Lecture # 10

JavaScript

Ms. Hafiza Alia
(alia@theprotec.com)

Js Dates

- Create the object of Date
  - var d = new Date();
  - new Date()
  - new Date(year, month, day, hours, minutes, seconds, milliseconds)

new Date()

- **new Date()** creates a new date object with the **current date and time**:
  - var d = new Date();

new Date(*year, month, ...*)

- 7 numbers specify year, month, day, hour, minute, second, and millisecond (in that order)
- 6 numbers specify year, month, day, hour, minute, second
- 5 numbers specify year, month, day, hour, and minute
- 4 numbers specify year, month, day, and hour
- var d = new Date(2018, 11, 24, 10, 33, 30, 0);

Note:
JavaScript counts months from 0 to 11.
January is 0. December is 11.

Date display methods

- d = new Date();
- The **toUTCString()** method converts a date to a UTC string (a date display standard).
  - document.getElementById("demo").innerHTML = d.toUTCString();
- The **toDateString()** method converts a date to a more readable format:
  - document.getElementById("demo").innerHTML = d.toDateString();

JavaScript Get Date Methods

Method	Description
getFullYear()	Get the **year** as a four-digit number (yyyy)
getMonth()	Get the **month** as a number (0-11)
getDate()	Get the **day** as a number (1-31)
getHours()	Get the **hour** (0-23)
getMinutes()	Get the **minute** (0-59)
getSeconds()	Get the **second** (0-59)
getMilliseconds()	Get the **millisecond** (0-999)
getTime()	Get the time (milliseconds since January 1, 1970)
getDay()	Get the weekday as a number (0-6)
Date.now()	Get the time. ECMAScript 5

JavaScript Set Date Methods

- The setFullYear() Method
- The setFullYear() method can **optionally** set month and day:

```
<script>
var d = new Date();
d.setFullYear(2020);
document.getElementById("demo").innerHTML = d;
</script>
```

JavaScript Set Date Methods

```
<script>
var d = new Date();
d.setFullYear(2020, 11, 3);
document.getElementById("demo").innerHTML =
d;
</script>
```

JavaScript Set Date Methods

```
<script>
var d = new Date();
d.setMonth(11);
document.getElementById("demo").innerHT
ML = d;
</script>
```

JavaScript Set Date Methods

```
<script>
var d = new Date();
d.setDate(20);
document.getElementById("demo").innerHTML =
d;
</script>
```

JavaScript Set Date Methods

- The setDate() method can also be used to **add days** to a date:
-

```
<script>
var d = new Date();
d.setDate(d.getDate() + 50);
document.getElementById("demo").innerHTML =
d;
</script>
```

JavaScript Set Date Methods

- The **setHours()** method sets the hours of a date object (0-23):
- The **setMinutes()** method sets the minutes of a date object (0-59):
- The **setSeconds()** method sets the seconds of a date object (0-59):

Assignment

- Get DOB from user
- And display his age in years months and days

JavaScript Math Object

- Math.PI; // returns 3.141592653589793
- Math.round(4.7); // returns 5
- Math.round(4.4); // returns 4
- Math.pow(8, 2); // returns 64
- Math.sqrt(64); // returns 8
- Math.abs(-4.7); // returns 4.7
- Math.ceil(4.4); // returns 5
- Math.floor(4.7); // returns 4
- Math.min(0, 150, 30, 20, -8, -200); // returns -200
- Math.max(0, 150, 30, 20, -8, -200); // returns 15

Math.random();

- Math.random() returns a random number between 0 (inclusive), and 1 (exclusive):

- Math.random() * 10; // returns a random integer from 0 to 9
- Math.random() * 11; // returns a random integer from 0 to 10
- Math.random() * 100; // returns a random integer from 0 to 99

JavaScript Errors

- JavaScript Errors - Throw and Try to Catch

- The try statement lets you test a block of code for errors.
- The catch statement lets you handle the error.
- The throw statement lets you create custom errors.
- The finally statement lets you execute code, after try and catch, regardless of the result.

JavaScript try and catch

- The **try** statement allows you to define a block of code to be tested for errors.

- The **catch** statement allows you to define a block of code to be executed, if an error occurs in the try block.
- Syntax
  - try {
    Block of code to try
    }
    catch(err) {
    Block of code to handle errors
    }

Example

```
<p id="demo"></p>
<script>
try {
    adddlert("Welcome guest!");
}
catch(err) {
    document.getElementById("demo").innerHTML = err.message;
}
</script>
```

Error Object (err.name, err.message)

Property	Description
name	Sets or returns an error name
message	Sets or returns an error message (a string)

Error Name Values

- different values can be returned by the error name property:

Error Name	Description
RangeError	A number "out of range" has occurred
ReferenceError	An illegal reference has occurred
SyntaxError	A syntax error has occurred
TypeError	A type error has occurred
URIError	An error in encodeURI() has occurred

Range Error

- A **RangeError** is thrown if you use a number that is outside the range of legal values.
-

```
var num = 1;
try {
    num.toPrecision(500);    // A number cannot have 500 significant digits
}
catch(err) {
    document.getElementById("demo").innerHTML = err.name;
}
```

Reference Error

- A **ReferenceError** is thrown if you use (reference) a variable/fnction that has not been declared:

```
var x;
try {
func();
Var b=a;
}
catch(err) {
    document.getElementById("demo").innerHTML = err.name;
}
```

Syntax Error

- A **SyntaxError** is thrown if you try to evaluate code with a syntax error.

```
try {
    eval("alert('Hello)");    // Missing ' will produce an error
}
catch(err) {
    document.getElementById("demo").innerHTML = err.name;
}
```

Type Error

- A **TypeError** is thrown if you use a value that is outside the range of expected types:

```
var num = 1;
try {
    num.toUpperCase();   // You cannot convert a number to upper case
}
catch(err) {
    document.getElementById("demo").innerHTML = err.name;
}
```

URI (Uniform Resource Identifier) Error

- A **URIError** is thrown if you use illegal characters in a URI function:

- ```
  try {
      enURI("%%%");    // You cannot URI decode percent signs
  }
  catch(err) {
      document.getElementById("demo").innerHTML = err.name;
  }
  ```
  ```

Maps

```
function mymap()
{
var mapdiv=document.getElementById("abc");
 var mappos = new google.maps.LatLng(24.8614622,67.0099388);
 var mapOptions = {center: mappos, zoom: 17};
 var map = new google.maps.Map(mapdiv, mapOptions);
 var marker =new google.maps.Marker({position: mappos});

marker.setMap(map);
 var myinfo=new google.maps.InfoWindow({content: "IBa City Campus"});
 myinfo.open(map,marker);
}
</script>
<script src="https://maps.googleapis.com/maps/api/js?callback=mymap"></script>
```

JavaScript
End Of Lecture 10