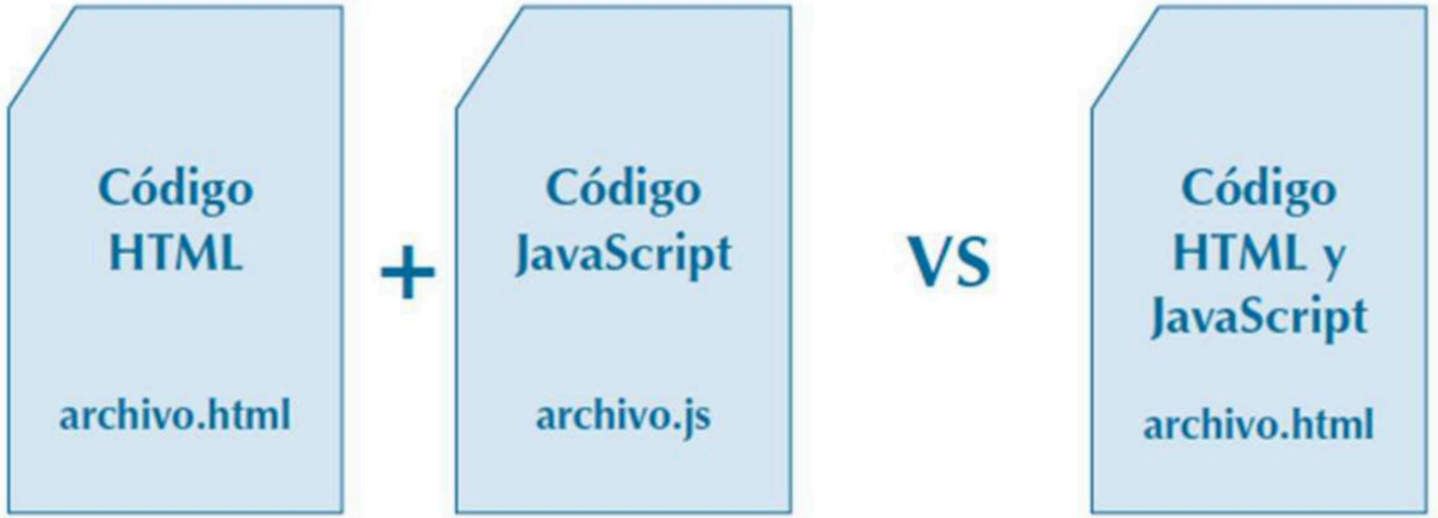
TEMA 1. JAVASCRIPT (II)

FRONTEND

DIANA GARCÍA-MIGUEL LÓPEZ

INTEGRACIÓN JAVASCRIPT EN HTML



SINTAXIS JAVASCRIPT (II)

FUNCIONES

Las funciones son uno de los elementos de la base de la programación estructurada. Reutilizar el código es algo básico en cualquier lenguaje de programación porque la regla que se debe seguir es "escribe una vez y utilízala tantas veces como puedas". La variación del resultado de las funciones dependerá de sus argumentos.

Un ejemplo de estructura básica de una función es el siguiente:

```
function suma(a, b) {
    return a + b;
}
```

Como se puede ver, se tiene la palabra reservada *function* para que JavaScript interprete que se encuentra frente a una función seguida de su nombre (en el ejemplo anterior, *suma*). Esta función aceptará dos parámetros *a* y *b* y devolverá (return) la suma de ambos.

* function nombre (parámetros o no parámetros){}
* Puede recibir o no parámetros (a,b,…) o ( )
* Puede devolver un valor -> return
* O puede no devolver nada

REGLAS BÁSICAS

- *Regla 1.* Las instrucciones en JavaScript terminan en un punto y coma. Ejemplo:

```
var s = "hola";
```

- *Regla 2.* Uso de decimales en JavaScript. Los números en JavaScript que tengan decimales utilizarán el punto como separador de las unidades con la parte decimal. Ejemplos de números:

```
var x = 4;
var pi = 3.14;
```

- *Regla 3.* Los literales se pueden escribir entre comillas dobles o simples. Ejemplo:

```
var s1 = "hola";
var s2 = 'hola';
```

