# JavaScript concepts:

1. **What are the ways to create the objects?**
   1. Using an object literal.

Example: var employee = {firstName:"Sudha", lastName:"Gunamgari", age:23, eyeColor:"black"};

* 1. With the keyword new.

Example: var employee = new Object();

employee.firstName = " Sudha ";

employee.lastName = " Gunamgari ";

employee.age = 23;

employee.eyeColor = " black ";

* 1. Defining an object constructor, and then create objects of the constructed type.

Example:

function employee(first, last, age, eye) {

this.firstName = first;

this.lastName = last;

this.age = age;

this.eyeColor = eye;

}

var employee = new employee("Malla Reddy", "Gunamgari", 51, "black");

1. **How many ways we can create the arrays.**
   1. Using an array literal

Example: var cars = ["Ferrari", "Toyota", "Mustang"];

* 1. Using new keyword

Example: var cars = new Array("Ferrari", "Toyota", "Mustang");

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

Function **arguments** are the actual **values** passed to the function as parameters in the function definition.

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

In JavaScript, the inheritance is prototype-based. That means that there are no classes. Instead, an object inherits from another object.

**Example:** When an object rabbit inherits from another object animal, in JavaScript that means that there is a special property rabbit.\_\_proto\_\_ = animal.

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

Enables enumeration of items in a collection. The Enumerator object provides a way to access any member of a collection and behaves similarly to the For...Each statement. The Enumerator object has no properties.

**Syntax:** enumObj = new Enumerator([collection])

**Parameters:** *enumObj :* Required. The variable name to which the Enumerator object is assigned.

*collection:*Optional. Any Collection object.

1. **Callbacks and closures.**

A callback function, also known as a higher-order function, is a function that is passed to another function (let’s call this other function “otherFunction”) as a parameter, and the callback function is called (or executed) inside the otherFunction. A callback function is essentially a pattern (an established solution to a common problem), and therefore, the use of a callback function is also known as a callback pattern.

A closure is an inner function that has access to the outer (enclosing) function’s variables—scope chain. The closure has three scope chains: it has access to its own scope (variables defined between its curly brackets), it has access to the outer function’s variables, and it has access to the global variables.

The inner function has access not only to the outer function’s variables, but also to the outer function’s parameters. Note that the inner function cannot call the outer function’s argumentsobject, however, even though it can call the outer function’s parameters directly.You create a closure by adding a function inside another function.

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

Modules divide programs into clusters of code that, by some criterion, belong together. This chapter explores some of the benefits that such division provides and shows techniques for building modules in JavaScript. The benefits of organizing a program into several files or modules are similar. Structure helps people who aren’t yet familiar with the code find what they are looking for and makes it easier for the programmer to keep things that are related close together.

1. **What is strict mode in JavaScript**

The "use strict" directive is new in JavaScript 1.8.5 (ECMAScript version 5).It is not a statement, but a literal expression, ignored by earlier versions of JavaScript.The purpose of "use strict" is to indicate that the code should be executed in "strict mode".With strict mode, you can not, for example, use undeclared variable.

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

== is just comparing the two values, and if they are of different types, type conversion is done

=== compares the values and well as their types - so no type conversion will be done here.

1. **What is 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

### **Parameters**

**condition**

An expression that evaluates to true or false.

**expr1, expr2**

Expressions with values of any type.

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

A variable declared outside a function, becomes **GLOBAL**.

A global variable has **global scope**: All scripts and functions on a web page can access it.

Private members are made by the constructor. Ordinary vars and parameters of the constructor becomes the private members.

A function is an object. That provides us with a nifty way to create static variables or, in other words, the variables which persist along multiple calls. There are languages which allow to put a static keyword before a variable, and then such variable is not cleared in next calls.