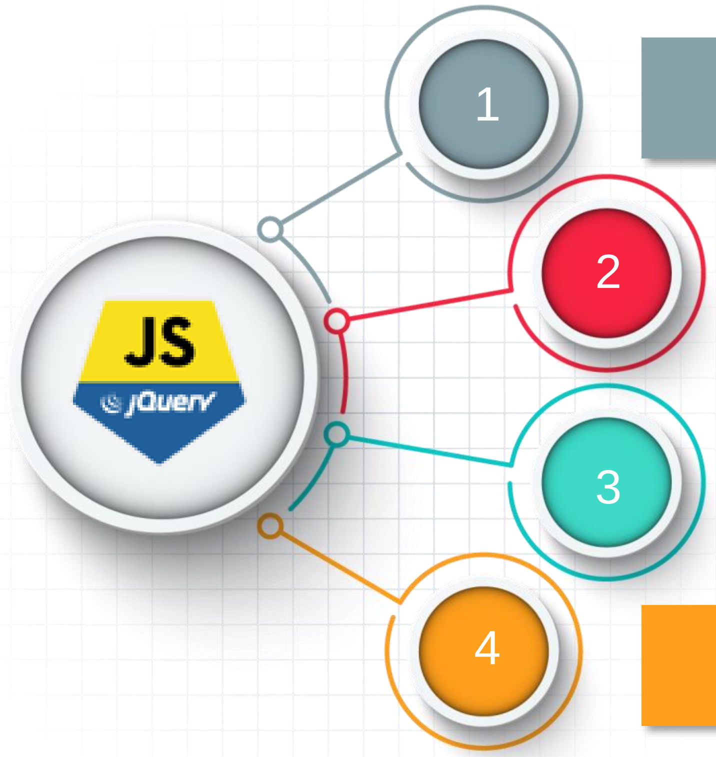
edureka!

JavaScript & JQuery

Course Outline



1. Introduction to JavaScript
2. Functions and Internal Memory Management
3. Event Handling and Objects
4. Libraries and Frameworks

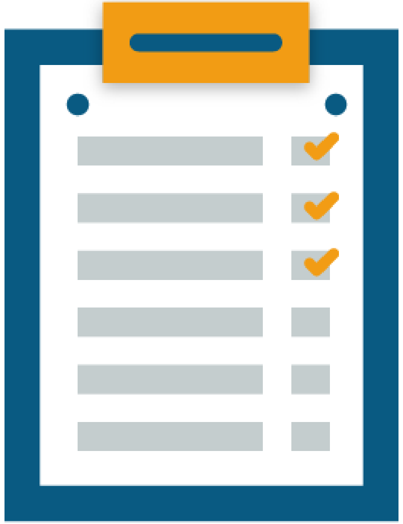
Module 1 – Introduction to JavaScript

edureka!

Objectives

After completing this module, you should be able to:

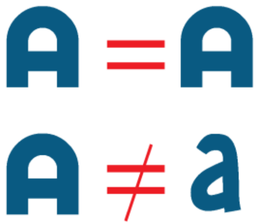
- Understand the basics of JavaScript
- Reduce the load of server using JavaScript in a server-client paradigm
- Define and use variables with different datatypes
- Handle conditional statements



edureka!

What is JavaScript?

Without any additional libraries JavaScript is also called as “Vanilla JavaScript”



Case - Sensitive language

Programming language which helps in making interactive web pages

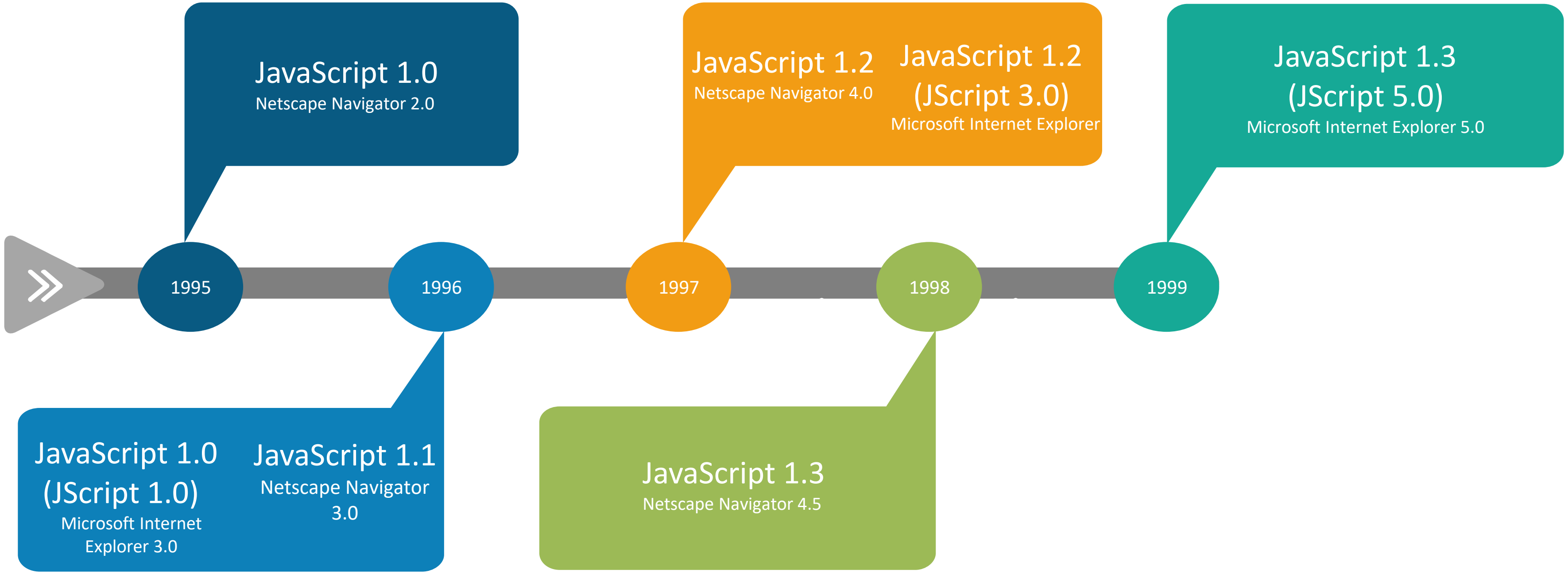
Interpreted and executed on the client machine

Used as default scripting language for HTML

Reduces the load on the server as some operations are done at the client-side

edureka!

JavaScript History



- In the late 1990's JavaScript has been standardized under the name *ECMAScript, in result*
  - A web developer, no longer care about the JavaScript version number
  - They just have to degrade the code if any feature is not supported by a browser

edureka!

