ADVANCE JAVASCRIPT

Outline

- Working with Browser Objects
  - Document Object Model (DOM)
    - Window, document, history, location Objects
    - Properties and Methods
  - Form Validation Script
- Creating Cookies in JavaScript
  - Constructing a standard cookie
  - Cookie Property
  - Interaction with the cookie
  - Using JavaScript to manipulate HTTP cookies

Using Browser Objects

- In the previous lectures, you were introduced to predefined objects in JavaScript
  - Math, String, Object, Boolean, Date, …
- JavaScript also provides you with objects that can control and manipulate the displays of browsers.
  - More dynamic and interactive.
- When a browser loads a webpage, it creates a number of JavaScript objects.

Document Object Model (DOM) history

- DOM is an object-oriented model that describes how all elements in an HTML page are arranged.
- It is used to locate any object in your HTML page (an unique address).
- There are different DOM levels
  - The Level 0 DOM (DOM0)
  - The Level 1 DOM (DOM1)
  - The Level 2 DOM (DOM2)
  - The Level 3 DOM (DOM3)

How the DOM works?

```
<script type="text/javascript">
function toggle( img ) {
  document.lamp1.src = img;     document.button1.src = img;
}  </script>

<a href="http://alphapeeler.sf.net" onmouseover="toggle('button_on.png')"
onmouseout="toggle('button_off.png')"><img name="button1"
src="button_off.png"></img></a> <BR>
<a href="http://alphapeeler.sf.net" > <img name="lamp1" src="button_off.png"
onmouseover="toggle('button_on.png')" onmouseout="toggle('button_off.png')" ></img></a>
```

