

JavaScript (JS)

*Prepared By: Eng. Jack Fayez
National Telecommunication Institute - NTI*



Introduction To JavaScript

- *JavaScript is an interpreted scripting language, and it is a free and Open Source.*
- *The most popular on the internet, and works in all major browsers.*
- *Object-based scripting language.*
- *Allows adding interactivity to a web page, such as Form Validation.*
- *JavaScript consists of three main parts:*
 - *ECMAScript that provides the core functionality.*
 - *The DOM that provides interacting with web pages elements.*
 - *The BOM that provides API for interacting with web browsers.*

Client-side vs. Server-side JavaScript

- *When JavaScript is used on a web page, it is executed in the web browsers of user's computers. In this case, JavaScript works as a client-side language.*
- *JavaScript can run on both web browsers and servers. A popular server-side environment for JavaScript is Node.js. Unlike the client-side JavaScript, the server-side JavaScript executes on the server that allows you to access databases, file systems, etc.*

Are Java and JavaScript the Same?

- *Java and JS are two completely different languages in Concept, Design and Coding.*
- *Java is developed by Sun Microsystems*
- *JavaScript is Developed by Netscape.*
- *Java is a powerful and complex programming language - in the same category as C and C++.*
- *JS is a lightweight programming language.*

- *JavaScript is used to:*
 - ✓ *Improve the design.*
 - ✓ *Add interactivity to HTML pages.*
 - ✓ *Client-side validation.*
 - ✓ *Displaying clocks, pop-up windows, and dialog boxes.*
 - ✓ *Create cookies.*
- *JavaScript can be inserted into the `<head>` section or into the `<body>` section of the HTML document using the `<script>` tag.*
- *Also the JavaScript can be attached to the HTML web page as a separated file.*

Adding JS to <head> Section.

Syntax

```
<html>
```

```
<head>
```

```
<title>.....</title>
```

```
<script language="javascript" type="text/javascript" >
```

JavaScript Code Goes Here

```
</script>
```

```
</head>
```

```
<body>.....</body>
```

```
</html>
```

Adding JS to <body> Section.

Syntax

`<html>`

`<head><title>.....</title></head>`

`<body>`

`<script language="javascript" type="text/javascript" >`

JavaScript Code Goes Here

`</script>`

`</body>`

`</html>`

Syntax

```
<html>
```

```
<head>
```

```
<title>.....</title>
```

```
<script type="text/JavaScript" src="file1.js"></script>
```

```
</head>
```

```
<body>
```

```
</body>
```

```
</html>
```


➤ Case Sensitivity

Everything in JavaScript including variables, function names, class names, and operators are case-sensitive. It means this means that “x”, “X” are not the same.

➤ Semicolons

All statements should end in a semicolon (;). The semicolon separates one statement from another.

➤ White Spaces

JavaScript, like HTML, ignores extra white spaces, tabs, and newlines that appear in statements.

JavaScript Comments are used to add information about the code to:

- *Make code easy to understand.*
- *Avoid the code from being executed as the JavaScript comments are ignored by the JavaScript engine.*

JavaScript supports both single-line and multiple line comments.

- *Single line comment*

Example

```
// this is a single-line comment
```

➤ Multiple line comment

A block comment starts with a forward slash and asterisk (/) and ends with the opposite (*/).*

Example

*/**

*This is a block comment that can
span multiple lines*

**/*



JavaScript's Events

Events are the triggers that call (start) functions. It could be an action such as clicking on a button, placing a mouse over an image or Submitting a form.

➤ The available event handlers in JavaScript are:

- `onClick()`
- `onSubmit()`
- `onMouseOver()`
- `onMouseOut()`
- `onFocus()`
- `onChange()`
- `onBlur()`
- `onLoad()`
- `onUnload()`



Js Alerts, Prompts, and Confirms.



JS Alerts

Alert is “modal window” (mini-window) with a message and “ok” button. The word “modal” means that the visitor can’t go through the rest of the page, until he press “OK”.

Syntax

```
alert("message");
```

Example

```
<body>
```

```
  <script>
```

```
    alert("Hello World!");
```

```
  </script>
```

```
</body>
```

From this page

Hello World!

OK

Example: With Button

```
<body>
```

```
<input type="button" value="click me"
```

```
onclick="alert('Welcome!');">
```

```
</body>
```

From this page

Hello World!

