LECTURE-2

- Functions review
- HTML Forms
- Arrays
- Exceptions
- Events

JAVASCRIPT FUNCTIONS, REVIEW

Syntax

```
function <functionName> (params)
{
    // code
}
```

Note: Parameters do NOT have variable type.

1. Recall: Function definition can be in
   - <head> part of HTML file.
   - <body> portion of HTML file
   - An external file.
2. "return" value of the function is optional.

FUNCTIONS – EXAMPLE1

```
<html>
  <head>
    <title> Example of a simple function </title>
    <script type="text/javascript">
        function factorial (input)
         {
            product = 1;
            for (i=1; i <= input; i++)
                product *= i;
            document.write ("factorial of i " + product);
         }
    </script>
  </head>
  <body>
    <h1> Example of a simple function </h1>
      <script type="text/javascript">
        factorial (9);
      </script>
  </body>
</html>
```

FUNCTIONS – EXAMPLE2

```
<html>
 <head>
  <title>Browser Information example</title>
  <script type="text/javascript">
   function BrowserInfoFn( )  {
    var browser = navigator.appName;
    var version = navigator.appVersion;
    var ver = parseFloat (version);
    document.write ("Broswer: " + browser + " version: “  + version + " ver: " + ver + "<br />");
   }
  </script>
 </head>
 <body>
  <h1>Browser Information example</h1>
  <script type="text/javascript">
   BrowserInfoFn( );
  </script>
  <hr>
</body>
</html>
```

SPECIAL FUNCTIONS IN JAVASCRIPT

encodeURI	encodes special characters of a URI, except: , / ? : @ & = + $ #
encodeComponentURI	Encodes special characters and , / ? : @ & = + $ # of a URI
decodeURI	Opposite of encodeURI
decodeComponentURI	Opposite of decodeComponentURI
escape	encodes special characters, except: * @ - _ + . /
unescape	Opposite of escape – decodes a string
eval	Evaluates and executes a string as Javascript code
isFinite	Finds out if argument is a finite, valid number
isNaN	Finds out if argument is not a number
Number	Converts a string to integer
String	Converts argument to string
parseFloat	Parses argument and returns a float value
parseInt	Parses argument and returns an integer value

ARRAYS

- Arrays: Hold multiple objects
  - E.g., array of strings, array of numbers, etc.

E.g., var mycars = [“Toyota”, “Honda”, “BMW”];

or

```
var mycars = new Array( );
mycars[0] = ”Toyota";
mycars[1] = ”Honda";
mycars[2] = "BMW";
```

or

```
var myCars=new Array(”Toyota","Honda","BMW");
myCars.push(“Acura”, “Lexus”);  // Add more cars
document.write (myCars);        // Toyota, Honea, BMW,  Acura, Lexus
myCars.pop( );                  // Get the last car – here Lexus
```

ARRAYS

- Useful array functions
  - push – Add an element at the end
  - pop – Remove the last element added
  - length – Get the number of elements added
  - toString – Convert to a string. Elements are ”,” separated
  - shift – Removes and returns first element
  - Unshift – adds an element at the beginning

- Ref: https://www.w3schools.com/js/js_array_methods.asp

VALUES OF JS POPUP BOXES

- Confirm box
  - var i = confirm (“Press OK or Cancel”)
- Prompt box
  - var i = prompt (“Enter some value”, “default”);
- Alert box
  - alert (“alert text”);

HTML FORMS

- We covered some HTML tags earlier.
- HTML form
  - Another HTML tag
  - Useful to send information from browser to server
  - Can use other HTML tags
    - <input>
    - <button>
    - <submit>
    - <select> and <option>
    - <textarea>
- Javascript functions can be used to verify HTML forms' input

HTML FORM EXAMPLE

```
<input id="id1" type="number" min="100" max="300" required>
<button onclick="myFunction()">OK</button>

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

<script>
function myFunction() {
    var inpObj = document.getElementById("id1");
    if (inpObj.checkValidity() == false) {
        document.getElementById("demo").innerHTML =
              inpObj.validationMessage;
    }
}
</script>
```

EXCEPTIONS

- Syntax and usage
  - Similar to Java/C++ exception handling

```
try
{
  // your code here
}
catch (excptn)
{
  // handle error
  // optional throw
}
```