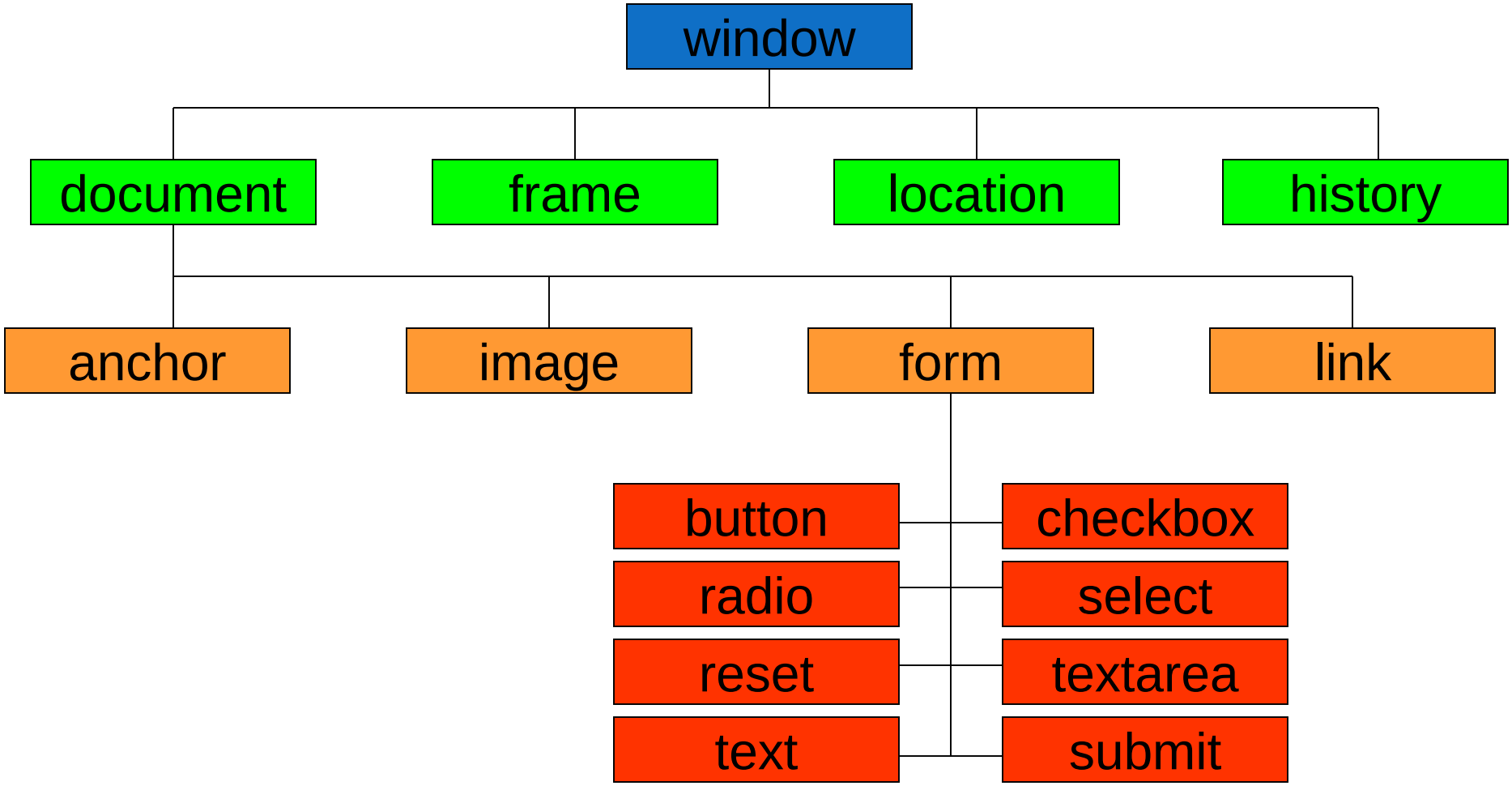
action

reaction

Action	Event	JavaScript	DOM	Reaction
src="button_off.png"	onmouseover	toggle()	document.img.button1	Src="button_on.gif"

1) User moves mouse over object
2) Event senses that something happened to the object
3) JavaScript tells the object what to do (Even handler)
4) Locates object on the web page
5) Object's image source is changed

Browser Hierarchy Model



The “window” Object

- It is the highest-level object in the JavaScript browser object hierarchy.
- It is the default object and is created automatically when a page is loaded.
- Since it is the default object, we may omit writing window explicitly.
  - document.write(“a test message”);
  - window.document.write(“a test message”);
- It also includes several properties and methods for us to manipulate the webpage.

Properties and methods of the "window" Object

Property	Description
length	An integer value representing the number of frames in the window
name	A string value containing the name of a window
parent	A string value containing the name of the parent window
status	A string value representing status bar text

Method	Description
alert(text)	Pop up a window with "text" as the message
close()	Closes the current window
open(url)	Open a new window populated by a URL.
setTimeout(expression, time)	Executes an expression after the elapse of the interval time.

Example of using the "window" Object

- Opening and closing windows
- Window attributes of the "open()" method

Attribute	Description
toolbar=yes|no|1|0	Creates the standard toolbar (IE & Firefox only)
location=yes|no|1|0	display the address field
status=yes|no|1|0	Creates the status bar
menubar=yes|no|1|0	Whether or not to display the menu bar
scrollbars=yes|no|1|0	Creates scrollbars when the document exceeds the window size (IE, Firefox & Opera only)
resizable=yes|no|1|0	Enables the user to resize the window (IE only)
width=pixels	Specifies the width of the window
height=pixels	Specifies the height of the window

http://www.w3schools.com/jsref/met_win_open.asp

The “document” Object

- It is one of the important objects in any window or frame.
- The document object represents a web document or a page in a browser window.
- When you access multiple sites simultaneously, there would be multiple windows opened.
  - Each window would have a corresponding window object, and each window object would have its own document object.

Properties and methods of the "document" Object

Property	Description
document.body.style.backgroundColor	background color of a document
document.alinkColor	color for active links
document.location	current URL
document.title	text specified by <title> tag

Method	Description
clear()	Clears the document window
open()	Open a document to receive data from a write() stream
write(content)	Writes the text of content to a document
writeln()	Writes the text and followed by a carriage return
close()	Closes a write() stream

The “history” Object

- Each time you visit a web page and click on the “Back” or “Forward” arrow buttons on your browser toolbar, you are accessing the history list.
- You can also add similar buttons / links that allow users to move backward and forward via the information stored in the history object.

Properties and methods of the "history" Object

Property	Description
length	number of urls in the history object

Method	Description
back()	Sends the user to the previous page in the history list
forward()	Sends the user to the next page in the history list
go(x)	Sends back or forward by "x" number of pages in the history list

The “form” Object

- The form object is accessed as a property of the document object.
- Each form element in a form (text input field, radio buttons), is further defined by other objects.
- The browser creates a unique “form” object for each form in a document.
- You can access the form object “form1”
  - document.form1

Form Element-Based Objects

- HTML forms can include eight types of input elements
  - Text fields, Textarea fields
  - Radio buttons
  - Check box buttons
  - Hidden fields
  - Password fields
  - Combo box select menu
  - List select menu
  - Each object has its own properties and methods.

