JavaScript

Embedding JS in HTML document

- <html>
- <head>
- <title>My title</title>
- <script >
- document.write(“Hello World");
- </script>
- </head>
- <body>
- </body>
- </html>

- <html>
- <head>
- <title>My title</title>
- </head>
- <body>
- </body>
- </html>
- <script >
- document.write("<h1>Outside HTML</h1>");
- </script>

- <html>
- <head>
- <title>My title</title>
- <script >
- document.write(“<h1>Hello World</h1>");
- </script>
- </head>
- <body>
- </body>
- </html>

External JavaScript File

- <html>
- <head>
- <title>My title</title>
- <script src="demo.js">
- </script>
- </head>
- <body>
- </body>
- </html>

Demo.js

- document.write("<h1>External file</h1>");

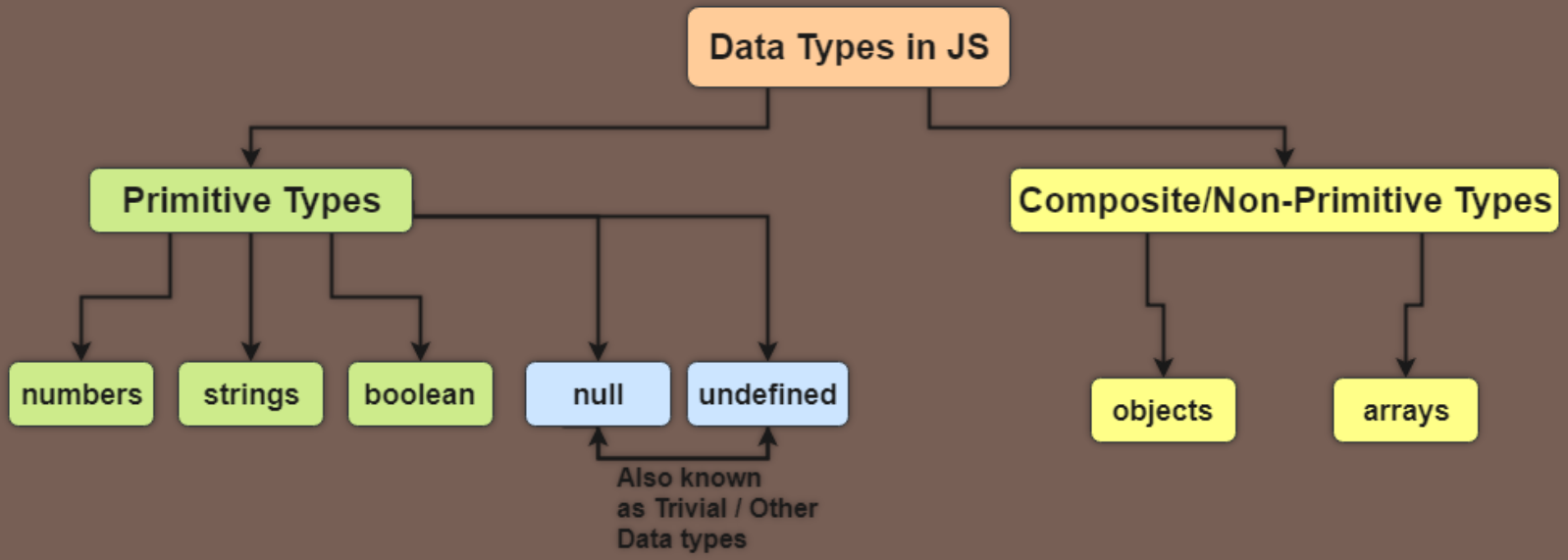
Statements in JavaScript

- <html>
- <head>
- <title>My title</title>
- <script>
- document.write("<h1>Hello World1</h1>");
- document.write("<h1>Hello World2</h1>");
- document.write("<h1>Hello World3</h1>");document.write("<h1>Hello World4</h1>");
- </script>
- </head>
- <body>
- </body> </html>

Comments in JavaScript

- <html><head><title>My title</title>
- <script>
- // Single line comment
- //document.write("<h1>Hello World1</h1>");
- document.write("<h1>Hello World2</h1>");
- /*Multiple line comment
- document.write("<h1>Hello World3</h1>");
- document.write("<h1>Hello World4</h1>");
- */
- </script>
- </head> <body> </body></html>

Data Types –


Data Types in JS
Primitive Types
Composite/Non-Primitive Types
numbers
strings
boolean
null
undefined
objects
arrays
Also known as Trivial / Other Data types

JavaScript Variables & Data Types

- **Variables –**
- Following are some basic definitions of variables in a typical programming language context.
- A variable provides us with a **named storage** that our programs can manipulate. It is the **basic unit of storage** in a program.
- Variables are used to store information to be referenced & manipulated in a computer program.
- In programming, a variable is a value that can be changed depending on the conditions or information being passed to the program.

