Javascript

Programación Web I

Comisión Miércoles Noche

Prof. Damián Spizzirri

Prof. Esteban David Alaníz

Comisión Viernes Mañana

Prof. Gabriel Panik

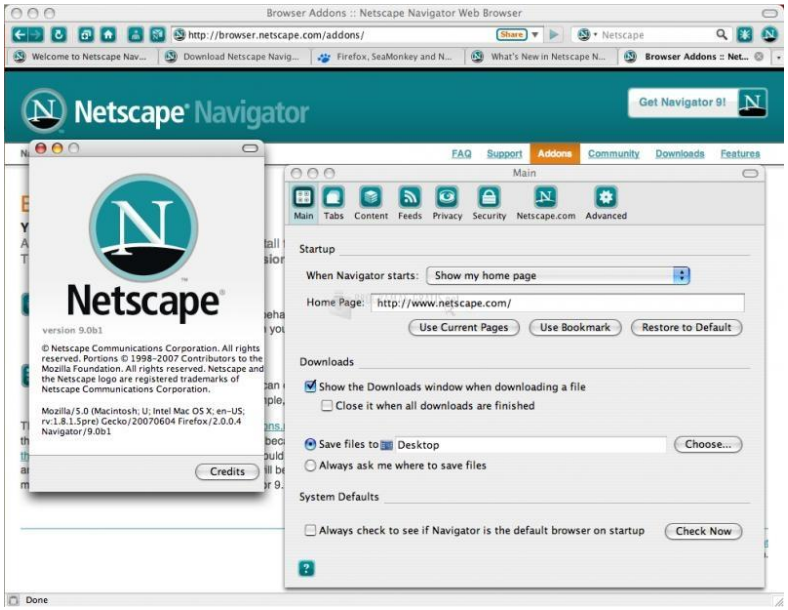
Prof. Javier Barraza

JAVASCRIPT

- Lenguaje Interpretado
- Multiplataforma
- Debilmente Tipado y Dinámico
- Orientado a Objetos y Eventos
- Imperativo
- Basado en prototipos

JAVASCRIPT

- Creado por Brendan Eich, programador de Netscape v2.0 en 1995. ->Livescript ->javascript



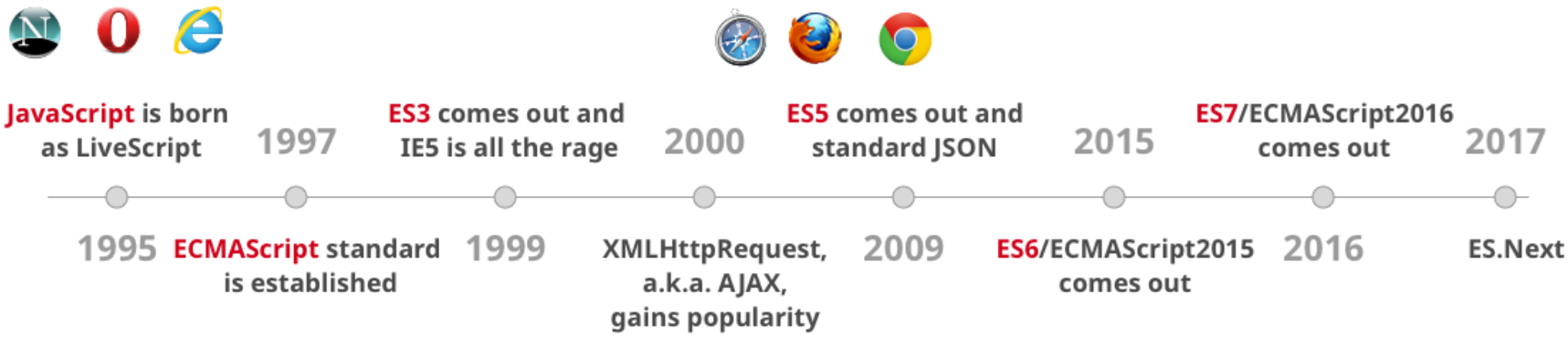
- Microsoft lanzó Jscript para Internet Explorer 3.0