- *Regla 4.* Cuando sea necesario declarar una variable, se utilizará la palabra reservada *var.*
- *Regla 5.* El operador de asignación, al igual que en la mayoría de lenguajes, es el símbolo igual (=).
- *Regla 6.* Se pueden utilizar los siguientes operadores aritméticos: (+ – * /). Ejemplo:

```
var x = (5*4)/2+1;
```

- *Regla 7.* En las expresiones, también se pueden utilizar variables. Ejemplo:

```
var t = 4;
var x = (5*t)/2+1;
var y;
y = x * 2;
```

- *Regla 8.* Comentarios en JavaScript. Existen dos opciones para comentar el código:

  a) // cuando se desea comentar el resto de la línea a partir de estas dos barras invertidas.
  b) /* y */. todo lo contenido entre ambas etiquetas quedará comentado.

- *Regla 9.* Los identificadores en JavaScript comienzan por una letra o la barra baja (_) o el símbolo del dólar ($).
- *Regla 10.* JavaScript es sensible a las mayúsculas y minúsculas (case-sensitive). Ejemplo:

```
var nombre = "Julio";
var Nombre = "Ramón";
```

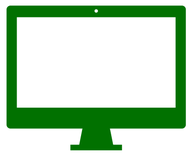
- *Nombre* y *nombre* son dos variables diferentes.
- Ten cuidado al escribir la palabra reservada *var*. Si escribes *Var* o *VAR*, tu código no funcionará.

ARRAYS

1. Creación de un array

0	1	2	3	4	5

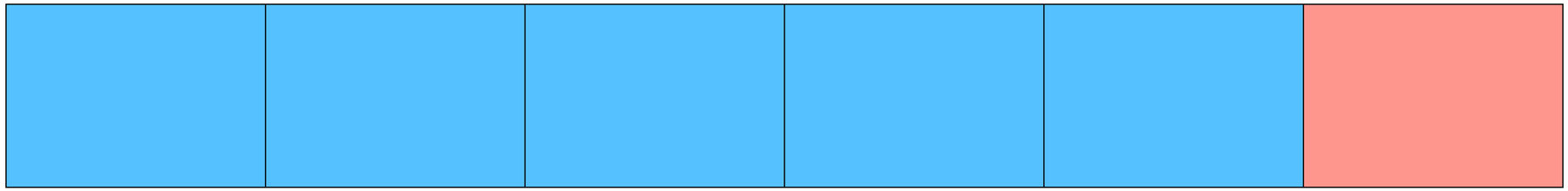
```
var array = new Array( );
var array=[ ];
```



```
var nombreArray = ["0", “1”,…];
```

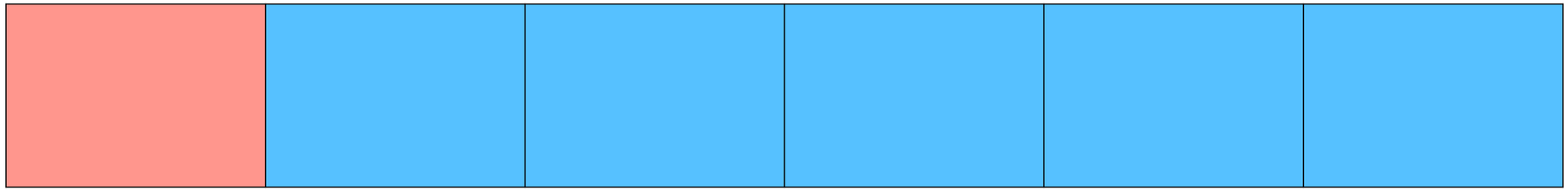
ARRAYS

2. Insertar elementos en el array



array.push(lo que sea);

3. Insertar elementos en el array al principio



array.unshift(lo que sea);

