شركــة تحالــف الإمـــارات للحـــلـــول الـتـقـنـيــة ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

JavaScript

Tahaluf Training Center 2021

شركــة تحالــف الإمـــارات للحـــلـــول الـتـقـنـيــة ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Day 4

JS Events

HTML events are **"things"** that happen to HTML elements.

When JavaScript is used in HTML pages, JavaScript can **"react"** on these events.

An HTML event can be something the browser does, or something a user does.

JS Events

JavaScript's interaction with HTML is handled through events that occur when the user or the browser manipulates a page.

When the page loads, it is called an event. When the user clicks a button, that click too is an event. Other examples include events like pressing any key, closing a window, resizing a window, etc.

JS Events

Developers can use these events to execute JavaScript coded responses, which cause buttons to close windows, messages to be displayed to users, data to be validated, and virtually any other type of response imaginable.

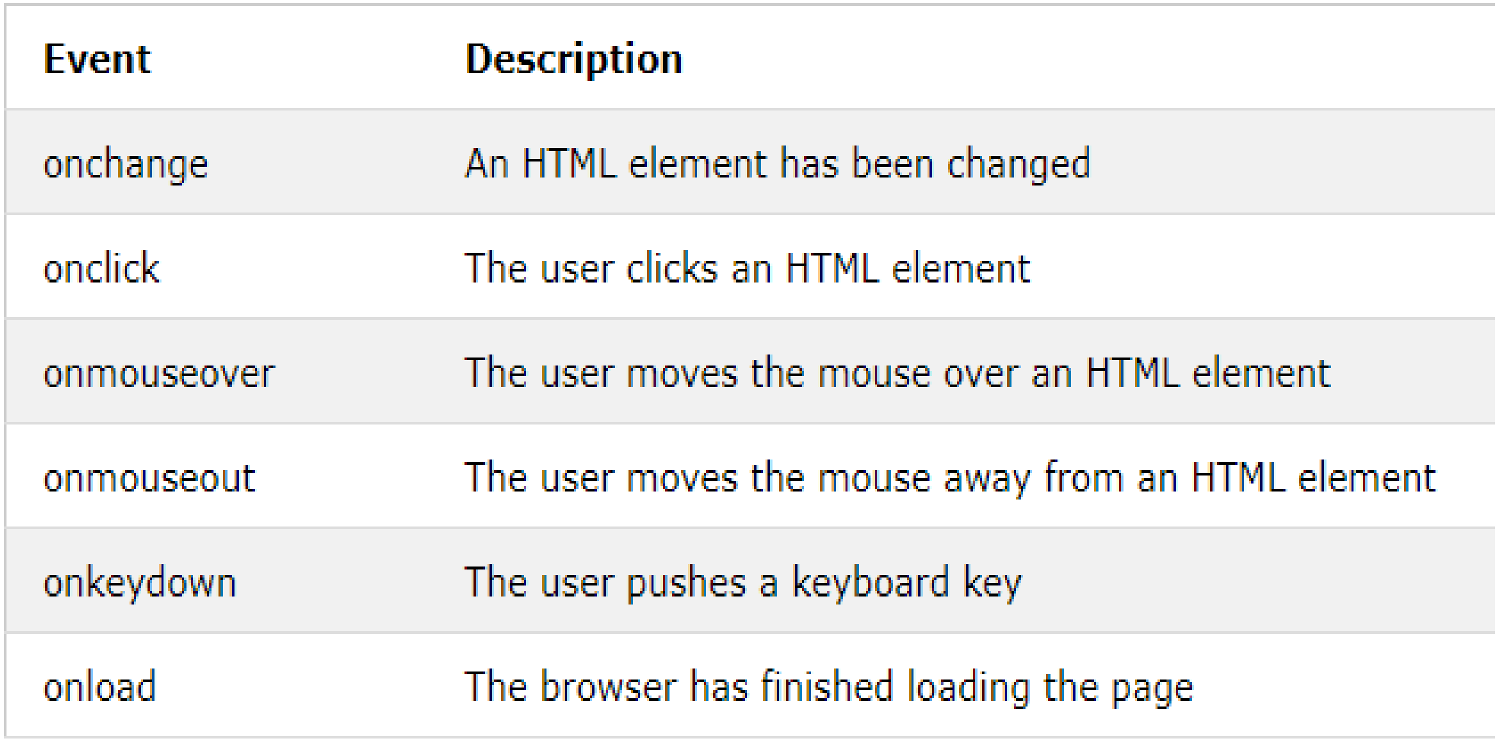
Events are a part of the Document Object Model (DOM) Level 3 and every HTML element contains a set of events which can trigger JavaScript Code.

