



Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

## Tutorial – Introducción a JavaScript

### ***Introducción***

- Con HTML y CSS es posible crear páginas web estáticas.
  - HTML lo utilizamos para definir la estructura de la página web
  - CSS lo utilizamos para definir la presentación de la página web
- Con JavaScript, se pueden diseñar páginas web dinámicas de cliente.
- Para esto, dentro del código fuente de la página web se puede incluir el script que permitirá modificar la página web dinámicamente en el lado del cliente, sin sobrecargar de trabajo al servidor web.
  - La inclusión del script puede realizarse de 3 maneras: en línea, embebido o por archivos externos, siendo esta última alternativa la más flexible de todas.

### ***El primer script***

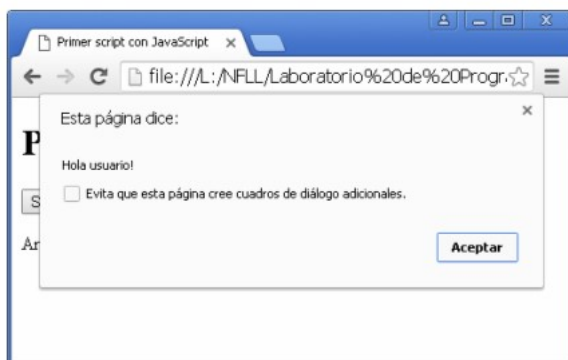
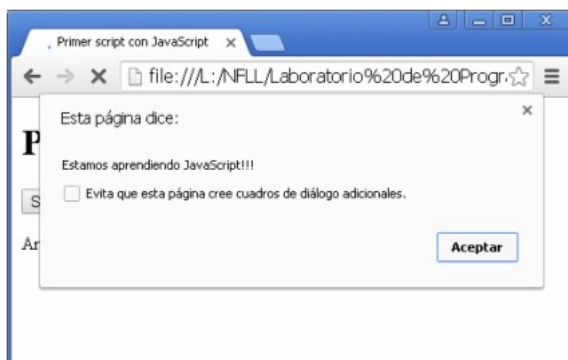
- El script puede incorporarse tanto dentro del cuerpo como del encabezado de la página web. Lo usual es definir las funciones del script en el encabezado de la página web (dentro de la etiqueta `<head>`).
- Para esto, HTML incluye la etiqueta `<script>...</script>`, la que tiene varios atributos. Algunos son:
  - `type`: indica tipo del script que se incluirá dentro de la etiqueta. Para JavaScript es: `"text/javascript"`.
  - `src`: indica el nombre del archivo `"js"` que contiene el código JavaScript. Este archivo puede incluir desde una lista de variables hasta una biblioteca completa de funciones. Si se define este atributo, entonces no se debe escribir código dentro de la etiqueta `<script>`.
- Dentro de los scripts, se puede utilizar el método `write` del objeto `document` (que representa a la página web actual) para imprimir texto dentro de la página.

Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

**Ejemplo 1: JavaScript incorporado en línea**

En este caso, no se utiliza la etiqueta <script>, sino que se declara un evento al cual reaccionará un elemento y la sentencia (o un pequeño bloque) que se ejecutará cuando se dispare el evento declarado:

```
1 <!-- Codigo_56.htm -->
2 <!DOCTYPE html>
3 <html lang="es">
4 <head>
5 <title>Primer script con JavaScript</title>
6 </head>
7 <body onLoad="alert('Estamos aprendiendo JavaScript!!!');">
8 <header>
9 <h1>Primer script con JavaScript</h1>
10 </header>
11 <section>
12 <article>
13 <button id="boton" type="button" onClick="alert('Hola usuario!');">
14   Saludame :)
15 </button>
16 </article>
17 </section>
18 <footer>
19 <p>Archivo: Codigo_56.htm</p>
20 </footer>
21 </body>
22 </html>
```



Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

### **Ejemplo 2: JavaScript incorporado embebido**

En este caso, se utiliza la etiqueta `<script>`, con su correspondiente etiqueta de cierre. Luego, dentro de ellas, se declara el bloque de sentencias que se ejecutará cuando se interprete la página en el navegador:

```
1 <!-- Codigo_55.htm -->
2 <!DOCTYPE html>
3 <html lang="es">
4 <head>
5 <title>Primer script con JavaScript</title>
6 </head>
7 <body>
8 <header>
9 <h1>Primer script con JavaScript</h1>
10 </header>
11 <section>
12 <article>
13 <script type="text/javascript">
14 document.write("<h1>Estamos aprendiendo JavaScript!!!</h1>");
15 </script>
16 </article>
17 </section>
18 <footer>
19 <p>Archivo: Codigo_55.htm</p>
20 </footer>
21 </body>
```



### **Ejemplo 3: JavaScript incorporado embebido**

En este ejemplo, el script JavaScript se incluye en el head de la página, y dentro de él, se define una función que luego puede ser invocada mediante la declaración de eventos en los elementos HTML:

```
1 <!-- Codigo_107.htm -->
2 <!DOCTYPE html>
3 <html lang="es">
4 <head>
5 <title>Primer script con JavaScript</title>
6 <script type="text/javascript">
7 function imprimoCuerpo()
8 {
9     document.write("<header>");
10    document.write("<h1>Primer script con JavaScript</h1>");
11    document.write("</header>");
12    document.write("<section>");
13    document.write("<article>");
14    document.write("<h2>Este texto es un artículo</h2>");
15    document.write("</article>");
16    document.write("</section>");
17    document.write("<footer>");
18    document.write("<p>Archivo: Codigo_107.htm</p>");
19    document.write("</footer>");
20 }
21 </script>
22 </head>
23 <body onload="imprimoCuerpo();">
24 </body>
```



Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

### Ejemplo 4: JavaScript con archivos externos

Finalmente, la manera más flexible y eficiente es separar el código JavaScript del código HTML. Esto se puede lograr a través de un archivo externo, con extensión “.js”, donde se definen variables, funciones, etc. El archivo js queda vinculado a la página web a través del atributo src de la etiqueta <script>:

```
1 <!-- Codigo_57.htm -->
2 <!DOCTYPE html>
3 <html lang="es">
4 <head>
5 <title>Primer script con JavaScript</title>
6 <script type="text/javascript" src="Codigo_57.js">
7 </script>
8 </head>
9 <body onLoad="imprimoCuerpo();">
10 </body>
11 </html>
```

```
1 <!-- Codigo_57.js -->
2 function imprimoCuerpo()
3 {
4     document.write("<header>");
5     document.write("<h1>Primer script con JavaScript</h1>");
6     document.write("</header>");
7     document.write("<section>");
8     document.write("<article>");
9     document.write("<h2>Este texto es un artículo</h2>");
10    document.write("<p>El texto es impreso desde afuera</p>");
11    document.write("</article>");
12    document.write("</section>");
13    document.write("<footer>");
14    document.write("<p>Archivo: Codigo_57.htm</p>");
15    document.write("</footer>");
16 }
```



### Variables

- En JavaScript, las variables se declaran usando la palabra reservada “var”, seguida por su identificador y, en lo posible, en el head del documento web o en el archivo js.
- El identificador es una secuencia de letras, números y/o guiones bajos, con las siguientes características:
  - Debe comenzar con una letra o con “\_”.
  - No debe coincidir con alguna de las palabras reservadas del lenguaje.
- Además, son case sensitive (respetan minúsculas y mayúsculas).
- No se declaran de un tipo de datos en particular y son dinámicas (es decir que pueden cambiar su tipo durante la ejecución del script).
- Pueden tener el valor NULL.