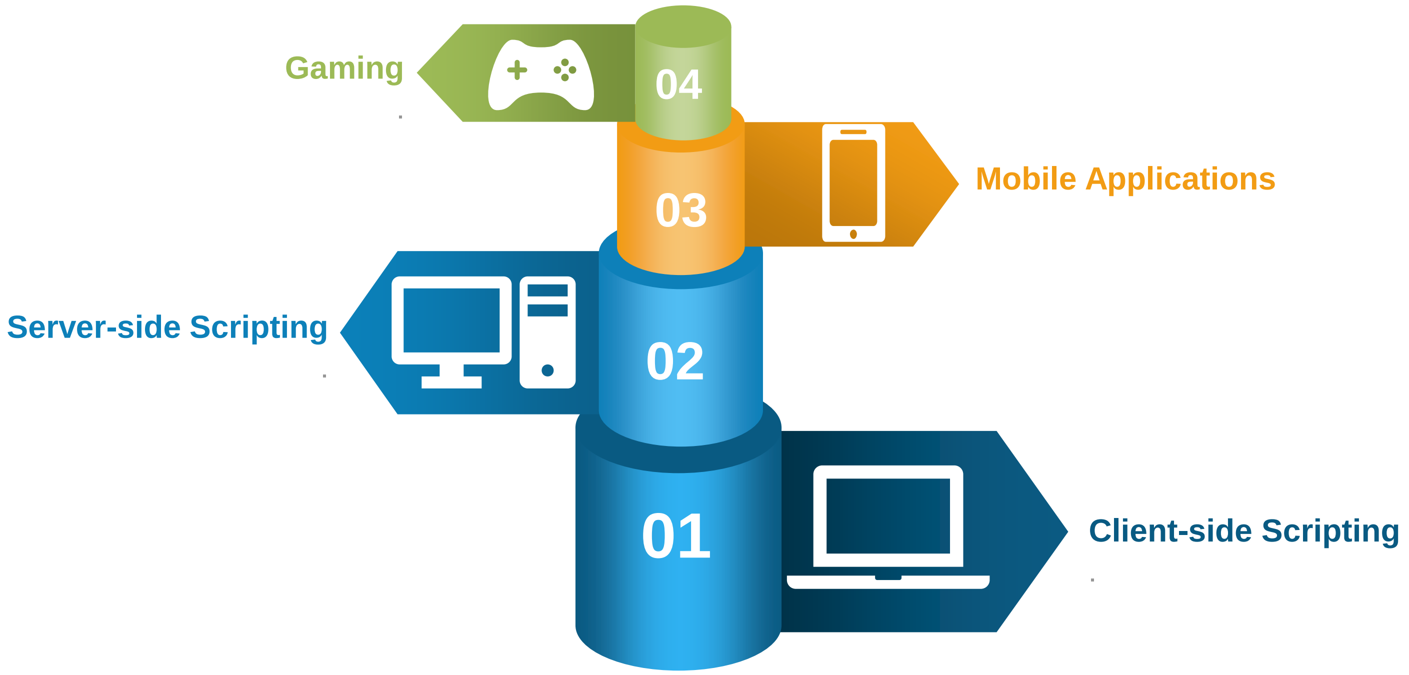
Java VS JavaScript

Java	JavaScript
▪ It is an OOP programming language	▪ It is an OOP scripting language
▪ Runs on a virtual machine or browser	▪ Runs on a browser only
▪ Code is compiled before execution	▪ Code is interpreted/Just In Time(JIT) compiled before execution
▪ Static type checking	▪ Dynamic type checking

edureka!

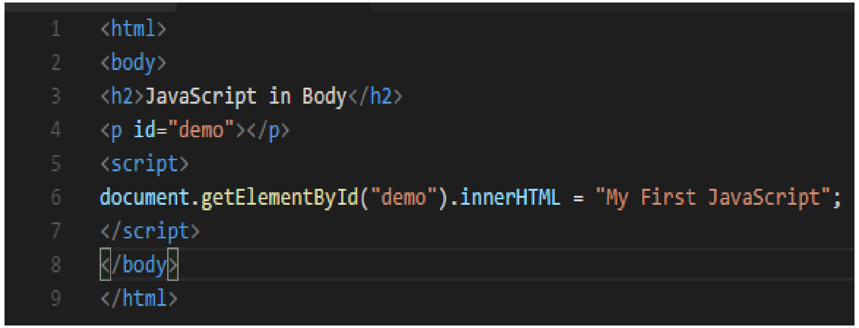
Uses of JavaScript

- Over the years, JavaScript has spread its use in the fields of:



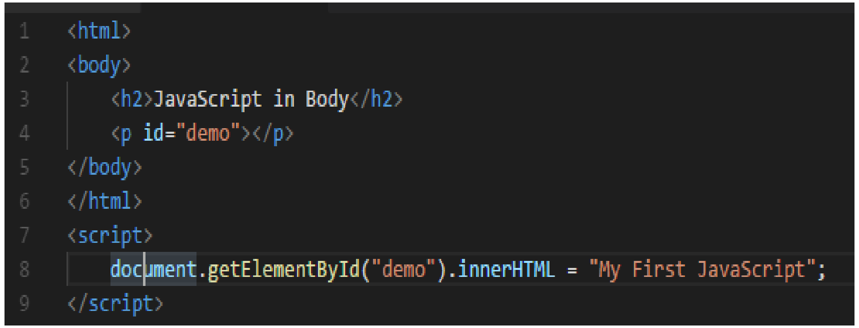
Getting Started with JavaScript

- JavaScript code must be inserted between the <script> tag, which could be placed in the
  - <head >
  - <body>
  - above / below the <html> code



```
<html>
<body>
<h2>JavaScript in Body</h2>
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML = "My First JavaScript";
</script>
</body>
</html>
```

JavaScript placed in <body> tag



```
<html>
<body>
    <h2>JavaScript in Body</h2>
    <p id="demo"></p>
</body>
</html>
<script>
    document.getElementById("demo").innerHTML = "My First JavaScript";
</script>
```

JavaScript placed below the <html> code

Getting Started with JavaScript (Contd.)

- JavaScript code can be inserted externally, in files having the extension .js
  - Example: **<script src= “myJs.js”></script>**

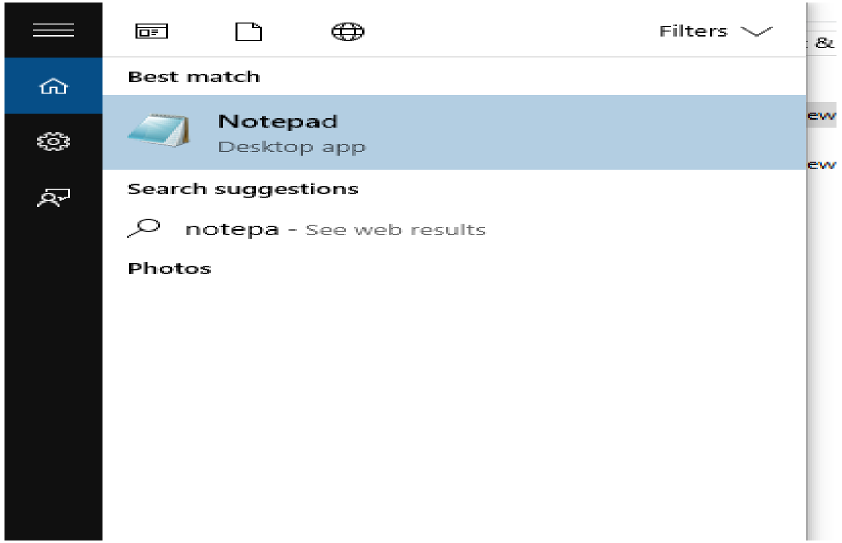
- For accessing external files from different folders use the path where the JavaScript file is located
  - Example: **<script src=“C: Desktop/js/myJs.js"></script>**

Demo – Implementing JavaScript on your HTML page

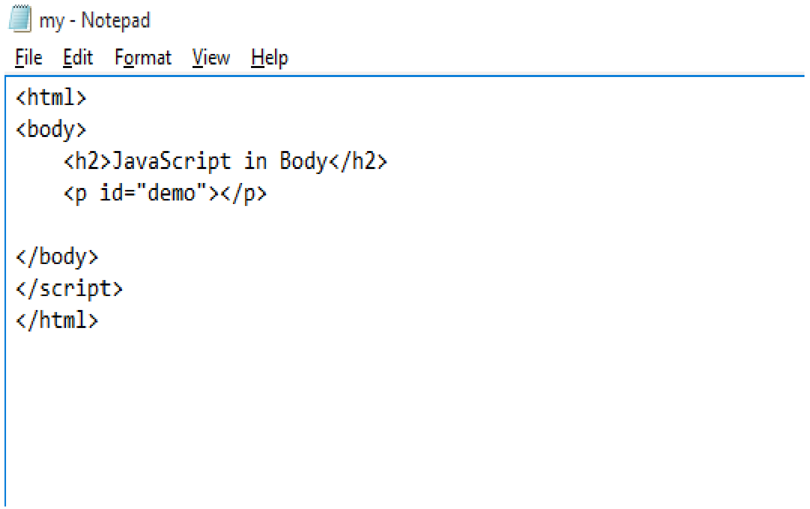
Display “sun” in a paragraph on your HTML