JS Events

```
<element event='some JavaScript'>

<element event="some JavaScript">
```

JS Events



Event	Description
onchange	An HTML element has been changed
onclick	The user clicks an HTML element
onmouseover	The user moves the mouse over an HTML element
onmouseout	The user moves the mouse away from an HTML element
onkeydown	The user pushes a keyboard key
onload	The browser has finished loading the page

JS Events

```
<button
onclick="document.getElementById('demo').
innerHTML=Date()">The time is?</button>

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

JS Events

```
<p>Click the button to display the date.</p>

<button onclick="displayDate()">The time is?</button>

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


<script>
        function displayDate() {
  document.getElementById("demo").innerHTML = Date();
}
</script>
```

JS Events

```
<p>Click the following button and see result</p>
<form>
      <input type = "button" onclick = "sayHello()"
      value = "Say Hello" />
</form>


          <script>
                function sayHello() {
                     alert('Hello!!')
                }
          </script>
```

JS Events

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

```
<label>Enter your name: </label>
<input type="text" id="fname" onchange="myFunction()“ />

<p>When you leave the input field, a function is triggered
which transforms the input text to upper case.</p>
```

JS Events

```
function myFunction() {
    var x = document.getElementById("fname");
    x.value = x.value.toUpperCase();
}
```

JS Events

addEventListener() method

```
<body>
    <p> using the addEventListener() to a button.</p>
    <button id="btnDate">Click Me</button>
    <p id="test"></p>
    <script>
document.getElementById("btnDate")
.addEventListener("click", displayDate);

function displayDate() {
document.getElementById("test").innerHTML = Date();  }
    </script>
</body>
```

JS Events

```
<p onmouseover="Over(this)" onmouseout="Out(this)“>

        Get the Mouse over me and check the change
    </p>

    <script>
        function Over(obj) {
            obj.innerHTML = "Mouse <b>Over</b>
the Element"
        }

        function Out(obj) {
            obj.innerHTML = "Mouse <b>Out</b>
the Element"
        }
    </script>
```

JavaScript strings

JavaScript strings are used for storing and manipulating text.

```
let studentName =  “Ahmad“;   // Double quotes
let studentName2 = 'Ahmad‘;  // Single quotes
```

String Methods

length

To find the length of a string, we use the built-in length property.

```
let str = ’We are Tahaluf students’;
let strln = str.length;
```

String Methods

toUpperCase() and toLowerCase()

A string is converted to upper case and to lower case.

```
let stName = ’Ahmad’;
let newStName = stName.toUpperCase();
```

String Methods

TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

slice()

Extracts a part of a string and returns the extracted part in a new string.

The method takes 2 parameters: the start position, and the end (If you omit the second parameter, the method will slice out the rest of the string).

```
let str = "Apple, Banana, Kiwi";
let res = str.slice(-12, -6);
```

String Methods

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

substring()

`substring()` is similar to `slice()`.
The difference is that `substring()` cannot accept negative indexes.

```
let str = "Apple, Banana, Kiwi";
let res = str.substring(7, 13);
```

String Methods

Read more about JS String Methods, we have more than the mentioned methods, such as:

indexOf(), search(), substr(), replace() and trim()

String Methods

Strings Can be Objects

Normally, JavaScript strings are primitive values, created from literals:

```
let stName = "Ahmad";
```

But strings can also be defined as objects with the keyword `new`:

```
let stName = new String("Ahmad");
```

JS Array

JavaScript arrays are used to store multiple values in a single variable.

An array can hold many values under a single name, and you can access the values by referring to an index number.

Arrays provide a lot of methods. To make things easier.

JS Array

```
<h2>JavaScript Arrays</h2>

 <p>JavaScript array elements are accessed using
  numeric indexs.</p>

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

 <script>
     var cars = ["Saab", "Volvo", "BMW"];
     cars[0] = "Opel";
     document.getElementById("demo").innerHTML = cars;
 </script>
```

JS Array

```
// The length property returns the number of elements
var x = cars.length;

// The sort() method sorts arrays
var y = cars.sort();

console.log(x);
console.log(y);
```

Array Methods

```
var fruits =
["Banana", "Orange", "Apple", "Mango"];

// First sort the elements of fruits
fruits.sort();

// Then reverse the order of the elements
fruits.reverse();
```

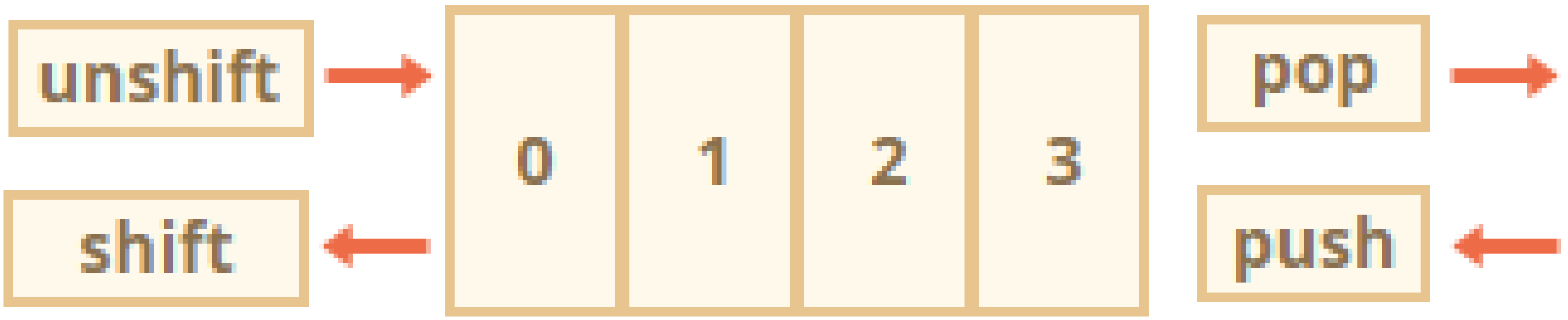
Array Methods

Add/Remove Items

We already know methods that add and remove items from the beginning or the end:

- arr.push(...items) – adds items to the end,
- arr.pop() – extracts an item from the end,
- arr.shift() – extracts an item from the beginning,
- arr.unshift(...items) – adds items to the beginning.

Array Methods



Array Methods