ARRAYS

0	1	2	3	4	5

4. Longitud de un array

```
array.length
```

5. Eliminar el último elemento de un array

```
array.pop( );
```

6. Eliminar el primer elemento de un array

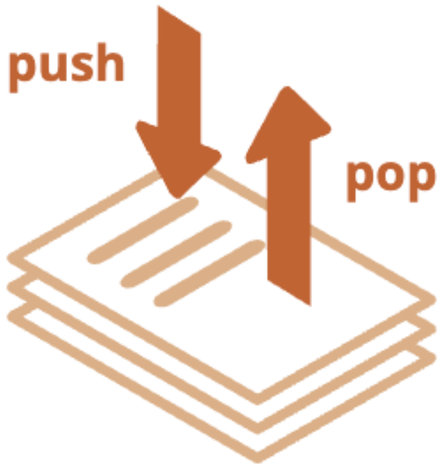
```
array.shift( );
```

ARRAYS

0	1	2	3	4	5

* push inserta un elemento al final. ~~Puede introducir varios elementos~~

* shift obtiene el elemento del principio, avanzando la cola, y así el segundo elemento se vuelve primero. unshift se utilizará para introducir un elemento en el principio. Puede introducir varios elementos



- push agrega un elemento al final.
- pop toma un elemento desde el final.

ARRAYS

0	1	2	3	4	5

7. Acceder a cada elemento del array

array[0,1, …]. -> ¿Cómo accedería al último? length-1, at(-1)

8. Acceso al array completo

array.foreach( );

```
frutas.forEach(function (elemento, indice, array) {
    alert(array);
});
```

ARRAYS

0	1	2	3	4	5

9. Saber el índice de un elemento en un array

array.indexOf(lo que buscamos)

¿Y si lo que buscamos no está?

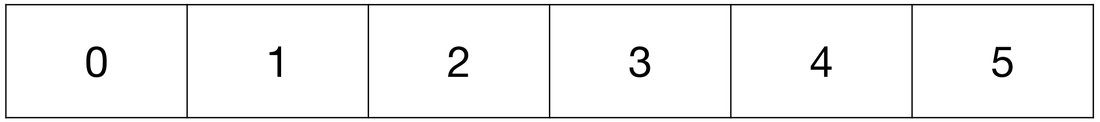
10. Eliminar uno o varios elementos desde una posición

array.splice(posición, numeroElementos)

11. Copias un array

array.splice( )

ARRAYS



12. Rellenar un array

array[posicion]=lo que quiera;

13. Rellenar un array dándole nombre a la posición

array.nombrePosicion= lo que quiera;

ARRAYS

0	1	2	3	4	5

12. Recorrer un array

forEach(function ())

```
numeros.forEach(function() {
    // código
});
```

ARRAYS

0	1	2	3	4	5

12. Recorrer un array. Ver cada elemento

forEach(function ())

```
array.forEach(function (elemento) {
  alert(elemento);
});
```

ARRAYS

0	1	2	3	4	5

12. Recorrer un array. Ver el elemento y la posición que ocupa

forEach(function ())

```
array.forEach(function (elemento, index) {
  alert(indice + ””+ elemento);
});
```

ARRAYS

0	1	2	3	4	5

12. Recorrer un array. Imprime el array

forEach(function ())

```
array.forEach(function (elemento, index, array) {
  alert(indice + ””+ elemento+ “ “+ array);
});
```

ARRAYS

0	1	2	3	4	5

12. Recorrer un array. Imprime el array

¿Qué ocurriría con el siguiente fragmento?

```
array.forEach(function (elemento, array) {
  alert(elemento+ “ “+ array);
});
```

SENTENCIAS CONDICIONALES

- La sentencia IF:

```
if (new Date().getHours() < 18) {
    saludo = "Buenos días";
}
```

El código anterior hace que la variable saludo valga "Buenos días" si es antes de las 18:00.

SENTENCIAS CONDICIONALES

- La sentencia ELSE:

```
if (new Date().getHours() < 18) {
    saludo = "Buenos días";
}else{
    saludo = "Buenas tardes";
}
```

El código anterior hace que la variable saludo valga "Buenos días" si es antes de las 18:00 y "Buenas tardes" en caso contrario.

SENTENCIAS CONDICIONALES

- La sentencia ELSE IF:

```
if (new Date().getHours() < 10) {
    saludo = "Al que madruga Dios le ayuda";
}else if (new Date().getHours() < 18) {
    saludo = "Buenos días";
}else {
    saludo = "Buenas tardes";
}
```

El código anterior hace que la variable saludo valga "Al que madruga Dios le ayuda" si la hora es antes de las 10 de la mañana, "Buenos días" si es antes de las 18:00 y "Buenas tardes" en caso contrario.

