

CSC 121 Computers and Scientific Thinking

Fall 2005

Event-Driven Programming

1

Event-driven Pages



one popular feature of the Web is its interactive nature

- e.g., you click on buttons to make windows appear
- e.g., you enter credit card information in a form and submit it

pages that respond to user actions such as mouse clicks or form entries are known as *event-driven pages*

- JavaScript code can be combined with HTML elements such as buttons, text fields, and text areas to produce event-driven pages

an *event handler* is an HTML element that can be programmed to respond to a user's actions

- the simplest event handler is a button
- a button can be associated with JavaScript code that will execute when the button is clicked

Click here for free money!

2

Buttons and Forms

general form of a button element:

```
<input type="button" value="BUTTON_LABEL" onClick="JAVASCRIPT_CODE" />
```

- the TYPE attribute of the INPUT element identifies the element to be a button
- the VALUE attribute specifies the text label that appears on the button
- the ONCLICK attribute specifies the action to take place
 - any JavaScript statement(s) can be assigned to the ONCLICK attribute
 - this can be (and frequently is) a call to a JavaScript function

event handlers like buttons must be placed inside *form* elements

- a *form* is a grouping of buttons and other event handlers within a page, delimited by tags

```
<form name="FORM_NAME"> . . . </form>
```

- if the page is to contain more than one event handler (e.g., multiple buttons), all can be placed within the same form

3

Random Number Example

recall the task of generating random numbers

- earlier, we did this by embedding JavaScript code in SCRIPT tags
- each time the page was loaded in the browser, the code was executed and the random number was written into the HTML text using `document.write`

DRAWBACK: the user had to reload for each random number

ALTERNATIVE: place a button in the page with associated code for generating and displaying the random number

- each time the user clicks the button, the code for generating and displaying the number is executed

4

```

1. <html>
2. <!-- lucky.html                                Dave Reed -->
3. <!-- This page displays a lucky number at the click of a button. -->
4. <!------->
5.
6. <head>
7.   <title> Dave's Lucky Number </title>
8.   <script type="text/javascript"
9.     src="http://www.prenhall.com/reed/random.js">
10. </script>
11.   <script type="text/javascript">
12.     function DisplayNumber()
13.     // Results: displays random number in an alert window
14.     {
15.       var LuckyNum;
16.
17.       luckyNum = RandomInt(1, 100);
18.
19.       alert("Your lucky number is: " + luckyNum);
20.     }
21.   </script>
22. </head>
23.
24. <body>
25.   <div style="text-align:center">
26.     <h2>Lucky Dave's Gift To You</h2>
27.
28.     <p>Numbers rule our lives. If you would like the benefit of Lucky Dave's
29.     amazing powers of prognostication, click on the button below to receive
30.     your guaranteed lucky number for the day!</p>
31.
32.     <form name="LuckyForm">
33.       <input type="button" value="Click Here for Lucky Number"
34.         onClick="DisplayNumber();" />
35.     </form>
36.   </div>
37. </body>
38. </html>

```

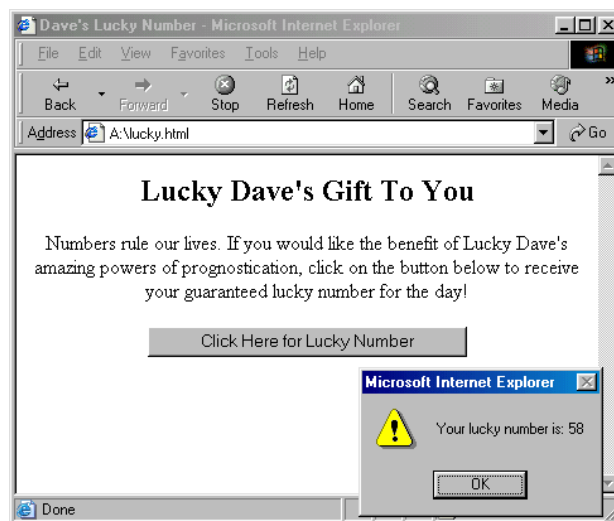
LuckyForm Example

when the button is clicked, the call `DisplayNumber();` is executed

the function generates a random number and displays it in an alert window

5

LuckyForm Example



6

Output vi a Text Boxes

a button provides a simple mechanism for user interaction in a Web page

- by clicking the button, the user initiates some action

a *text box* is an event-handler that can display text (a word or phrase)