- In this Program we will use to our HTML code

Step1: Open Notepad or any Text Editor



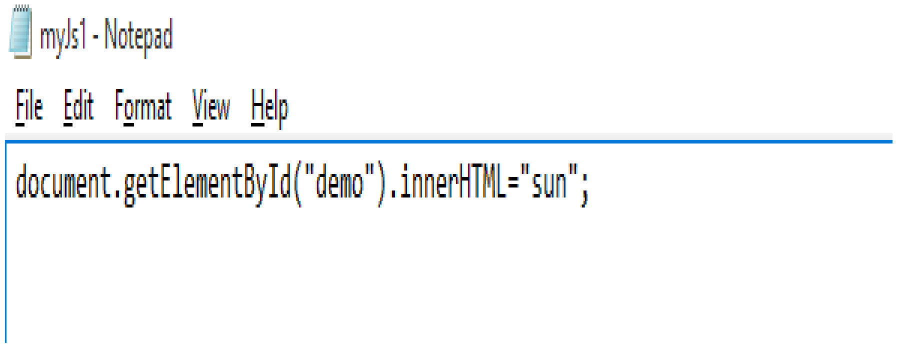
Step2: Write HTML code as shown below and as save your file with extension .html



```
<html>
<body>
    <h2>JavaScript in Body</h2>
    <p id="demo"></p>

</body>
</script>
</html>
```

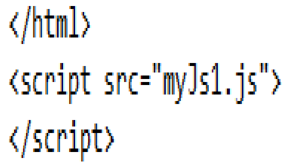
Display “sun” in a paragraph on your HTML (Contd.)

Step 3 : Save your JavaScript code in a file with extension .js



```
document.getElementById("demo").innerHTML="sun";
```

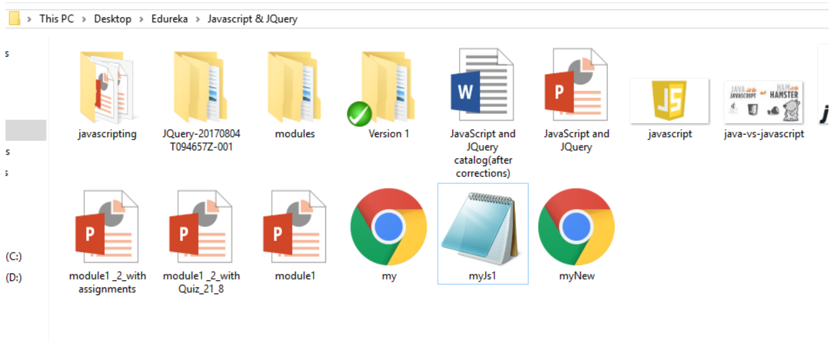
Step 4 : Link your Html code with your .js file



```
</html>
<script src="myJs1.js">
</script>
```

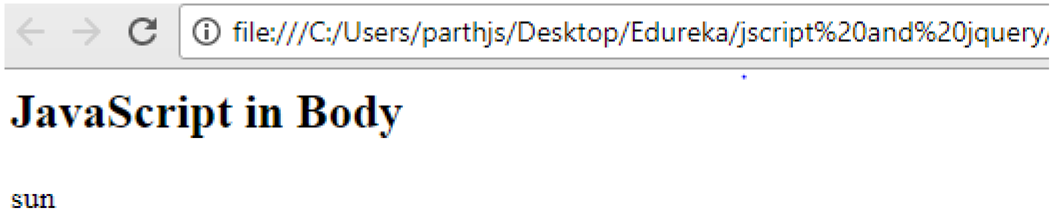
Display “sun” in a paragraph on your HTML (Contd.)

Step 5 : Check that your html and JavaScript file are located in the same folder



This PC › Desktop › Edureka › Javascript & JQuery
javascripting
JQuery-20170804T094657Z-001
modules
Version 1
JavaScript and JQuery catalog(after corrections)
JavaScript and JQuery
javascript
java-vs-javascript
module1 _2_with assignments
module1 _2_with Quiz_21_8
module1
my
myJs1
myNew
(C:)
(D:)

Step 6 : Open your html file with a browser and check the input

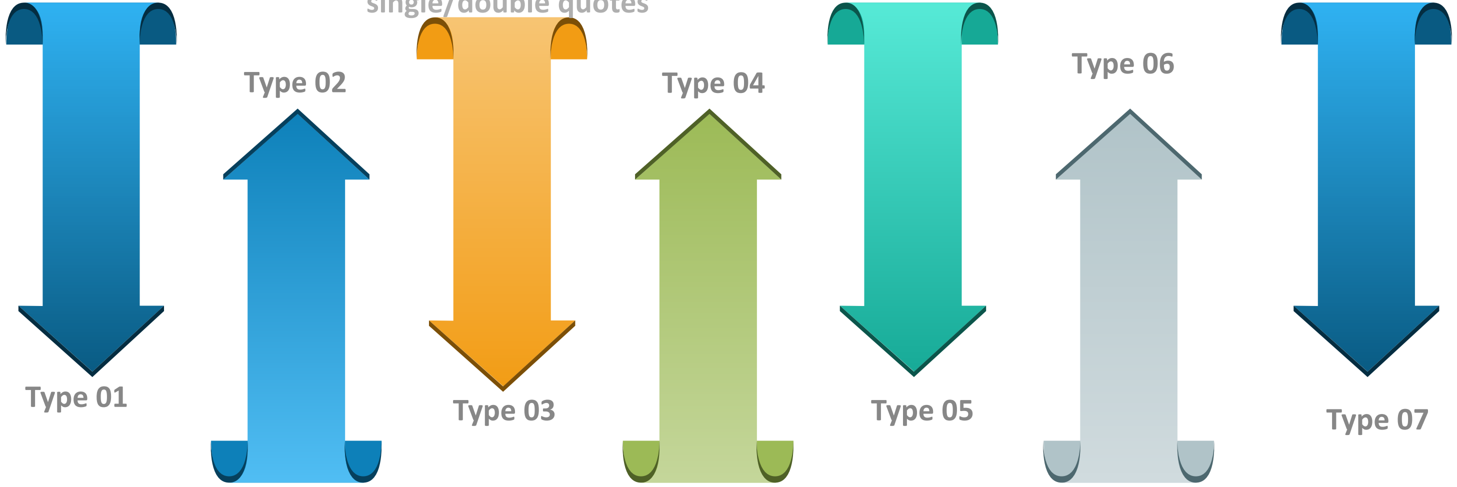

file:///C:/Users/parthjs/Desktop/Edureka/jscript%20and%20jquery/

JavaScript in Body

sun

JavaScript – Data Types

Primitive Data Types



Number
Numeric Value

Boolean
True or **False**

String
Characters in single/double quotes

Null
It as one specific value **null**

Arrays
Indexable list of items

Object
Group of attribute value pairs

Special values
NaN- Not a number
Undefined- a declared variable is not defined

edureka!

JavaScript – Variables

- Javascript variables can be considered as containers, which store a particular value or name for a particular block of memory
- JavaScript variables has standardized naming conventions:
  - Do not use JavaScript language Keywords such as **if, for, do** and **function**
  - Do not start with a digit **0, 1, 2, ... 9**
  - Do not use special characters **(%, $, &**) inside the name
  - Start with an alphabet or followed by an alphabet or digits or underscore
  - Can use **uppercase** or **lowercase** alphabets

Valid Variable Names	Invalid Variable Names
Sum	1nd_sum
first_name	function
unit_test1	last$name

JavaScript – Variables (Contd.)