Universidad Nacional de la Patagonia San Juan Bosco  
Facultad de Ingeniería  
LABORATORIO DE PROGRAMACION Y LENGUAJES

Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

### ***Tipos de datos***

- Los tipos de datos incluidos en JavaScript son:

Tipo	Descripción
Number	Almacena cualquier valor entero o real. El separador decimal es el "."
Boolean	Puede tener los valores <i>true</i> o <i>false</i> .
String	Es una secuencia de caracteres alfanuméricos encerrados entre comillas dobles.
Function	Son variables que hacen referencia a funciones propias del lenguaje o a las definidas por el usuario.
Object	Son variables que almacenan objetos, ya sean los propios del lenguaje o los definidos por el usuario.

### ***Operadores***

- El operador de asignación se denota con el símbolo "=".
- Los operadores aritméticos son:

Operador	Operación
+	Suma de números
-	Resta de números
*	Multiplicación de números
/	División de números
%	Módulo o resto de la división entera de dos números

- El operador "+" también se utiliza para concatenar (es un operador sobrecargado).
- El operador de asignación se puede combinar con los operadores aritméticos/concatenación, a fin de simplificar la sintaxis de éstas operaciones:

Sentencia	Sintaxis reducida	Ejemplo
<var> = <var> + <valor>	<var> += <valor>	a = a + 5 se puede escribir a += 5
<var> = <var> + <cadena>	<var> += <cadena>	b = b + "mundo" se puede escribir b += "mundo"

- Los operadores de incremento y decremento son:

Operador	Operación	Ejemplo
++<var>	Incrementa en 1 y luego retorna el valor de <var>	++a
<var>++	Retorna el valor de <var> y luego lo incrementa en 1	a++
--<var>	Decrementa en 1 y luego retorna el valor de <var>	--a
<var>--	Retorna el valor de <var> y luego lo decrementa en 1	a--



Universidad Nacional de la Patagonia San Juan Bosco  
Facultad de Ingeniería  
LABORATORIO DE PROGRAMACION Y LENGUAJES

Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

- Los operadores relacionales son:

Operador	Operación
==	Devuelve True si el valor de una variable es igual al de otra.
===	Devuelve True si una variable es idéntica al de otra (igual valor y tipo de datos)
!=	Devuelve True si el valor de una variable es distinto al de otra.
!==	Devuelve True si el valor de una variable no es idéntico al de la otra (igual valor y tipo).
< / <=	Devuelve True si el valor de la primer variable es menor / menor o igual al de la segunda
> / >=	Devuelve True si el valor de la primer variable es mayor / mayor o igual al de la segunda

- Los operadores lógicos incorporados son:

Operador	Operación
&&	Devuelve True si todos los operandos son verdaderos.
	Devuelve True si alguno de los operandos es verdadero.
!	Operador de negación. Devuelve True si el operando es falso y viceversa.

- El operador condicional es una forma resumida de la sentencia condicional dicotómica. Se forma con una condición, la acción a realizar si la condición es verdadera y la acción a realizar si es falsa. La sintaxis es:

$$\langle \text{var} \rangle = (\text{condicion}) ? \text{accion\_true} : \text{accion\_false}$$

- El operador *typeof* se utiliza para determinar el tipo de datos actual de una variable. Devuelve una cadena con el tipo actual.

### **Ejemplo 5: uso de variables y operadores**

En este ejemplo, se muestra como se pueden declarar e inicializar variables, y luego su uso con operadores. Para las variables "a" y "b", sus valores se generan aleatoriamente a partir de los métodos random y round de la clase Math. El valor devuelto por el método random es multiplicado por la expresión (maximo - minimo) y, al resultado obtenido, se le suma el valor mínimo esperado. Concretamente, entonces, la forma de obtener un valor aleatorio que se encuentre dentro del rango [A,B] es:  $\text{Math.round}(\text{Math.random()} * (B - A) + A)$



Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

```
1 <!-- Codigo_58a.js -->
2 function calculos()
3 {
4 <!-- Declaración de variables -->
5 var a, b, c = 0, d = 0.0, e = "0", f = false, g, comparacion, resu;
6 var h = new Array();
7 <!-- Generación de valores enteros aleatorios para las variables a y b -->
8 a = Math.round(Math.random() * (20 - 1) + 1);
9 b = Math.round(Math.random() * (10 - 1) + 1);
10 document.write("<section>");
11 document.write("<article>");
12 <!-- Operadores aritméticos -->
13 document.write("a = " + a + "<br>");
14 document.write("b = " + b + "<br>");
15 resu = a+b;
16 document.write("a + b = " + resu + "<br>");
17 resu = a-b;
18 document.write("a - b = " + resu + "<br>");
19 resu = a*b;
20 document.write("a * b = " + resu + "<br>");
21 resu = a/b;
22 document.write("a / b = " + resu + "<br>");
23 resu = a%b;
24 document.write("resto(a / b) = " + resu + "<br>");
25 <!-- Comparaciones -->
26 (c==d)? document.write("c y d son iguales<br>") : document.write("c y d no son iguales<br>");
27 comparacion = (c==e)? "c y e son iguales<br>" : "c y e no son iguales<br>";
28 document.write(comparacion);
29 comparacion = (c==f)? "c y f son iguales<br>" : "c y f no son iguales<br>";
30 document.write(comparacion);
31 comparacion = (c===e)? "c y e son identicas<br>" : "c y e no son identicas<br>";
32 document.write(comparacion);
33 <!-- Tipo de datos de una variable -->
34 document.write("Tipo de datos de a: " + typeof(a) + "<br>");
35 document.write("Tipo de datos de e: " + typeof(e) + "<br>");
36 document.write("Tipo de datos de f: " + typeof(f) + "<br>");
37 document.write("Tipo de datos de g: " + typeof(g) + "<br>");
38 document.write("Tipo de datos de h: " + typeof(h) + "<br>");
39 document.write("</article>");
40 document.write("</section>");
41 }
```

```
1 <!-- Codigo_58.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta charset="utf-8" />
6 <title>Variables en JavaScript</title>
7 <script type="text/javascript" src="Codigo_58a.js">
8 </script>
9 </head>
10 <body onLoad="calculos();">
11 </body>
12 </html>
```

Archivo Editar Ver Historial Marcadores Herramientas Ayuda

Variables en JavaScript

file:///

a = 12  
b = 8  
a + b = 20  
a - b = 4  
a \* b = 96  
a / b = 1.5  
resto(a / b) = 4  
c y d son iguales  
c y e son iguales  
c y f son iguales  
c y e no son identicas  
Tipo de datos de a: number  
Tipo de datos de e: string  
Tipo de datos de f: boolean  
Tipo de datos de g: undefined  
Tipo de datos de h: object

## Constantes

- JavaScript no incorpora constantes ni permite al usuario definir las propias.
- Sin embargo, se puede simular utilizando la declaración y asignación de variables en una misma sentencia.



Universidad Nacional de la Patagonia San Juan Bosco  
Facultad de Ingeniería  
LABORATORIO DE PROGRAMACION Y LENGUAJES

Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

- Esto puede traer problemas, porque para el interprete del lenguaje, no es una constante, por lo tanto se puede manipular como una variable.
- Esta simulación se puede lograr nombrando a las constantes con letras mayúsculas.

