Outlines of today's lecture

- Events
- Assigning events using DOM
- Dom nodes
- Browser Object Model (BOM)

IS 242 Web Application Development 1

Lecture 11: Introduction to JavaScript (Part 5)

Event Attributes

Attribute	The event occurs when...
onchange	**The content of a field changes**
onclick	**Mouse clicks an object**
ondblclick	**Mouse double-clicks an object**
oerror	**An error occurs when loading a document or an image**
onload	**A page or image is finished loading**
onmouseout	**The mouse is moved off an element**
onmouseover	**The mouse is moved over an element**

onchange Event

- The onchange event occurs when the content of a field changes.
- often used in combination with validation of input fields

```
<html>
  <head>
       <script type="text/javascript">
        function upperCase(x)
        {
        var y=document.getElementById(x).value
        document.getElementById(x).value=y.toUpperCase()
        }
      </script>
  </head>
  <body>
  Enter your name:
  <input type="text" id=“fname" onchange=“upperCase(this.id)" />

  </body>
</html>
```

onclick Event

The onclick event occurs when an object gets clicked.

```
<html>
    <body>

      Field1: <input type="text" id="field1" value="Hello World!" />
      <br />
      Field2: <input type="text" id="field2" />
      <br /><br />
      Click the button to copy the content of Field1 to Field2.
      <br />
      <button
onclick="document.getElementById('field2').value=document.getEleme
ntById('field1').value">Copy Text</button>

    </body>
</html>
```

ondbclick Event

The ondblclick event occurs when an object gets double-clicked.

```
<html>
 <body>
  Field1: <input type="text" id="field1" value="Hello World!" />
  <br />
  Field2: <input type="text" id="field2" />
  <br /><br />
  Double Click the button to copy the content of Field1 to Field2.
  <br />
  <button
 ondblclick="document.getElementById('field2').value=documen
 t.getElementById('field1').value">Copy Text</button>
 </body>
</html>
```

onerror Event

The onerror event is triggered when an error occurs with an element.

```
<html>
 <body>

  <img src="image.gif" onerror="alert('The image could not be loaded.')" />

  <p>In this example we refer to an image that does not exist, therefore we will get the alert box.</p>

 </body>
</html>
```

onload Event

The onload event occurs immediately after a page or an image is loaded.

```
<html>
 <head>
  <script type=“text/javascript">
  function load()
  {
  alert("Page is loaded”);
   }
  </script>
 </head>
 <body onload=“load()">
  <h1>Hello World!</h1>
 </body>
</html>
```

onmouseover and onmouseout Events

- The **onmouseover** event occurs when the mouse pointer moves over a specified object.
- The **onmouseout** event occurs when the mouse pointer moves away from a specified object.

```
<html>
<body>

<img src="image_w3default.gif" name="mousetest"
onMouseOver="document.images['mousetest'].src='image_w3default2.gif'"
onMouseOut="document.images['mousetest'].src='image_w3default.gif'" width="234" height="91" />

</body>
</html>
```

Assigning Events Using DOM

The HTML DOM allows you to assign events to HTML elements using JavaScript:

Assign an onclick event to a button element:

```
<button id="myBtn">Try it</button>
<p id="demo"></p>
<script>
document.getElementById("myBtn").onclick = displayDate;
function displayDate() {
    document.getElementById("demo").innerHTML = Date();
}
</script>
```

Checkbox checked attribute

```
<html>
  <head>
  <script type="text/javascript">
  function check() { document.getElementById("check1").checked=true; }
  function uncheck() { document.getElementById("check1").checked=false;}
  </script>
  </head>
<body>
  <form>
    <input type="checkbox" id="check1" />Do you like summer?
    <input type="button" value="Check Checkbox" onclick="check()" />
    <input type="button" value="Uncheck Checkbox" onclick="uncheck()" />
  </form>
</body>
</html>
```