- unlike an alert window, the text box appears as a box embedded in the page
- text can be written to the box by JavaScript code (i.e., the box displays output)



for example, we could reimplement the lucky number page using a text box
the text box containing the random number is embedded in the page
doesn't require the user to close the alert window after each number

7

Output vi a Text Boxes

general form of a text box element:

```
<input type="text" name="TEXTBOX_NAME" size=NUM_CHARS value="INITIAL_TEXT" />
```

- the TYPE attribute of the INPUT element identifies the element to be a text box
- the NAME attribute gives the element a name so that it can be referenced
- the SIZE attribute specifies the size of the box (number of characters that fit)
- the VALUE attribute specifies text that initially appears in the box

to display text in a text box, a JavaScript assignment is used to assign to its value attribute

- as part of the assignment, must specify the *absolute name* of the box
- the general form is:

```
document.FORM_NAME.TEXTBOX_NAME.value = VALUE_TO_BE_DISPLAYED;
```

8

```

1. <html>
2. <!-- lucky1.html Dave Reed -->
3. <!-- This page displays a lucky number in a text box. -->
4. <!------->
5.
6. <head>
7. <title> Fun with Forms </title>
8. <script type="text/javascript"
9.   src="http://www.prenhall.com/reed/random.js">
10. </script>
11. <script type="text/javascript">
12.   function DisplayNumber()
13.     // Assumes: document.LuckyForm.number exists in the page
14.     // Results: displays random number in the text box
15.     {
16.       var LuckyNum;
17.       LuckyNum = RandomInt(1, 100);
18.       document.LuckyForm.number.value = LuckyNum;
19.     }
20.   }
21. </script>
22. </head>
23.
24. <body>
25. <div style="text-align:center">
26. <h2>Lucky Dave's Gift To You</h2>
27.
28. <p>Numbers rule our lives. If you would like the benefit of Lucky Dave's
29.   amazing powers of prognostication, click on the button below to receive
30.   your guaranteed lucky number for the day!</p>
31.
32. <form name="LuckyForm">
33.   <input type="button" value="Click Here for Lucky Number"
34.     onClick="DisplayNumber();" />
35.   <br />
36.   <hr />
37.   Your lucky number is: <input type="text" name="number" size=4 value="" />
38. </form>
39. </div>
40. </body>
41. </html>
42.

```

Text Box for Displaying Output

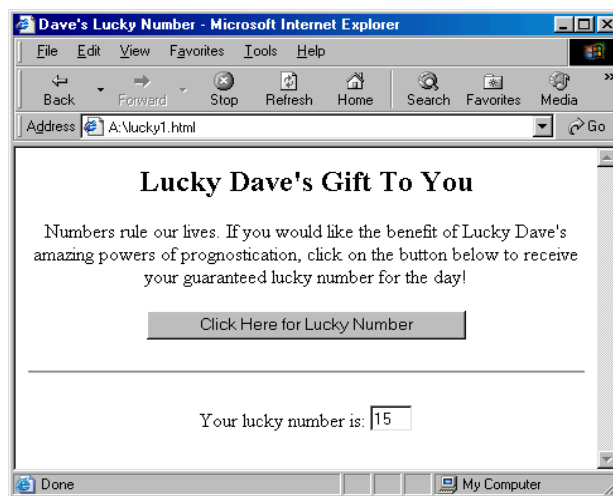
when the button is clicked, the function call `DisplayNumber()` is executed

the function generates the random number and assigns it to the text box

as a result, each button click yields a new number in the box

9

Text Box Example



10

Input via Text Boxes

text boxes can also be used for receiving user input

- the user can enter text directly into the box
- that text can then be accessed by JavaScript code via the absolute name of the box

```
document.FORM_NAME.TEXTBOX_NAME.value
```