***Algunas funciones propias del lenguaje***

- JavaScript incorpora algunas funciones que son útiles para el manejo de variables y expresiones aritmético-lógicas:

Función	Descripción
Eval	Recibe como parámetro una cadena que representa una expresión, una sentencia o un bloque. JavaScript la/s ejecutará y, en el caso de las expresiones aritmético-lógica, devolverá el resultado a una variable.
parseInt	Es una función para realizar la conversión explícita de una cadena a un valor entero. Tiene un segundo parámetro para especificar la base.
parseFloat	Es una función utilizada para convertir explícitamente una cadena a un número real.
isNaN	Es una función para determinar si una variable <u>no</u> es numérica.

***Ejemplo 6: uso de funciones propias del lenguaje***

```
1  <!-- Codigo_59.htm -->
2  <!DOCTYPE html>
3  <html>
4  <head>
5  <meta charset="utf-8" />
6  <title>Funciones de JavaScript</title>
7  <script type="text/javascript" src="Codigo_59.js"></script>
8  </head>
9  <body onLoad="invocoFunciones();">
10 </body>
11 </html>
```



Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

```

1 <!-- Codigo_59.js -->
2 function invocoFunciones()
3 {
4 <!-- Declaración de variables -->
5 var a, b, c, resu;
6 var d = "25sd", e = "0.25asd";
7
8 a = Math.round(Math.random() * (20 - 5) + 5);
9 b = Math.round(Math.random() * (10 - 1) + 1);
10 c = Math.round(Math.random() * 5);
11 document.write("<section>");
12 document.write("<article>");
13 document.write("Valor de a: " + a + "<br>");
14 document.write("Valor de b: " + b + "<br>");
15 document.write("Valor de c: " + c + "<br>");
16 resu = eval("a + b");
17 eval("c++");
18 document.write("Suma de a y b: " + resu + "<br>");
19 document.write("Valor de c incrementado en 1: " + c + "<br>");
20 document.write("Valor entero de la variable d: " + parseInt(d) + "<br>");
21 document.write("Valor real de la variable e: " + parseFloat(e));
22 document.write("</article>");
23 document.write("</section>");
24 }
  
```

Archivos Editar Ver Historial Marcadores Herramientas Ayuda

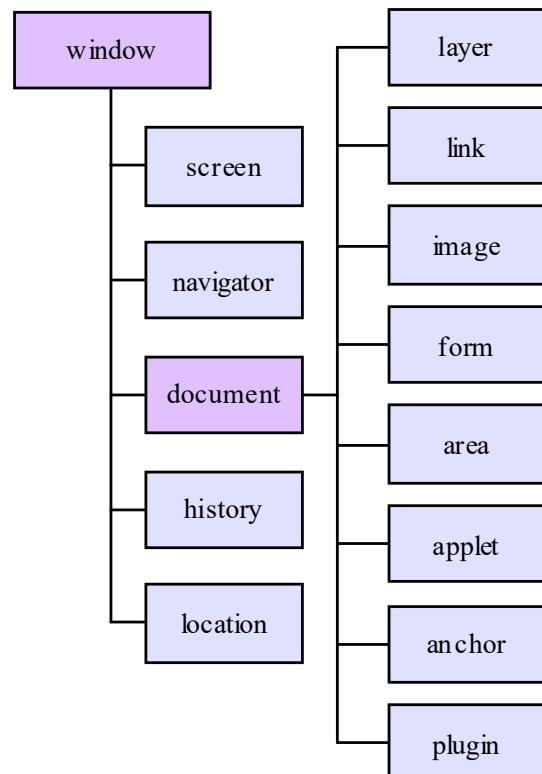
Funciones de JavaScript

file:///

Valor de a: 13  
Valor de b: 2  
Valor de c: 2  
Suma de a y b: 15  
Valor de c incrementado en 1: 3  
Valor entero de la variable d: 25  
Valor real de la variable e: 0.25

### Jerarquía de objetos en JavaScript

- JavaScript incorpora una jerarquía de objetos que representan a todos y cada uno de los elementos que se pueden incorporar en una página web o bien manejar en el browser.
- Dentro de esta jerarquía, se destacan los objetos:
  - Window: representa a la ventana del browser.
  - Document: contiene el documento web actual y todos los métodos asociados. Se genera cuando se encuentra la etiqueta <body> en el documento HTML.



Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

- Algunos métodos importantes del objeto Window son:

Método	Descripción
Alert	Se utiliza para informar al usuario sobre alguna situación. Muestra una ventana emergente (de dialogo) con un mensaje que es el parámetro del método.
Prompt	Se utiliza para todo tipo de ingreso de datos. Tiene como parámetro un mensaje que se muestra en una ventana de diálogo, junto a un cuadro de texto y a los botones "Aceptar" y "Cancelar". Opcionalmente, se puede agregar el valor por defecto en el cuadro de texto, como un segundo parámetro. Devuelve el valor ingresado por el usuario. Esto permite hacer la entrada de datos.
Confirm	Se utiliza para ingreso de datos booleanos o lógicos. Tiene como parámetro un mensaje que se mostrará en una ventana de diálogo junto a los botones "Aceptar" y "Cancelar". Devuelve true si el usuario pulsa "Aceptar" y false en caso contrario.

**Ejemplo 7: Uso de los métodos del objeto Window para ingreso de datos**

```

1 <!-- Codigo_60.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta charset="utf-8" />
6 <title>Objeto Window</title>
7 <script type="text/javascript" src="Codigo_60.js"></script>
8 </head>
9 <body onLoad="analizoNumero();">
10 </body>
11 </html>

```

```

1 // Codigo_60.js
2 function analizoNumero()
3 {
4     var valor, mensaje;
5
6     valor = prompt("Ingrese un valor numerico:");
7     mensaje = (isNaN(valor)) ? "El valor ingresado no es numerico" : "El valor ingresado es numerico";
8     alert(mensaje);
9 }

```



**Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente**



**Ejercicio a**

Programe una página web dinámica de cliente que permita al usuario ingresar 2 valores numéricos num1 y num2 y calcule la suma y el producto de ambos. Luego se deben mostrar los resultados.

**Estructuras de control**

- JavaScript incorpora bloques, sentencias de selección y sentencias iterativas.

**Bloques**

- JavaScript permite agrupar sentencias a fines de formar bloques. La sintaxis para definirlos es:

```
{
    Sentencia_1;
    Sentencia_2;
    .....
    Sentencia_n; }
```

**Estructuras de control selectivas**

- JavaScript incluye 2 variantes para las sentencias de selección simple y dicotómica:

Sintaxis 1 (selección simple)	Sintaxis 2 (selección dicotómica)
<pre>if (Expresion) {     Sentencias; }</pre>	<pre>if (Expresion) { Sentencias; } else { Sentencias; }</pre>

Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

- Además, incluye una sentencia de selección múltiple.
- La variable a evaluar puede ser de tipo number o string.
- La sentencia se comporta igual que en C++, por lo que se debe usar la sentencia break para salir de la opción.
- Incluye un default, que se ejecutará cuando el valor de la variable no se encuentre en la lista de opciones.