JavaScript Primitive Data Types

- Data TypeDescription
- **String** : represents sequence of characters e.g. “hello”
- **Number** :represents numeric values e.g. 100
- **Boolean**: represents boolean value either false or true
- **Undefined**: represents undefined value
- **Null**: represents null i.e. no value at all

JavaScript Numbers –

- JavaScript has only one type of numbers. Numbers can be written with, or without decimals:
- var x1 = 34.00;   // Written with decimals
- var x2 = 34;   // Written without decimals
- **JavaScript Booleans –**
- Booleans can only have two values: true or false. Booleans are often used in conditional testing.
- var flag1 = true;
- var flag2 = false;

JavaScript Strings –

- A string (or a text string) is a series of characters like “Simple Snippets”. Strings are written with quotes. You can use single or double quotes:
- var carName = "Mercedes"; // Using double quotes
- var carName = 'BMW'; // Using single quotes
- You can use quotes inside a string, as long as they don’t match the quotes surrounding the string:
- var answer = “ It's alright "; // Single quote inside double quotes
- var answer = “ He is called 'Mark‘ "; // Single quotes inside double quotes
- var answer = ‘ He is called "Mark“ '; // Double quotes inside single quotes

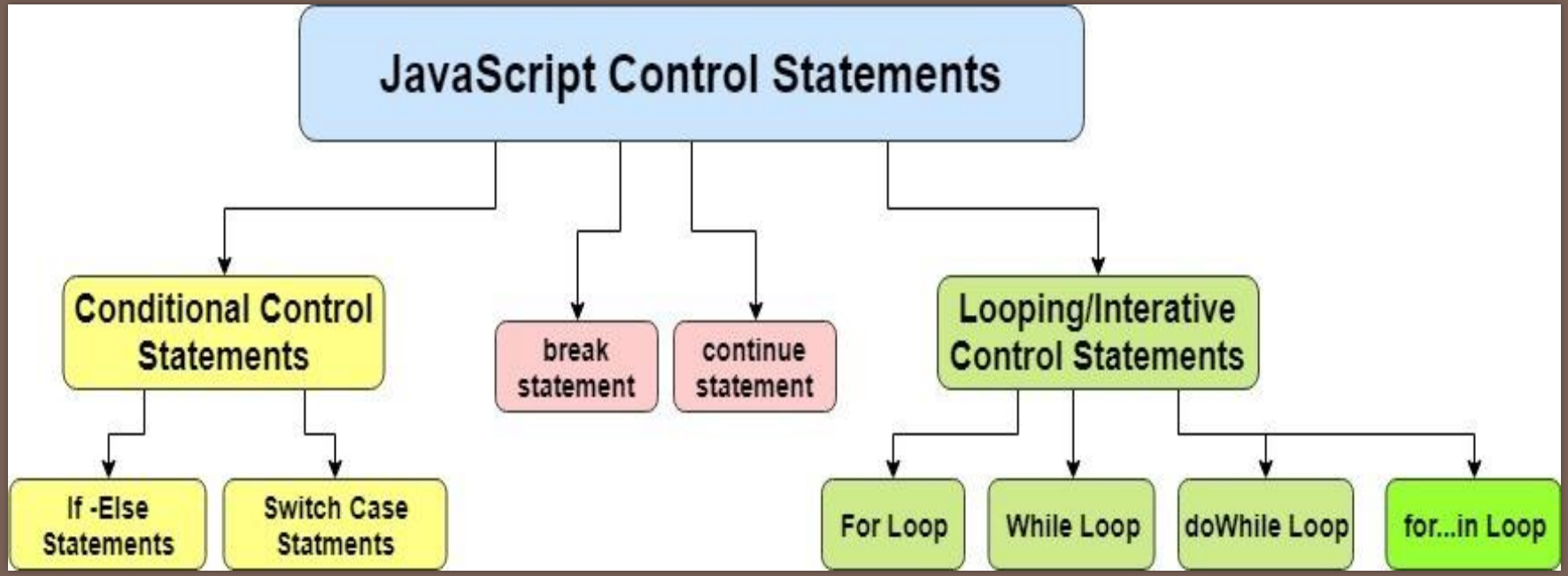
- **JavaScript is LOOSELY/WEAKLY TYPED:**
- Javascript is known as **untyped/loosely/weakly typed** language. This means that we do not have to specify the data type in advance unlike other languages like C++, Java C# etc.
- Example –
- var x;
- x = 5;
- In the above example we can see that we did not have to specify any data type for the variable x in advance.