- note that the value retrieved from a text box is always a string
 - if the user enters a number, say 93, then the absolute name will access "93"
 - similar to `prompt`, you must use `parseFloat` to convert the string to its numeric value

example: we can revisit our temperature conversion page

- the user enters the Fahrenheit temperature in a text box
- at the click of a button, the input is accessed and converted to Celsius
- another text box is used to display the converted temperature

11

```

1. <html>
2. <!-- convert1.html                               Dave Reed -->
3. <!-- This page converts temperatures from Fahrenheit to Celsius. -->
4. <!------->
5. <head>
6. <title>Temperature Conversion Page</title>
7.
8. <script type="text/javascript">
9.     function FahrToCelsius(tempInFahr)
10.         // Assumes: tempInFahr is a temperature in Fahrenheit
11.         // Returns: the equivalent temperature in Celsius
12.         {
13.             return (5/9) * (tempInFahr - 32);
14.         }
15.
16.     function Convert()
17.         // Assumes: document.TempForm.fahrBox contains degrees Fahrenheit
18.         // Results: assigns document.TempForm.celsiusBox the equiv. temperature
19.         {
20.             var tempF, tempC;
21.
22.             tempF = parseFloat(document.TempForm.fahrBox.value);
23.             tempC = FahrToCelsius(tempF);
24.
25.             document.TempForm.celsiusBox.value = tempC;
26.         }
27.     </script>
28. </head>
29.
30. <body>
31. <h2>Temperature Conversion Page</h2>
32. <hr /><br />
33. <form name="TempForm">
34.     Enter a temperature in degrees Fahrenheit:
35.     <input type="text" name="fahrBox" size=10 value="" />
36.     <br />
37.     <input type="button" value="Convert to Celsius" onClick="Convert();" />
38.     <br />
39.     Equivalent temperature in degrees Celsius:
40.     <input type="text" name="celsiusBox" size=10 value="" />
41. </form>
42. </body>
43. </html>

```

Text Boxes for Input

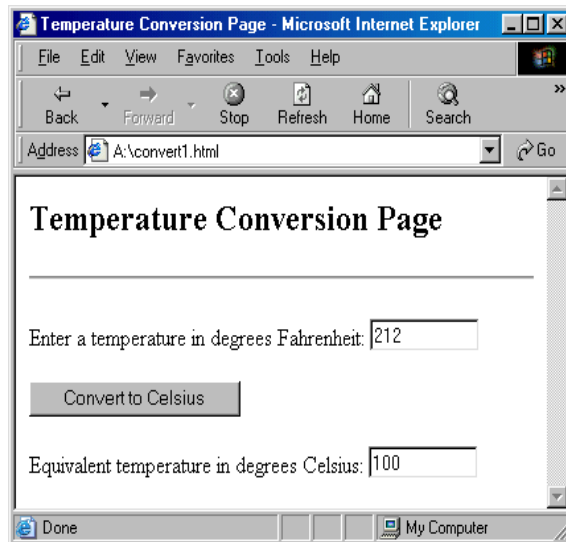
fahrBox is used for input

Convert is called when the button is clicked

celsiusBox is used for output

12

Text Boxes for Input



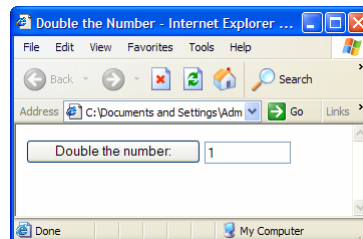
13

Input and Output



note: the same text box can be used for both input and output

- can modify the conversion page to allow for entering a temperature in either box, then convert to the other
- can write a simple page in which the user can enter a number, then double it by clicking a button