Variable Declaration or Creation	• **var star**;
Assigning value to variables	• **star="sun";**
Assigning variable value to other variables	• **var moon=star;** /*variable moon will also have the value "sun"*/

JavaScript – Arrays

- An **array** stores a fixed-size sequential collection of elements of the same type

Declaration

Example-

```
var space=[" moon ", " star ", " sun "];
OR
var space= new Array(" moon ", " star ", " sun ");
```

Accessing elements

Example-

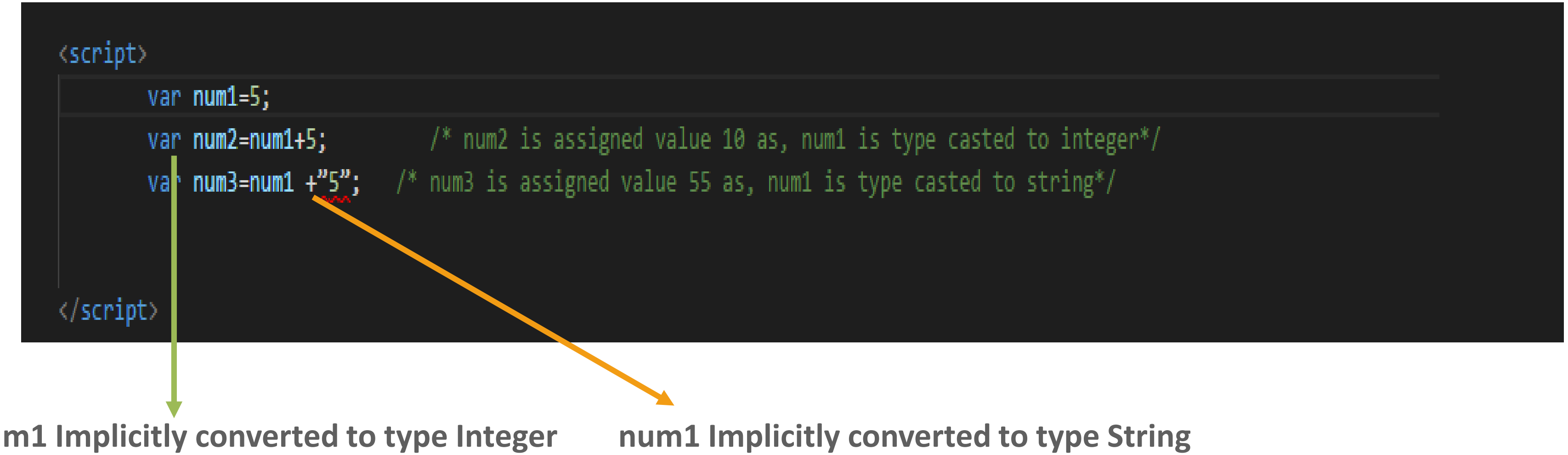
```
var bodies= space[0] +space[1]+space[3];
/*bodies will have the value "moon star sun" */
OR
space[0]=" planet ";    /*the first element of the space array will have the value "planet" */
```

edureka!

JavaScript – Type Conversions

- Type Conversions/ Type Casting: A process where an entity of one data type is converted to another
- There are two ways in which Type Conversion is done in JavaScript

1. Implicit Conversion - Integers converted to Strings and back automatically



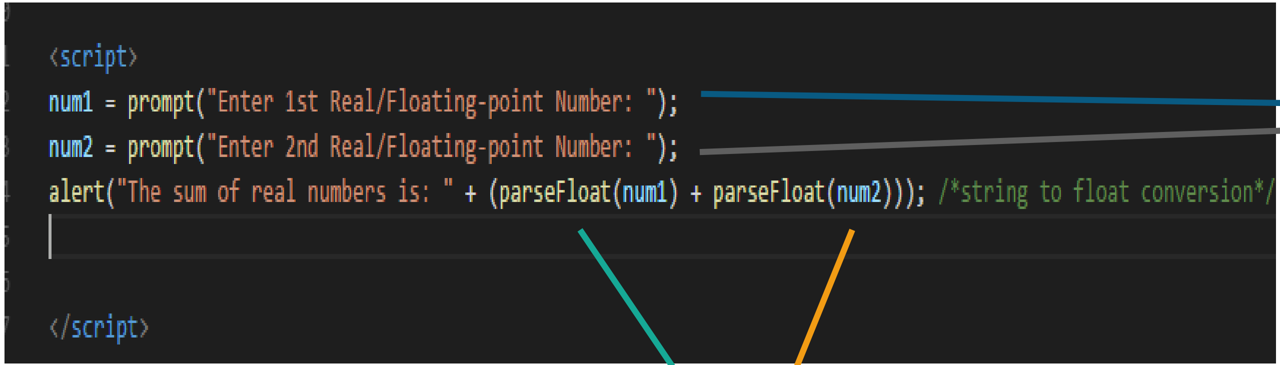
```
<script>
        var num1=5;
        var num2=num1+5;        /* num2 is assigned value 10 as, num1 is type casted to integer*/
        var num3=num1 +"5";   /* num3 is assigned value 55 as, num1 is type casted to string*/


</script>
```

num1 Implicitly converted to type Integer

num1 Implicitly converted to type String

JavaScript – Type Conversions (Contd.)

2. Explicit Conversion - Use JavaScript functions like **parseInt()**, **parseFloat()** etc.



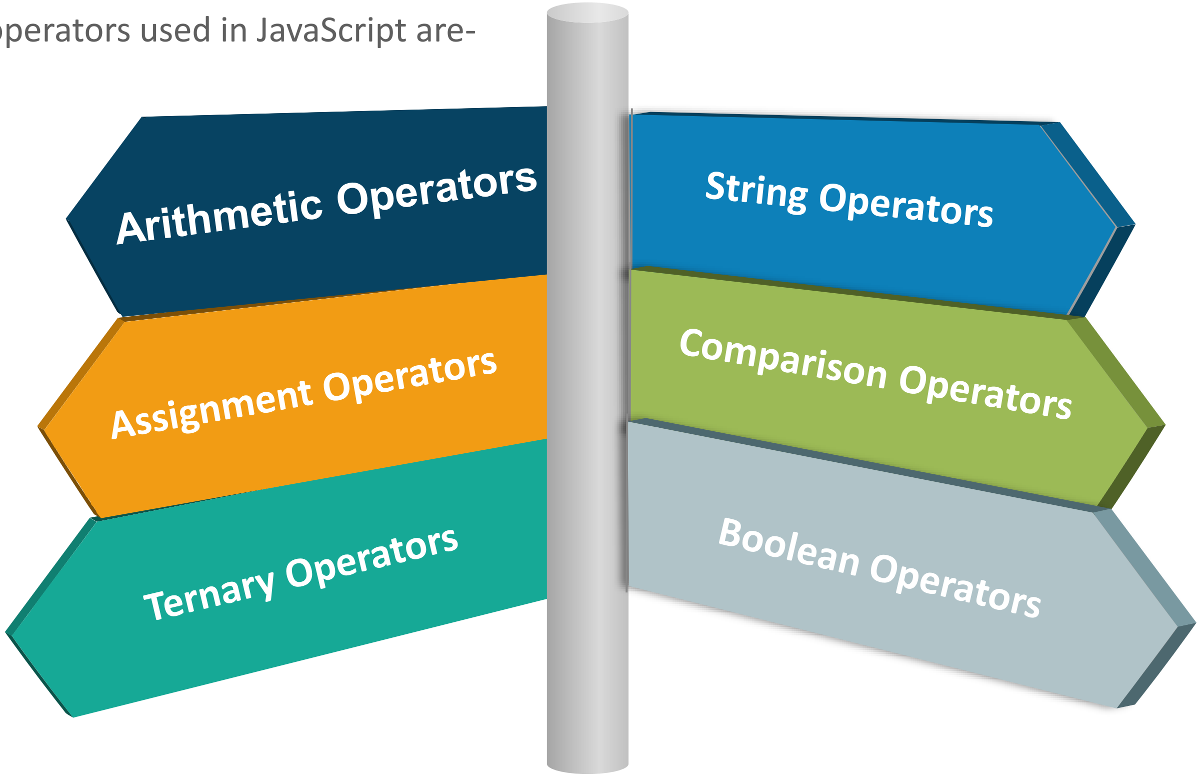
```
<script>
num1 = prompt("Enter 1st Real/Floating-point Number: ");
num2 = prompt("Enter 2nd Real/Floating-point Number: ");
alert("The sum of real numbers is: " + (parseFloat(num1) + parseFloat(num2))); /*string to float conversion*/


</script>
```

Inputs num1 and num2 of type String

num1 and num2 of type String type casted to Float

JavaScript – Operators

- Some of the operators used in JavaScript are-



JavaScript – Operators (Contd.)

