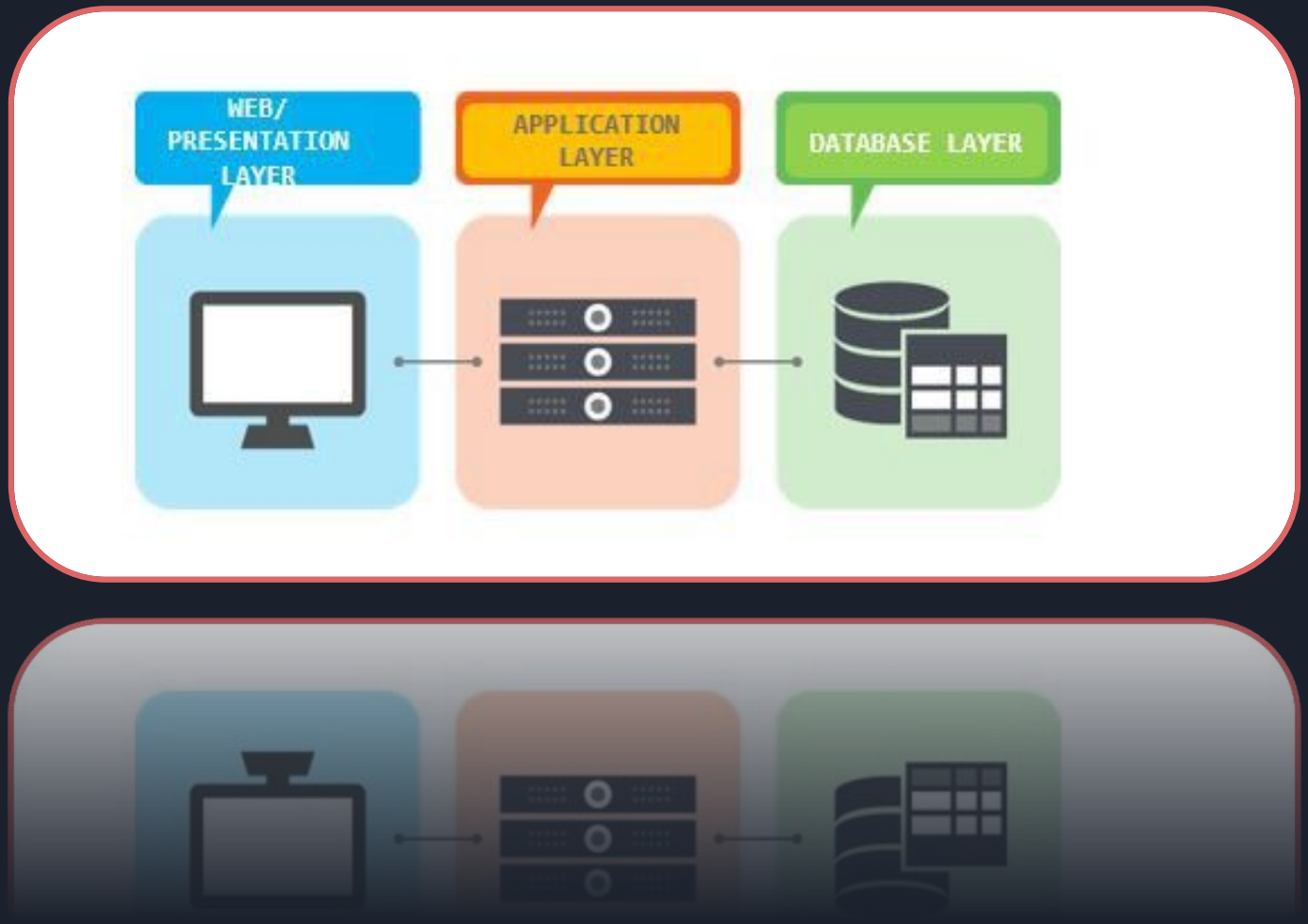
Desarrollo Web en Entorno Cliente

Javascript

Aplicaciones web

Arquitectura de tres capas



Front - end vs. Back - end

Tecnologías Front - end

Entorno de desarrollo

Básicos de JavaScript

Ejecución

Etiqueta <script>

Ejecuta código directamente

O carga ficheros .js con código.