OK

The background features a dynamic splash of blue water on the left side. Overlaid on this is a circular clock face with a blue globe as its center. The globe shows the Americas. The clock has black hands and a red second hand. Numbers 1 through 12 are visible around the perimeter of the clock face.

JS Prompt

Prompt is a modal window with a text message, an input field, and “OK/Cancel” buttons.

Syntax

```
prompt("message", "default value");
```

Example

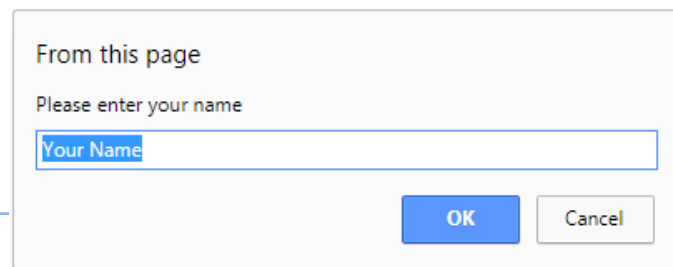
```
<body>
```

```
<script>
```

```
prompt('Please enter your name', 'Your Name');
```

```
</script>
```

```
</body>
```



A screenshot of a JavaScript prompt dialog box. The dialog has a title bar that says "From this page". Below the title bar, the text "Please enter your name" is displayed. There is a text input field containing the text "Your Name". At the bottom right of the dialog, there are two buttons: "OK" and "Cancel".

A decorative background on the left side of the slide. It features a blue and white water splash graphic that flows upwards. Overlaid on this is a clock face where the numbers are represented by a globe showing the Americas. The clock has black hands and a red second hand. The overall theme is dynamic and modern.

JS Confirm

Prompt is a modal window with a question and two buttons: “OK” and “Cancel”.

Syntax

```
confirm("question");
```

Example

```
<body>
```

```
<script >
```

```
confirm('Are you really over 21 years of age?');
```

```
</script>
```

```
</body>
```

From this page

Are you really over 21 years of age?

OK

Cancel



Generating Js Outputs

In JavaScript there are several different ways of generating output including:

- *Displaying Output in Alert Dialog Boxes.*
- *Writing Output to the Browser Window.*
- *Writing Output to Browser Console.*
- *Inserting Output Inside an HTML Element.*

Displaying Output in Alert Dialog Boxes

Alert dialog boxes can be used to display the message or output data to the user. An alert dialog box is created using the `alert()` method.

Syntax

```
alert("output");
```

Example

```
alert("Hello World!");
```


Writing Output to the Browser Window

- You can use the `document.write()` method to write the content to the current document.
- The `write()` method is mostly used for testing: If it is used after an HTML document is fully loaded, it will delete all existing HTML.

Syntax

```
document.write("output");
```

Syntax

```
document.write("output");
```

Example

```
<body>
```

```
<script>
```

```
document.write("<h1>This is a heading</h1>");
```

```
document.write('<p>This is a paragraph</p>');
```

```
</script>
```

```
</body>
```

Writing Output to Browser Console

- *Message and Data can be easily written to the browser console using the `console.log()` method. This is a simple, but very powerful method for generating detailed output.*

Syntax

```
console.log("output");
```

Example

```
console.log("Hello World!");
```

A decorative background on the left side of the slide. It features a blue and white water splash graphic that flows from the top left towards the bottom. Overlaid on this splash is a circular clock face. The clock face has a blue globe as its background, showing continents in white. The clock has black hands and a red second hand. The numbers 1 through 12 are visible around the perimeter of the clock face, though some are partially obscured by the water splash.

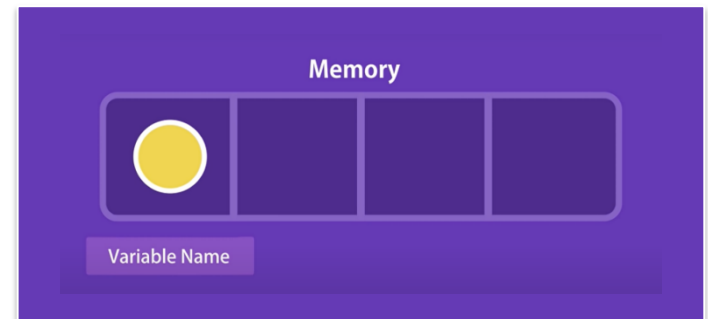
JavaScript's Variables

Variables are just named placeholders for values.

