Pérez Molinito Cristian José
Programación Web

Conceptos básicos sobre jQuery & JavaScript

JavaScript

JavaScript es un lenguaje de programación que añade interactividad a una página Web. Funciona en conjunto con HTML y CSS para crear sitios atractivos e interactivos. Algunos de los usos y funcionalidades que tiene pueden ser por ejemplo:

- Mostrar información basada en la hora del día.
- Identificar el navegador del usuario.
- Validar datos de formularios.
- Cambiar dinámicamente el contenido que se muestra en las páginas.
- Etc.

JavaScript es un lenguaje del lado del cliente, es decir, los códigos escritos en este lenguaje se ejecutan en la computadora del usuario una vez que es cargado el sitio web. Un gran beneficio que ofrece JavaScript es que no requiere de herramientas especiales para poder escribir código, en otras palabras, un simple editor de texto bastará.

El código de JavaScript se puede escribir dentro del archivo HTML, usando las etiquetas: **<script> … </script>** o bien escribiendo el código en un archivo con extensión **.js** independiente del archivo **.html**, de esta forma evitamos mezclar más de un lenguaje en un solo archivo, además de que se vuelve más fácil su uso cuando un mismo método se requiere en distintas partes del sitio Web, permitirá facilitar su mantenimiento y/o corrección e incluso puede hacer más fácil leer el código. Para poder hacer referencia a un archivo **.js** que contiene una función en específico, dentro del archivo **.html** se usará una etiqueta como la siguiente:

```
<script src="js/loteria.js"></script>
```

Funciones

JavaScript permite el uso de funciones, que pueden ser ejecutadas cuando son llamadas. Algunos eventos como un “click” pueden llamar a la ejecución de un método.

JavaScript Output

Pérez Molinito Cristian José
Programación Web

El código en JavaScript puede mostrar la información de distintas formas:

- Escribiendo un elemento HTML **innerHTML**.
- Mediante una ventana de alerta **window.alert()**.
- A través de la consola del navegador **console.log()**.

Mediante **innerHTML**:

```
<p id="demo"></p>
<script>
    document.getElementById("demo").innerHTML = 5+6;
</script>
```

Con este método JavaScript usa un elemento HTMl mediante su id y define su contenido.

Mediante **window.alert()**:

```
<script>
    window.alert(5+6);
</script>
```

Con este método, se muestra una caja de alerta o pop-up en el navegador, y muestra información. No se recomienda su uso.

Mediante **console.log()**:

```
<script>
    console.log(5+6);
</script>
```

La información se muestra en la consola del navegador, su utilidad principal está enfocada a la depuración de errores o **debugging**.

Sintaxis

En resumen, la sintaxis de un lenguaje es un conjunto de reglas que definen cómo se debe escribir. En el caso de los lenguajes de programación, definen cómo es que los programas están construidos.

Un programa es una lista de instrucciones que una máquina debe seguir, las instrucciones se conocen como “declaraciones” o en inglés **statements**.

Cada declaración está separada por un **;** (Semicolon).

Las declaraciones en JavaScript se componen de valores, operadores, expresiones, palabras clave o reservadas y comentarios.

Valores en JavaScript:

- Valores fijos: **literales**.
- Valores variables: **Variables**.

Los valores fijos pueden estar definidor por números (Decimales o enteros) y cadenas (“Hola”) por mencionar lo más básico.

```
document.getElementById("demo").innerHTML = 10.50;
```

Los variables pueden contener números y cadenas, por mencionar lo más básico, las variables se definen de la siguiente forma:

```
var x, y;
x = 5 + 6;
y = x * 10;
```

Donde: **var** es una palabra reservada que indica que los identificadores “x” y “y” serán **variables**.

Los comentarios son un componente importante en todo programa de computadora, nos permitirán revisar nuestros códigos en futuros próximos y saber qué es lo que hace. Su uso en la documentación nos ahorrará muchos dolores de cabeza.