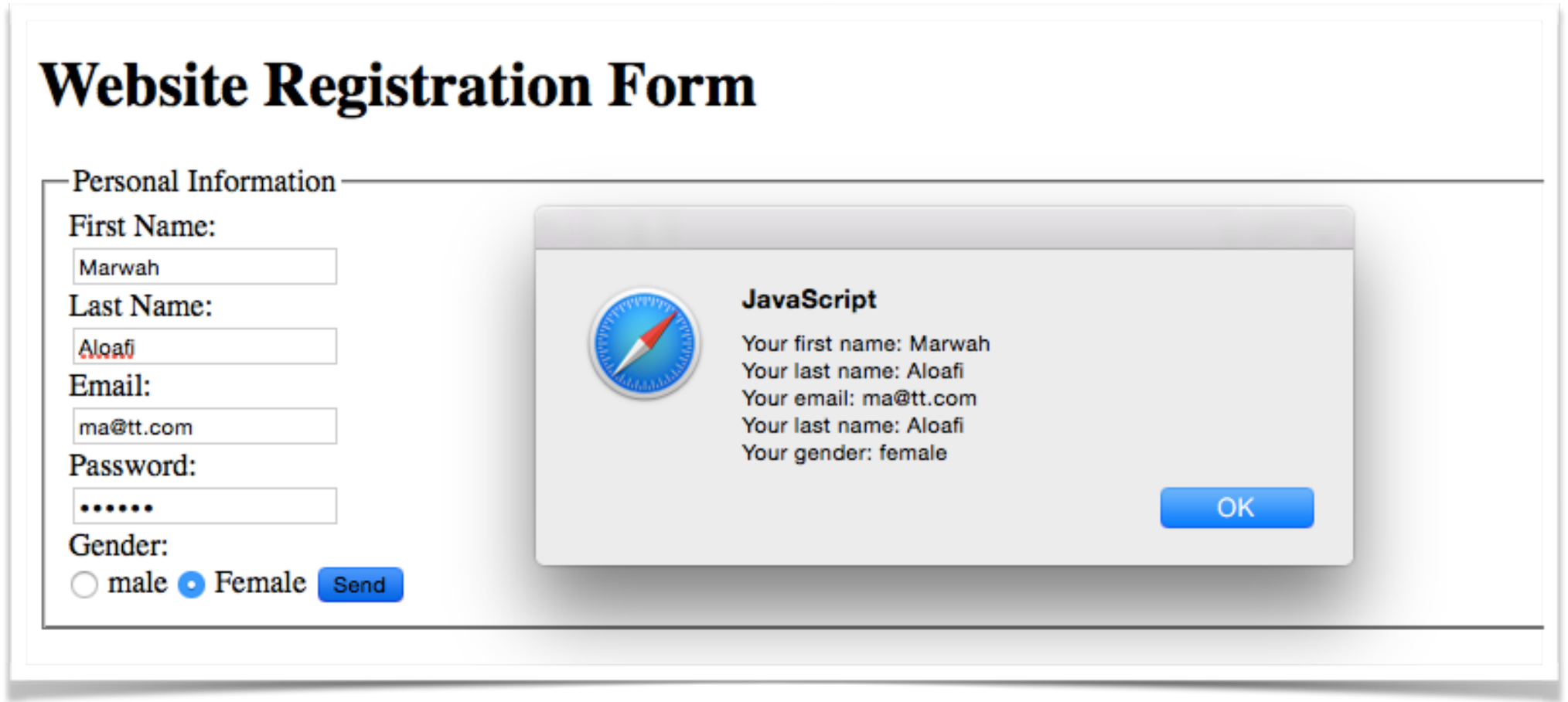
Radio Button value and checked attributes

```
<html>
  <head>
     <script type="text/javascript">
      function showFun()
     {
     var x="";
     if(document.getElementById("r1").checked==true)
     x=document.getElementById("r1").value;
     else if(document.getElementById("r2").checked==true)
     x=document.getElementById("r2").value;
     alert(x);
     }
     </script>
   </head>
  <body>
     Gender:<br/>

     <input type="radio" name="gender" value="Male" id="r1" /> Male <br/>
     <input type="radio" name="gender" value="Female" id="r2" /> Female <br/>
     <input type="button" value="show gender"  onclick="showFun()"/>

  </body>
</html>
```

Select selectedIndex Property

```
<html> <head>
<script type="text/javascript">
function displayResult()
{
var y=document.getElementById("car").options;
var x=document.getElementById("car").selectedIndex;
alert("x= " + x + " y[" + x + "]= " + y[x].value);
}
</script> </head>
<body>
Select your Car:<br />
<select id="car">
<option value ="Kia"> KIA </option>
<option value ="bmw"> BMW </option>
<option value ="jaguar"> JAGUAR </option>
<option value ="golf"> GOLF </option>
</select>
<input type="button" value="showCar" onclick="displayResult()">
</body> </html>
```