JavaScript variables are loosely typed; i.e. variables can hold values with any type of data.

JavaScript's Variables

- *Like other programming languages, JS has variables. Variable is simply a name of storage container (location) used to store data value and then refer to the data simply by naming this container.*
- *The Rules for constructing variables names:*
 - ✓ *Cannot be a reserved word.*
 - ✓ *Can only contain letters, digits, underscores, and dollar signs and cannot contain space or hyphen (-).*
 - ✓ *Must begin with a letter or with \$ or with _*
 - ✓ *Variable names are case-sensitive.*



JavaScript's Variables

JavaScript Reserved Words

<i>abstract</i>	<i>else</i>	<i>instanceof</i>	<i>switch</i>	<i>volatile</i>	<i>void</i>
<i>boolean</i>	<i>enum</i>	<i>int</i>	<i>synchronized</i>	<i>while</i>	<i>double</i>
<i>break</i>	<i>export</i>	<i>interface</i>	<i>this</i>	<i>with</i>	<i>Public</i>
<i>Byte</i>	<i>extends</i>	<i>long</i>	<i>throw</i>	<i>short</i>	<i>return</i>
<i>case</i>	<i>false</i>	<i>native</i>	<i>throws</i>	<i>static</i>	<i>goto</i>
<i>catch</i>	<i>final</i>	<i>new</i>	<i>transient</i>	<i>super</i>	<i>if</i>
<i>char</i>	<i>finally</i>	<i>null</i>	<i>true</i>	<i>implements</i>	<i>debugger</i>
<i>class</i>	<i>float</i>	<i>package</i>	<i>try</i>	<i>import</i>	<i>default</i>
<i>const</i>	<i>for</i>	<i>private</i>	<i>typeof</i>	<i>in</i>	<i>do</i>
<i>continue</i>	<i>function</i>	<i>protected</i>	<i>var</i>	<i>delete</i>	

A decorative background on the left side of the slide. It features a blue and white water splash graphic that flows from the top left towards the bottom. Overlaid on this splash is a circular clock face. The clock face has a blue globe as its center, showing continents in white. The clock hands are black, and there is a red second hand. The numbers 1 through 12 are visible around the perimeter of the clock face, though some are partially obscured by the water splash.

Using “var” keyword

“var” keyword is used for variable declaration or initialization, once for the life of any variable name in a document.

Declare JavaScript variables using var keyword

To declare a variable, the “var” keyword is used followed by the variable name.

Syntax

```
var variableName;
```

Example

```
<script>  
    var x;  
</script>
```

- Declares an empty variable named x.
- The value of x will be “undefined”.

Assigning Values to Variables

Syntax

```
variableName=value;
```

Example

```
X = "Yahoo";
```

- The value “yahoo” is assigned to variable *x*.

Declaring and Assigning in one step

Syntax

```
var variableName=value;
```

Example

```
var x="yahoo";
```

- *Declares a variable named x and assigns the value “yahoo” to the variable x in one step.*

Declaring and Assigning many variables in one step

- Two or more variables can be declared using one statement, each variable declaration is separated by a comma (,).

Example

```
var lastname="Ahmed", age=30, job="Driver";
```

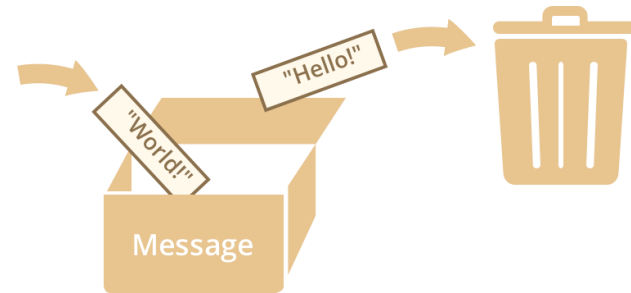
Same as

```
var lastname="Ahmed",  
age=30,  
job="Driver";
```

Variable Re-declaring

Example

```
var message="Hello";  
var message="World";
```



➤ Note that:

Re-declaring a JavaScript variable, doesn't make the variable lose its value.

