Arrays

Presenter:
Agubata Odinaka

What is an Array?

Definition

An array is a special variable that stores one or more values at once.

For example if you want to store 100 numbers then instead of defining 100 variables its easy to define an array of 100 length

Definition

There are three different kinds of arrays and each array value is accessed using a key which is called array index.

- ❖Numeric array
- ❖Associative array
- ❖Multidimensional array

To create an array we make use of the array() function.

Numeric Array

Numeric array is an array with a numeric index.

Values are stored and accessed in linear fashion.

These arrays can store numbers, strings and any object but their index will be represented by numbers. By default array index starts from zero.

Let’s look at the example below:

Numeric Array

Example 1

```
/* First method to create array. */
$numbers = array( 1, 2, 3, 4, 5);
echo “The first number is $numbers[0]”;
echo “The last number is $numbers[4]”;
```

Numeric Array

Example 2

```
/* First method to create array. */
$numbers = array( 1, 2, 3, 4, 5);
foreach( $numbers as $value )
{
  echo "Value is $value <br />";
}
```

Numeric Array

Example 3

```
/* Second method to create array. */
$numbers[0] = "one";
$numbers[1] = "two";
$numbers[2] = "three";
$numbers[3] = "four";
$numbers[4] = "five";
foreach( $numbers as $value )
{
  echo "Value is $value <br />";
}
```

Numeric Array

Example 4

```
<?php
$cars = array("Volvo", "BMW", "Toyota");
$arrlength = count($cars);

for($x = 0; $x < $arrlength; $x++) {
   echo $cars[$x]. "<br>";
}
?>
```

Associative Array

The associative arrays are very similar to numeric arrays in terms of functionality but they are different in terms of their index. Associative array will have their index as string so that you can establish a strong association between key and values.

Associative Array

To store the salaries of employees in an array, a numerically indexed array would not be the best choice. Instead, we could use the employees names as the keys in our associative array, and the value would be their respective salary.

Associative Array

Example 1:

```
/* First method to associate create array. */
$salaries = array(
                    "mohammad" => 2000,
                    "qadir" => 1000,
                    "zara" => 500
                   );

echo "Salary of mohammad is ". $salaries['mohammad'] . "<br />";
echo "Salary of qadir is ".  $salaries['qadir']. "<br />";
echo "Salary of zara is ".  $salaries['zara']. "<br />";
```

Associative Array

Example 2:

```
/* Second method to create array. */
$salaries['mohammad'] = "high";
$salaries['qadir'] = "medium";
$salaries['zara'] = "low";

echo "Salary of mohammad is ". $salaries['mohammad'] . "<br />";
echo "Salary of qadir is ".  $salaries['qadir']. "<br />";
echo "Salary of zara is ".  $salaries['zara']. "<br />";
```

Associative Array

Example 3:

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");

foreach($age as $name => $name_value) {
    echo $name . " is " . $name_value ." years old <br>“;
}
?>
```

Multidimensional Arrays

In a multi-dimensional array each element in the main array can also be an array. And each element in the sub-array can be an array, and so on. Values in the multi-dimensional array are accessed using multiple index.

Example:

In this example we create a two dimensional array to store marks of three students in three subjects:

Multidimensional Arrays

$marks = array("Mohammad" => array("physics" => 35,"maths" => 30, "chemistry" => 39), “Qadir" => array("physics" => 30, "maths" => 32, "chemistry" => 29), “Zara" => array("physics" => 31, "maths" => 22, "chemistry" => 39));

To access a multi-dimentional array:

echo $marks[‘Mohammad']['physics'];

Multidimensional Arrays

```
$cars = array( array("Volvo",22,18), array("BMW",15,13), array("Saab",5,2),
 array("Land Rover",17,15)
 );
 for ($row = 0; $row < 4; $row++) {
        echo "<p><b>Row number $row</b></p>";
        echo "<ul>";
        for ($col = 0; $col < 3; $col++) {
          echo "<li>".$cars[$row][$col]."</li>";
        }
        echo "</ul>";
}
```

Array Functions

Array Functions

array_sum()

array_sum() returns the sum of values in an array as an integer or float.

Example:

```
<?php
$a=2;
$as = array(3, 4, 5, 6, 2);
echo array_sum($a);

?>
```

The above example will output:

```
sum(a) = 20
```

array_reverse()

array_reverse() takes input array and returns a new array with the order of the elements reversed.

Example:

```
<?php
$cars = array("Volvo", "BMW", "Toyota");
 $reversed_array = array_reverse($cars);

print_r($cars) ;
print_r($reversed_array ) ;
?>
```

The above example will output:

```
Array ( [0] => Volvo [1] => BMW [2] => Toyota )
Array ( [0] => Toyota [1] => BMW [2] => Volvo )
```

array_push()

array_push() treats array as a stack, and pushes the passed variables onto the end of array. The length of array increases by the number of variables pushed.

Example:

```
<?php
$arr = array("orange", "banana");
array_push($arr, "apple", "raspberry");
print_r($arr);
?>
```

count

Returns the number of elements in an array

Example:

```
<?php

$as = array(3, 4, 5, 6, 2);

$result = count($as);

echo $result ;

?>
```

sort

This function sorts an array. Elements will be arranged in ascending order when this function has completed.

Example:

```
<?php
$fruits = array("lemon", "orange", "banana", "apple");
sort($fruits);
$clength = count($fruits);
for($x = 0; $x < $clength; $x++) {
   echo $fruits[$x].”<br>”;
}
?>
```

QUESTIONS?