**JavaScript Assignment**

1. **What are the ways to create the objects?**

There are 3 ways to create an object in Java script

1. Define and create a single object, using an object literal.
2. Define and create a single object, with the keyword new.
3. Define an object constructor, and then create objects of the constructed type.

Using Simple Object Literal :

This is very simple one and most common one.

Using new keyword :

Simple the name itself says object is created using new key word.

Constructor using Object Literal :

If we don’t want to add the behavior to the object means just it’s a data, then we will go with the object literal whereas if we want to add some behavior to it then it is better to go with constructor using Object Literal.

1. **How many ways we can create the arrays ?**
2. Using an array literal (Easiest Way)

var cars = ["Saab", "Volvo", "BMW"];

1. A declaration can span multiple lines

var cars = [

* + - * 1. "Saab",
        2. "Volvo",
        3. "BMW"];

1. Using new keywords

var cars = new Array ("Saab", "Volvo", "BMW");

1. **What are arguments in JavaScript functions ?**

* JavaScript functions have a built-in object called the arguments object. The argument object contains an array of the arguments used when the function was invoked.
* JavaScript arguments are passed by value: The function only gets to know the values, not the argument's locations. If a function changes an argument's value, it does not change the parameter's original value.

1. **What is prototypal inheritance in JavaScript ?**

Prototypal inheritance means, a prototype is a property of functions and of objects that are created by constructor functions. The prototype of a function is an object. Its main use is when a function is used as a constructor.

1. **What are enumerators in JavaScript?**

Due to the changes made in the JavaScript object model of IE version 9 JavaScript properties may be enumerated differently from how they are enumerated in earlier versions.

Example:

The possible values of a gender attribute are restricted to one of the following: "male", "female", or "undetermined". Instead of using these strings as the internal values of the enumeration attribute gender, it is preferable to use the positive integers 1, 2 and 3, which enumerate the possible values.

1. **Callbacks and closures .**

|  |  |
| --- | --- |
| Callbacks | Closures |
| Functions are first-class objects. One of the consequences of this fact is that functions can be passed as arguments to other functions and can also be returned by other  functions.  A function that takes other functions as arguments or returns functions as its result is called a higher-order function, and the function that is passed as an argument is called a callback function. It’s named “callback” because at some point in time it is “called back” by the higher-order function.  Example:  - setTimeout()  - setInterval() | a closure is any function that keeps reference to variables from its parent’s scope even after the parent has returned.  That is practically any function can be considered a closure.  A function can refer to, or have access to –   1. own function scope 2. outer (parent) functions 3. global scope |

1. **Module based programming in JavaScript .**

The module pattern is a common JavaScript coding pattern.

A module fulfills two purposes:

1. It holds content, by mapping identifiers to values.
2. It provides a namespace for those identifiers, to prevent them from clashing with identifiers in other modules.

In JavaScript, modules are implemented via objects.

1. Name spacing: A top-level module is put into a global variable. That variable is the namespace of the module content.
2. Holding content: Each property of the module holds a value.
3. Nesting modules: One achieves nesting by putting a module inside another one.
4. **What is strict mode in Java Script ?**

Strict mode is declared by adding "use strict"; to the beginning of a script or a function. Declared at the beginning of a script, it has global scope. All code in the script will execute in strict mode.

Example:

"use strict";

x = 3.14;

1. **What is the difference between == and ===?**

Loose equality (==)

1. If the two variables are not in the same type, it converts to a common type and then the equality check is done. For example, if (1=='1') it returns true.
2. Loose equality is usually not encouraged to use.

Strict equality (===)

1. No type conversion is done prior to the comparison.
2. If the values are string and a number, it can't be equal. For example, if (1==='1') returns false.
3. **Ternary operator ?**

The **conditional (ternary) operator** is the only JavaScript operator that takes three operands. This operator is frequently used as a shortcut for the [if](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/if...else) statement.

**Syntax**

*condition* ? *expr1* : *expr2*

1. **Difference between public, private and static variables and their use cases ?**

Public variable - a public variable will be accessible by other scripts.

Private variable - a private variable will not be accessible by other scripts.

Static variable - a static variable will be shared between all the instances of your script.