ECMAScript

ECMA

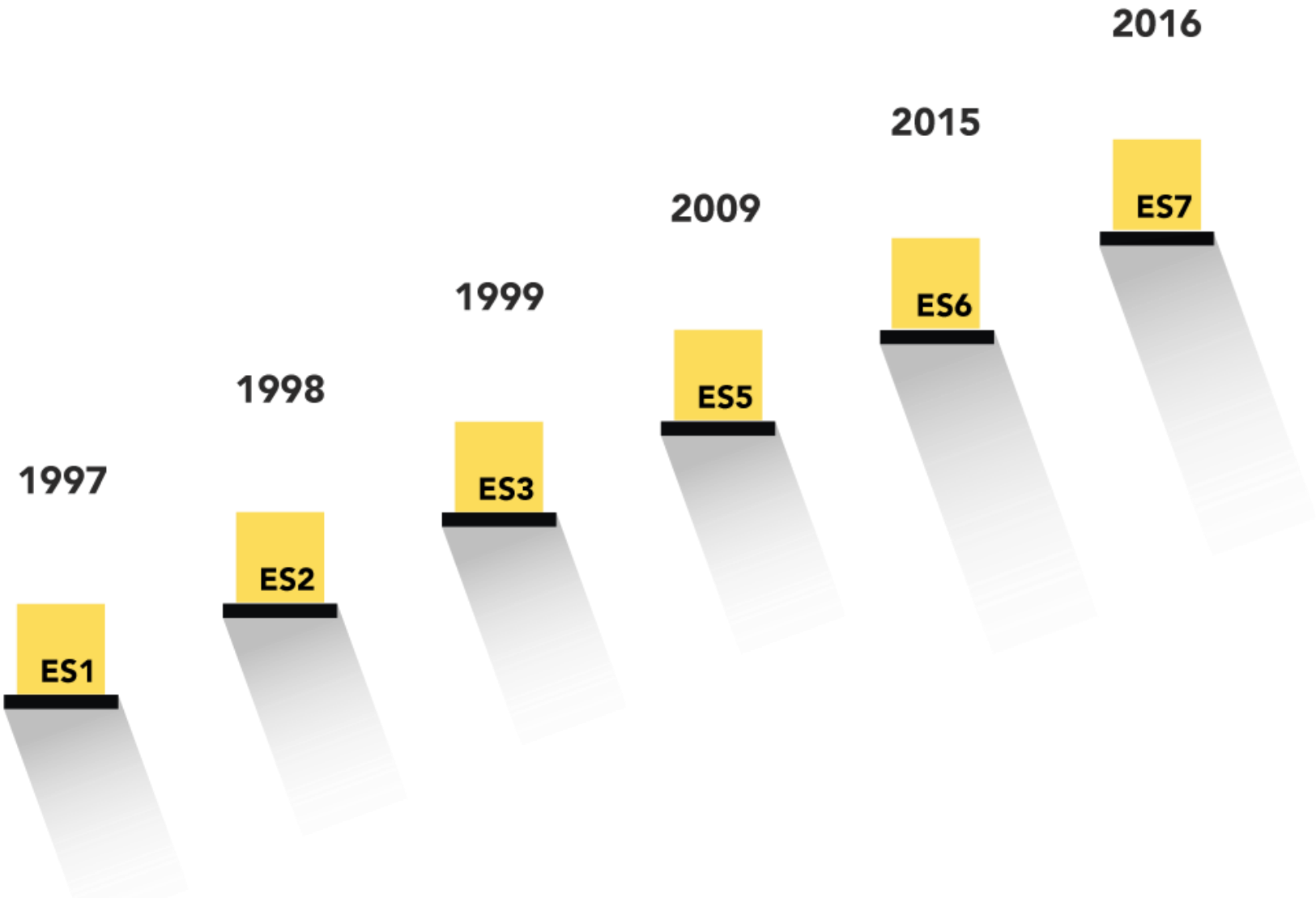
European Computer Manufacturers Association