Arithmetic Operators	String Operator	Assignment Operators	Comparison Operators
+: addition - : subtraction * : multiplication / : division % : modulus ++ : increment - - : decrement - : unary minus	+ : concatenation	= : assignment += : add, assign -= : subtract, assign *= : multiply, assign /= : division, assign %= : mod, assign	== : equal != : not equal > : greater < : lesser >= : greater/equal <= : lesser/equal === : equal value and same type !== : not equal value or not same type

edureka!

JavaScript – Operators (Contd.)

Boolean Operators

&& : AND
|| : OR
! : NOT

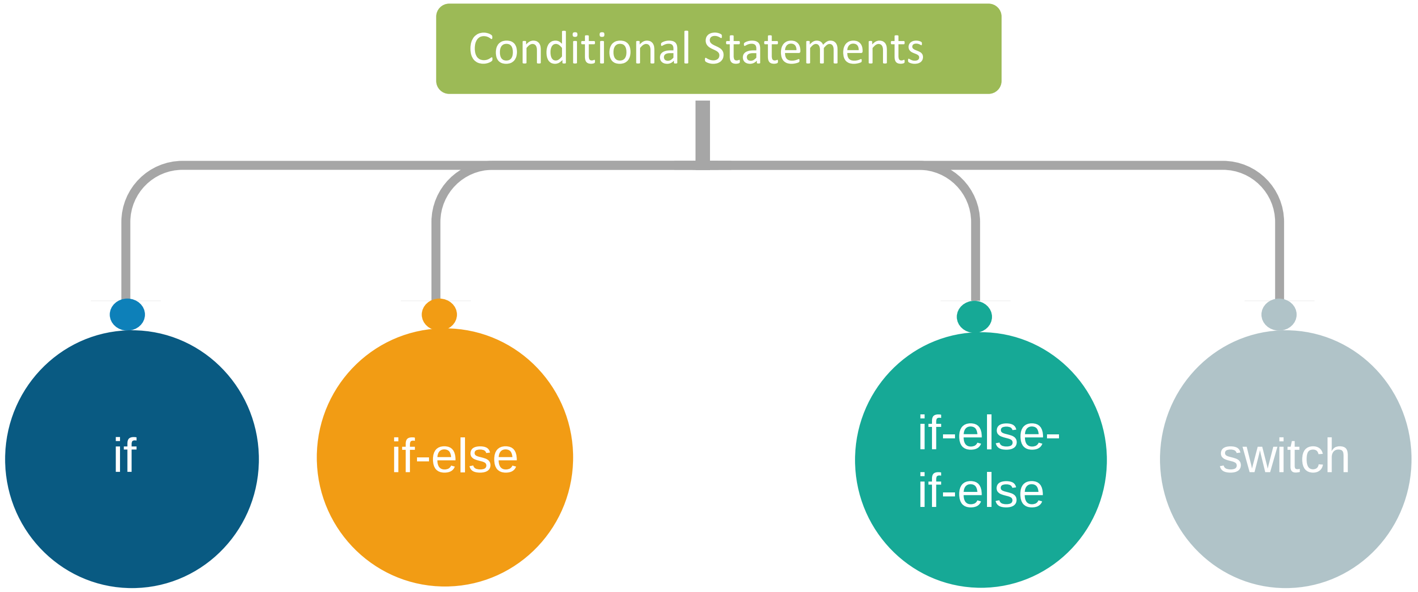
Ternary Operators

variable_name= (condition)? value1:value2;

/* if the condition is true variable_name will be assigned value1,or else value2 */

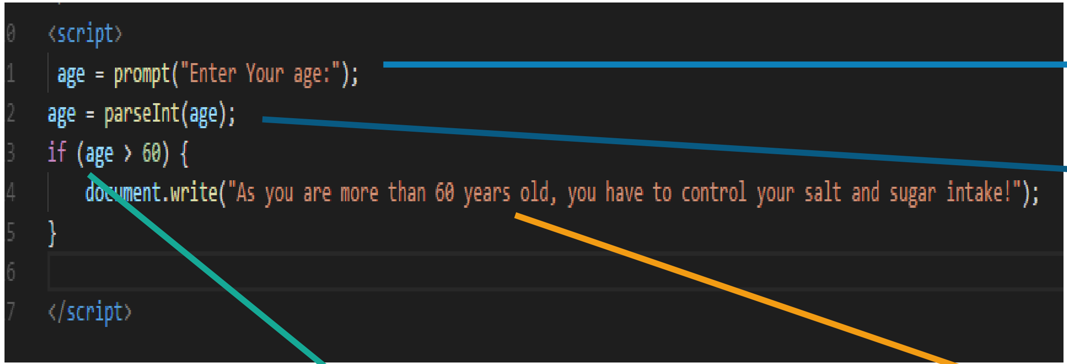
edureka!

JavaScript – Conditional Statements



if – Conditional Statement

- Condition expression evaluates to a Boolean value
- If the condition expression evaluates to true, then the block is executed
- If condition expression evaluates to false, then the block is skipped



```
<script>
 age = prompt("Enter Your age:");
age = parseInt(age);
if (age > 60) {
    document.write("As you are more than 60 years old, you have to control your salt and sugar intake!");
}

</script>
```

Step 1: Input of type String, lets say "65"

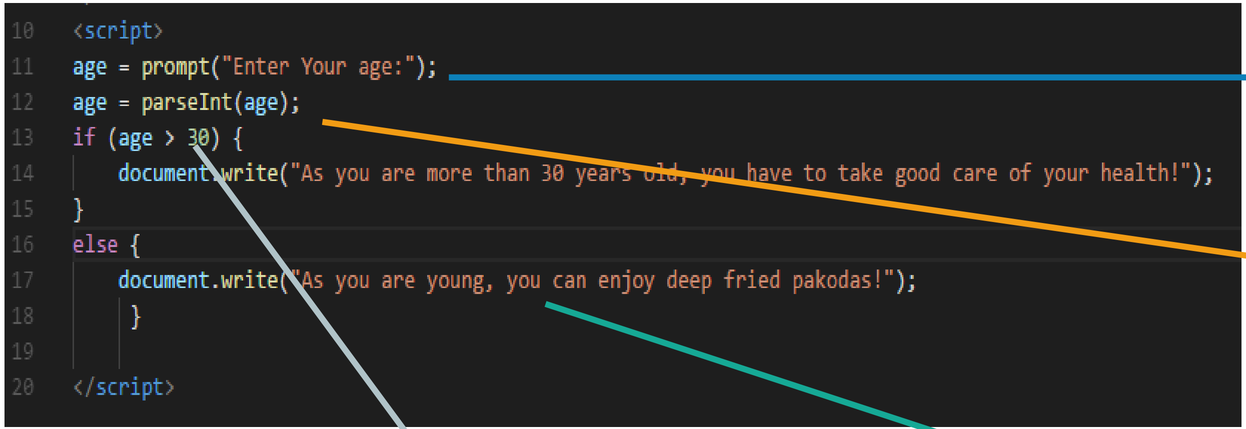
Step 2: Type casted to Number that is, "65" to 65

