An array is a collection of items. Each slot in the array can hold an object or a primitive value. Arrays in Java are objects that can be treated just like other objects in the language.

Arrays can contain any type of element value (primitive types or objects), but you can't store different types in a single array. You can have an array of integers or an array of strings or an array of arrays, but you can't have an array that contains, for example, both strings and integers.

To create an array in Java, you use three steps:

1. Declare a variable to hold the array.
2. Create a new array object and assign it to the array variable.
3. Store things in that array.

The first step in creating an array is creating a variable that will hold the array, just as you would any other variable. Array variables indicate the type of object the array will hold (just as they do for any variable) and the name of the array, followed by empty brackets ([]). The following are all typical array variable declarations:

String difficultWords[];

Point hits[];

int temps[];

An alternate method of defining an array variable is to put the brackets after the type instead of after the variable. They are equivalent, but this latter form is often much more readable. So, for example, these three declarations could be written like this:

String[] difficultWords;

Point[] hits;

int[] temps;

**Creating Array Objects**

The second step is to create an array object and assign it to that variable. There are two ways to do this:

* Using new
* Directly initializing the contents of that array

The first way is to use the new operator to create a new instance of an array:

String[] names = new String[10];

That line creates a new array of Strings with 10 slots (sometimes called elements). When you create a new array object using new, you must indicate how many slots that array will hold. This line does not put actual String objects in the slots-you'll have to do that later.

### Changing Array Elements

To assign an element value to a particular array slot, merely put an assignment statement after the array access expression:

myarray[1] = 15;

sentence[0] = "The";

sentence[10] = sentence[0];

An important thing to note is that an array of objects in Java is an array of references to those objects (similar in some ways to an array of pointers in C or C++). When you assign a value to a slot in an array, you're creating a reference to that object, just as you do for a plain variable. When you move values around inside arrays (as in that last line), you just reassign the reference; you don't copy the value from one slot to another. Arrays of primitive types such as ints or floats do copy the values from one slot to another.

Arrays of references to objects, as opposed to the objects themselves, are particularly useful because you can have multiple references to the same objects both inside and outside arrays. For example, you can assign an object contained in an array to a variable and refer to that same object by using either the variable or the array position.