https://www.w3schools.com/js/js_versions.asp



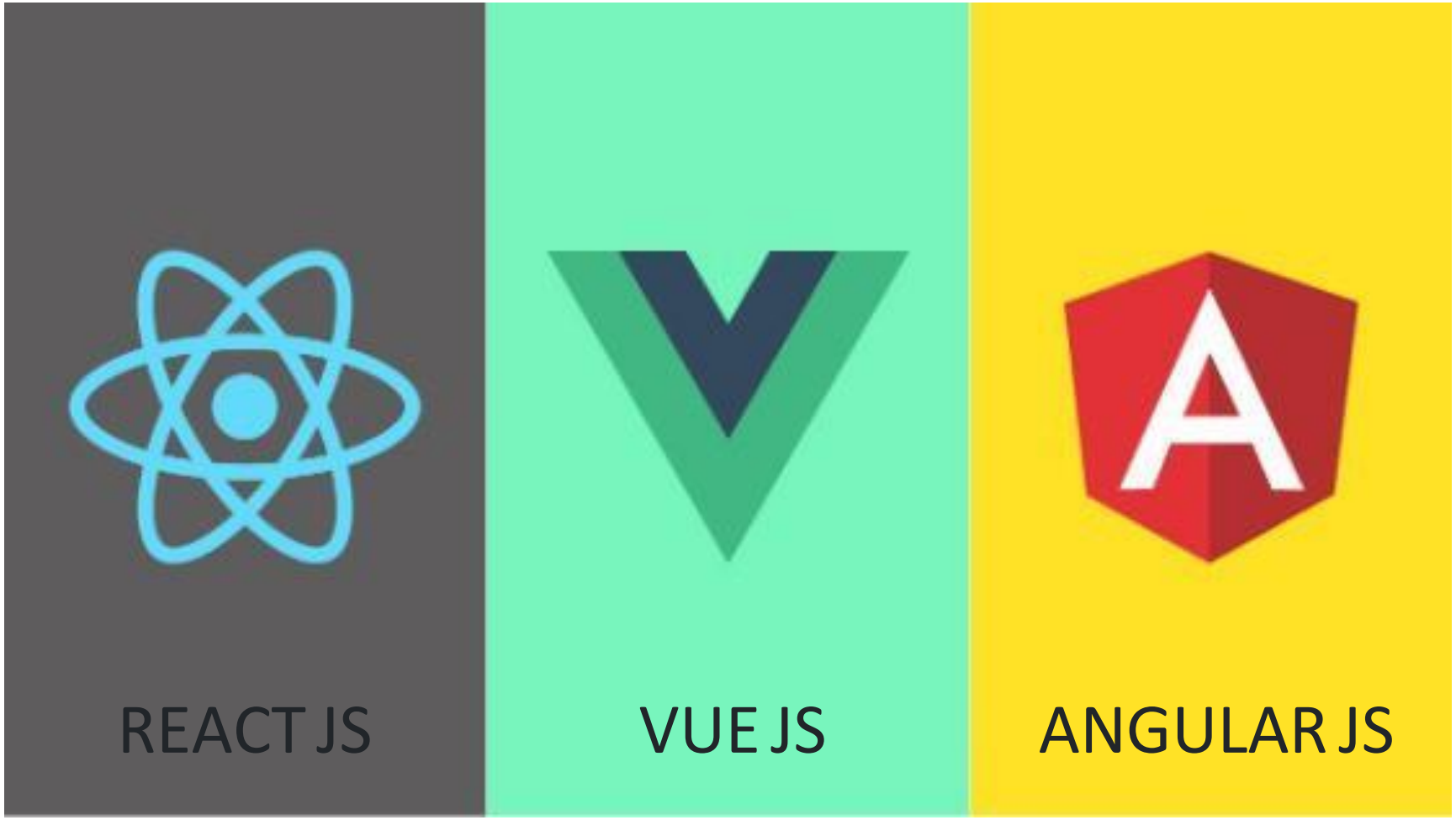
Estandarizar de un lenguaje de script multiplataforma e independiente de cualquier empresa

