**ΠΙΝΑΚΕΣ – ARRAYS IN PHP**

<?php
  $animal1 = "dog";
  $animal2 = "iguana";
  $animal3 = "giraffe";
  $animal4 = "fish";
  $animal5 = "tiger";
?>

<?php
  $animals[0] = "dog";
  $animals[1] = "iguana";
  $animals[2] = "giraffe";
  $animals[3] = "fish";
  $animals[4] = "tiger";
?>

<?php
  $animals[0] = "Monkey";
  $animals[1] = "Panda";
  $animals[2] = "Spider";
  $animals[3] = "Elephant";
  $animals[4] = "Ferret";

  $animals = array("Monkey", "Panda", "Spider", "Elephant", "Ferret");

  $animals = array(1 => "Monkey", 2 => "Panda", 3 => "Spider", 4 => "Elephant", 5 => "Ferret");
?>

<?php
  $animals = array("Monkey", "Panda", "Spider", "Elephant", "Ferret");
  echo "Result: ";
  print\_r($animals);
?>

**Result:** Array ( [0] => Monkey [1] => Panda [2] => Spider [3] => Elephant [4] => Ferret )

<?php
  $answer = array("lizard", "panda", "mouse", "snake", "cat");
  echo "What's the definition of a narrow squeak?";
  echo "A thin " . $answer[2] . "!";
?>

**Associative Arrays**

<?php
  $zoo\_animals['Monkey'] = 15;
  $zoo\_animals['Panda'] = 3;
  $zoo\_animals['Spider'] = 167;
  $zoo\_animals['Elephant'] = 5;
  $zoo\_animals['Ferret'] = 7;

  $zoo\_animals = array("Monkey" => 15, "Panda" => 3, "Spider" => 167, "Elephant" => 5, "Ferret" => 7);
?>

**Multidimensional Arrays**

<?php
  $animals = array("Pets"=>array("dog", "cat", "hamster"), "Farm Animals"=>array("Horse", "Cow", "Pig"), "Wildlife"=>array("Elephant", "Deer", "Camel"));
?>

In the example we have an array called "animals" which contains three different arrays ("Pets", "Farm Animals" & "Wildlife"), each of which contain three array values of their own. The "animals" array is an associative array, but each of the three other arrays are numerical arrays, meaning that the following example will echo "Horse":

<?php
  $animals = array(
    "Pets"=>array("dog", "cat", "hamster"),
    "Farm Animals"=>array("Horse", "Cow", "Pig"),
    "Wildlife"=>array("Elephant", "Deer", "Camel")
  );

  echo $animals['Farm Animals'][0];
?>

**Array Functions**

There are over 70 array functions available in PHP, but listed below are 40 of the most useful (and understandable) functions.

