# Creación de Arrays en Javascript

El primer paso para utilizar un array es crearlo. Para ello utilizamos un objeto Javascript ya implementado en el navegador. Veremos en adelante un tema para explicar lo que es la orientación a objetos, aunque no será necesario para poder entender el uso de los arrays. Esta es la sentencia para crear un objeto array:

```
var miArray = new Array()
```

Esto crea un array en la página que esta ejecutándose. El array se crea sin ningún contenido, es decir, no tendrá ninguna casilla o compartimiento creado. También podemos crear el array Javascript especificando el número de compartimentos que va a tener.

```
var miArray = new Array(10)
```

En este caso indicamos que el array va a tener 10 posiciones, es decir, 10 casillas donde guardar datos.
Es importante que nos fijemos que la palabra Array en código Javascript se escribe con la primera letra en mayúscula. Como en Javascript las mayúsculas y minúsculas si que importan, si lo escribimos en minúscula no funcionará.
Tanto se indique o no el número de casillas del **array javascript**, podemos introducir en el array cualquier dato. Si la casilla está creada se introduce simplemente y si la casilla no estaba creada se crea y luego se introduce el dato, con lo que el resultado final es el mismo. Esta creación de casillas es dinámica y se produce al mismo tiempo que los scripts se ejecutan. Veamos a continuación cómo introducir valores en nuestros arrays.

```
miArray[0] = 290
miArray[1] = 97
miArray[2] = 127
```

Se introducen indicando entre corchetes el índice de la posición donde queríamos guardar el dato. En este caso introducimos 290 en la posición 0, 97 en la posición 1 y 127 en la 2.

**Los arrays en Javascript empiezan siempre en la posición 0**, así que un array que tenga por ejemplo 10 posiciones, tendrá casillas de la 0 a la 9. Para recoger datos de un array lo hacemos igual: poniendo entre corchetes el índice de la posición a la que queremos acceder. Veamos cómo se imprimiría en la pantalla el contenido de un array.

```
var miArray = new Array(3)
miArray[0] = 155
miArray[1] = 4
miArray[2] = 499

for (i=0;i<3;i++){
   document.write("Posición " + i + " del array: " + miArray[i])
```

```
    document.write("<br>")
}
```

Hemos creado un array con tres posiciones, luego hemos introducido un valor en cada una de las posiciones del array y finalmente las hemos impreso. En general, el recorrido por arrays para imprimir sus posiciones, o cualquier otra cosa, se hace utilizando bucles. En este caso utilizamos un bucle FOR que va desde el 0 hasta el 2.

## Tipos de datos en los arrays

En las casillas de los arrays podemos guardar datos de cualquier tipo. Podemos ver un array donde introducimos datos de tipo carácter.

```
miArray[0] = "Hola"
miArray[1] = "a"
miArray[2] = "todos"
```

Incluso, en Javascript podemos guardar distintos tipos de datos en las casillas de un mismo array. Es decir, podemos introducir números en unas casillas, textos en otras, boleanos o cualquier otra cosa que deseemos.

```
miArray[0] = "desarrolloweb.com"
miArray[1] = 1275
miArray[1] = 0.78
miArray[2] = true
```

## Declaración e inicialización resumida de Arrays

En Javascript tenemos a nuestra disposición una manera resumida de declarar un array y cargar valores en un mismo paso. Fijémonos en el código siguiente:

```
var arrayRapido = [12,45,"array inicializado en su declaración"]
```

Como se puede ver, se está definiendo una variable llamada arrayRapido y estamos indicando en los corchetes varios valores separados por comas. Esto es lo mismo que haver declarado el array con la función Array() y luego haberle cargado los valores uno a uno.

En el próximo artículo seguiremos viendo cosas relacionadas con los arrays, en concreto aprenderemos a acceder a la longitud de un array.

# Ciclos en JavaScript

## Sentencia For

Esta sentencia se utiliza comunmente para ejecutar lineas de codigo un numero determiando de veces, y funciona por medio de un conteo.
Vea el siguiente pseudocodigo para ver su estructura:

```
for(contador = 0; mientras el contador es menor a 10; contador se incrementa en 1)
{

/*codigo que se ejecuta varias veces*/

}
```

La sentencia for consta de 3 parametros, donde el parametro 1 es la variable numerica que contaremos, el parametro 2 es una condicion que determina cuando debe contarse esta variable, y el ultimo parámetro (3) indica si la variable deberá contarse en incrementos o decrementos.

Estructura:

```
for(parámetro 1; parámetro 2; parámetro 3)
{/*código*/
}
```

Para comprender mejor su uso veamos este ejemplo para el cual contamos números del 1 al 5

```
/*Especificamos un contador que inicia en 1, que solo se contara mientras sea menor o igual a 5, y que se contara a razón de incrementos de 1*/

for(var contador=1;contador <=5;contador++)

{

alert(“el contador esta en:”+x);

}

/*fin del programa*/
```

## Sentencia While

La sentencia while lleva la siguietne estructura, y se utiliza comúnmente para crear bucles indefinidos hasta que ciertas condiciones se den en el programa.
Para realizar esto lo escribimos de la siguiente manera. Veamos el pseudo código:

```
while(condición = verdadera) /*mientras una condición sea verdadera*/
{

 /*Código que se ejecuta varias veces*/
 /*Si la condición es aun verdadera repetimos el ciclo, de lo contrario termina el bucle*/
}

/*Este código se ejecuta cuando la “condición” sea diferente a la especificada en el parámetro de while*/
```

En el siguiente ejemplo contamos números del 1 al 5 utilizando la sentencia while

### Ejemplo:

```
var x = 1;
```

```
while(x!=5)
{

 x ++;/*la x es incrementada en 1, cada vez que se repite el ciclo*/
 alert(“la cuenta esta en :”+x);
}/*cuando x alcance el valor de 5, terminara el bucle*/
```

Recordemos que a diferencia de la sentencia for, cuando utilizamos while solo tenemos un parámetro, que se reduce a una condición verdadera o falsa. En este caso tenemos x!=5, la cual es verdadera ya que cuando el programa inicia, x tiene un valor de 1, por lo tanto es diferente de 5 y la condición se hace verdadera, cuando se incrementa x varias veces hasta alcanzar el valor de 5, la condición es falsa porque ya no es igual a 5, y el ciclo termina.