Example 8: Form Validation Script

```
<script LANGUAGE="JavaScript">
function validate() {
    if (document.form1.yourname.value.length < 10) {
        alert("Please enter your full name.");
        return false;
    }
   if (document.form1.address.value.length < 15) {
       alert("Please enter your address.");
       return false;
   }
   if (document.form1.phone.value.length < 7) {
       alert("Please enter your phone number.");
       return false;
   }
   return true;
}
</script>
```

```
<body>
<h1>Form Example</h1>
<p>Enter the following information. When you press the Display
button, the data you entered will be validated, then sent by email.</p>

<form name="form1"  onSubmit="validate();">
<p><b>Name:</b> <input type="text" length="20" name="yourname">
</p>
<p><b>Address:</b> <input type="text" length="30" name="address">
</p>
<p><b>Phone: </b> <input type="text" length="15" name="phone">
</p>
<p><input type="submit" value="Submit"></p>
</form>
</body>
</html>
```

HTML DOM forms Collection

Property	Description
length	Returns the number of <form> elements in the collection. **Note:** This property is read-only

Method	Description
[*index*]	Returns the <form> element from the collection with the specified index (starts at 0). **Note:** Returns null if the index number is out of range
item(*index*)	Returns the <form> element from the collection with the specified index (starts at 0). **Note:** Returns null if the index number is out of range
namedItem(*id*)	Returns the <form> element from the collection with the specified id. **Note:** Returns null if the id does not exist

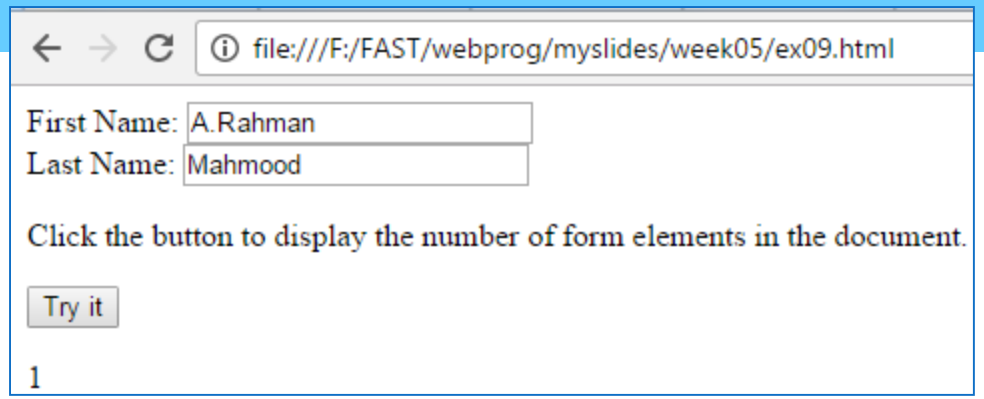
- Find out how many <form> elements there are in the document:
- var x = document.forms.length;

HTML DOM forms Collection

- Get the id of the first <form> element (index 0) in the document:
  - var x = document.forms[0].id;
- Get the id of the first <form> element (index 0) in the document:
  - var x = document.forms.item(0).id;
- Get the HTML content of the <form> element with id="myForm" in the document:
  - var x = document.forms.namedItem("myCarForm").innerHTML;

Example 9

```
<form>
  First Name: <input type="text" name="fname" value="A.Rahman"><br>
  Last Name: <input type="text" name="lname" value="Mahmood">
</form>
<p>Click the button to display the number of form elements in the document.</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    var x = document.forms.length;
    document.getElementById("demo").innerHTML = x; }
</script>
```



Example 10

```
<form id="myCarForm">
  Favorite Car: <input type="text" name="fname" value="Volvo">
</form>
<form id="myColorForm">
  Favorite Color: <input type="text" name="favcolor" value="Blue">
</form>
<p>Click to display id of 1st form element in the doc.</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    var x = document.forms[0].id;
    document.getElementById("demo").innerHTML = x;
}
</script>
```