```
<h2>JavaScript Arrays</h2>
<p>The push method appends a new element to an array.</p>
<button onclick="myFunction()">Push to Array</button>
<p id="demo"></p>
```

```
<script>
     var fruits = ["Banana", "Orange", "Apple", "Mango"];
     document.getElementById("demo").innerHTML = fruits;

     function myFunction() {
         fruits.push("Lemon");
         document.getElementById("demo").innerHTML = fruits;
     }
 </script>
```

Array Methods

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];

// Removes the last element ("Mango") from fruits
fruits.pop();
```

Array Methods

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];

// Removes the first element "Banana" from fruits
fruits.shift();
```

Array Methods

```
var fruits =
["Banana", "Orange", "Apple", "Mango"];

// Adds a new element "Lemon" to fruits
fruits.unshift("Lemon");
```

Array Methods

How to delete an element from the array?
The arrays are objects, so we can try to use delete:

```
let arr = ["I", "go", "home"];
delete arr[1]; // remove "go"
console.log( arr[1] ); // undefined
// now arr = ["I", , "home"];
console.log( arr.length ); // 3
```

The element was removed, but the array still has 3 elements, we can see that arr.length == 3.

Array Methods

```
arr.splice(start[, deleteCount, elem1, ..., elemN])
```

It modifies arr starting from the index start, removes deleteCount elements and then inserts elem1, ..., elemN at their place. Returns the array of removed elements.

Array Methods

```
let arr = ["I", "study", "JavaScript"];
arr.splice(1, 1); // from index 1 remove 1 element
console.log( arr ); // ["I", "JavaScript"]
```

Easy, right? Starting from the index 1 it removed 1 element.

Array Methods

```
let arr = ["I", "study", "JavaScript", "right", "now"];

// remove 3 first elements and replace them with another
arr.splice(0, 3, "Let's", "dance");

console.log(arr);
// now ["Let's", "dance", "right", "now"]
```

Array Methods

The method arr.slice is much simpler than similar-looking arr.splice.

arr.slice([start], [end])

It returns a new array copying to it all items from index start to end (not including end).

Both start and end can be negative, in that case position from array end is assumed.

Array Methods

شركـــة تحالـف الإمـــارات للحــلـــول التـقـنيـــة ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

```
let arr = ["t", "e", "s", "t"];

alert( arr.slice(1, 3) );
// e,s (copy from 1 to 3)

alert( arr.slice(-2) );
// s,t (copy from -2 till the end)
```

Array Methods

The JavaScript method toString() converts an array to a string of (comma separated) array values.

```
<h2>JavaScript Array Methods</h2>

<h2>toString()</h2>

<p>The toString() method returns an array as a
   comma  separated string:</p>

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

Array Methods

```
 var fruits = ["Banana", "Orange", "Apple", "Mango"];

document.getElementById("demo").innerHTML =
fruits.toString();
```

Array Methods

The join() method also joins all array elements into a string.

It behaves just like toString(), but in addition you can specify the separator.

Array Methods

```
<h2>JavaScript Array Methods</h2>

<h2>join()</h2>

<p>The join() method joins array elements into a
  string.</p>

<p>It this example we have used " * " as a
separator between the elements:</p>

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

Array Methods

```
var fruits =
["Banana", "Orange", "Apple", "Mango"];

document.getElementById("demo").innerHTML =
fruits.join("*");
```

Array Methods

The map() method creates a new array by performing a function on each array element.

The map() method does not execute the function for array elements without values.

The map() method does not change the original array.

Array Methods

```
<h2>JavaScript Array.map()</h2>
<p>Creates a new array by performing a function on each
array element.</p>
<p id="demo"></p>

<script>
    var numbers1 = [45, 4, 9, 16, 25];
    var numbers2 = numbers1.map(myFunction);

    document.getElementById("demo").innerHTML = numbers2;

    function myFunction(value, index, array) {
        return value * 2;
    }
</script>
```

Array Methods

```
let result = arr.map(function(item, index, array)
{ // returns the new value instead of item });
```

```
let lengths = ["Bilbo", "Gandalf",
"Nazgul"].map(item => item.length);

alert(lengths); // 5,7,6
```

Day Four Task

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

On the E-Learning Portal