- **2. JavaScript is DYNAMICALLY TYPED :**
- Javascript is known as **dynamically typed** language. This means, that once a variable is created in javascript using the keyword **var**, we can store any type of value in this variable supported by javascript.
- Example –
- // creating variable to store a number
- var num = 5;
-
- // store string in the variable num
- num = "Simple Snippets";
- In this example above, you can see that the data type of the variable num changes from number to string as we pass string data. This it is flexible in nature.

```
<!DOCTYPE html><html><body>
<h2>JavaScript Variables</h2>
<p>Strings are written with quotes.</p>
<p>Numbers are written without quotes.</p>
<script>
var pi = 3.14; var x=5; var person = "John Doe";
var answer = 'Yes I am!';
var result= pi+x + person+ answer +x;
document.write(result+"<br>");
 var x= "Now I am a String"
document.write("<br>"+x);
</script>
</body></html>
```

```
<!DOCTYPE html><html><body>
<h2>JavaScript Variables</h2>
<p>Strings are written with quotes.</p>
<p>Numbers are written without quotes.</p>
<script>
var pi = 3.14;
var x=5
var person = "John Doe";
var answer = 'Yes I am!';
var result= pi+x + person+ answer +x;
document.write(result+"<br>");
 var x= "Now I am a String"
document.write("<br>"+x);
alert(x);
</script>
</body></html>
```

Control Statement



```
<html> <head><title>IF-Else if - Else Control Statments in javascript</title>
<script >
/*Q3) Find if a number is positive and even*/
var x=8;
if(x>0)
{
document.write("<h3>Positive Number</h3>");
if(x%2==0)
{
document.write("<h3>Positive & Even Number</h3>");
}
}
</script>
</head><body></body></html>
```

```
<!DOCTYPE html>
<html>
<body>
<h2>JavaScript IF Else</h2>
<script>
// JavaScript program to illustrate If statement
var i = 10;
if (i%2==0)
  document.write("Number is Even");
// This statement will be executed
// as if considers one statement by default
else
document.write("number is odd");
  </script>
</body></html>
```

```
<html><head><title>IF-Else if - Else Control Statments in javascript<
<script type="text/javascript">
/*Q1) Find if a number is positive, negative or 0*/
var x=8;
if(x>0)
{
document.write("<h3>Positive Number</h3>");
}
else if(x<0)
{
document.write("<h3>Negative Number</h3>");
}
else
{
document.write("<h3>Number is 0</h3>");
}
</script>
</head><body> </body></html>
```

JavaScript Functions

- **JavaScript functions** are used to perform operations. We can call JavaScript function many times to reuse the code.
- Advantage of JavaScript function
- There are mainly two advantages of JavaScript functions.
- **Code reusability**: We can call a function several times so it save coding.
- **Less coding**: It makes our program compact. We don't need to write many lines of code each time to perform a common task.

- JavaScript Function Syntax
- The syntax of declaring function is given below.
- function functionName([arg1, arg2, ...argN]){
- //code to be executed
- }
- JavaScript Functions can have 0 or more arguments.

```
<html>
<body>
<script>
function msg(){
alert("hello! this is message");
}
</script>
<input type="button" onclick="msg()" value="call function"/>
</body>
</html>
```

JavaScript Function Arguments

```
<html>
<body>
<script>
function getcube(number){
alert(number*number*number);
}
</script>
<form>
<input type="button" value="click" onclick="getcube(4)"/>
</form>
</body>
</html>
```

Function with Return Value

- <html>
- <body>
- <script>
- function getInfo(){
- return "hello javatpoint! How r u?";
- }
- </script>
- <script>
- document.write(getInfo());
- </script>
- </body>
- </html>

Function

- <html> <head><title>Functions in JS</title>
- <script type="text/javascript">
- /*Q1) write a function to add 2 numbers and print the result */
- function addNumber(a , b)
- {
- var total = a+b;
- return total;
- }
- var output = addNumber(4,2); // function call
- document.write("<h1>The total is : "+output+"</h1>");
- </script>
- </head> <body></body></html>