Sintaxis
<pre>switch( variable ) { case valor_1: sentencias;                break; case valor_2: sentencias;                break; ... case valor_N: sentencias;                break; default:      sentencias; }</pre>

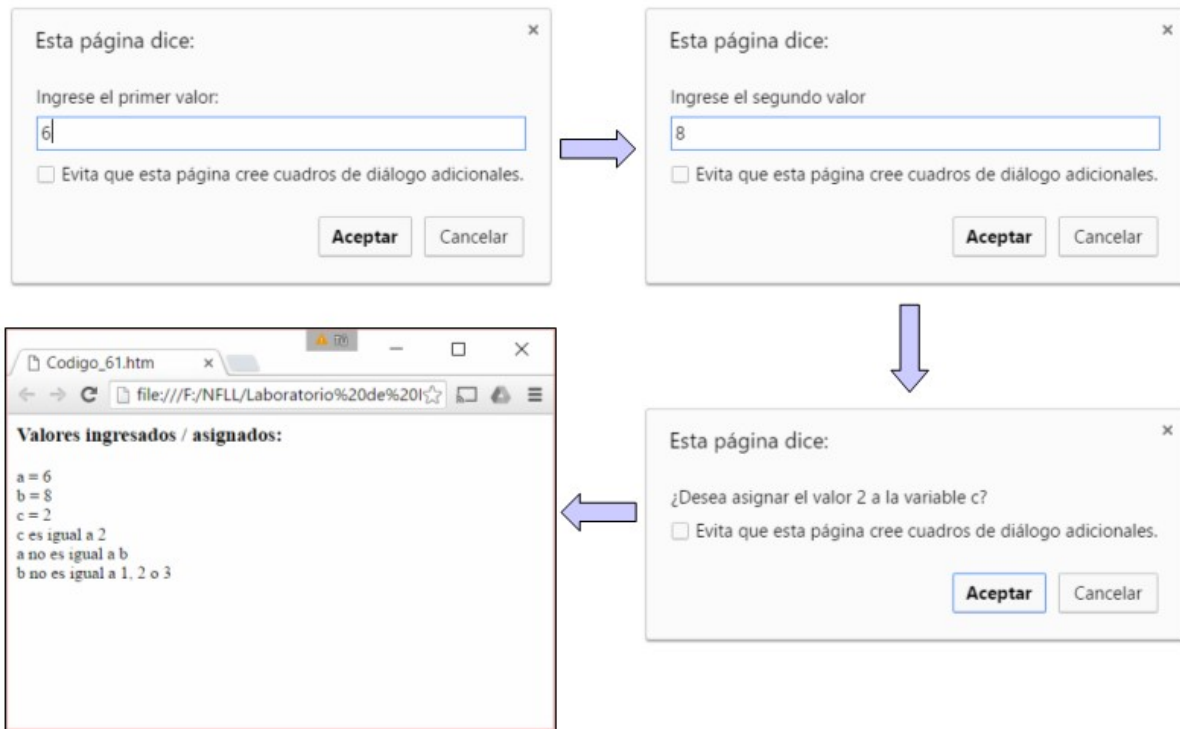
**Ejemplo 8: Implementación de las estructuras de control selectivas**

```
1  <!-- Codigo_61.js -->
2  function selecciones()
3  {
4  var a = prompt("Ingrese el primer valor: ");
5  var b = prompt("Ingrese el segundo valor", "0");
6  var c;
7
8  if (confirm("¿Desea asignar el valor 2 a la variable c?"))
9  {   c = 2;  }
10 else
11 {   c = a;  }
12 document.write("<section>");
13 document.write("<article>");
14 document.write("<h1>Valores ingresados / asignados:</h1>");
15 document.write("<p>");
16 document.write("a = " + a + "<br>");
17 document.write("b = " + b + "<br>");
18 document.write("c = " + c + "<br>");
19 <!-- If solo -->
20 if (c == 2)
21 {   document.write("c es igual a 2<br>");   }
22 <!-- If con then y else -->
23 if (a == b)
24 {   document.write("a es igual a b<br>");   }
25 else
26 {   document.write("a no es igual a b<br>");   }
27 <!-- switch -->
28 switch(b)
29 {
30     case 1: document.write("b es igual a 1<br>"); break;
31     case 2: document.write("b es igual a 2<br>"); break;
32     case 3: document.write("b es igual a 3<br>"); break;
33     default: document.write("b no es igual a 1, 2 o 3<br>"); break;
34 }
35 document.write("</p>");
36 document.write("</article>");
37 document.write("</section>");
38 }
```



Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

```
1 <!-- Codigo_61.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta charset="utf-8" />
6 <title>Estructuras de selecci&oacute;n</title>
7 <script type="text/javascript" src="Codigo_61.js"></script>
8 </head>
9 <body onload="selecciones();">
10 </body>
11 </html>
```



**Estructuras de control iterativas**

- JavaScript incluye 4 sentencias de iteración: while, do-while, for y for..in
- La sintaxis de las tres primeras es:

While	Do-While	For
<pre>while (condición) {     Sentencia_1;     Sentencia_2;     ....     Sentencia_N; }</pre>	<pre>do {     Sentencia_1;     Sentencia_2;     ....     Sentencia_N; } while (condición)</pre>	<pre>for (inicialización; condición; incremento) {     Sentencia_1;     Sentencia_2;     ....     Sentencia_N; }</pre>

Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

- La evaluación de la condición se realiza:
  - While y For: antes de la ejecución de cada ciclo. Puede no ejecutarse nunca.
  - Do-while: Luego de la ejecución de cada ciclo. Entonces, se ejecuta al menos una vez
- La sentencia For tiene las siguientes partes:
  - Inicialización: asignación que establece el valor inicial de la variable de control.
  - Condición: expresión que comprueba la variable de control.
  - Incremento: define como cambia el valor de la variable de control.
- La sentencia For..in se utiliza para recorrer objetos o arreglos.

**Ejemplo 9: Implementación de las estructuras de control iterativas while, do-while y for**

```
1 <!-- Codigo_64.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta charset="utf-8" />
6 <title>Ejemplo - Sentencia WHILE</title>
7 <script type="text/javascript">
8 function sentenciaWhile()
9 {
10 var i = 6;
11 while (i != 0)
12 { document.write(i--);
13   document.write("<br>");
14 }
15 }
16 </script>
17 </head>
18 <body onLoad="sentenciaWhile();">
19 </body>
20 </html>
```

```
1 <!-- Codigo_63.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta charset="utf-8" />
6 <title>Ejemplo - Sentencia DO - WHILE</title>
7 <script type="text/javascript">
8 function sentenciaDoWhile()
9 {
10 var i = 1, j = 10, resu;
11 do
12 { resu = i + j;
13   document.write(resu + "<br>");
14   j--;
15   i++;
16 } while (i<=10);
17 }
18 </script>
19 </head>
20 <body onLoad="sentenciaDoWhile();">
21 </body>
22 </html>
```

```
1 <!-- Codigo_62.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta charset="utf-8" />
6 <title>Ejemplo - Sentencia FOR</title>
7 <script type="text/javascript">
8 function sentenciaFor()
9 {
10 var i, valor;
11
12 for (i=0;i<6;i++)
13 { valor = i + 5;
14   document.write("Para " + i + ", el resultado es: " + valor + "<br>"); }
15 }
16 </script>
17 </head>
18
19 <body onLoad="sentenciaFor();">
20 </body>
21 </html>
```





Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

**Ejercicio b**

Modifique la página del ejercicio 1 de manera tal que, si num1 es mayor a num2, entonces calcule e informe  $\text{num1}^{\text{num2}}$ . Si num1 es menor o igual a num2, entonces debe calcular, utilizando el método de restas sucesivas, la división de num2 por num1.