Example

```
var lastname="Ahmed";  
var lastname;
```

Using “let” keyword



Declare JavaScript variables using let keyword

Starting From ES6, “let” or “const” keywords can be used to declare one or more variables.

Syntax

```
let variableName=value;
```

Example

```
let x="yahoo";
```

- Declares a variable named x and assigns a value equals to “yahoo” to the “x” variable.



Using “const” keyword

Declare JavaScript variables using const keyword

The const keyword works like the let keyword, excepts that the variable declared must be initialized immediately with a value, and that value can't be changed ever.

Syntax

```
const variableName = value;
```

Example

```
const x="yahoo";  
const x="hotmail";           // retruns error
```


A decorative background on the left side of the slide. It features a blue and white water splash graphic that flows from the top left towards the bottom. Overlaid on this splash is a circular clock face where the numbers are replaced by a map of the world. The clock has three hands: a black hour hand, a black minute hand, and a red second hand. The overall aesthetic is clean and modern, with a focus on time and global connectivity.

JS Data Types

Data types basically specify what kind of data can be stored and manipulated within a program.

- *JavaScript provides different data types to hold different types of values. There are three types of data types in JavaScript*
 - *Primitive (primary) data types.*
 - *String, Number, Boolean.*
 - *Reference (composite) data types.*
 - *Array, Object, Function.*
 - *Special data types.*
 - *Undefined and Null.*
- *Js is a dynamic language, i.e. you don't need to specify type of the variable because it is dynamically identified by the JavaScript engine.*

The background features a dynamic splash of blue water on the left side. Overlaid on this is a circular clock face with a blue globe as its center. The globe shows the Americas. The clock has black hands and a red second hand. Numbers 1 through 12 are visible around the clock's perimeter.

Primitive (primary) Data Types

Primitive data types are the data types that can hold only one value at a time.

JS String Data Type

- *The string data type is used to represent textual data (i.e. sequences of characters). Strings are created using single or double quotes surrounding one or more characters.*

Example 1

```
var str = 'JavaScript';           // Single quotes
```

Example 2

```
var str = "JavaScript";          // Double quotes
```

The “typeof” Operator

The “typeof” operator is used to get the data type of a JavaScript variable.

Syntax

```
typeof variableName
```

Example

```
var x='JavaScript';  
console.log(typeof x);           // string
```

JS Number Data Type

- *The number data type is used to represent positive or negative numbers with or without decimal place, or numbers written using exponential notation.*
- *To use numbers don't use any type of quotes.*

Integers

```
var num = 100;
```

Floating-point numbers (decimal)

```
var num = 15.2;
```

```
var num = 18.00;           // interpreted as integer 18
```

JS Number Data Type

- To represent octal (base 8) numbers, use first digit as zero (0) followed by octal digit numbers (0 to 7), or use first two digits as (0o) followed by octal digit numbers.

Integers - octal

```
var oct = 060;
```

```
    console.log(oct);           // 48
```

```
var oct = 0o60;
```

```
    console.log(oct);           // 48
```

JS Number Data Type

- To represent hexadecimal (base 16) numbers, use (0x) (in lowercase) as the first two digits followed by any number of hexadecimal digits.

Hexadecimal Numbers

```
var num = 0xff;  
console.log(num);           // 255
```

- JavaScript allows you to use the e-notation to represent very large or small numbers as in the following example.

e-notation Numbers

```
var num = 2.15e6;  
console.log(num);           // 2150000
```


JS Boolean Data Type

- The Boolean data type can hold only two values: true or false. It is typically used to store values like yes (true) or no (false), on (true) or off (false).

Boolean

```
var num1 = true;
```

```
var num2 = false;
```

```
    console.log(num1);           // true
```

```
    console.log(typeof num1);    // Boolean
```

JS Number Data Type - Special Values

➤ *The JS Number Data Type also includes some special values:*

- *Infinity, -Infinity:*

represents the mathematical Infinity ∞ , which is greater than any number. Infinity is the result of dividing a non-zero number by 0.

- *NaN:*

represents Not-a-Number value. It is a result of an invalid or an undefined mathematical operation, like taking the square root of -1 or dividing 0 by 0, or using strings inside a mathematical equations.