SENTENCIAS CONDICIONALES

- La sentencia SWITCH:

```
var = new Date().getDay();
switch (var) {
    case 0:
    dia = “Domingo”;
    break;
    case 1:
    dia = “Lunes”;
    break;
    case 2:
    dia = “Martes”;
    break;
    case 3:
    dia = “Miércoles”;
    break;
    case 4:
    dia = “Jueves”;
    break;
    case 5:
    dia = “Viernes”;
    break;
    case 6:
    dia = “Sábado”;
    default:
    dia = “ - “;
}
```

Como se puede ver, dependiendo del valor de la variable var, la variable dia tendrá un valor u otro en función del día de la semana en el que se esté.

BUCLES

En cualquier lenguaje de programación, se necesita ejecutar un trozo de código repetidas veces. Para ello, se usan los bucles y los hay de distintos tipos, los comunes a todos los lenguajes son el bucle for y while. A continuación, se hará un repaso de cada una de ellos:

a) *Bucle for.* Se utiliza esta sentencia cuando se quiere ejecutar un bloque de código una serie de veces determinada (generalmente, se conoce de antemano el número de veces).
b) *Bucle while.* Se emplea esta sentencia cuando se desea ejecutar un bloque de código una serie de veces sin conocer de antemano cuántas. El bloque de código se ejecutará mientras se cumpla una condición determinada.

BUCLES

- La sentencia o bucle FOR:

```
for (i = 0; i <= 10; i++) {
    text += "Número: " + i + " ";
}
```

En el bucle for, se inicializa la variable *i* a 0 (i=0). Dicho bucle se ejecutará mientras que la variable *i* sea menor o igual que 10 (i<=10) y la variable *i* a su vez se irá incrementando en una unidad (i++) en cada pasada del bucle.

BUCLES

Existe una variante del bucle for que es el bucle for in. A continuación, se muestra un ejemplo sencillo de comprender:

```
var persona = {nombre:"Dimas", apellidos:"Moreno", edad:25};
var texto = "";
var x;
for (x in persona) {
    texto += persona[x];
}
```

BUCLES

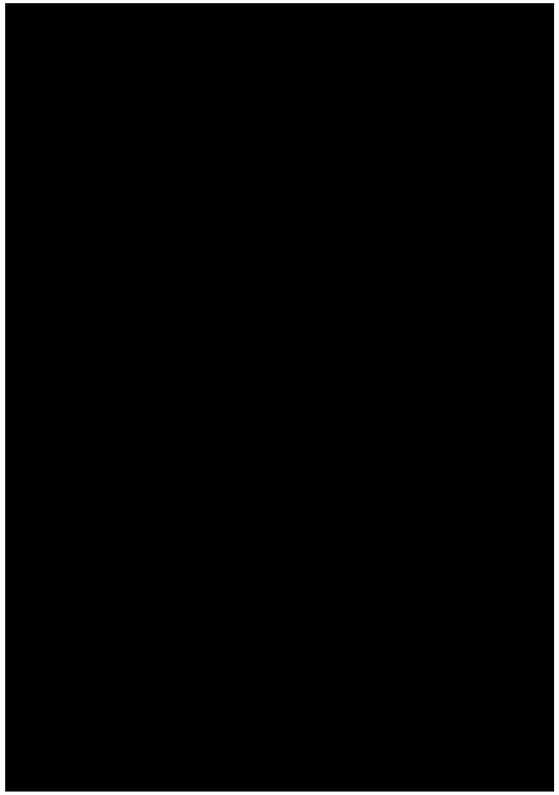
- La sentencia o bucle WHILE:

```
i = 0;
while (i <= 10) {
    text += "Número " + i;
    i++;
}
```

Como puede observarse, es el anterior ejemplo del bucle for, pero ahora reconvertido a while.