**El objeto Array**

- El objeto *Array* es un conjunto ordenado de elementos, que pueden ser del mismo tipo o no.
- Para crearlo, se utiliza el operador *new*, con o sin parámetros. Si se incluyen parámetros, serán los elementos del arreglo.
- El índice para acceder a los elementos es numérico, y el primer elemento se asocia con el índice 0.
- Tiene el atributo *length*, que indica la cantidad de elementos en el arreglo.
- Algunos métodos que contiene son:

Método	Descripción
Concat	Anexa el arreglo que recibe como parámetro al arreglo que lo invoca
Join	Forma una cadena con los elementos del arreglo
Pop	Elimina el último elemento y lo devuelve
Shift	Elimina y devuelve el primer elemento
Push	Agrega a un arreglo la lista de elementos que se pasan como parámetros
Unshift	Agrega al comienzo de un arreglo la lista de elementos que se pasan como parámetros
Reverse	Cambia el orden de los elementos
Slice	Devuelve un arreglo con los elementos ubicados entre las 2 posiciones pasadas como parámetros
Sort	Ordena los elementos del arreglo de manera ascendente, comparando cada elemento como si fuera una cadena

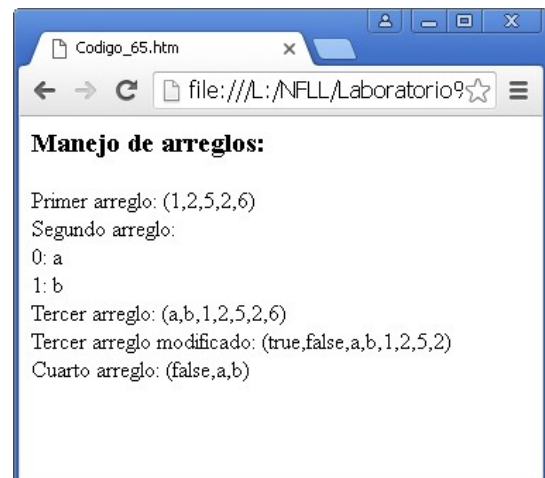
**Ejemplo 10: Creación y manejo de arreglos**

En este ejemplo, se crean arreglos y se recorren con la sentencia *for...in*, que es la más apropiada para ello:

Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

```
1 <!-- Codigo_65.js -->
2 function arreglos()
3 {
4
5   var Arreglo1 = new Array(1,2,5,2,6);
6   var Arreglo2 = new Array("a","b");
7   var indice, Arreglo3, Arreglo4;
8
9   document.write("<section>");
10  document.write("<article>");
11  document.write("<h1>Manejo de arreglos:</h1>");
12  document.write("<p>");
13  document.write("Primer arreglo: (" + Arreglo1 + ")<br>");
14  document.write("Segundo arreglo: <br>");
15  for (indice in Arreglo2)
16  {   document.write(indice + ": " + Arreglo2[indice] + "<br>"); }
17
18  Arreglo3 = Arreglo2.concat(Arreglo1);
19  document.write("Tercer arreglo: (" + Arreglo3 + ")<br>");
20
21  Arreglo3.pop();
22  Arreglo3.unshift(true,false);
23  document.write("Tercer arreglo modificado: (" + Arreglo3 + ")<br>");
24
25  Arreglo4 = Arreglo3.slice(1,4);
26  document.write("Cuarto arreglo: (" + Arreglo4 + ")<br>");
27
28  document.write("</p>");
29  document.write("</article>");
30  document.write("</section>");
31 }
```

```
1 <!-- Codigo_65.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5   <meta charset="utf-8" />
6   <title>Arreglos</title>
7   <script type="text/javascript" src="Codigo_65.js">
8   </script>
9 </head>
10 <body onLoad="arreglos();">
11 </body>
12 </html>
```



### El objeto String

- Este objeto permite crear y manipular cadenas de caracteres.
- Tiene asociado el atributo *length* que contiene la cantidad de caracteres.
- Un objeto *string* es un arreglo, por lo tanto el primer carácter se encuentra en la posición 0.
- Algunos métodos del objeto *string* son:



Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

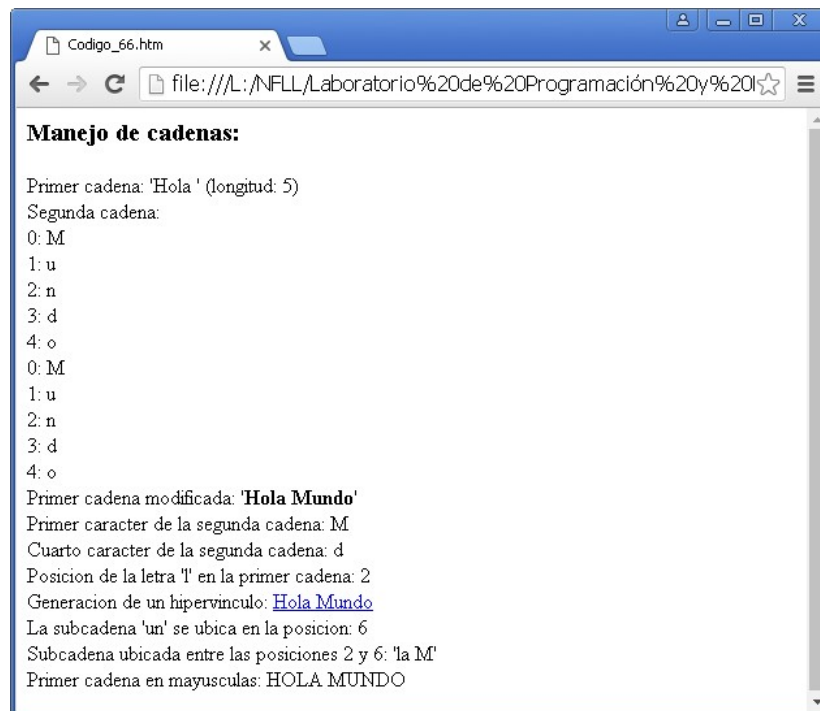
Método	Descripción
Bold	Devuelve el string aplicándole el estilo negrita
Italics	Devuelve el string aplicándole el estilo cursiva
Concat	Concatena la cadena actual con la que recibe como parámetro
charAt	Devuelve el carácter ubicado en la posición pasada como parámetro
indexOf	Devuelve la posición de la subcadena que se pasa como parámetro. La búsqueda se realiza a partir de la posición indicada como segundo parámetro
lastIndexOf	Igual a indexOf, pero la búsqueda se realiza desde el final y hacia el principio
Link	Devuelve un hipervínculo a partir de la URL que recibe como parámetro
Replace	Devuelve un string en el que han sustituido uno o varios caracteres por otros, pasados todos como parámetros
Search	Devuelve la posición donde se encuentra la primera ocurrencia de una subcadena (parámetro)
Substring	Devuelve la subcadena ubicada entre 2 posiciones pasadas como parámetros
Substr	Devuelve la subcadena formada por los caracteres que se encuentran desde la posición pasada como parámetro hasta una longitud -1
toLowerCase	Devuelve la cadena con todos los caracteres en minúscula
toUpperCase	Devuelve la cadena con todos los caracteres en mayúsculas
Split	Divide la cadena en subcadenas, cuyo separador es un carácter pasado como parámetro