Favorite Car: Volvo

Favorite Color: Blue

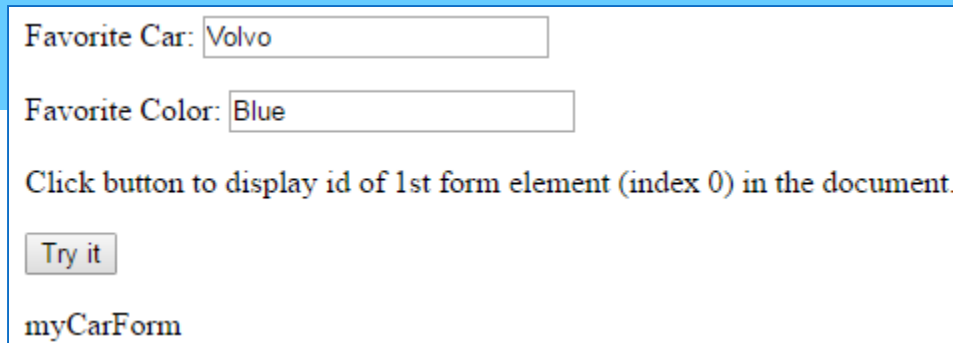
Click to display id of 1st form element in the doc.

Try it

myCarForm

Example 11

```
<form id="myCarForm">
  Favorite Car: <input type="text" name="fname" value="Volvo">
</form>
<form id="myColorForm">
  Favorite Color: <input type="text" name="favcolor" value="Blue">
</form>
<p>Click button to display id of 1st form element (index 0) in the document.</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    var x = document.forms.item(0).id;
    document.getElementById("demo").innerHTML = x;
}
</script>
```



Example 12

```
<form id="myCarForm">
  Favorite Car: <input type="text" name="fname" value="Volvo">
</form>
<form id="myColorForm">
  Favorite Color: <input type="text" name="favcolor" value="Blue">
</form>
<p>Click to display HTML content of form with id.</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    var x = document.forms.namedItem("myCarForm").innerHTML;
    document.getElementById("demo").innerHTML = x; }
</script>
```

Favorite Car: Volvo

Favorite Color: Blue

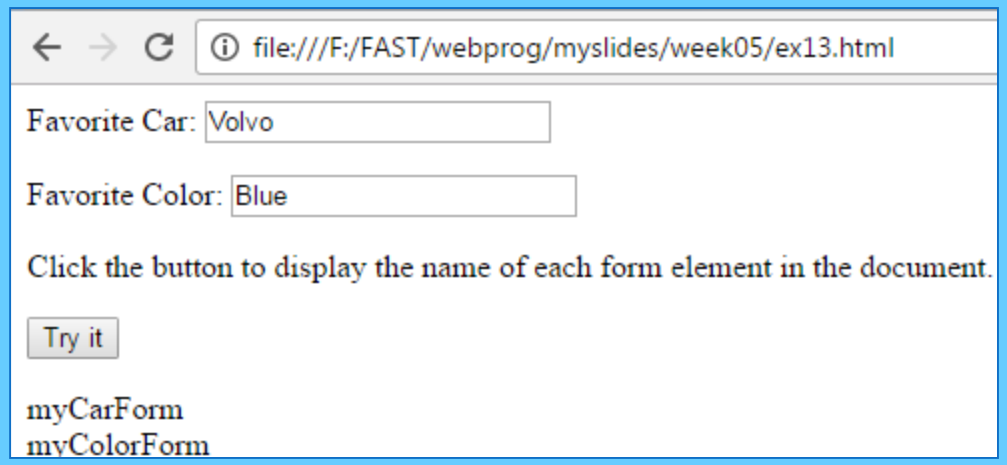
Click to display HTML content of form with id.

Try it

Favorite Car: Volvo

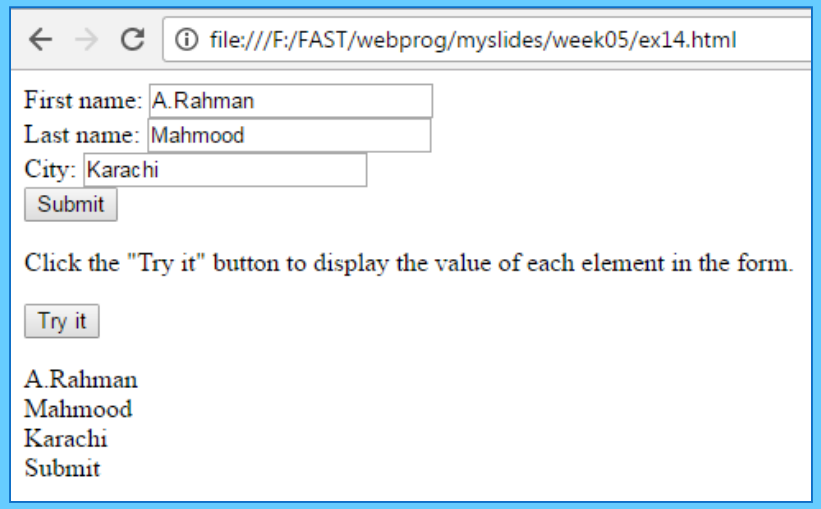
Ex13: HTML DOM forms Collection

```
<html> <body>
<form id="myCarForm">
  Favorite Car: <input type="text" name="fname" value="Volvo">
</form>
<form id="myColorForm">
  Favorite Color: <input type="text" name="favcolor" value="Blue">
</form>
<p>Click the button to display the name of each form element in the document.</p><button
type="button" onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    var x = document.forms;
    var txt = "";
    var i;
    for (i = 0; i < x.length; i++) {
        txt = txt +  x[i].id + "<br>";
    }
    document.getElementById("demo").innerHTML = txt;
}
</script> </body> </html>
```