- <!DOCTYPE html><html>
- <body>
- <h2>JavaScript in Body</h2>
- 1st Paragraph.
- A Paragraph.
- <button type="button" onclick="myFunction()">Click Me </button>
- <script>
- function myFunction() {
- document.getElementById("demo").innerHTML = "Paragraph chan
- }
- </script></body></html>

```
<!doctype html><html>  <head>
<script>
function add(){
var a,b,c;
a=Number(document.getElementById("first").value);
b=Number(document.getElementById("second").value);
c= a + b;
document.getElementById("answer").value= c;
}
</script>
</head>
<body>
Enter the First number : <input id="first" placeholder="input number
Enter the Second number: <input id="second" placeholder="input
<button onclick="add()">Add</button>
<input id="answer" placeholder="Result will be display here">
</body></html>
```

```
<!DOCTYPE html><html><head><title> Form Validation </title>
<body>
<script>
function validate()
{
var username = document.getElementById("uname");
var password = document.getElementById("pass");
if(username.value == "" || password.value =="")
{
alert("No blank Value allowed");
return false;
}
}
</script>
<form onsubmit="return validate()" action="message.html">
<input id="uname" placeholder="Username" type="text" />
<br><br>
<input id="pass" placeholder="Password" type="password" /> <br><br>
<button  type="submit"> Login </button>
</form></body></html>
```

```
<html><head><title> Mobile Validation</title>
<script>
function validate(){
var text=document.getElementById("text1").value;
var regx= /^[7-9][0-9]{9}$/;
if(regx.test(text)) {
document.getElementById("lbltext").innerHTML="Valid";
document.getElementById("lbltext").style.visibility="visible";
document.getElementById("lbltext").style.color="green";
return true;
} else {
document.getElementById("lbltext").innerHTML="InValid";
document.getElementById("lbltext").style.visibility="visible";
document.getElementById("lbltext").style.color="red";
return false;
}}
</script></head><body>
<form>
<input id="text1" placeholder="Enter Mobile Number" type="text"/><br>
<label id="lbltext" style="color:red;visibility:hidden">Invalid</label>
<br>
<button onclick=" return validate()" type="button">Submit</button>
</form>
</body></html>
```

```
<html><head><title> Email Validation</title>
<script>
function validate()
{
var text=document.getElementById("text1").value;
var regx= /^([a-zA-Z0-9\._]+)@([a-zA-Z0-9\._]+).([a-zA-Z]{2,8})?$/;
if(regx.test(text))
{
document.getElementById("lbltext").innerHTML="Valid";
document.getElementById("lbltext").style.visibility="visible";
document.getElementById("lbltext").style.color="green";
return true;
} else
{
document.getElementById("lbltext").innerHTML="InValid";
document.getElementById("lbltext").style.visibility="visible";
document.getElementById("lbltext").style.color="red";
return false;}}
</script></head><body>
<form>
<input id="text1" placeholder="Email" type="text"/><br>
<label id="lbltext" style="color:red;visibility:hidden">Invalid</label>   <br>
<button onclick=" return validate()" type="button">Submit</button>
</form>
</body></html>
```

JavaScript Objects

- A javaScript object is an entity having state and behavior (properties and method). For example: car, pen, bike, chair, glass, keyboard, monitor etc
- JavaScript is an object-based language. Everything is an object in JavaScript.
- JavaScript is template based not class based. Here, we don't create class to get the object. But, we direct create objects.
- Creating Objects in JavaScript
- There are 3 ways to create objects.
- By object literal
- By creating instance of Object directly (using new keyword)
- By using an object constructor (using new keyword)

JavaScript Object by object literal

- The syntax of creating object using object literal is given below:
- object={property1:value1,property2:value2.....propertyN:valueN}
- As you can see, property and value is separated by : (colon).
- Let's see the simple example of creating object in JavaScript.
- **<script>**
- emp={id:102,name:"Shyam Kumar",salary:40000}
- document.write(emp.id+" "+emp.name+" "+emp.salary);
- **</script>**

Output of the above example
102 Shyam Kumar 40000

By creating instance of Object

- The syntax of creating object directly is given below:
- var objectname=new Object();
- Here, **new keyword** is used to create object.
- Let's see the example of creating object directly.
- **<script>**
- var emp=new Object();
- emp.id=101;
- emp.name="Ravi Malik";
- emp.salary=50000;
- document.write(emp.id+" "+emp.name+" "+emp.salary);
- **</script>**

Output of the above example
101 Ravi 50000

By using an Object constructor

- Here, you need to create function with arguments. Each argument value can be assigned in the current object by using this keyword.
- The **this keyword** refers to the current object.
- The example of creating object by object constructor is given below.
- **<script>**
- function emp(id,name,salary){
- this.id=id;
- this.name=name;
- this.salary=salary;
- }
- e=new emp(103,"Vimal Jaiswal",30000);
- document.write(e.id+" "+e.name+" "+e.salary);
- **</script>**

Output of the above example
103 Vimal Jaiswal 30000