ECMAScript

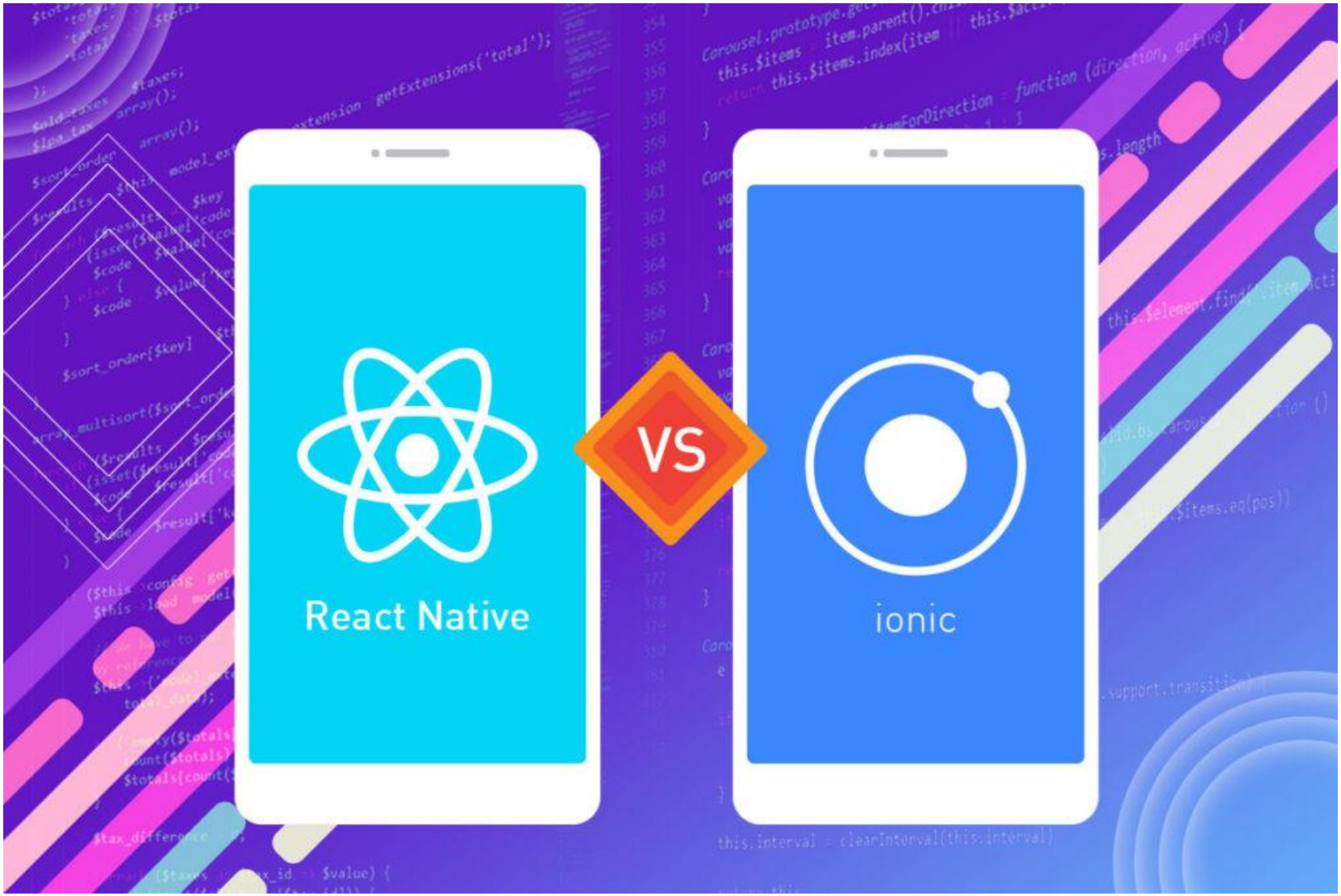


Librerías JAVASCRIPT

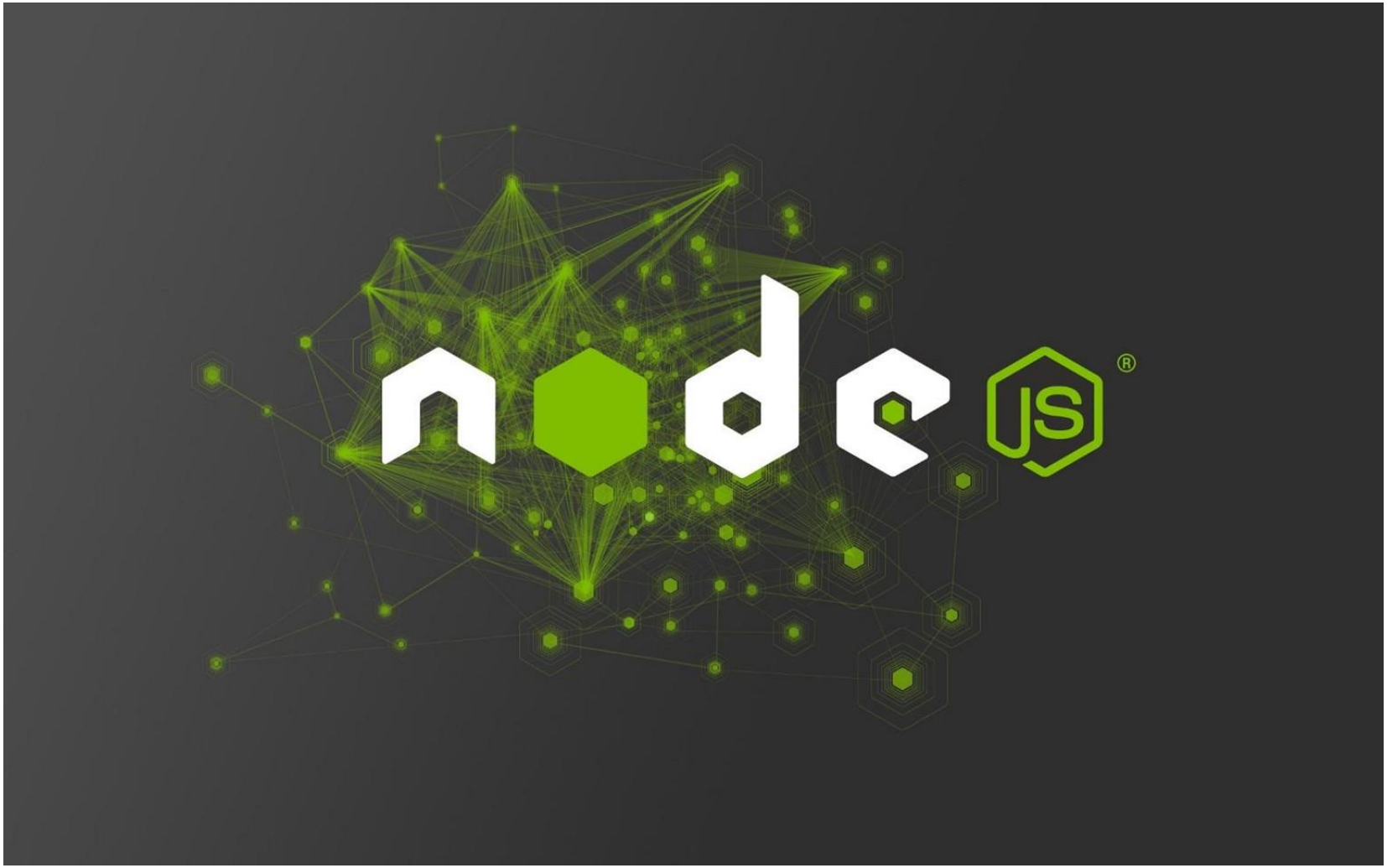
Frameworks JAVASCRIPT



Frameworks JAVASCRIPT



Frameworks JAVASCRIPT



INCLUIR JS en HTML

```
<head>
    <meta charset="UTF-8">
    <script>
        alert ("Primer Script");
    </script>
</head>
```

INCLUIR JS en Archivo Externo

1. Crear un archivo con la extensión .js dentro de una carpeta llamada js
2. Vincular el código javascript a nuestro html

```
<head>
    <meta charset="UTF-8">
    <script src="js/codigo.js">
    </script>
</head>
```