**Ejemplo 11: Creación y manejo de objetos string**

```
1 <!-- Codigo_66.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta charset="utf-8" />
6 <title>Cadenas</title>
7 <script type="text/javascript" src="Codigo_66.js">
8 </script>
9 </head>
10 <body onLoad="cadenas();" >
11 </body>
12 </html>
```

Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

```
1 <!-- Codigo_66.js -->
2 function cadenas()
3 {
4   var Cadena1 = new String("Hola ");
5   var Cadena2 = "Mundo";
6   var indice, ind;
7   document.write("<section>");
8   document.write("<article>");
9   document.write("<h1>Manejo de cadenas:</h1>");
10  document.write("<p>");
11  <!-- Impresion de la primer cadena -->
12  document.write("Primer cadena: ' " + Cadena1 + "' (longitud: " + Cadena1.length + ")<br>");
13  <!-- Impresion de la segunda cadena -->
14  document.write("Segunda cadena: <br>");
15  for (ind=0; ind<Cadena2.length; ind++)
16  {   document.write(ind + ": " + Cadena2.charAt(ind) + "<br>");   }
17  for (indice in Cadena2)
18  {   document.write(indice + ": " + Cadena2[indice] + "<br>");   }
19  <!-- Concatenacion de cadenas -->
20  Cadena1 = Cadena1.concat(Cadena2);
21  document.write("Primer cadena modificada: ' " + Cadena1.bold() + "'<br>");
22  <!-- Primer y cuarto caracter de la cadena2 -->
23  document.write("Primer caracter de la segunda cadena: " + Cadena2.charAt(0) + "<br>");
24  document.write("Cuarto caracter de la segunda cadena: " + Cadena2.charAt(3) + "<br>");
25  <!-- Posicin del caracter "l" en la cadena1 -->
26  document.write("Posicion de la letra 'l' en la primer cadena: " + Cadena1.indexOf("l") + "<br>");
27  <!-- Generacion de un hipervinculo -->
28  document.write("Generacion de un hipervinculo: " + Cadena1.link("http://www.ing.unp.edu.ar") + "<br>");
29  <!-- Busqueda de una subcadena -->
30  document.write("La subcadena 'un' se ubica en la posicion: " + Cadena1.search("un") + "<br>");
31  <!-- Subcadena -->
32  document.write("Subcadena ubicada entre las posiciones 2 y 6: ' " + Cadena1.substring(2,6) + "'<br>");
33  <!-- Mayusculas -->
34  Cadena1 = Cadena1.toUpperCase();
35  document.write("Primer cadena en mayusculas: " + Cadena1);
36  document.write("</p>");
37  document.write("</article>");
38  document.write("</section>");
39 }
```



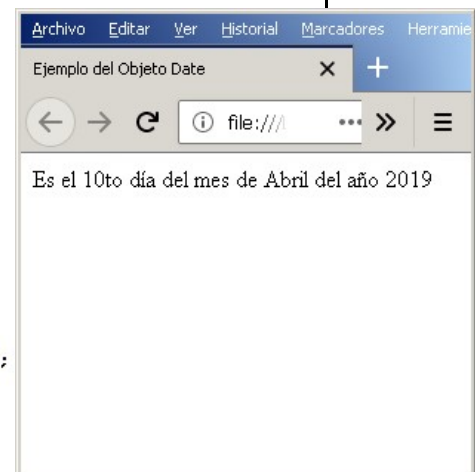
Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

### ***El objeto Date***

- Es un objeto que contiene información sobre la fecha y hora actual en el reloj del cliente.
- También puede crearse con parámetros particulares, como el año, el mes, el día, o un literal que represente una fecha.
- No tiene atributos y algunos métodos que contiene son:
  - getDate, getDay, getHours, getMinutes, getMonth, getSeconds, getTime, getYear, getFullYear.
  - setDate, setHours, setMinutes, setMonth, setSeconds, setTime, setYear.

### ***Ejemplo 12: Uso del objeto Date***

```
1 <!-- Codigo_120.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <title>Ejemplo del Objeto Date</title> </head>
6 <body>
7 <section>
8 <article>
9 <script language=JavaScript>
10 var meses = new Array("Enero", "Febrero", "Marzo", "Abril", "Mayo", "Junio", "Julio",
11                        "Agosto", "Septiembre", "Octubre", "Noviembre", "Diciembre");
12 var dateAct = new Date();
13 var yearAct = dateAct.getFullYear();
14 var monthAct = meses[dateAct.getMonth()];
15 var dayAct = dateAct.getDate();
16 var daySuffix;
17 switch (dayAct){
18 case 1: case 21: case 31:   daySuffix = "er";
19                             break;
20 case 2: case 22:           daySuffix = "do";
21                             break;
22 case 3: case 23:           daySuffix = "er";
23                             break;
24 default:                   daySuffix = "to";
25 }
26 document.write("Es el " + dayAct + daySuffix + " día ");
27 document.write("del mes de " + monthAct);
28 document.write(" del año " + yearAct);
29 </script>
30 </article>
31 </section>
32 </body>
33 </html>
```





Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

## ***Funciones en JavaScript***

### ***Definición de funciones***

- JavaScript incorpora solamente funciones, las que pueden o no tener parámetros y devolver o no un único valor.
- Se pueden definir dentro de la página web en el encabezado, o en un archivo “.js” externo.
- La sintaxis para definir las es:

```
function nombre([parametro1, parametro2, ..., parametroN])
{
    Sentencia_1;
    ...
    Sentencia_N;
    [return valor_a_retornar;]
}
```

### ***Invocación de las funciones***

- Las funciones definidas en el encabezado, estarán disponibles cuando se cargue la página web.
- Dependiendo si la función devuelve o no un valor, deberá invocarse de manera directa o indirecta:
  - Si devuelve un valor, entonces se invocará de manera indirecta.
  - Si no devuelve ningún valor, entonces se invocará de manera directa.

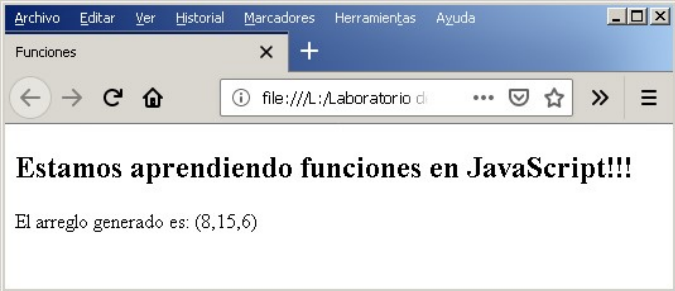
### ***Ejemplo 13: Definición e invocación de funciones en JavaScript***

```
1 <!-- Codigo_67.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta charset="utf-8" />
6 <title>Funciones</title>
7 <script type="text/javascript" src="Codigo_67.js">
8 </script>
9 <script>
10 window.onload = principal();
11 </script>
12 </head>
13 <body>
14 </body>
15 </html>
```



### Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

```
1 <!-- Codigo_67.js -->
2 var Arreglo;
3
4 function abroSeccion()
5 {
6     document.write("<section>");
7     document.write("<article>");
8     document.write("<h2>Estamos aprendiendo funciones en JavaScript!!!</h2>");
9     document.write("<p>");
10 }
11
12 function cierroSeccion()
13 {
14     document.write("</p>");
15     document.write("</article>");
16     document.write("</section>");
17 }
18
19 function CreoArreglo()
20 {
21     var aux = new Array();
22     aux[0] = Math.round(Math.random() * (10 - 1) + 1);
23     aux[1] = Math.round(Math.random() * (20 - 5) + 5);
24     aux[2] = Math.round(Math.random() * (15 - 2) + 2);
25     return aux;
26 }
27
28 function principal()
29 {
30     abroSeccion();
31     Arreglo = CreoArreglo();
32     document.write("El arreglo generado es: (" + Arreglo + ")");
33     cierroSeccion();
34 }
```



#### ***Pasaje de parámetros***

- JavaScript admite el pasaje de parámetros. Durante la invocación, si el parámetro actual es de tipo *number*, *string* o *boolean*, entonces cualquier modificación que se haga sobre él dentro de la función no se verá reflejada cuando la función finalice su ejecución. Por el contrario, si el parámetro actual es un arreglo o un objeto, entonces cualquier modificación que se realice sobre él en el cuerpo de la función permanecerá en la variable pasada como parámetro aún cuando la función haya finalizado su ejecución.
  - Tener precaución porque los tipos de parámetros no se distinguen de ninguna manera.
- Al igual que en las variables, los parámetros no se declaran de un tipo determinado, sino como parámetros “genéricos”.
- Importante: las funciones tienen asociadas un arreglo llamado *arguments*, que permite manejar una cantidad variable de parámetros. En este caso, la función se define sin parámetros. Luego, al momento de la invocación, ésta se realiza enviando los parámetros actuales que correspondan.

### Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

Finalmente, el acceso a los parámetros actuales dentro del cuerpo de la función se realiza utilizando el arreglo *arguments*.

#### Ejemplo 14: Modificación de parámetros

En los siguientes dos ejemplos se muestra como la modificación de los parámetros actuales se mantiene luego de la invocación de una función dependiendo del tipo de datos del parámetro actual:

```
1 <!-- Codigo_130.js -->
2 var Arreglo;
3 var numero;
4
5 function abroSeccion()
6 {
7     document.write("<section>");
8     document.write("<article>");
9     document.write("<h2>Parametros en funciones de JavaScript</h2>");
10    document.write("<p>");
11 }
12
13 function cierroSeccion()
14 {
15     document.write("</p>");
16     document.write("</article>");
17     document.write("</section>");
18 }
19
20 function CreoArreglo()
21 {
22     var aux = new Array();
23     aux[0] = Math.round(Math.random() * (10 - 1) + 1);
24     aux[1] = Math.round(Math.random() * (20 - 5) + 5);
25     aux[2] = Math.round(Math.random() * (15 - 2) + 2);
26     return aux;
27 }
28
29 function ModificoArreglo(vector)
30 {
31     vector[0] = vector[1] + vector[2];
32     vector[1] = vector[0] * 2;
33 }
34
35 function ModificoNumero(num)
36 {
37     num = num * 2;
38 }
39
40 function principal()
41 {
42     abroSeccion();
43     numero = 5;
44     document.write("La variable 'numero' tiene el valor: " + numero + "<br>");
45     ModificoNumero(numero);
46     document.write("Luego de la invocación, la variable 'numero' tiene el valor: " + numero + "<br>");
47     Arreglo = CreoArreglo();
48     document.write("El arreglo generado es: (" + Arreglo + ")<br>");
49     ModificoArreglo(Arreglo);
50     document.write("El arreglo modificado es: (" + Arreglo + ")");
51     cierroSeccion();
52 }
```

```
1 <!-- Codigo_130.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta charset="utf-8" />
6 <title>Parametros en Funciones</title>
7 <script type="text/javascript" src="Codigo_130.js">
8 </script>
9 <script>
10 window.onload = principal();
11 </script>
12 </head>
13 <body>
14 </body>
15 </html>
```

Archivo Editar Ver Historial Marcadores Herramientas Ayuda

Parametros en Funciones x +

file:///F:/Laboratorio de Programació

### Parametros en funciones de JavaScript

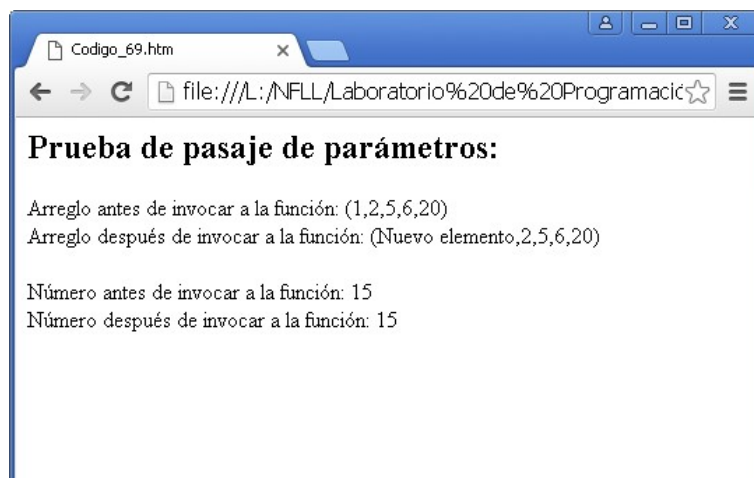
La variable 'numero' tiene el valor: 5  
Luego de la invocación, la variable 'numero' tiene el valor: 5  
El arreglo generado es: (9,13,8)  
El arreglo modificado es: (21,42,8)

Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

**Ejemplo 15: Modificación de parámetros**

```
1 <!-- Codigo_69.js -->
2 var Vector = new Array(1,2,5,6,20);
3 var Numero = 15;
4
5 function abroSeccion(titulo)
6 { document.write("<section>");
7   document.write("<article>");
8   document.write(titulo);
9   document.write("<p>"); }
10
11 function cierreSeccion()
12 { document.write("</p>");
13   document.write("</article>");
14   document.write("</section>"); }
15
16 function pruebaParametro1(arreglo)
17 { arreglo[0]="Nuevo elemento"; }
18
19 function pruebaParametro2(nro)
20 { nro++; }
21
22 function principal()
23 {
24   abroSeccion("<h2>Prueba de pasaje de parámetros:</h2>");
25   <!-- Primer pasaje: se pasa un objeto -->
26   document.write("Arreglo antes de invocar a la función: (" + Vector + ")<br>");
27   pruebaParametro1(Vector);
28   document.write("Arreglo después de invocar a la función: (" + Vector + ")<br><br>");
29   <!-- Segundo pasaje: se pasa un número -->
30   document.write("Número antes de invocar a la función: " + Numero + "<br>");
31   pruebaParametro2(Numero);
32   document.write("Número después de invocar a la función: " + Numero + "<br>");
33   cierreSeccion();
34 }
```

```
1 <!-- Codigo_69.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5   <meta charset="utf-8" />
6   <title>Pasaje de parámetros:</title>
7   <script src="Codigo_69.js">
8   </script>
9 </head>
10 <body onLoad="principal();">
11 </body>
12 </html>
```



**Ejemplo 16: Parámetros variables (uso de arguments)**

En este ejemplo, la función *Sumatoria* puede invocarse con diferentes cantidades de parámetros. Los actuales, se obtienen a través del arreglo *arguments*.