JS Number Data Type - Special Values

Infinity, -Infinity

```
console.log(16 / 0);           // Infinity
console.log(-16 / 0);          // -Infinity
console.log(16 / -0);          // -Infinity
```

NaN

```
console.log("Text" / 2);       // NaN
console.log("Text" / 2 + 10);   // NaN
```



Special Data Types

The “undefined” Data Type

The undefined data type can only have one value named “undefined”. If a variable has been declared, and no value is assigned to it, then its value will be “undefined”.

X value is Undefined

```
var x;  
console.log(x);    // undefined
```

- “null” is another special data type that can have only one value the “null” value. A null value means that there is no value. It is not equivalent to an empty string (“”) or 0, it is simply nothing.
- A variable can be explicitly emptied of its current contents by assigning it the null value.

null

```
var y=50;  
    console.log(y);           // 50 → number  
y=null;  
    console.log(y);           // explicitly emptied  
                                // null
```

The background features a dynamic splash of blue water on the left side. Overlaid on this is a circular clock face where the numbers are replaced by a world map. The clock has three hands: a black hour hand, a black minute hand, and a red second hand. The overall aesthetic is clean and modern, with a light blue and white color palette.

Reference (composite) Data Types

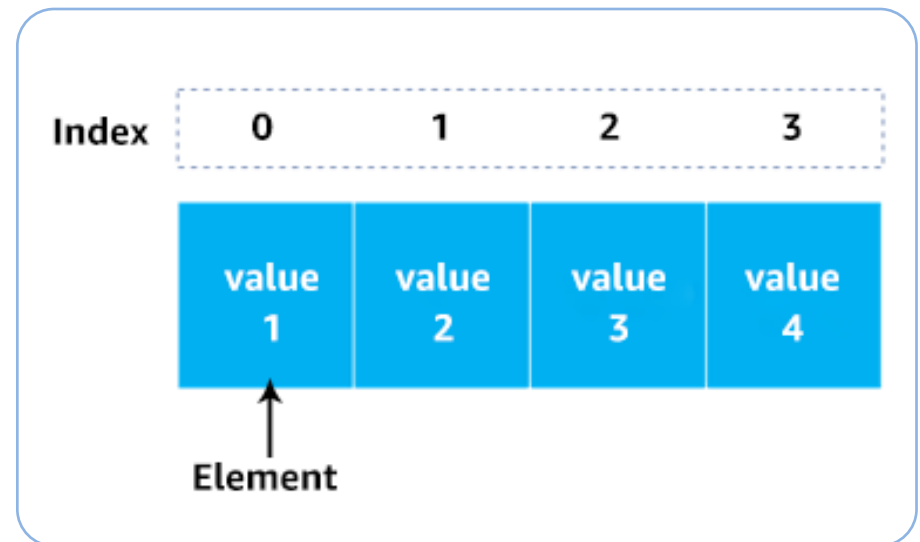
Reference data types are the data types that can hold collections of values and more complex entities.



JS Array Data Type

Arrays are complex variables that allows to store more than one value or a group of values under a single variable name.

- *In JavaScript, an array is an ordered list of values. Each value is called an element specified by an index.*
- *There are three ways to create array in JavaScript:*
 - ✓ *Array Literal.*
 - ✓ *Array Direct Instance.*
 - ✓ *Array Constructor.*
- *Let us create an array named “colors” and Insert the (Red-Green-Blue) values inside this array.*



Create Array by Array Literal

Syntax

```
var arrayName = [element0, element1, ... , elementN];
```

Example

```
var colors=['Red','Green','Blue'];
```

Create Array by Array Direct Instance

Syntax

```
var arrayName = new Array()  
arrayName[index] = "value";
```

Example

```
var colors = new Array()  
colors[0] = 'Red';  
colors[1] = 'Green';  
colors[2] = 'Blue';
```

Create Array Array Constructor

Syntax

```
var arrayname = new Array(element0, element1, ... , elementN);
```

Example

```
var colors = new Array('Red' , 'Green' , 'Blue');
```

Accessing array elements

Syntax

```
arrayName[index];
```

Example

```
colors[0];
```

Example

```
var colors = ['Red', 'Green', 'Blue'];  
    // Creates new array and assigns values to it  
console.log(colors);  
    // prints all the array values separated by comma  
console.log(colors[0]);  
    // prints the first value in the array  
console.log(colors[1]);  
    // prints the second value in the array
```