EXCEPTIONS – EXAMPLE

```
<html>
 <head>
   <title> Error handling example </title>
   <script type="text/javascript">
    function handleError( )  {
       try  {
          i = foo( ); // Calling an undefined function raises an exception
          return i;
        }
       catch (err)  {
          alert ("Something was wrong");
          return ("Error!");
        }
     }
   </script>
 </head>

 <body>
   <script type="text/javascript">
     var i = handleError();
     document.write ("i: " + i);
   </script>
 </body>
</html>
```

THROW

- Syntax
  - throw (exception)

Example

```
try {
   // some code
}
catch (error) {
   // handle error
   throw (error2)
}
```

THROW EXAMPLE

```
<html>
 <body>
  <script type="text/javascript">
    var x = prompt ("Enter a number between 0 and 10:","");
     try {
         if (x > 10)
            throw "Err1";
         else if (x < 0)
            throw "Err2";
      }
       catch (er) {
         if (er == "Err1")
            alert ("Error! The value is too high");
         if (er == "Err2")
            alert ("Error! The value is too low");
     }
  </script>
 </body>
</html>
```

HANDLING ERRORS – ONERROR

- onerror – an event to handle errors.

onerror = handleErr // Call handleErr on errors

```
function handleErr(msg,url,l)
// msg – error msg, url – current URL, l – line #
{
  //Handle the error here
  return true or false
}
```

EVENTS

- Events in Javascript – "something" happening. Examples
  - Web page is loaded/unloaded
  - Mouse key clicked/double-clicked
  - Mouse hovering over/out-of a region
  - Any keyboard key is pressed/released
  - An error has occurred
  - A "submit" or "reset" button is pressed.
  - An element gets or loses focus.
- Complete list of events are given at

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

EVENTS – ONLOAD( )

Called (if defined) when a web page is loaded.

Simple Example

```
<html>
  <head>
    <title> On Load event example </title>
    <script type="text/javascript">
    function onloadFn( )          // Function definition
     {
        alert ("Web page finished loading");
     }
    </script>
  </head>
  <body onload="onloadFn( )">// Call it when page is loaded
</body>
</html>
```

EVENTS – ONUNLOAD

- Opposite of onload
  - Called when
    - We go out of a web page or
    - A web page is re-loaded.
- Example
  - Same example as before
    - Except replace onload with onunload.

EVENTS – ONERROR

- Recall our example of error handling

Example:

```
onerror=handleErr // Call handleErr on errors

function handleErr(msg,url,l)
// msg – error msg, url – current URL, l – line #
{
   //Handle the error here
   return true or false
}
```

EVENTS – ONSUBMIT, ONRESET

- onsubmit – Event when a submit button is pressed.
  - E.g., When using forms.
- onreset – event when a reset button is pressed.
  - Typically, used to cancel/reset the values of all fields.

EVENTS – ONMOUSEUP, ONMOUSEDOWN, ONMOUSEOVER, ONMOUSEOUT

- onmousedown – event when a mouse button is pressed down.
- onmouseup – event when a mouse button is pressed up.
- onmouseover – event when a mouse hovers over (a specific region).
- onmouseout – event when a mouse comes out (of a specific region).

EVENTS – ONKEYPRESS, ONKEYDOWN, ONKEYUP

- onkeypress – Event when a key is pressed
- onkeydown – Event when a key is pressed or held down
  - Similar to onkeypress
- onkeyup – Event when a key is released (after being pressed)

EVENTS – ONCLICK, ONDBLCLICK, ONCHANGE

- onclick – event when a button is clicked
- ondblclick – event when a button is double clicked
  - Try to avoid onclick when ondblclick is defined

EVENTS – ONFOCUS, ONBLUR, ONRESIZE

- onfocus – event when an element gets focus
- onblur – event when an element loses focus
  - Opposite of onfocus

EVENTS – ONRESIZE, ONCHANGE, ONABORT

- onresize – event when a browser window is resized (changed).
- onchange – event when the value of a field changes
- onabort – event when loading of an image is interrupted.

TIMING EVENTS

- setTimeout: execute something after a given time
  - Syntax: var t = setTimeout (code, time_in_msec);
  - Similar to sleep
  - Difference with sleep: Code "setTimeout" is executed immediately, no timeout there.
- clearTimeout: Cancel a timeout condition
  - Syntax: clearTimeout (t)
    - "t" was the variable returned by setTimeout