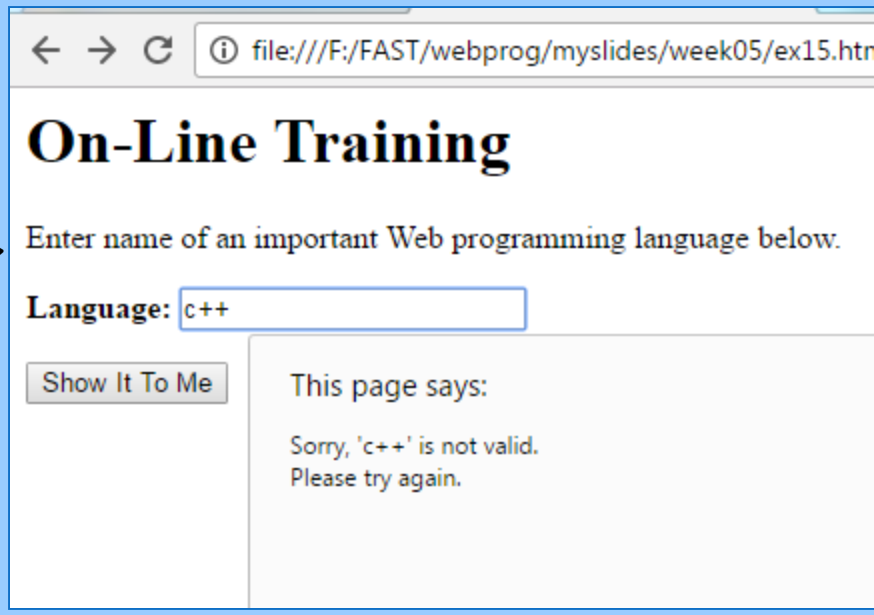
Ex 14: Using the **elements** collection together with document.forms to get the value of each element in the form:

```
<html> <body>
<form action="form_action.asp">
  First name: <input type="text" name="fname" value="A.Rahman"><br>
  Last name: <input type="text" name="lname" value="Mahmood"><br>
  City: <input type="text" name="city" value="Karachi"><br>
  <input type="submit" value="Submit">
</form>
<p>Click the "Try it" button to display the value of each element in the form.</p>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction() {
    var x = document.forms[0];
    var txt = "";
    var i;
    for (i = 0; i < x.length; i++) {
        txt = txt + x.elements[i].value + "<br>";
    }
    document.getElementById("demo").innerHTML = txt;
}
</script></body> </html>
```



Ex 15: Checking Form Values Individually

```
<html><head><title>On-Line Training</title>
<script language="JavaScript">
function checkLanguage() {
  // or document.forms["langForm"].elements["langField"]
  var field = document.langForm.langField;
  var lang = field.value;
  var prefix = lang.substring(0, 4).toUpperCase();
  if (prefix != "JAVA") {
    alert("Sorry, '" + lang + "' is not valid.\n" +
          "Please try again.");
    field.value = "";  // Erase old value
    field.focus();     // Give keyboard focus
  }
}
</script></head> <body><h1>On-Line Training</h1>
<form name="langForm">
Enter name of an important Web
programming language below.
<p> <b>Language:</b>
<input type="text" name="langField"
      onChange="checkLanguage()"> <p>
<input type="submit" value="Show It To Me">
</form> </body> </html>
```