MOSTRAR DATOS

- //popup
```
alert ("");
```

- //en el documento HTML
```
document.write ("");
```

- //en la consola
```
console.log("");
```

SINTAXIS JAVASCRIPT

- **No** se tienen en cuenta los espacios en blanco y las nuevas líneas
- Se distinguen las mayúsculas y minúsculas
- **No** se define el tipo de las variables
- **No** es necesario terminar cada sentencia con el carácter de punto y coma
- Se pueden incluir comentarios
  - Una sola línea //
  - Varias líneas (entre /* y */)

Variables

- Almacenar un dato
- Se crean con la palabra reservada var

```
//declaración
let nombre;

//inicialización
nombre ="Pablo";
```

Variables

El nombre de una variable:

- **NO** puede empezar con un número
- **NO** puede usar carácteres especiales, ni espacios en blanco
- **Se puede** utilizar el signo $ y el guión bajo (_)
- Distingue Mayúsculas de Minúsculas

Variables

Convención para los nombres de Variables:

camelCase

Primera palabra toda en minúscula, primera letra de cada palabra siguiente Mayúsculas

Por ejemplo:

```
let nombreApellido;
let numeroDocumento;
```

Tipos de Datos Primitivos

TIPO	Valores
Number	Numéricos incluidos números negativos y decimales
String	Cadenas de Texto. Comillas simples o dobles.
Boolean	Valores posibles true o false
Null	Nulos o vacíos
Undefined	Sin definir, cuando una variable no fue inicializada.

PEDIR DATOS

Palabra reservada prompt

```
prompt("Mensaje al usuario");
```

```
let nombre;
```

```
nombre = prompt("Ingrese su nombre");
```

Constantes

- Almacenar un dato NO Variable
- Se crean con la palabra reservada const

```
//declaración
const PI = 3.14;
```

Constantes

Convención para los nombres de Constantes:

UPPERCASE

Toda la palabra en MAYÚSCULA

Por ejemplo:

```
const PI;
const IVA;
```

Template Strings

```
let nombre1 = "JavaScript";
let nombre2 = "ES6";

console.log(`Concateno en
${nombre1} con ${nombre2}`);
```

Operadores de Asignación

- =
- +=, -=, *=, /=

Ejemplos:

```
numeroUno = 8;
numeroDos = 3;
numeroUno += 3;
```

Operadores Aritméticos

- +
- -
- *
- /
- %
- ++ (Incremento)
- -- (decremento)

Operadores Lógicos

- Negación (!)
- AND (&&)
- OR (||)

Operadores Lógicos

Ejemplos:

```
a = 8;
b = 3;
c = 3;
document.write( (a == b) && (c > b) );
document.write( (a == b) && (b == c) );
document.write( (a == b) || (b == c) );
document.write( (b <= c) );
document.write( !(b <= c) );
```

Operadores Relacionales

- >, >= Mayor, mayor o igual
- <, <= Menor, menor o igual
- == Igual
- != Distinto

Operadores Relacionales

```
let numero1 = 3;
let numero2 = 5;
resultado = numero1 > numero2;
resultado = numero1 < numero2;
numero1 = 5; numero2 = 5;
resultado = numero1 >= numero2;
resultado = numero1 <= numero2;
resultado = numero1 == numero2;
resultado = numero1 != numero2;
numero1 = 5; numero2 = 4;
resultado = numero1 == numero2 ;
resultado = numero1 === numero2;
```

PARSEAR DATOS

- `isNaN(variable);`
  `//devuelve true si no lo es`
- `parseInt(variable);`
  `//convierte en número entero`
- `parseFloat(variable);`
  `//convierte en número flotante`
- `String(variable);`
  `//convierte en cadena de texto`

CONDICIONALES IF

```
if(condicion){
    //instrucciones a ejecutar
    si se cumple la condición
}
```

CONDICIONALES ELSE

```
if(condicion){


} else{
   //instrucciones a ejecutar
  si NO se cumple la condición
  anterior
}
```

CONDICIONALES ELSE if

```
if(condicion){

} else if(condicion){
   //instrucciones a ejecutar
   si NO se cumple la condición
   anterior y SI esta
}
```