Los comentarios son partes de código **ignoradas** por el intérprete y se pueden definir de la siguiente forma:

```
//Simple comment

/*
Complex comment
*/
```

Los identificadores en JavaScript nos permiten “identificar” elementos dentro de nuestro código, que van desde variables, funciones, palabras clave, etc.

`var x, y;` En este caso, “x” y “y” son identificadores para dos variables.

Pérez Molinito Cristian José
Programación Web

Es importante recordar que los identificadores en JavaScript son **case sensitive**, esto quiere decir que son sensibles a mayúsculas y minúsculas.

```
var hola, Hola;
```
En este caso, “hola” es distinto de “Hola”.

El código en JavaScript se puede agrupar en bloques de código:

```
function hola(){
    //Statement 1;
    //Statement 2;
    //Statement 3;
}
```

Está permitida la declaración de muchas variables en una sola línea de código:

```
var person = "Kriz", carName = "BMW", price = 200000;
```

JavaScript cuenta con los operadores aritméticos básicos:

+	Suma
-	Resta
*	Multiplicación
/	División
%	Módulo
++	Incremento
--	Decremento

Operadores de asignación:

Operator	Example	Same As
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y

Pérez Molinito Cristian José
Programación Web

Operadores de comparación:

Operator	Description
==	equal to
===	equal value and equal type
!=	not equal
!==	not equal value or not equal type
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
?	ternary operator

Operadores lógicos:

Operator	Description
&&	logical and
||	logical or
!	logical not

Operadores que devuelven datos específicos:

Operator	Description
typeof	Returns the type of a variable
instanceof	Returns true if an object is an instance of an object type

Pérez Molinito Cristian José
Programación Web

Operadores de nivel de bits:

Operator	Description	Example	Same as	Result	Decimal
&	AND	5 & 1	0101 & 0001	0001	1
|	OR	5 | 1	0101 | 0001	0101	5
~	NOT	~ 5	~0101	1010	10
^	XOR	5 ^ 1	0101 ^ 0001	0100	4
<<	Zero fill left shift	5 << 1	0101 << 1	1010	10
>>	Signed right shift	5 >> 1	0101 >> 1	0010	2
>>>	Zero fill right shift	5 >>> 1	0101 >>> 1	0010	2

Mencionado anteriormente, JavaScript acepta distintos tipos de datos para las variables, que van desde números, cadenas y hasta objetos.

```
var length = 16;                                    // Number
var lastName = "Johnson";                           // String
var x = {firstName:"John", lastName:"Doe"};         // Object
```

Funciones

Las funciones en JavaScript están conformadas por un bloque de código que realiza una acción en específico y devuelven un resultado (No siempre devuelven algo).

```
function hola(p1){
    //Statement 1;
    //Statement 2;
    //Statement 3;
    return p1;
}
```

Las funciones pueden recibir parámetros que serán de utilidad al código que contienen. La sentencia **return** detendrá la ejecución de la función y devolverá el valor que le sea definido al elemento que haya invocado la función.

Arreglos

Sirven para almacenar múltiples valores en un solo contenedor, haciendo más fácil su uso en determinadas aplicaciones.

```
var cars = ["Saab", "Volvo", "BMW"];
```

Sentencias condicionales

Pérez Molinito Cristian José
Programación Web

Las sentencias condicionales son elementos que podemos encontrar en prácticamente cualquier lenguaje de programación, nos permiten evaluar determinados valores y tomar acciones respecto a ello.

```
//Sentencia if
if (condition) {
    //block of code to be executed if the condition is true
}
```

```
//Sentencia if else
if (condition) {
    //block of code to be executed if the condition is true
} else {
    //block of code to be executed if the condition is false
}
```

```
//switch
switch(expression) {
    case n:
        //code block
        break;
    case n:
        //code block
        break;
    default:
        //code block
}
```

Ciclos

El uso de ciclos nos permite ejecutar un bloque de código una y otra vez. El ciclo de ejecución se detendrá cuando se cumplan los requisitos definidos por el programador, para ello existen algunas formas de definir un ciclo:

El ciclo for define un contador, y se detendrá cuando el contador alcance su límite.

```
//Ciclo for
for (statement 1; statement 2; statement 3) {
    //code block to be executed
}
```