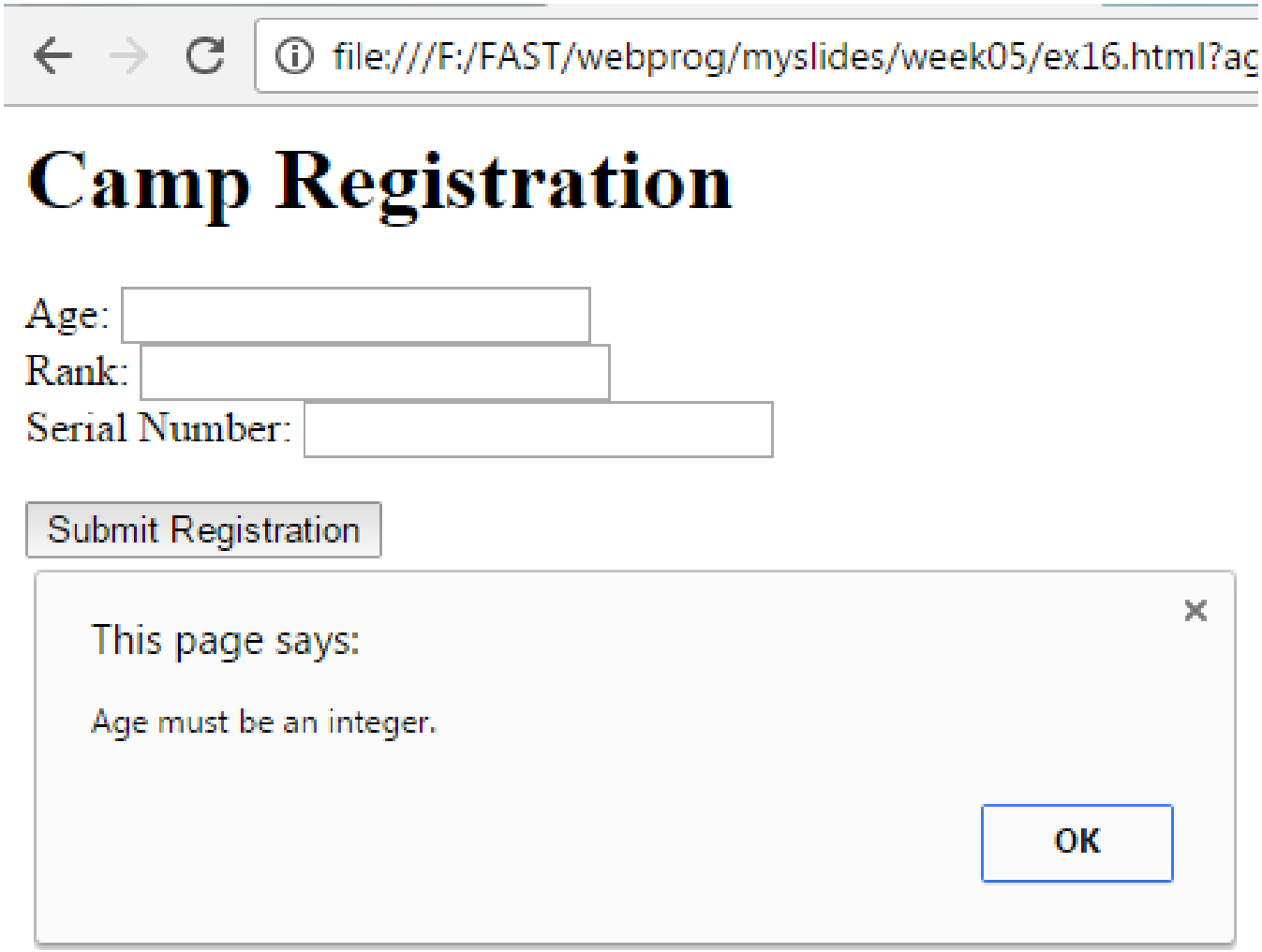


Ex: 16 Checking Values When Form is Submitted

```
<html><head><title>Camp
   Registration</title>
<script language="JavaScript">
function isInt(string) {
  var val = parseInt(string);
  return(val > 0); }
function checkRegistration() {
  var ageField =
    document.registerForm.ageField;
  if (!isInt(ageField.value)) {
    alert("Age must be an integer.");
    return(false);
  }
return(true);
}
</script>
```

```
<body><h1>Camp Registration</h1>
<form name="registerForm"
      onSubmit="return(checkRegistration())">
Age: <input type="text" name="ageField"><br>
Rank: <input type="text"
   name="rankField"><br>
Serial Number: <input type="text"
   name="serialField"><p>
<input type="submit" value="Submit
   Registration">
</form> </body> </html>
```

Checking Values When Form is Submitted, Results



Cookie Basics

- When a user closes the browser, the information contained in a hidden form field will be lost.
- A cookie is used to store information on the user's computer even when the user switches off his/her computer.
- It is a data that is sent from a web server to a web browser when the user visits a site on a server.
  - It is just a .txt file in a user's computer.