CONDICIONALES SWITCH

```
switch(nomVariable) {
case "opcion1" :
    instrucciones;
   break;
case "opcion2" :
    instrucciones;
    break;
.....................
case "opcionN" :
    instrucciones;
    break;
default :
    instrucciones;
}
```

CONDICIONALES SWITCH

```
switch(numero) {
        case 5:
        ...
        break;
        case 8:
         ...
         break;
        case 20:
        ...
        break;
        default:
        ...
        break;
}
```

CONDICIONALES SWITCH

```
switch(numero) {
        case 5:
        ...
        break;
        case 8:
         ...
         break;
        case 20:
        ...
        break;
        default:
        ...
        break;
}
```

BUCLE FOR

```
for(let i=0;i<=valor;i++){
    //instrucciones a repetir
}
```

BUCLE WHILE

```
while(condicion a evaluar){
  // bloque a ejecutar
        mientras la condición
      sea verdadera
}
```

CONDICIONALES ELSE if

```
do{

} while(condicion)
```

SINTAXIS

Declaración de una función:

```
function nombreFuncion( ){
    //instrucciones
}
```

SINTAXIS

Llamado a esa función:

```
nombreFuncion();
```

Ejemplo

```
function sumar(){
  var num1 = 5;
  var num2 = 8;
  var suma = num1 + num2;
  document.write("Total: "+suma);

}

sumar();
```

FUNCIONES CON PARAMETROS

```
function nombreFunc(parametros...){


 }
 nombreFuncion(parametro1, prametro2);
```

FUNCIONES CON PARAMETROS

```
function sumar(num1,num2){
  var suma = num1 + num2;
  document.write("Total: "+suma);
}

sumar(3,6);
```

FUNCIONES VALOR RETORNO

```
function nombreFuncion(){
  return variable;
}
```

FUNCIONES VALOR RETORNO

```
function sumar(num1,num2){
  let suma = num1 + num2;
  return suma;

}
 document.write(sumar(4,5));
```

FUNCIONES útiles Números

`toFixed();`

Devuelve el número original con tantos decimales como los indicados por el parámetro `digitos` y realiza los redondeos necesarios.

```
let numero1 = 4564.34567;
numero1.toFixed(2); // 4564.35
numero1.toFixed(6); // 4564.345670

numero1.toFixed(); // 4564
```

FUNCIONES útiles Números

isNaN();

Permite proteger a la aplicación de posibles valores numéricos no definidos

```
isNaN(numero1);// devuelve true si no es un número.
```

FUNCIONES útiles Cadenas

`length`

Calcula la longitud de una cadena de texto (el número de caracteres que la forman)

```
let mensaje = "Hola Mundo";

let numeroLetras = mensaje.length;
```

FUNCIONES útiles Cadenas

`+ o concat`

(Se emplean para concatenar varias cadenas de texto)

```
let mensaje1 = "Hola";
let mensaje2 = " Mundo";

let mensaje = mensaje1 + mensaje2;

let mensaje3 = mensaje1.concat(mensaje2);
```

FUNCIONES útiles Cadenas

toUpperCase()

Transforma el texto en mayúscula

```
let mensaje1 = "Hola";
let mensaje2 = mensaje1.toUpperCase();
```

toLowerCase()

Transforma el texto en minúscula

```
let mensaje1 = "Hola";
let mensaje2 = mensaje1.toLowerCase();
```

FUNCIONES útiles Cadenas

charAt()

Obtiene el caracter que en la posición indicada

```
let mensaje = "Hola";
let letra = mensaje.charAt(0); // letra = H
letra = mensaje.charAt(2);     // letra = l
```

substring(inicio, final)

Toma una porción del texto

```
   let mensaje = "Hola Mundo";
let porcion = mensaje.substring(2) //"la Mundo"
porcion = mensaje.substring(1, 8); //"ola Mun"
```

FUNCIONES útiles Cadenas

`indexOf(caracter)`

(Calcula la posición en la que se encuentra el carácter indicado dentro de la cadena de texto. Si el carácter se incluye varias veces dentro de la cadena de texto, se devuelve su primera posición empezando a buscar desde la izquierda. Si la cadena no contiene el carácter, la función devuelve el valor -1)

```
let mensaje = "Hola Carola";
let posicion = mensaje.indexOf('a');
//posicion =3
posicion = mensaje.indexOf('b'); // posicion=-1
```