BÁSICO DOM

HTML



document

Elementos id

Elementos id

Elementos id

Elementos id

getElementoById(id)

document.getElementById("text")

<input type="text" id="text"></input>

Tengo que tener algo con id=text

BÁSICO DOM

Introducir

document

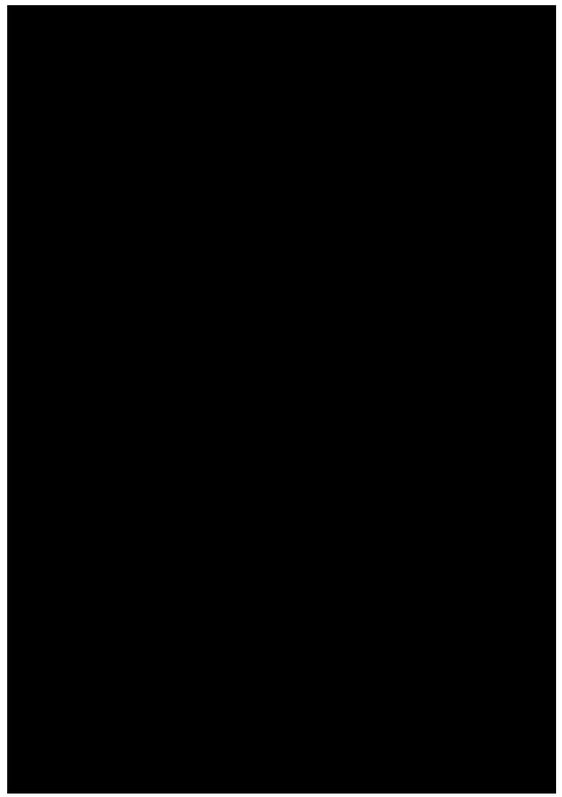
Elementos id

Elementos id

Elementos id

Elementos id

HTML



getElementoById(id)

<input type="text" id=“resultado"></input>

Tengo que tener algo con id=resultado

document.getElementById(“resultado”.innerHTML=lo que vaya a meter;)

BÁSICO DOM

HTML

document

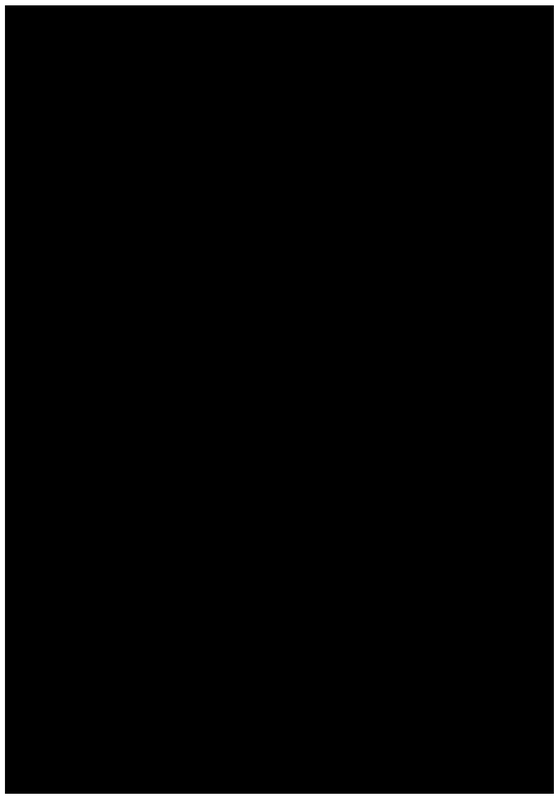
Elementos id

Elementos id

Elementos id

Elementos id

getElementoById(id)



<input type="text" id=“resultado"></input>

Tengo que tener algo con id=resultado

document.getElementById(“resultado”.innerHTML=lo que vaya a meter;)

OJO: Estamos sacando el elemento, o accediendo al elemento, pero si quiero meterle un valor o extraerle el valor, tengo que acceder a su propiedad value

BÁSICO DOM

OJO: Estamos sacando el elemento, o accediendo al elemento, pero si quiero meterle un valor o extraerle el valor, tengo que acceder a su propiedad value