Características de JavaScript

Lenguaje interpretado

Sentencias delimitadas por ;

Palabras reservadas

Sensible a mayus-minus

Comentarios: de una línea //
de bloque /* */

Bloques de sentencias: { ... }

Cuadros de diálogo y consola

Alert .- saca un mensaje y un botón

Confirm .- saca un mensaje y dos botones, Aceptar y Cancelar

Prompt .- saca un cuadro para meter texto y dos botones, Aceptar y Cancelar

console.log .- saca mensaje por la consola de la herramienta de desarrollador

Variables y tipos de datos

Tipos básicos: strings, numbers y booleanos

Tipo complejo: Objeto (un array es un objeto)

undefined

null

Operador typeof para saber el tipo de una variable

Declaración de variables

Con las palabras reservadas ***var, let y const***

var.- declara una variable de ámbito global o local a una función

let .- declara una variable en un bloque de código

const .- declara una constante

Tipo de dato number

var x = 1;

var x = 1.0;

var x = Infinity //número especial infinito

NaN; representa resultados de operaciones que no son números.

Strings - declaración

var cad1=”hola”;
var cad2=’hola’;

Esto permite usar las comillas en texto:

“entre comillas dobles uso la simple ‘ por ejemplo”

‘entre comillas simples uso la doble “ por ejemplo’

Strings - plantillas

Se pueden usar plantillas con ${...}

var nombre=”alberto”
console.log(“El se llama ${nombre}”);

Y con expresiones más complejas:

console.log(“dos por cuatro son ${2*4}”);

Strings - secuencias de escape

\n - salto de línea
\t - tabulador
\r - retorno de carro
\f - salto de página
\v - tabulador vertical
\” - comillas dobles
\’ - comillas simples
\b - retroceso
\\ - barra invertida

Booleanos

true

false

Resultado de operadores de comparación

Se consideran false: texto vacío, 0, false, NaN, undefined y null

Operadores

Aritméticos .- operaciones matemáticas

Relacionales .- comparaciones

Encadenamiento .- operador +

Lógicos .- negación, and y or

De bit .- operan a nivel de bit

Asignación .- incremento, decremento y comprimidos

Conversiones de tipos

Conversión automática .- javascript intenta convertir los tipos datos de forma automática.

“2” * 3 = 6

“2” + 3 = 23

“hola” * 3 = NaN // se evalúa con isNan

true * 3 = 3

Forzar conversión

Number(“3”) .- devuelve el número 3

String(3) .- devuelve la cadena “3”

parseInt .- devuelve un entero y permite convertir de números de otras bases:
parseInt(“1010”,2) devuelve 10

parseFloat .- devuelve decimales

Control de flujo

condicional simple: if
condicional compuesto: if ... else
condicional anidado: if ... else if ... else
bucle while
bucle do ... while
bucle for
bucle for ... in
bucle for ... of

sentencias break y continue

Strings - comparación

Comparación de strings estricta: ==
La secuencia unicode da problemas en la ordenación.
Comparación de strings sin mayus/minus y con el orden alfabético correcto:
str1.localecompare(str2)

- -1 si str1 < str2
- 0 si str1 = str2
- 1 si str1 > str2

Strings - métodos y propiedades

str.length .- tamaño del string

str.charAt(3) .- carácter en la posición 3

str.toUpperCase() .- texto en mayúsculas

str.toLowerCase() .- texto en minúsculas

str.indexOf() .- posición donde empieza un subtexto

str.lastIndexOf() .- última posición de subtexto

str.endsWith()/str.startsWith() .- cierto si finaliza/empieza con un texto determinado

Strings - métodos y propiedades

str.replace() .- cambia texto

str.trim() .- quita espacios iniciales y finales

str.slice() .- extrae texto

str.substr() .- como slice

str.split() .- divide una cadena

String.fromCharCode() .- convierte código unicode en texto

Arrays - declaración

```
var a = []; //array vacío

var a = new array(); //array vacío

a[3]="hola"; //coloca en la posición 3 la palabra hola
a[0]="adios"; //coloca en la posición 1 la palabra adios

console.log(a[0]); //imprime adios
console.log(a[1]); //imprime undefined

var a = [1,3,5,7,9]; //crear array con valores
var a = new array(1,3,5,7,9); //crear array con valores
```

Cuidado con la declaración const

Arrays

delete .- para borrar elementos de un array

Los elementos de un array puede ser de cualquier tipo, incluso puede haber arrays dentro de arrays.