El ciclo while evalúa una expresión, mientras el resultado sea verdadero, continuará la ejecución del ciclo, éste se detendrá cuando el resultado de la evaluación sea falso.

```
//ciclo while
while (condition) {
    //code block to be executed
}
```

Existe una variante del ciclo while, que ejecuta el código una vez antes de realizar la evaluación.

```
//ciclo do while
do {
    //code block to be executed
}
while (condition);
```

jQuery

jQuery es una herramienta cuyo propósito es facilitar la implementación y uso de JavaScript durante el desarrollo de un sitio Web.

Se trata de una librería de JavaScript que permite “Hacer más escribiendo menos”.

jQuery condensa tareas comunes que requerirían muchas líneas de código a simples métodos que pueden ser invocados en una sola línea de código. Estas tareas contenidas en la librearía pueden ser:

- Manipulación de **CSS**.
- Control de eventos **HTML**.
- Efectos y animaciones.
- Manipulación **HTML/DOM**.
- **AJAX**.
- Algunas utilidades más.

El uso de jQuery requiere de la instalación de la librearía que será utilizada en el proyecto, de la cual existen 2 versiones:

- **Production versión**: De uso general.
- **Development versión**: De uso para pruebas y desarrollo.

También se puede usar la librería desde un host externo, Google y Microsoft proveen este servicio, para ello, se debe incluir el código siguiente:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>
```

La sintaxis básica de jQuery nos permite seleccionar elementos HTML y ejecutar acciones sobre ellos.

```
$(selector).action()
```

Donde:

- **$** es un símbolo para definir o acceder a jQuery.
- **(selector)** busca los elementos HTML.
- **.action()** ejecuta una acción sobre los elementos.

```
//Ejemplos
$(this).hide()  //hides the current element.

$("p").hide()  //hides all <p> elements.

$(".test").hide()  //hides all elements with class="test".

$("#test").hide()  //hides the element with id="test".
```

Es importante mencionar que todos los métodos de jQuery pueden estar dentro de un “document ready event”. Esto con la finalidad de ejecutar cualquier código jQuery antes de que el documento haya finalizado de cargar en el navegador.

```
//Document ready event statement
$(document).ready(function(){

   // jQuery methods go here...

});
```

jQuery selectors

Los selectores de jQuery nos permiten seleccionar y manipular elementos HTML en específico. Se usan para encontrar los elementos HTML basándose en su nombre, id, clase, tipo, atributo o valor.

Por ejemplo, podemos seleccionar todos los elementos con la etiqueta **<p>**:

```
$("p")
```

Guía rápida para los selectores en jQuery:

Syntax	Description
$("*")	Selects all elements
$(this)	Selects the current HTML element
$("p.intro")	Selects all <p> elements with class="intro"
$("p:first")	Selects the first <p> element
$("ul li:first")	Selects the first element of the first
$("ul li:first-child")	Selects the first element of every
$("[href]")	Selects all elements with an href attribute
$("a[target='_blank']")	Selects all <a> elements with a target attribute value equal to "_blank"
$("a[target!='_blank']")	Selects all <a> elements with a target attribute value NOT equal to "_blank"
$(":button")	Selects all <button> elements and <input> elements of type="button"
$("tr:even")	Selects all even <tr> elements
$("tr:odd")	Selects all odd <tr> elements

Eventos en jQuery

Todas las acciones realizadas en la página web en respuesta a una acción generada por el usuario se pueden denominar **eventos**.

Un evento representa el preciso momento en que algo ocurre. Por ejemplo:

- Mover el cursor sobre un elemento.
- Seleccionar una opción.
- Click sobre un elemento.

Pérez Molinito Cristian José
Programación Web

Incluso existen eventos relacionados a acciones de teclado.

Guía rápida para eventos en jQuery:

Mouse Events	Keyboard Events	Form Events	Document/Window Events
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload

Ejemplo:

```
//Ejemplo simple de click con jQuery
$("p").click(function(){
    $(this).hide();
});
```