

Associative Arrays

How to use associative arrays in Perl

Page Resource

[CGI Resources](#) | [Perl Tutorials](#)

[Home/CGI/Perl/Arrays 3](#)

Associative arrays are yet another way to store variables in a group. Unlike a regular array, however, you get to use your own text strings to access elements in the array. Associative arrays are created with a set of key/value pairs. A key is a text string of your choice that will help you remember the value later. The value, then, is the value of the variable you want to store.

Define an Associative Array

Let's take a look at how an associative array is created in Perl:

```
%array_name = ('key1', 'value1', 'key2', 'value2');
```

Of course, you can make it longer or shorter depending on your needs. Notice the % sign at the beginning in front of the array name. This indicates that what follows is an associative array, so the interpreter knows to use the keys and values, rather than assign index numbers to each string. Let's take a look at an example:

```
%our_friends = ('best', 'Don', 'good', 'Robert', 'worst', 'Joe');
```

Here, you have a list of friends associated with keys for remembering them when you access the array later. So, your 'best' friend would be Don, you would have a 'good' friend named Robert, and your 'worst' friend is Joe (poor Joe).

Access Your Elements

To access an element, you use your key string in place of a number. Otherwise, it is like using a normal array. You define a plain variable with the array name followed by its key. So if we wanted to get the name of our 'good' friend, we would use:

```
$our_friends{'good'}
```

Notice the key string 'good', which will give us back our good friend Robert. These are often used by setting the value to another variable, like this:

```
$good_friend = $our_friends{'good'};  
print "I have a good friend named $good_friend.\n";
```

Adding to the Array

Like a regular array, you can add a value to an associative array by simply defining a new value

in your script. In this case, you would define it using a new key string and a value:

```
%our_friends = ('best', 'Don', 'good', 'Robert', 'worst', 'Joe');  
$our_friends{'cool'} = "Karen";
```

This adds the key/value pair of 'cool' and 'Karen' to the %our_friends array.

Deleting from the Array

You can also delete a key/value pair from an associative array using the delete function, which is a little different than the regular array. However, it is a fairly easy command to use, look at the example below:

```
%our_friends = ('best', 'Don', 'good', 'Robert', 'worst', 'Joe');  
delete ($our_friends{'worst'});
```

Now, your 'worst' friend Joe is deleted from the associative array. Again, poor Joe.

Associative arrays are very handy when you are trying to get input from forms on a web page. Most scripts that do this read the form values in as key/value pairs to an associative array, thus making it easy to use the values you need. We'll do a little more with this later when we discuss form input.

Well, that's it for now. Let's go on to: [Chop, Length, and Substrings](#).

Partners

[CoolHomepages](#) | [Web Design Library](#) | [Website Content](#)

The tutorials and articles on these pages are © 1997-2005 by John Pollock and may not be reposted without written permission from the author, and may not be reprinted for profit.



[Previous](#)

By: [John Pollock](#)



[Next](#)

[Main Page](#) | [HTML](#) | [JavaScript](#) | [Graphics](#) | [DHTML/Style Sheets](#) | [ASP/PHP](#)
[PutWeb/FTP](#) | [CGI/Perl](#) | [Promotion](#) | [Java](#) | [Design Articles](#)
[Support Forums](#) | [Site Search](#) | [FAQs](#) | [Privacy](#) | [Contact](#)

Copyright © 1997-2005 [The Web Design Resource](#). All rights reserved.