Isomorfismo

Tres capas de la progra web: Frontend, backend y persistencia de datos

Lenguaje Bajo Nivel: Instrucciones más complejas y específicas.

Lenguaje Alto Nivel: Instrucciones menos complejas y detalladas.

Javascript es el único que puede estar en las tres capas: Frontend, Backend, Persistencia de Datos (mongo DB, Couch DB, Firebase, entre otros). Es dinámico, interpretado, de alto nivel, débilmente tipado, multi paradigma. Sensible a MAYÚSCULAS y minúsculas, no necesita los puntos y comas al final de cada línea.

Las variables deben empezar con (‘letra’, ‘\_’, ‘&’)

Javascript usa snake\_case: Separar las palabras con ‘\_’. Ej. Mi\_archivo\_javascript.js;

Javascript usa UPPER\_CASE en constantes. const UNA\_CONSTANTE = “una constante”;

Javascript usa UpperCamelCase\_CASE en clases. class SerHumano {……..}

Javascript usa lowerCamelCase en Objetos. const unObjeto = {…};

Javascript usa lowerCamelCase en Primitivos. unaCadena = ;

Palabras reservadas:

* A: abstract
* B: boolean, break, byte
* C: case, catch, char, class, const, continue
* D: debugger, default, delete, do, double
* E: else, enum, export, extends
* F: false, final, finally, float, for, function
* G: goto
* I: if, implements, import, int, instanceof, int, interface
* L: let, long
* P: package, private, protected, public
* R: return
* S: short, static, super, switch, synchronized
* T: this, throw, throws, transient, true, try, typeof
* V: var, volatile, void
* W: while, with

Tipos de datos en Jacascript:

* Primitivos: se accede directamente al valor.

1. string
2. number
3. null 🡪 equivale a un valor nulo
4. boolean
5. undefined 🡪 cuando una variable no tiene ningún valor.
6. NaN 🡪 La operación no es posible. (es como Syntaxis error, hay que impedir que suceda, hay que hacer validaciones)

* Compuestos: se accede a la referencia del valor.

1. object = {}
2. array = {}
3. function () {}
4. Class {}
5. Etc.