Step 3: Condition 65>60 is checked and returns the Boolean value true

Step 4: As the returned value is true, this statement will be executed

if-else – Conditional Statement

- If the condition expression evaluates to true, then the block following the condition is executed
- If condition expression evaluates to false, then the block following the else keyword is executed



```
<script>
age = prompt("Enter Your age:");
age = parseInt(age);
if (age > 30) {
    document.write("As you are more than 30 years old, you have to take good care of your health!");
}
else {
    document.write("As you are young, you can enjoy deep fried pakodas!");
    }

</script>
```

Step 1: Input of type String, lets say "20"

Step 2: Type casted to Number that is, "20" to 20

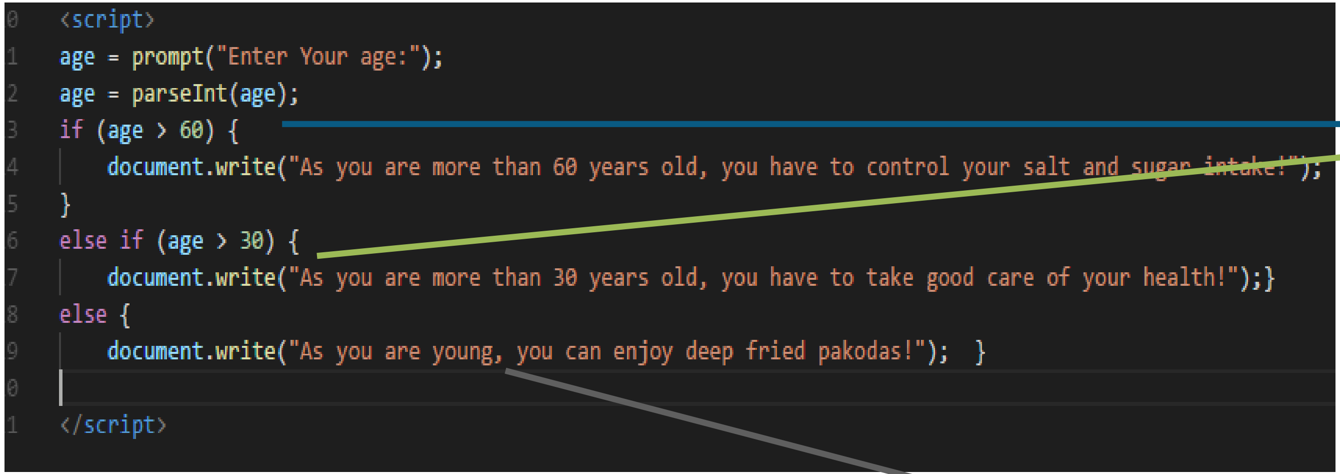
Step 3: Condition 20>60 is checked and returns the Boolean value false

Step 4: As the returned value is false, this statement in the else block will be executed

edureka!

if-else-if-else – Conditional Statement

- if-else statements can be cascaded



```
<script>
age = prompt("Enter Your age:");
age = parseInt(age);
if (age > 60) {
    document.write("As you are more than 60 years old, you have to control your salt and sugar intake!");
}
else if (age > 30) {
    document.write("As you are more than 30 years old, you have to take good care of your health!");}
else {
    document.write("As you are young, you can enjoy deep fried pakodas!");  }

</script>
```

Two if conditions are checked, and returns a Boolean value

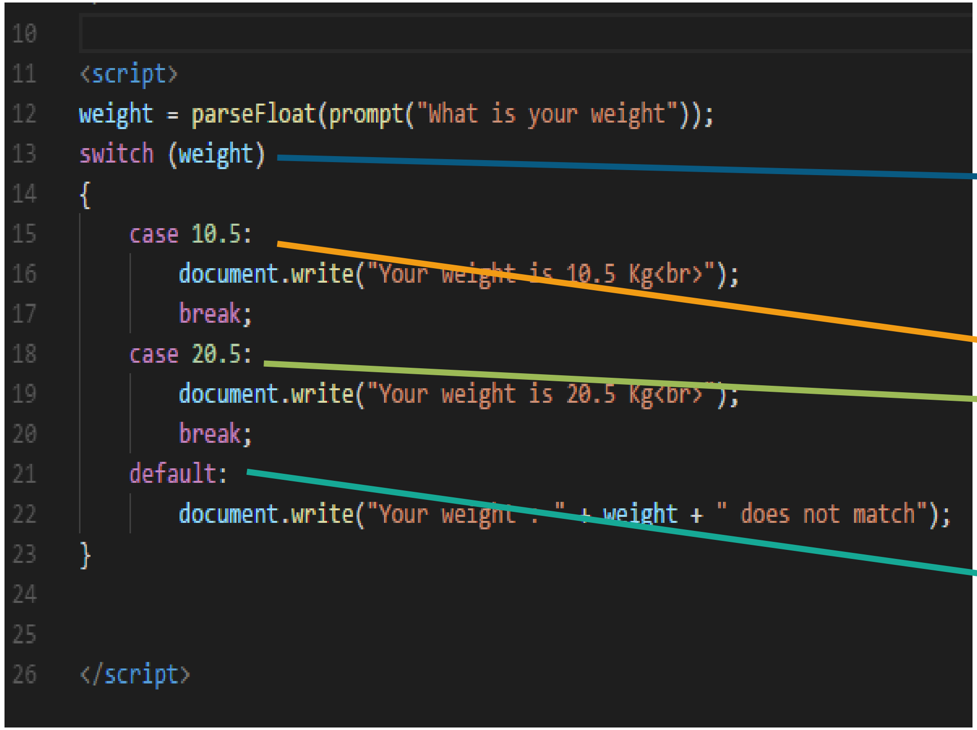
If both the if conditions return false, then this else block statement is executed

edureka!

Switch – Conditional Statements

- Test Expression
  - Evaluated to a integer, floating-point number, string or Boolean value
- **case** Statements
  - Contains the different values a test expression evaluates to
  - Permitted case values
    - Integer, floating-point number, string or Boolean values
  - A group of statements are executed (followed by break statement)
- **break** Statements
  - Breaks the execution of group of statements following case
- **default** Statement
  - Matched when test expression does not match the listed case statements

Switch – Conditional Statement (Contd.)



```
<script>
weight = parseFloat(prompt("What is your weight"));
switch (weight)
{
    case 10.5:
        document.write("Your weight is 10.5 Kg<br>");
        break;
    case 20.5:
        document.write("Your weight is 20.5 Kg<br>");
        break;
    default:
        document.write("Your weight : " + weight + " does not match");
}


</script>
```

Value of the weight variable is passed in the switch statement

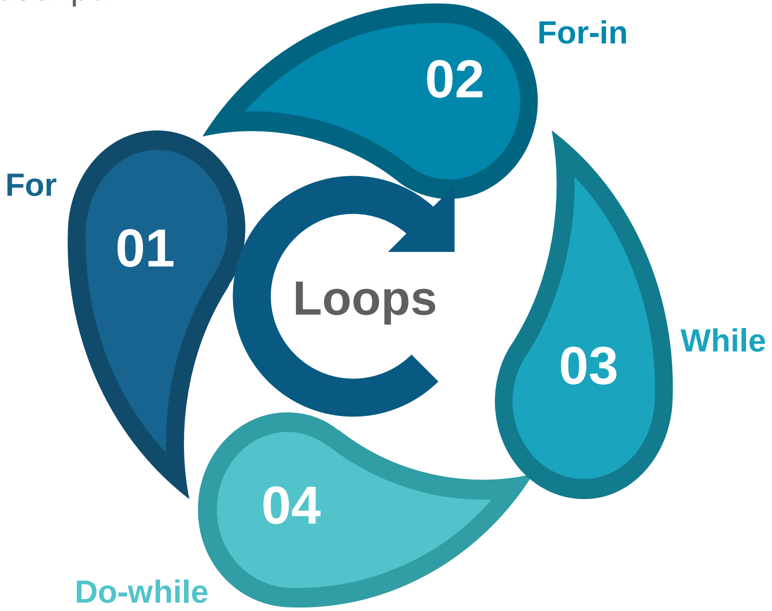
If weight matches the value of the case condition, the following block is executed

If none of the cases match with the weight variable, default case block is executed

edureka!