Features of Cookies

- A cookie can be associated with one or more documents on the web server.
- More than one cookie can be associated with a document on the web server.
- Every cookie has a NAME-VALUE pair associated with it.
- Cookies have an expiration date associated with them.

Cookies Applications

- Web page customization for each user
- Form information storage
- Shopping carts for order information
- Store userID and password
- Track how many times the user has visited
- Maintain a past score for each player on a test or online games

Cookie

- Create a Cookie with JavaScript
  - document.cookie = "username=John Doe";
- You can also add an expiry date (in UTC time). By default, the cookie is deleted when the browser is closed:
  - document.cookie = "username=John Doe; expires=Thu, 18 Dec 2013 12:00:00 UTC";
- Path parameter, you can tell the browser what path the cookie belongs to.
  - document.cookie = "username=John Doe; expires=Thu, 18 Dec 2013 12:00:00 UTC; path=/";

Cookie

- Read a Cookie with JavaScript
  - var x = document.cookie;
- Change a Cookie with JavaScript: change cookie same way as you create it:
- document.cookie = "username=John Smith; expires=Thu, 18 Dec 2013 12:00:00 UTC; path=/";
- Delete a Cookie with JavaScript: Just set the expires parameter to a passed date:
- document.cookie = "username=; expires=Thu, 01 Jan 1970 00:00:00 UTC; path=/;";

Cookie, Ex 17

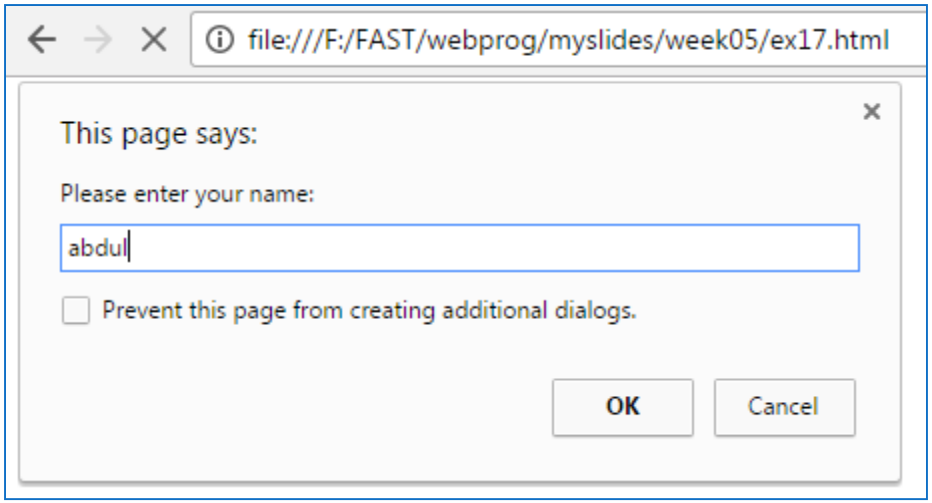
```
<html> <head> <script>
function setCookie(cname,cvalue,exdays) {
    var d = new Date();
    d.setTime(d.getTime() + (exdays*24*60*60*1000));
    var expires = "expires=" + d.toGMTString();
    document.cookie = cname + "=" + cvalue + ";" + expires + ";path=/";
}
function getCookie(cname) {
    var name = cname + "=";
    var decodedCookie = decodeURIComponent(document.cookie);
    var ca = decodedCookie.split(';');
    for(var i = 0; i < ca.length; i++) {
        var c = ca[i];
        while (c.charAt(0) == ' ') {
            c = c.substring(1);
        }
        if (c.indexOf(name) == 0) {
            return c.substring(name.length, c.length);
        }
    }
    return "";
}
```