Estructuras para recorrer arrays:

for .- estructura normal
for..in .- recorre los índices con valor de un array
for..of .- recorre los valores de un array, no salta los indefinidos
(Funcionan con strings)

Arrays - propiedades y métodos

length .- número de elementos

instanceof .- permite saber si una variable es de tipo array

push() .- pone un elemento al final del array
pop() .- quita el último elemento
unshift() .- pone un elemento al principio del array
shit() .- quita el primer elemento

concat() .- une dos arrays
slice() .- obtiene una parte del array
splice .- elimina elementos de un array

Arrays - propiedades y métodos

join() .- convierte un array en un string

indexOf() .- busca un elemento en un array y da la posición
lastIndexOf() .- busca el último elemento en un array y da la posición
include() .- cierto si está el elemento, falso si no lo está

reverse() .- da la vuelta al array
sort() .- ordena el array, este método tiene la opción de poder usar una función anónima para ordenar.

Arrays - asignación de valores

propiedad que presentan los arrays que permiten la asignación de varios valores a varias variables:

//asignar coordenadas de “golpe”

```
var [x,y,z]=[2,5,1];
```

x vale 2 y vale 5 z vale 1

Se pueden asignar expresiones:

```
var [suma,resta]=[x+y,x-y];
```

Muy útil para hacer intercambio de valores:

```
[a,b]=[b,a];
```

Arrays - propagación (spread)

Característica de javascript para asignación de variables. El operador son tres puntos —> ...

```
var arrayEjem=[1,2,3];
var [x,y,z]=[...array];  //x vale 1      y vale 2        z vale 3
```

Otro ejemplo:

```
var a, b, unArray;
[a,b,...unArray]=[1,2,3,4,5];    a vale 1      b vale 2
unArray vale [3,4,5]
```

Sets- conjuntos de datos

Declaración

var datos = new Set();

En esta estructura de datos se guardan valores pero no admite duplicados.

Insertar valores con el método add()
datos.add(5);
datos.add(“hola”);
datos.add(5);

la variable datos contiene el número 5 y la cadena “hola”

Sets- conjuntos de datos

Inicialización con array:

var datos = new Set([1,3,5,7,9]);

Cuidado con inicializar con un string
var datos = new Set(“hola”);

datos contiene ‘h’, ’o’ ,’l’ ,’a’
para añadir una cadena se usa el método add().

Sets- métodos y propiedades

datos.size; //propiedad que da el tamaño del conjunto

datos.delete(5); //método que elimina un valor

datos.has(“hola”); //cierto si el elemento existe
//falso si el elemento no existe

Para convertir un set en array se usa la propagación ...
var unArray = [...datos]

Para recorrer un conjunto se usa for..of

```
for (var dat of datos)
    console.log(dat);
```

Maps - clave-valor

Son estructuras de datos de tipo array asociativo, es decir, asocia a una clave un valor.

Declaración:
var edad = new Map();

Añadir valores:
edad.set(“juan”, 15);
edad.set(“ana”, 17);

Las claves no se pueden repetir, se usan para cambiar valores.

Maps - clave-valor

Declaración con array:

var edad = new Map([[“juan”,15],[“ana”,17],[“jose”,14]]);

esto crea el mapa:

```
Map {
    “juan” => 15,
    “ana” => 17,
    “jose” => 14
}
```

Maps -métodos y propiedades

var num=edad.get(“juan”); //obtener el valor de la clave “juan”

edad.has(“ana”); //devuelve cierto porque “ana” es clave del mapa edad

edad.delete(“jose”); //borra el elemento con clave “jose”

edad.keys(); //devuelve un array de las claves

edad.values(); //devuelve un array de los valores

var unArray = [...edad]; // crea un array desde el mapa

Maps - recorrer

La estructura for..of es la indicada para recorrer bucles.

```
for (var dat of edad)
    console.log(dat);   //dat contiene un array con la forma
                        // [clave. valor]

for ( [clave,valor] of edad )
    // la variable clave contiene en cada iteración un
nombre
    // la variable valor contiene en cada iteración una
edad
```