Arrays

En ocasiones, a los arrays se les llama vectores, matrices e incluso *arreglos*. No obstante, el término array es el más utilizado y es una palabra comúnmente aceptada en el entorno de la programación.

Un array es una colección de variables, que pueden ser todas del mismo tipo o cada una de un tipo diferente. Su utilidad se comprende mejor con un ejemplo sencillo: si una aplicación necesita manejar los días de la semana, se podrían crear siete variables de tipo texto:

```
var dia1 = "Lunes";

var dia2 = "Martes";

...

var dia7 = "Domingo";
```

Aunque el código anterior no es incorrecto, sí que es poco eficiente y complica en exceso la programación. Si en vez de los días de la semana se tuviera que guardar el nombre de los meses del año, el nombre de todos los países del mundo o las mediciones diarias de temperatura de los últimos 100 años, se tendrían que crear decenas o cientos de variables.

En este tipo de casos, se pueden agrupar todas las variables relacionadas en una colección de variables o array. El ejemplo anterior se puede rehacer de la siguiente forma:

```
var dias = ["Lunes", "Martes", "Miércoles", "Jueves", "Viernes",
"Sábado", "Domingo"];
```

Ahora, una única variable llamada `dias` almacena todos los valores relacionados entre sí, en este caso los días de la semana. Para definir un array, se utilizan los caracteres `[` y `]` para delimitar su comienzo y su final y se utiliza el carácter `,` (coma) para separar sus elementos:

```
var nombre_array = [valor1, valor2, ..., valorN];
```

Una vez definido un array, es muy sencillo acceder a cada uno de sus elementos. Cada elemento se accede indicando su posición dentro del array. La única complicación, que es responsable de muchos errores cuando se empieza a programar, es que las posiciones de los elementos empiezan a contarse en el `0` y no en el `1`:

```
var diaSeleccionado = dias[0];    // diaSeleccionado = "Lunes"

var otroDia = dias[5];            // otroDia = "Sábado"
```

En el ejemplo anterior, la primera instrucción quiere obtener el primer elemento del array. Para ello, se indica el nombre del array y entre corchetes la posición del elemento dentro del array. Como se ha comentado, las posiciones se empiezan a contar en el `0`, por lo que el primer elemento ocupa la posición `0` y se accede a el mediante `dias[0]`.

El valor `dias[5]` hace referencia al elemento que ocupa la sexta posición dentro del array `dias`. Como las posiciones empiezan a contarse en `0`, la posición `5` hace referencia al sexto elemento, en este caso, el valor `Sábado`.

Funciones útiles para arrays

A continuación se muestran algunas de las funciones más útiles para el manejo de arrays:

`length`, calcula el número de elementos de un array

```
var vocales = ["a", "e", "i", "o", "u"];

var numeroVocales = vocales.length; // numeroVocales = 5
```

`concat()`, se emplea para concatenar los elementos de varios arrays

```
var array1 = [1, 2, 3];

array2 = array1.concat(4, 5, 6);   // array2 = [1, 2, 3, 4, 5, 6
]

array3 = array1.concat([4, 5, 6]); // array3 = [1, 2, 3, 4, 5, 6
]
```

`join(separador)`, es la función contraria a `split()`. Une todos los elementos de un array para formar una cadena de texto. Para unir los elementos se utiliza el carácter `separador` indicado

```
var array = ["hola", "mundo"];

var mensaje = array.join(""); // mensaje = "holamundo"

mensaje = array.join(" ");    // mensaje = "hola mundo"
```

`pop()`, elimina el último elemento del array y lo devuelve. El array original se modifica y su longitud disminuye en 1 elemento.

```
var array = [1, 2, 3];

var ultimo = array.pop();

// ahora array = [1, 2], ultimo = 3
```

`push()`, añade un elemento al final del array. El array original se modifica y aumenta su longitud en 1 elemento. (También es posible añadir más de un elemento a la vez)

```
var array = [1, 2, 3];

array.push(4);

// ahora array = [1, 2, 3, 4]
```

`shift()`, elimina el primer elemento del array y lo devuelve. El array original se ve modificado y su longitud disminuida en 1 elemento.

```
var array = [1, 2, 3];

var primero = array.shift();

// ahora array = [2, 3], primero = 1
```

`unshift()`, añade un elemento al principio del array. El array original se modifica y aumenta su longitud en 1 elemento. (También es posible añadir más de un elemento a la vez)

```
var array = [1, 2, 3];

array.unshift(0);

// ahora array = [0, 1, 2, 3]
```

`reverse()`, modifica un array colocando sus elementos en el orden inverso a su posición original:

```
var array = [1, 2, 3];

array.reverse();

// ahora array = [3, 2, 1]
```