|  |  |
| --- | --- |
| **Function** | **Description** |
| [array()](http://www.php.net/manual/en/function.array.php) | Create an array |
| [array\_change\_key\_case()](http://www.php.net/manual/en/function.array-change-key-case.php) | Changes all keys in an array |
| [array\_chunk()](http://www.php.net/manual/en/function.array-chunk.php) | Split an array into chunks |
| [array\_combine()](http://www.php.net/manual/en/function.array-combine.php) | Creates an array by using one array for keys and another for its values |
| [array\_count\_values()](http://www.php.net/manual/en/function.array-count-values.php) | Counts all the values of an array |
| [array\_flip()](http://www.php.net/manual/en/function.array-flip.php) | Exchanges all keys with their associated values in an array |
| [array\_keys()](http://www.php.net/manual/en/function.array-keys.php) | Return all the keys or a subset of the keys of an array |
| [array\_merge()](http://www.php.net/manual/en/function.array-merge.php) | Merge one or more arrays |
| [array\_multisort()](http://www.php.net/manual/en/function.array-multisort.php) | Sort multiple or multi-dimensional arrays |
| [array\_pad()](http://www.php.net/manual/en/function.array-pad.php) | Pad array to the specified length with a value |
| [array\_product()](http://www.php.net/manual/en/function.array-product.php) | Calculate the product of values in an array |
| [array\_rand()](http://www.php.net/manual/en/function.array-rand.php) | Pick one or more random entries out of an array |
| [array\_reverse()](http://www.php.net/manual/en/function.array-reverse.php) | Return an array with elements in reverse order |
| [array\_search()](http://www.php.net/manual/en/function.array-search.php) | Searches the array for a given value and returns the corresponding key if successful |
| [array\_slice()](http://www.php.net/manual/en/function.array-slice.php) | Extract a slice of the array |
| [array\_splice()](http://www.php.net/manual/en/function.array-splice.php) | Remove a portion of the array and replace it with something else |
| [array\_sum()](http://www.php.net/manual/en/function.array-sum.php) | Calculate the sum of values in an array |
| [array\_unique()](http://www.php.net/manual/en/function.array-unique.php) | Removes duplicate values from an array |
| [arsort()](http://www.php.net/manual/en/function.arsort.php) | Sort an array in reverse order and maintain index association |
| [asort()](http://www.php.net/manual/en/function.asort.php) | Sort an array and maintain index association |
| [compact()](http://www.php.net/manual/en/function.compact.php) | Create array containing variables and their values |
| [count()](http://www.php.net/manual/en/function.count.php) | Count all elements in an array, or properties in an object |
| [current()](http://www.php.net/manual/en/function.current.php) | Return the current element in an array |
| [each()](http://www.php.net/manual/en/function.each.php) | Return the current key and value pair from an array and advance the array cursor |
| [end()](http://www.php.net/manual/en/function.end.php) | Set the internal pointer of an array to its last element |
| [in\_array()](http://www.php.net/manual/en/function.in-array.php) | Checks if a value exists in an array |
| [is\_array()](http://www.php.net/manual/en/function.is-array.php) | Checks if a variable is an array |
| [key()](http://www.php.net/manual/en/function.key.php) | Fetch a key from an array |
| [krsort()](http://www.php.net/manual/en/function.krsort.php) | Sort an array by key in reverse order |
| [ksort()](http://www.php.net/manual/en/function.ksort.php) | Sort an array by key |
| [list()](http://www.php.net/manual/en/function.list.php) | Assign variables as if they were an array |
| [natcasesort()](http://www.php.net/manual/en/function.natcasesort.php) | Sort an array using a case insensitive "natural order" algorithm |
| [natsort()](http://www.php.net/manual/en/function.natsort.php) | Sort an array using a "natural order" algorithm |
| [next()](http://www.php.net/manual/en/function.next.php) | Advance the internal array pointer of an array |
| [prev()](http://www.php.net/manual/en/function.prev.php) | Rewind the internal array pointer |
| [range()](http://www.php.net/manual/en/function.range.php) | Create an array containing a range of elements |
| [reset()](http://www.php.net/manual/en/function.reset.php) | Set the internal pointer of an array to its first element |
| [rsort()](http://www.php.net/manual/en/function.rsort.php) | Sort an array in reverse order |
| [shuffle()](http://www.php.net/manual/en/function.shuffle.php) | Shuffle an array |
| [sizeof()](http://www.php.net/manual/en/function.sizeof.php) | Alias of count |
| [sort()](http://www.php.net/manual/en/function.sort.php) | Sort an array |

Let's take a look at these three functions in action

<?php
  $fruit = array("orange", "pineapple", "peach", "apple", "pear", "cherry");

  sort($fruit);
  print\_r($fruit);

  echo count($fruit);

  $range\_array = range(0, 50, 5);
  print\_r($range\_array);
?>

And the results are:

Array ( [0] => apple [1] => cherry [2] => orange [3] => peach [4] => pear [5] => pineapple )

6

Array ( [0] => 0 [1] => 5 [2] => 10 [3] => 15 [4] => 20 [5] => 25 [6] => 30 [7] => 35 [8] => 40 [9] => 45 [10] => 50 )

**Array Loops (Foreach)**

<?php
  $characters[0] = "Bugs Bunny";
  $characters[1] = "Tweety";
  $characters[2] = "Wile E. Coyote";
  $characters[3] = "Elmer Fud";
  $characters[4] = "Sylvester";
  $characters[5] = "Road Runner";

  foreach ($characters as $value) {
    echo $value . "<br />";
  }
?>

<?php
  $characters['pig'] = "Porky Pig";
  $characters['duck'] = "Daffy Duck";
  $characters['mouse'] = "Speedy Gonzales";

  foreach ($characters as $key => $value) {
    echo $value . " is a " . $key . ".<br />";
  }
?>

And the result is:

Porky Pig is a pig.
Daffy Duck is a duck.
Speedy Gonzales is a mouse.

<form action="<?php echo $\_SERVER['PHP\_SELF']; ?>" method="**GET**">

 Type In Something: <input name="random-info" type="text" size="25">

 <input type="submit" value="Submit">

</form> <br>

<?php

 echo "You Typed: " . $\_GET['random-info'];

?>

<form action="<?php echo $\_SERVER['PHP\_SELF']; ?>" method="POST">

 Type In Something: <input name="random-info" type="text" size="25">

 <input type="submit" value="Submit">

</form> <br>

<?php

 echo "You Typed: " . $\_POST['random-info'];

?>