Node JS - Educacion IT - Apuntes.

**NPM -** node package manager - sirve para organizar las librerías de node.

**package.json** - tiene todo los frameworks que voy a utilizar.

**npm install** lee el package y empieza a instalar todo.

**npm init** (primer comando) es el comando para empezar a generar el proyecto. Es medio interactivo, me sugiere el nombre, versiones, test, autos, licencia y etc.

Importantes: Entry point - punto de entrada. Por default es index.js

Con que exista el package node ya es considerado un proyecto node.

**npm start** (segundo comando) ejecuta el proyecto. Para que funcione tengo que en script.start tener la ruta a mi index => "start": "node index.js",

Para asignar variables se puede usar VAR - CONST - LET

Si uso el **use strict** tengo que si o si utilizar var. Te exije que le pongas el var adelante o no.

Conviene usar let o const.

**El const es mas recomendable porque es más facil para debugger.**

La variables pueden ser números, strings, booleanos y function

Declarar funciones existen dos maneras:

Una función es como una variable.

const miFuncion = function(){

var a = 23; // vive solo dentro de la funcion

console.log('Hola');

}

miFuncion();

En este entorno si se respeta el var. El var fuera de esa funcion no esta definido.

Solo vive dentro del scope de esa función.

Hay dos maneras para declarar una función, una es la anterior en donde le asigno una función a una variable.

La segunda no la guardo dentro de una variable y la puedo ejecutar antes de declarar la función. Con esta nomenclatura el nombre de la función existe en todo el archivo.

Escribiéndola como variable estoy obligada a llamarla luego de haberla creado. (funciones anónimas

otraFuncion()

function otraFuncion(){

console.log('Chau');

}

Adentro de las funciones puedo pasarle parametros, que estan asignados por fuera de la función. Las funciones también pueden estar dentro de objetos.

const alumno = {

inscribir : function(mensaje){

console.log(mensaje);

}

}

alumno.inscribir("este es un mensaje");

Todas las funciones tiene un arguments. Es como si fuese una array que contiene todos los parametros que le estoy pasando.

const alumno2 = {

inscribir : function(arguments){

console.log(arguments[0]);

}

}

alumno2.inscribir(23 , true);

**Objetos**: Agrupo propiedades relacionadas de un mismo elemento. Es una variable que adentro tiene propiedades mas chiquitas.

const alumno1 = {

nombre: 'Pilar',

dni : '23',

}

Puedo acceder a esas propiedades mediante el nombre de la variable.lapropiedad.

console.log(alumno1.nombre);

**Funciones anónimas:**

Si en mi proyecto tengo dos variables de mismo nombre, una va a estar pisando a la otra.

la función anónima me permite ‘elegir’ cual quiero que sea compartida y cual no.

Por una buena práctica yo solo debo tener una variable global.

Js no permite que una variable empiece con function solamente, por eso se le pone el paréntesis adelante y luego lo () para autoejecutarse. De alguna manera ‘hackeo’ a js para que me permita tener solo una variable.

var a = (function(){

var alumno

console.log('fn anonima');

return 45

}());

**Closures**