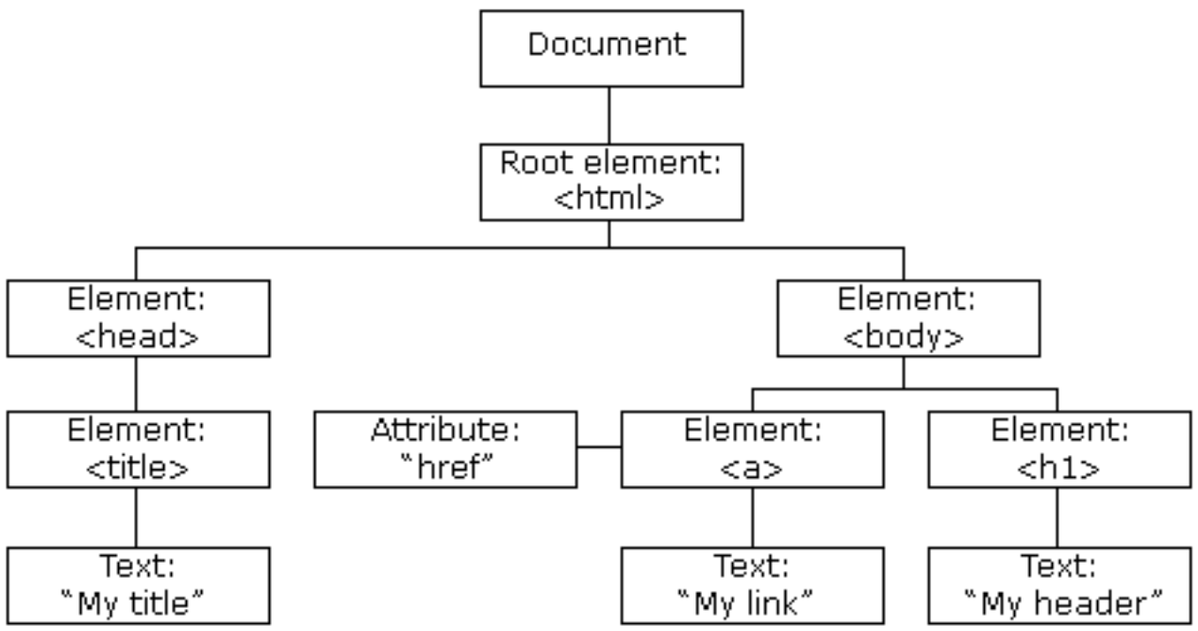
Demo !


Website Registration Form
Personal Information
First Name:
Marwah
Last Name:
Aloafi
Email:
ma@tt.com
Password:
Gender:
male
Female
Send
JavaScript
Your first name: Marwah
Your last name: Aloafi
Your email: ma@tt.com
Your last name: Aloafi
Your gender: female
OK

DOM Nodes

According to the W3C HTML DOM standard, everything in an HTML document is a node:

- The entire document is a document node
- Every HTML element is an element node
- The text inside HTML elements are text nodes
- Every HTML attribute is an attribute node
- All comments are comment nodes



Navigating Between Nodes

You can use the following node properties to navigate between nodes with JavaScript:

- parentNode
- childNodes[nodenumber]
- firstChild
- lastChild
- nextSibling
- previousSibling

Node Properties-Examples

In addition to the innerHTML property, you can also use the childNodes and nodeValue properties to get the content of an element.

```
<html>
  <body>

    <h1 id="intro">My First Page</h1>

    <p id="demo">Hello!</p>

    <script>
    var myText =
document.getElementById("intro").childNodes[0].nodeValue;
    document.getElementById("demo").innerHTML = myText;
    </script>

  </body>
</html>
```

Node Properties-Examples

Using the firstChild property is the same as using childNodes[0]:

```
<html>
 <body>

   <h1 id="intro">My First Page</h1>

   <p id="demo">Hello World!</p>

   <script>
   myText =
document.getElementById("intro").firstChild.nodeValue;
   document.getElementById("demo").innerHTML = myText;
   </script>

 </body>
</html>
```

The Browser Object Model (BOM)

- **The Browser Object Model (BOM)** is a collection of objects that give you access to the browser and the computer screen. These objects are accessible through the global object **window**

- The **window object** is supported by all browsers. It represent the browser's window.

- There's a **window** object for every frame, iframe, pop up, or browser tab.

- All global JavaScript **objects**, **functions**, and **variables** automatically become members of the window object.

The setInterval () Method

- The **setInterval()** method will wait a specified number of milliseconds, and then execute a specified function, and it will continue to execute the function, once at every given time-interval.

- **Syntax**

```
window.setInterval("javascript function", milliseconds);
```

- The **window.setInterval()** method can be written without the window prefix.
- The **first parameter** of setInterval() should be a **function**.
- The **second parameter** indicates the length of the **time-intervals** between each execution.

- **Note:** There are 1000 milliseconds in one second.
- **Example**

Alert "hello" every 3 seconds:

```
setInterval(function () {alert("Hello")}, 3000);
```

The setTimeout () Method

- Syntax
- window.setTimeout("javascript function", milliseconds);
- The window.setTimeout() method can be written without the window prefix.
- The **setTimeout()** method will wait the specified number of milliseconds, and then execute the specified function.
- The **first parameter** of setTimeout() should be a **function.**
- The **second parameter** indicates how many **milliseconds**, from now, you want to execute the first parameter.
- **Example**

Click a button. Wait 3 seconds. The page will alert "Hello":

```
<button onclick =
"setTimeout(function(){alert('Hello')},3000)">Try
it</button>
```

References

- www.w3schools.com
- Stefanov, S. (2013). Object-Oriented JavaScript. Packt Publishing Ltd.
- Deitel & Deitel (2011). *Internet and World Wide Web How to Program, 5th Edition, Harvey & Paul Deitel & Associates.*