* ES6: Stand for ECMAScript 6 is is a programming standard language by ECMAScript International. It is used by applications to enable client-side scripting. But it will be generate that script to JavaScript at the end of the process.
* Basic
  1. let: similar to var but let has scope. let is only accessible in the block level it is defined. Ex: let age=20; if(true){let age=10; consle.log(age);}
  2. const: is a keywrod that we use to declare variabe which user can’t change or reassigned the value to that variable. It fix value. Ex: const x=50; var x=20; //it will be error
  3. loop:
     1. for … in: is use to loop through an object’s property. Ex: var obj = {a:4, b:6, c:8, d:10}; for (var p in obj){console.log(obj[p]);}
     2. for … of : is used to iterates through list of elements. Ex: let arr = [2,3,4,1];  
        for (let value of arr) {console.log(value);}
  4. function:
     1. rest parameter :act as placeholders for multiple arguments of the same type. Ex: function func(…params){console.log(params.legth);}func(); // 0 func(3, 4, 23, 34); // 4
     2. anonymous function : is a functions that are not bound to an identifier (function name). Ex: var func = function(x,y){return x + y} function item() { var result; result = func(46,74); console.log("item result : " +result);} item();
     3. object:
  5. object:

is an instance which contains a set of key value pairs. Ex : var person ={ firstname: “savong” , lastname:”sam” , func:function(){return "Hello!!"},};

* 1. class:

Classes can be created using the class keyword in ES6. Classes can be included in the code either by declaring them or by using class expressions. A class definition can include the following −

* + 1. **Constructors** − Responsible for allocating memory for the objects of the class.
    2. **Functions** − Functions represent actions an object can take. They are also at times referred to as methods.

These components put together are termed as the data members of the class.

**Note** − A class body can only contain methods, but not data properties.

ES6 supports the concept of **Inheritance**. Inheritance is the ability of a program to create new entities from an existing entity - here a class. Inheritance can be classified as −

* **Single** − Every class can at the most extend from one parent class.
* **Multiple** − A class can inherit from multiple classes. ES6 doesn’t support multiple inheritance.

**Multi-level** − Consider the following example.

* 1. Promises:

**Promises** are a clean way to implement async programming in JavaScript (ES6 new feature). Prior to promises, Callbacks were used to implement async programming.

* + 1. Callback:

A function may be passed as a parameter to another function. This mechanism is termed as a **Callback**.

* + 1. Asynccallback:

The setTimeout() method takes two parameters −

* + - * A callback function.
      * The number of seconds after which the method will be called.
  1. Module:
     1. exporting module:

To make available certain parts of the module, use the export keyword.

Export a single value or element - Use export default

export default element\_name Export multi value or elements export {element\_name1, …}

* + 1. importing a module:

To be able to consume a module, use the import keyword. Following is the syntax for the same.

Import a single value or element

import element\_name from module\_name

Export multiple values or elements

import {element\_name1, …} from module\_name