ARRAYS

An array is a special type of variable. It doesn't just store one value; it stores a list of values.

You should consider using an array whenever you are working with a **list** or a set of values that are **related** to each other.

Arrays are especially helpful when you do not know how many items a list will contain because, when you create the array, you do not need to specify how many values it will hold.

If you don't know how many items a list will contain, rather than creating enough variables for a long list (when you might only use a small percentage of them), using an array is considered a better solution.

For example, an array can be suited to storing the individual items on a shopping list because it is a list of related items.

Additionally, each time you write a new shopping list, the number of items on it may differ.

As you will see on the next page, values in an array are separated by commas.

In Chapter 12, you will see that arrays can be very helpful when representing complex data.

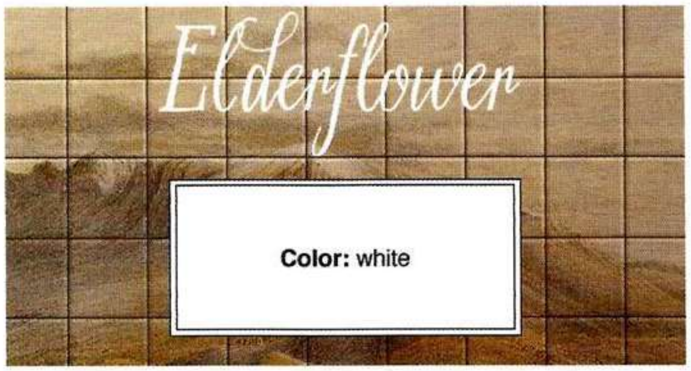
CREATING AN ARRAY

JAVASCRIPT c02/js/array-literal.js

```
var colors;
colors = ['white', 'black', 'custom'];

var el = document.getElementById('colors');
el.textContent = colors[0];
```

RESULT



JAVASCRIPT c02/js/array-constructor.js

```
var colors = new Array('white',
                       'black',
                       'custom');

var el = document.getElementById('colors');
el.innerHTML = colors.item(0);
```

The array literal (shown in the first code sample) is preferred over the array constructor when creating arrays.

You create an array and give it a name just like you would any other variable (using the `var` keyword followed by the name of the array).

The values are assigned to the array inside a pair of square brackets, and each value is separated by a comma. The values in the array do not need to be the same data type, so you can store a string, a number and a Boolean all in the same array.

This technique for creating an array is known as an **array literal**. It is usually the preferred method for creating an array. You can also write each value on a separate line:

```
colors = ['white',
          'black',
          'custom'];
```

On the left, you can see an array created using a different technique called an **array constructor**. This uses the `new` keyword followed by `Array();` The values are then specified in parentheses (not square brackets), and each value is separated by a comma. You can also use a method called `item()` to retrieve data from the array. (The index number of the item is specified in the parentheses.)

VALUES IN ARRAYS

Values in an array are accessed as if they are in a numbered list. It is important to know that the numbering of this list starts at zero (not one).

NUMBERING ITEMS IN AN ARRAY

Each item in an array is automatically given a number called an **index**. This can be used to access specific items in the array. Consider the following array which holds three colors:

```
var colors;
colors = ['white',
          'black',
          'custom'];
```

Confusingly, index values start at 0 (not 1), so the following table shows items from the array and their corresponding index values:

INDEX	VALUE
0	`'white'`
1	`'black'`
2	`'custom'`

ACCESSING ITEMS IN AN ARRAY

To retrieve the third item on the list, the array name is specified along with the index number in square brackets.

Here you can see a variable called `itemThree` is declared. Its value is set to be the third color from the `colors` array.

```
var itemThree;
itemThree = colors[2];
```

NUMBER OF ITEMS IN AN ARRAY

Each array has a property called `length`, which holds the number of items in the array.

Below you can see that a variable called `numColors` is declared. Its value is set to be the number of the items in the array.

The name of the array is followed by a period symbol (or full stop) which is then followed by the `length` keyword.

```
var numColors;
numColors = colors.length;
```

Throughout the book (especially in Chapter 12) you meet more features of arrays, which are a very flexible and powerful feature of JavaScript.

ACCESSING & CHANGING VALUES IN AN ARRAY

JAVASCRIPT

c02/js/update-array.js

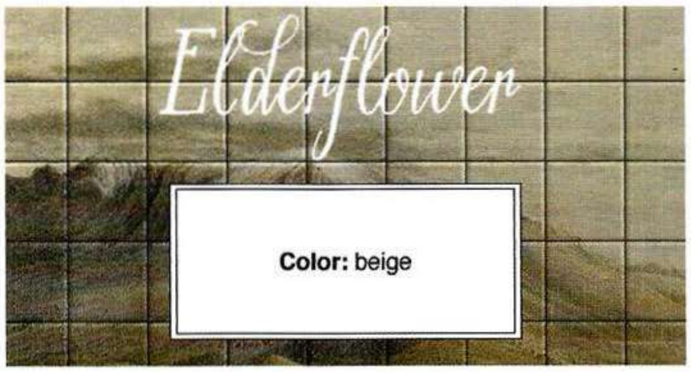
```
// Create the array
var colors = ['white',
              'black',
              'custom'];

// Update the third item in the array
colors[2] = 'beige';

// Get the element with an id of colors
var el = document.getElementById('colors');

// Replace with third item from the array
el.textContent = colors[2];
```

RESULT



The first lines of code on the left create an array containing a list of three colors. (The values can be added on the same line or on separate lines as shown here.)

Having created the array, the third item on the list is changed from `'custom'` to `'beige'`.

To access a value from an array, after the array name you specify the index number for that value inside square brackets.

You can change the value of an item an array by selecting it and assigning it a new value just as you would any other variable (using the equals sign and the new value for that item).

In the last two statements, the newly updated third item in the array is added to the page.

If you wanted to write out *all* of the items in an array, you would use a loop, which you will meet on p170.