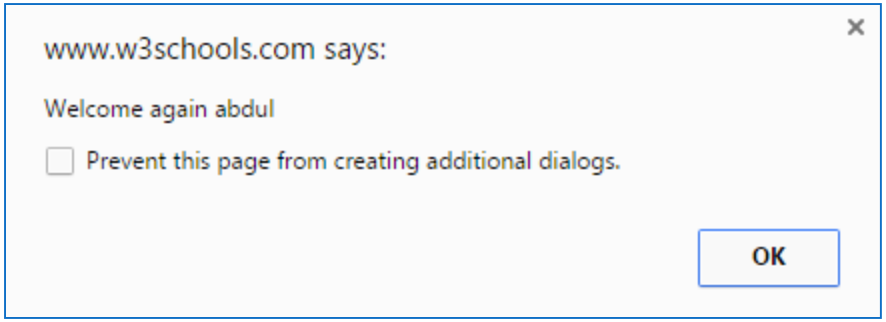
Cookie, Ex 17

```
function checkCookie() {
    var user=getCookie("username");
    if (user != "") {
        alert("Welcome again " + user);
    } else {
       user = prompt("Please enter your name:","");
       if (user != "" && user != null) {
           setCookie("username", user, 30);
       }
    }
}
</script> </head>
<body onload="checkCookie()"> </body> </html>
```

http://www.w3schools.com/js/tryit.asp?filename=tryjs_cookie_username

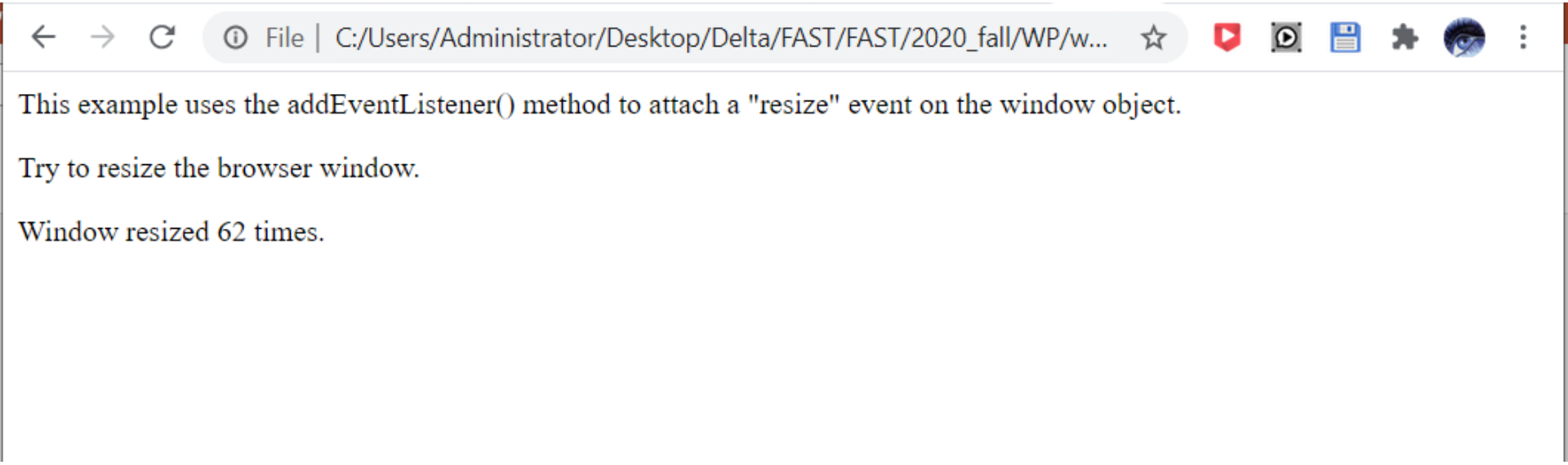
Cookie, Ex 17


file:///F:/FAST/webprog/myslides/week05/ex17.html
This page says:
Please enter your name:
abdul
Prevent this page from creating additional dialogs.
OK
Cancel


www.w3schools.com says:
Welcome again abdul
Prevent this page from creating additional dialogs.
OK

```
<body>
<p>This example uses the addEventListener() method to attach a
"resize" event on the window object.</p>
<p>Try to resize the browser window.</p>
<p>Window resized <span id="demo">0</span> times.</p>

<script>
window.addEventListener("resize", myFunction);
var x = 0;
function myFunction() {
    var txt = x += 1;
    document.getElementById("demo").innerHTML = txt;
}
</script>
</body>
```


File | C:/Users/Administrator/Desktop/Delta/FAST/FAST/2020_fall/WP/w...
This example uses the addEventListener() method to attach a "resize" event on the window object.
Try to resize the browser window.
Window resized 62 times.