FUNCIONES útiles Cadenas

`lastIndexOf(caracter)`

(calcula la última posición en la que se encuentra el carácter indicado dentro de la cadena de texto. Si la cadena no contiene el carácter, la función devuelve el valor -1. La posición devuelta se calcula empezando a contar desde el principio de la palabra)

```
let mensaje = "Hola Carola";
let posicion = mensaje.lastIndexOf('a');
//posicion =10
posicion = mensaje.lastIndexOf('b'); //
posicion=-1
```

Arrays

```
let variable1 = new Array();

let variable1 = new Array(10);

let variable1 = new Array(2, "hola",

true, 45.34);
```

Arrays

```
let variable1 = new Array();

variable1[0] = 2;

variable1[1] = "hola";

variable1[2] = true;

variable1[3] = 45.34;
```

Arrays

```
let dias = ["Lunes", "Martes", "Miércoles",

"Jueves", "Viernes", "Sábado", "Domingo"];



for(var i=0; i<7; i++) {

  alert(dias[i]);

}
```

Arrays

```
let dias = ["Lunes", "Martes", "Miércoles",

"Jueves", "Viernes", "Sábado", "Domingo"];



for(i in dias) {

  alert(dias[i]);

}
```

Arrays

```
let dias = ["Lunes", "Martes", "Miércoles",

"Jueves", "Viernes", "Sábado", "Domingo"];



dias.forEach((item)=>{

    alert(item);

});
```

Funciones útiles para arrays

Funciones útiles para arrays

- **length** (Calcula el número de elementos de un array)

```
var vocales = ["a", "e", "i", "o", "u"];
var numeroVocales = vocales.length;                    // numeroVocales = 5
```

- **concat()** (Se emplea para concatenar los elementos de varios array)

```
var array1 = [1, 2, 3];
array2 = array1.concat(4, 5, 6);          // array2 = [1, 2, 3, 4, 5, 6]
array3 = array1.concat([4, 5, 6]);        // array3 = [1, 2, 3, 4, 5, 6]
```

- **join(separador)** (Es la función contraria a split(). Une todos los elementos de un array para formar una cadena de texto. Para unir los elementos se utiliza el carácter separador indicado)

```
var array = ["hola", "mundo"];
var mensaje = array.join("");             // mensaje = "holamundo"
mensaje = array.join(" ");                // mensaje = "hola mundo"
```

Funciones útiles para arrays

Funciones útiles para arrays

- **pop() (**Elimina el último elemento del array y lo devuelve. El array original se modifica y su longitud disminuye en 1 elemento)

```
var array = [1, 2, 3];
var ultimo = array.pop();        // ahora array = [1, 2], ultimo = 3
```

- **push()** (Añade un elemento al final del array. El array original se modifica y aumenta su longitud en 1 elemento. También es posible añadir más de un elemento a la vez)

```
var array = [1, 2, 3];
array.push(4);                   // ahora array = [1, 2, 3, 4]
```

- **shift()** (Elimina el primer elemento del array y lo devuelve. El array original se ve modificado y su longitud disminuida en 1 elemento)

```
var array = [1, 2, 3];
var primero = array.shift();              // ahora array = [2, 3], primero = 1
```

Funciones útiles para arrays

Funciones útiles para arrays

- **unshift()** (Añade un elemento al principio del array. El array original se modifica y aumenta su longitud en 1 elemento. También es posible añadir más de un elemento a la vez)

```
var array = [1, 2, 3];
array.unshift(0);                    // ahora array = [0, 1, 2, 3]
```

- **reverse()** (Modifica un array colocando sus elementos en el orden inverso a su posición original)

```
var array = [1, 2, 3];
array.reverse();                     // ahora array = [3, 2, 1]
```

Juego piedra, papel y tijera

Cree un archivo html con un javascript asociado. El cual le deberá pedir al usuario que ingrese las palabras piedra, papel o tijera.
Cree un array con estos 3 valores. El programa deberá elegir aleatoriamente un valor del array y compararlo con lo ingresado por el usuario