitted and an alert box outputs "Thanks for taking the test".

11. This script is modified from your textbook, see page 60 for an explanation of each line and is used to validate an e-mail address. If an invalid e-mail address is typed in an alert box pops up and says invalid e-mail.

12. This program displays the name of each active element on a form as its event handler triggered in a textarea box. This form should help you understand some of the events that occur when you select and different form elements.

13. Optional - Calculator using form elements script by Richard Tymko Assignment for next week.

Many large web sites have a password protected areas that can only be viewed by those knowing the password. You can add a password protected area quite easily to a website using the form and window objects and an understanding of the previous exercises.

Part I

Using a form, create a textbox where a user must type in a password and click on a button in order to access another web page. You will need to create another web page to test that the password works.

Part II

Once you complete this, create two versions of the password script so that the "hidden" web page:

- a) opens in new window leaving the current password window open
- b) opens the hidden page in the same document window, i.e. within the same the password window document.

Part III

c) On the hidden page you access with a password, create a button which returns you to the original page where you need to add a password.

HINT look up window.open() object