14

Text Areas

a text area is similar to a text box but it can contain any number of text lines

general form of a text area element:

```
<textarea name="TEXTAREA_NAME" rows=NUM_ROWS cols=NUM_COLS wrap="virtual">
INITIAL_TEXT
</textarea>
```

- the NAME attribute gives the element a name so that it can be referenced
- the ROWS attribute specifies the height (number of text lines) of the area
- the COLS attribute specifies the width (number of characters) of the area
- the WRAP attribute ensures that the text will wrap from one line to the next instead of running off the edge of the text area

unlike a text box, opening and closing tags are used to define a text area

- any text appearing between the tags will be the initial text in the text area
- otherwise, the contents of a text area are accessed/assigned in the same way

15

Input/Output via Text Areas

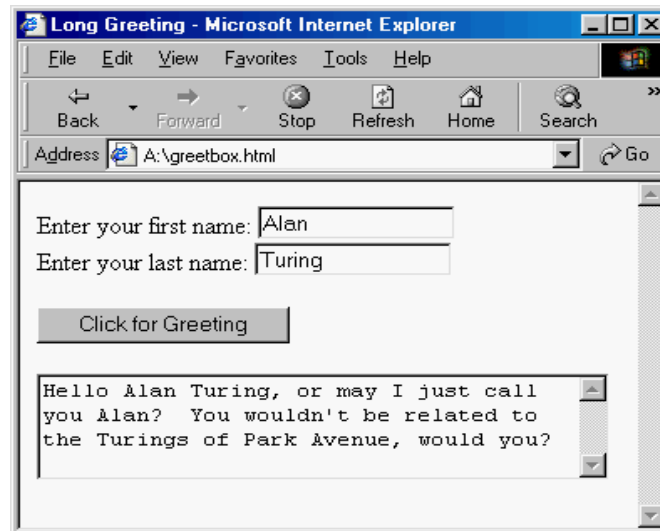
```
1. <html>
2. <!-- greetbox.html                                Dave Reed -->
3. <!-- This page displays a personalized greeting in a text area. -->
4. <!------->
5.
6. <head>
7. <title> Long Greeting </title>
8. <script type="text/javascript">
9.     function Greet()
10.        // Assumes: document.NameForm.firstName and
11.        //           document.NameForm.lastName contain names
12.        // Results: writes a message with those names in document.NameForm.message
13.        {
14.            var firstName, lastName, greeting;
15.
16.            firstName = document.NameForm.firstName.value;
17.            lastName = document.NameForm.lastName.value;
18.
19.            greeting = "Hello " + firstName + " " + lastName +
20.                ", or may I just call you " + firstName +
21.                "? You wouldn't be related to the " + lastName +
22.                "'s of Park Avenue, would you?";
23.
24.            document.NameForm.message.value = greeting;
25.        }
26.    </script>
27. </head>
28.
29. <body>
30. <form name="NameForm">
31.     Enter your first name: <input type="text" size=15 name="firstName" />
32.     <br />
33.     Enter your last name: <input type="text" size=15 name="lastName" />
34.     <br />
35.     <input type="button" value="Click for Greeting" onClick="Greet();" />
36.     <br />
37.     <textarea name="message" rows=4 cols=40 wrap="virtual"></textarea>
38. </form>
39. </body>
40. </html>
```

the user enters
first and last
names into text
boxes

a long greeting is
constructed using
the names and
assigned to the
text area

16

Text Area Example



17

Dynamic Images



just as you can use user-initiated events to change the contents of text areas and text boxes, you can also dynamically modify images

```

```

causes the image stored in the file `happy.gif` to appear in the page

you can change the image by reassigning its SRC attribute

- similar to the way that text boxes/areas have their VALUE attribute reassigned
- absolute name of an image: `document.images.IMAGE_NAME.src`

```
document.images.photo.src = "sad.gif";
```

replaces `happy.gif` with `sad.gif`

note: images do not need to be embedded in forms

18