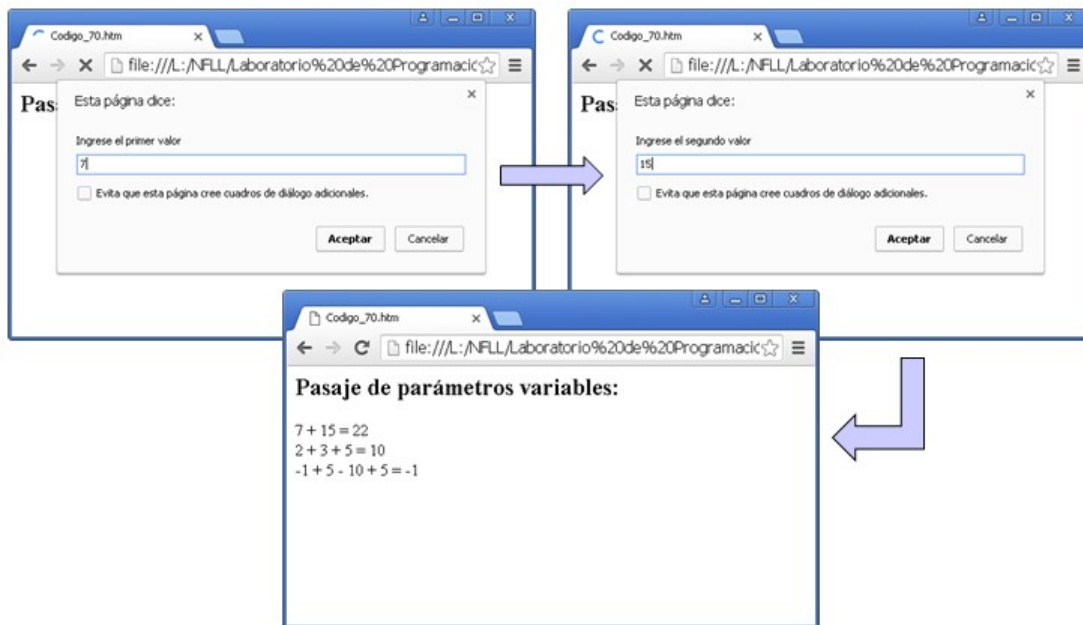
## LABORATORIO DE PROGRAMACION Y LENGUAJES

### Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

```
1 <!-- Codigo_70.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta charset="utf-8" />
6 <title>Pasaje de parametros variables</title>
7 <script src="Codigo_70.js"></script>
8 </head>
9 <body onLoad="principal();">
10 </body>
11 </html>
```

```
1 <!-- Codigo_70.js -->
2 var n1, n2;
3
4 function abroSeccion(titulo)
5 { document.write("<section>");
6   document.write("<article>");
7   document.write(titulo);
8   document.write("<p>"); }
9
10 function cierreSeccion()
11 { document.write("</p>");
12   document.write("</article>");
13   document.write("</section>"); }
14
15 function leoNumero(texto)
16 { numero = prompt(texto);
17   return numero; }
18
19 function Sumatoria()
20 { var i, cantiElem, resu;
21
22   resu = 0;
23   cantiElem = Sumatoria.arguments.length;
24   for(i=0;i<cantiElem;i++)
25   { resu = resu + Sumatoria.arguments[i]; }
26   return resu; }
27
28 function principal()
29 {
30   abroSeccion("<h2>Pasaje de par&aacute;metros variables:</h2>");
31   n1 = parseInt(leoNumero("Ingrese el primer valor"));
32   n2 = parseInt(leoNumero("Ingrese el segundo valor"));
33   document.write(n1 + " + " + n2 + " = " + Sumatoria(n1,n2) + "<br>");
34   document.write("2 + 3 + 5 = " + Sumatoria(2,3,5) + "<br>");
35   document.write("-1 + 5 - 10 + 5 = " + Sumatoria(-1,5,-10,5) + "<br>");
36   cierreSeccion();
37 }
```

Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente



**Alcance de las variables**

- Dentro de las funciones, es posible utilizar variables locales, cuyo ámbito será solamente la función.
- Al igual que las globales, las variables locales se definen utilizando la palabra reservada *var*.
- Se debe tener precaución con los identificadores a utilizar, ya que pueden coincidir con los de las variables globales y se generarán dos variables diferentes con el mismo nombre.
  - En este caso, la referencia a la variable será hacia la local, por lo cual no será posible acceder a la global.

**Ejemplo 17: Alcance de las variables globales y locales**

```

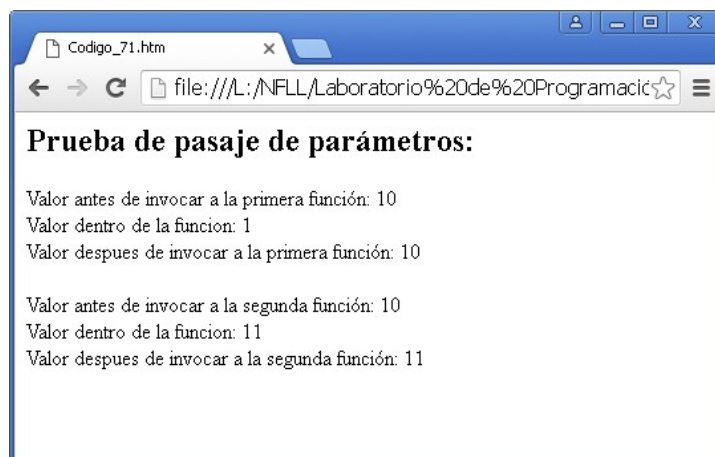
1 <!-- Codigo_71.htm -->
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta charset="utf-8" />
6 <title>Pasaje de parametros</title>
7 <script src="Codigo_71.js"></script>
8 </head>
9 <body onLoad="principal();">
10 </body>
11 </html>

```



### Trabajo Practico Nro. 3. Tecnología de desarrollo web del lado del cliente

```
1 <!-- Codigo_71.js -->
2 var nro = 10;
3
4 function prueba()
5 { <!-- Funcion con una variable local con identificador igual al de la variable global -->
6   var nro = 0;
7   nro++;
8   document.write("Valor dentro de la funcion: " + nro + "<br>"); }
9
10 function prueba2()
11 { <!-- Funcion que modifica la variable global -->
12   nro++;
13   document.write("Valor dentro de la funcion: " + nro + "<br>"); }
14
15 function abroSeccion(titulo)
16 { document.write("<section>");
17   document.write("<article>");
18   document.write(titulo);
19   document.write("<p>"); }
20
21 function cierroSeccion()
22 { document.write("</p>");
23   document.write("</article>");
24   document.write("</section>"); }
25
26 function principal()
27 {
28   abroSeccion("<h2>Prueba de pasaje de parámetros:</h2>");
29   document.write("Valor antes de invocar a la primera función: " + nro + "<br>");
30   prueba();
31   document.write("Valor despues de invocar a la primera función: " + nro + "<br><br>");
32   document.write("Valor antes de invocar a la segunda función: " + nro + "<br>");
33   prueba2();
34   document.write("Valor despues de invocar a la segunda función: " + nro);
35   cierroSeccion();
36 }
```



#### Ejercicio c

Modifique la página del ejercicio anterior programando funciones para cada uno de los cálculos realizados.

Además, incorpore en la página algún recurso para ver la fecha actual de la computadora del cliente.