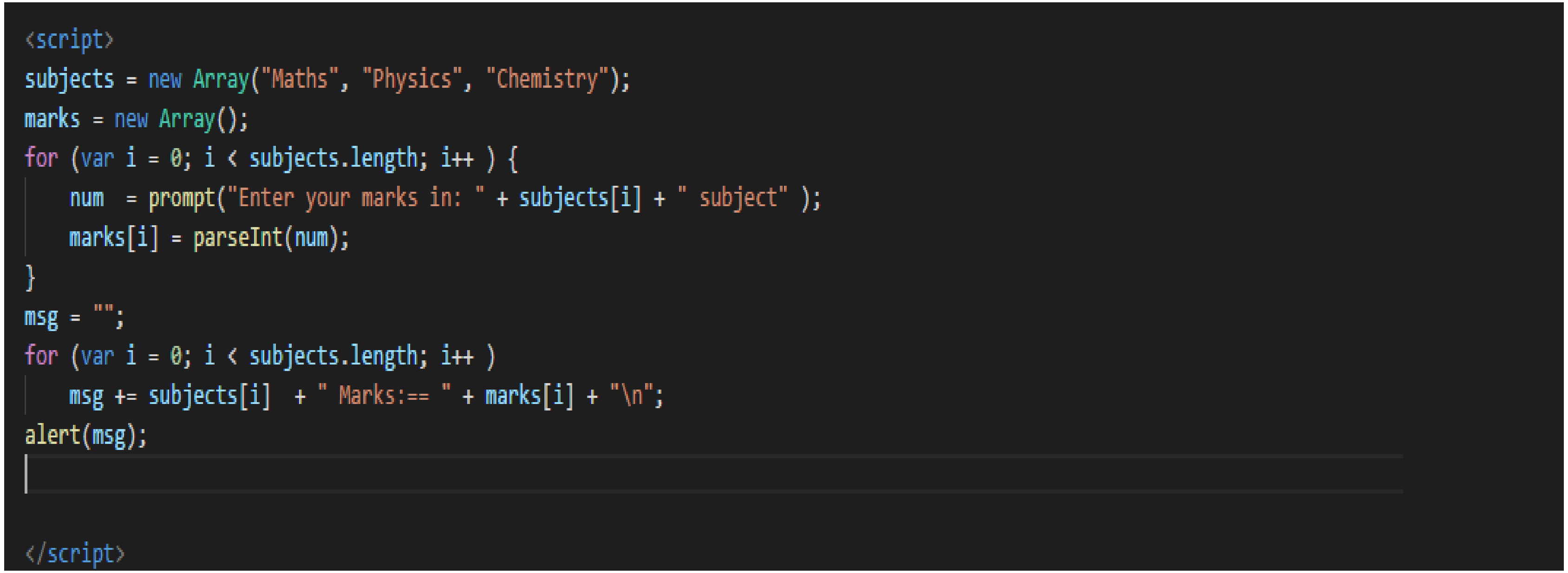
JavaScript – Loops

- Loops are basically blocks of code that are to be executed for a number of times
- Different loops in JavaScript:



for Loop

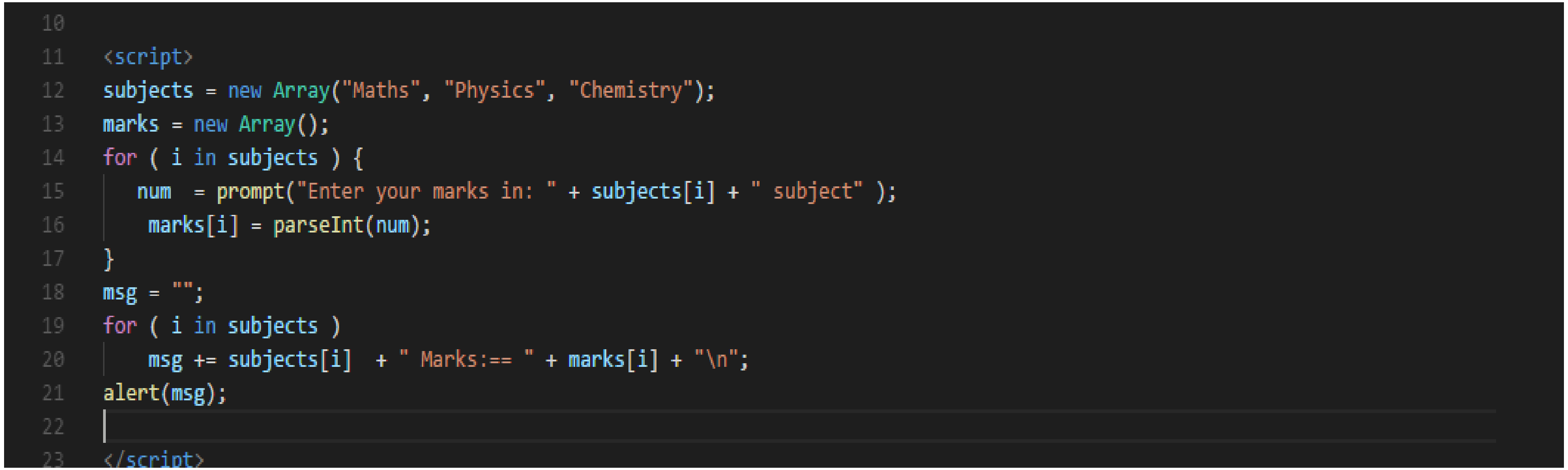
- **for (initialization; condition; updation)** – Statement
- The Loop will keep on executing until the condition check returns false



```
<script>
subjects = new Array("Maths", "Physics", "Chemistry");
marks = new Array();
for (var i = 0; i < subjects.length; i++ ) {
    num  = prompt("Enter your marks in: " + subjects[i] + " subject" );
    marks[i] = parseInt(num);
}
msg = "";
for (var i = 0; i < subjects.length; i++ )
    msg += subjects[i]  + " Marks:== " + marks[i] + "\n";
alert(msg);


</script>
```

for-in Loop

- **for (variable in object)** – Statement
- The loop will keep on executing until the all variables in the object are passed in the for statement