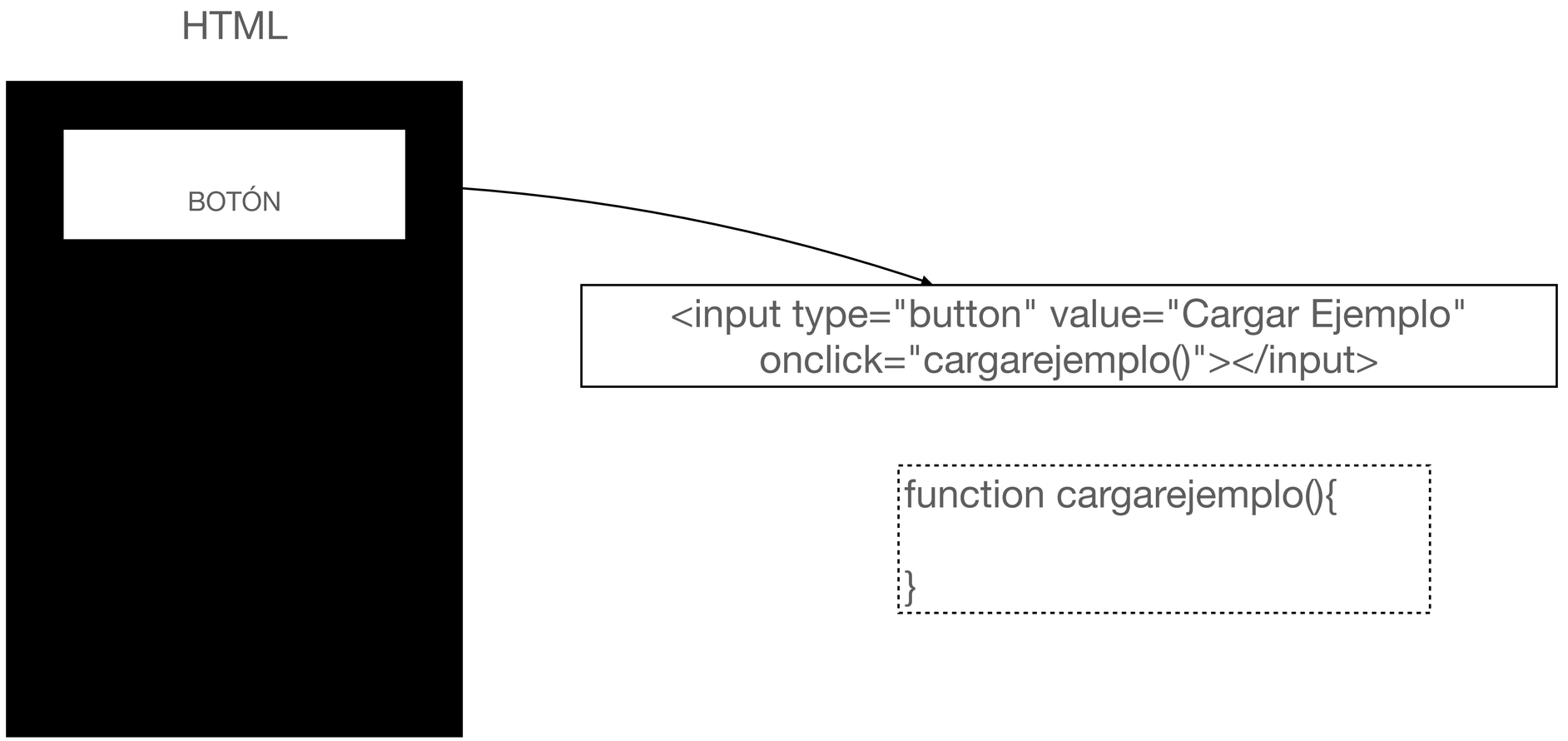
```
document.getElementById("text").value="5,99,43,12,37";
```

¿Cómo elimino las comas?

```
document.getElementById("text").value.split(/,/);
```

BÁSICO EVENTOS

Introducir



BÁSICO FOR

- La sentencia o bucle FOR:

```
for (i = 0; i <= 10; i++) {
    text += "Número: " + i + " ";
}
```

En el bucle for, se inicializa la variable *i* a 0 (i=0). Dicho bucle se ejecutará mientras que la variable *i* sea menor o igual que 10 (i<=10) y la variable *i* a su vez se irá incrementando en una unidad (i++) en cada pasada del bucle.

BÁSICO FOR-PARA ARRAY

```
for(i=0;i<array.length;i++){
    …;
}
```