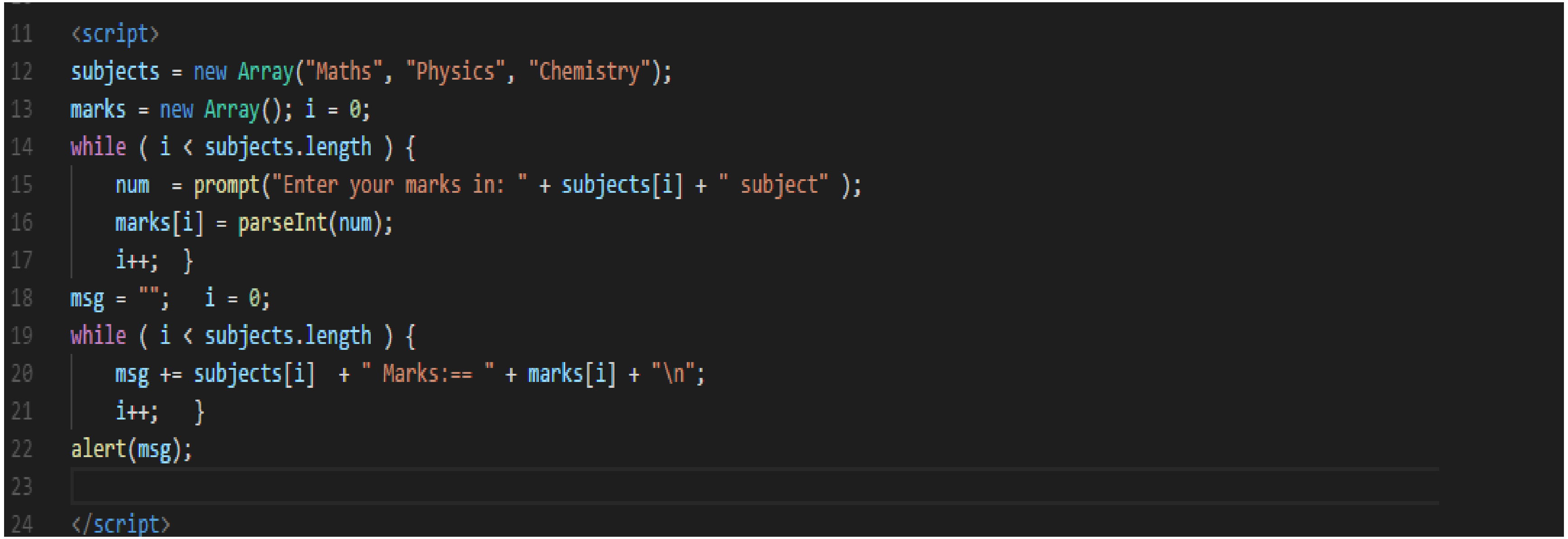


```
<script>
subjects = new Array("Maths", "Physics", "Chemistry");
marks = new Array();
for ( i in subjects ) {
    num  = prompt("Enter your marks in: " + subjects[i] + " subject" );
     marks[i] = parseInt(num);
}
msg = "";
for ( i in subjects )
     msg += subjects[i]  + " Marks:== " + marks[i] + "\n";
alert(msg);

</script>
```

edureka!

while Loop

- **while (condition)** – Statement
- The while block will be executed unless the condition statement returns a Boolean false

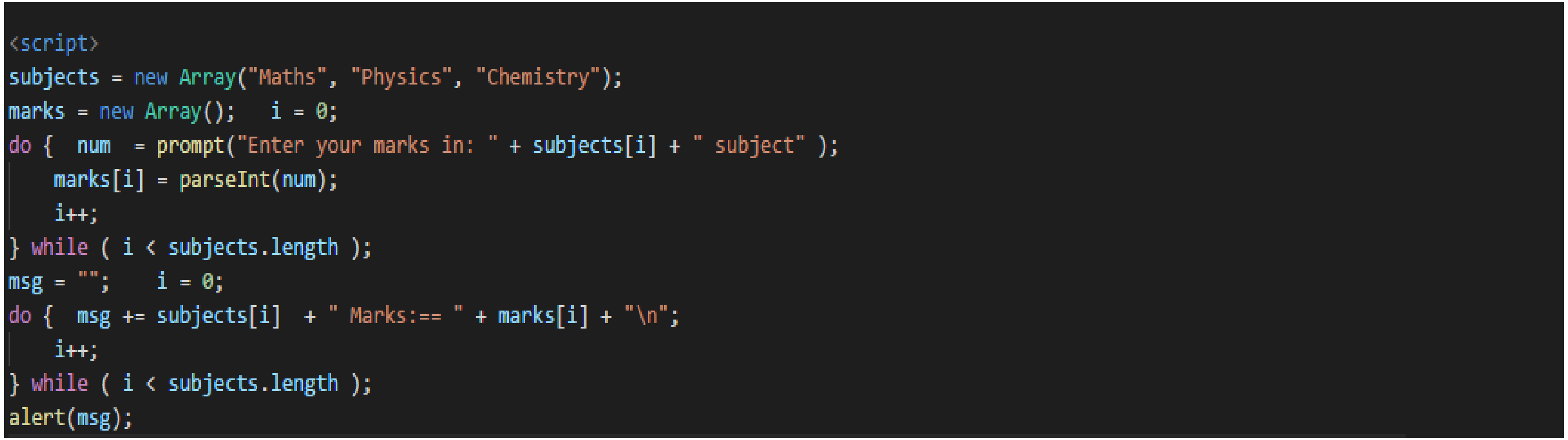


```
<script>
subjects = new Array("Maths", "Physics", "Chemistry");
marks = new Array(); i = 0;
while ( i < subjects.length ) {
    num  = prompt("Enter your marks in: " + subjects[i] + " subject" );
    marks[i] = parseInt(num);
    i++;  }
msg = "";   i = 0;
while ( i < subjects.length ) {
    msg += subjects[i]  + " Marks:== " + marks[i] + "\n";
    i++;   }
alert(msg);

</script>
```

edureka!

do-while Loop

- **do { statements } while (condition);** – Statement
  - First the do block will be executed once
  - the condition will be checked then it will be executed until the condition in the while
  - If the statement returns false, it stops execution
  - If true it executes the do loop again, and checks the condition again.



```
<script>
subjects = new Array("Maths", "Physics", "Chemistry");
marks = new Array();   i = 0;
do {  num  = prompt("Enter your marks in: " + subjects[i] + " subject" );
    marks[i] = parseInt(num);
    i++;
} while ( i < subjects.length );
msg = "";    i = 0;
do {  msg += subjects[i]  + " Marks:== " + marks[i] + "\n";
    i++;
} while ( i < subjects.length );
alert(msg);
```

edureka!

Quiz

1. What is the correct JavaScript syntax for a **for** loop

   a. for(var i=0;i<star.length;i++)

   b. for(var i in star);

   c. for var i=0;i<star.length;i++;

   d. None of the above;

Answers

1. What is the correct JavaScript syntax for a **for** loop?

   a. **for(var i=0;i<star.length;i++)**

   b. for(var i in star);

   c. for var i=0;i<star.length;i++;

   d. None of the above;

Answer a:

Explanation:

Correct syntax for **for** loop is –

```
for( var varname= start_value ; condition; Increment/decrement){
          Statement 1;
          Statement n;
          }
```

So, 1st option would be the correct choice

edureka!

Quiz

2. What is the correct syntax for declaring an Array with elements sun, moon and planet:

a. var myArray[]=[“sun”, “moon”, “planet”];

b. var myArray=[“sun”,“moon”, “planet”];

c. var myArray=new Array[“sun”,“moon”, “planet”];

d. Array myArray=new Array(“sun”,“moon”, “planet”);

Answers

2. What is the correct syntax for declaring an Array with elements sun, moon and planet:

   a. var myArray[]=[“sun”, “moon”, “planet”];

   b. var myArray=[“sun”,“moon”, “planet”];

   c. var myArray=new Array[“sun”,“moon”, “planet”];

   d. Array myArray=new Array(“sun”,“moon”, “planet”);

Answer b:

Explanation:

Syntax for array declaration is var name_array =[“element 1”, “element 2”,..., “element n”];

Quiz

3. Java and JavaScript are related. :

   a. True

   b. False

Answers

3. Java and JavaScript are related.

a. True

b. **False**

Answer b:
Explanation:
Java and JavaScript are not related considering the following points-

- Java is an OOP programming language, while JavaScript is an OOP scripting language
- Java runs on a virtual machine or browser, where as JavaScript runs on a browser only
- In Java code is compiled before execution, while in JavaScript code is interpreted/Just In Time(JIT) compiled before execution
- In Java Static type checking is done while in JavaScript Dynamic type checking

edureka!

Summary

In this module, you should have learnt:

- To Execute a simple JavaScript code
- The syntax for defining and using variables
- Different ways of Type Conversion
- To work with Conditional Statements and Loops

Comments

Ratings

Suggestions

FEEDBACK

Survey

Likes

Ideas

edureka!

Thank You

For more information please visit our website
www.edureka.co