# Array

In computer science an **array** is a data structure consisting of a group of elements that are accessed by indexing. In most programming languages each element has the same data type and the array occupies a contiguous area of storage. Most programming languages have a built-in array data type.

Some programming languages support array programming which generalizes operations and functions to work transparently over arrays as they do with scalars, instead of requiring looping over array members.

Well as we can see here guys, array can be divided into 3 categories:

- **Numeric array**
- **Associative array**
- **Multidimensional array** (sorry, will not be discussed here)

We're going to define one by one of the categories that provided above with the example for each of it.

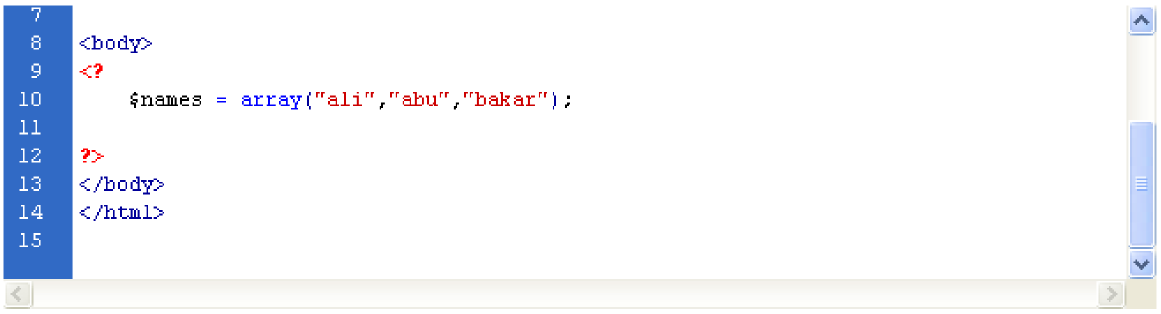
## Numeric array

When we discuss about numeric ID it can be done using 2 ways. It is :

- Assign using Automatic method
- Assign using Manual method

A numeric array stores each element with a numeric ID key. Using the numeric ID, the index of the array can be stored using 2 ways. The below example is automatically assigned into the array.

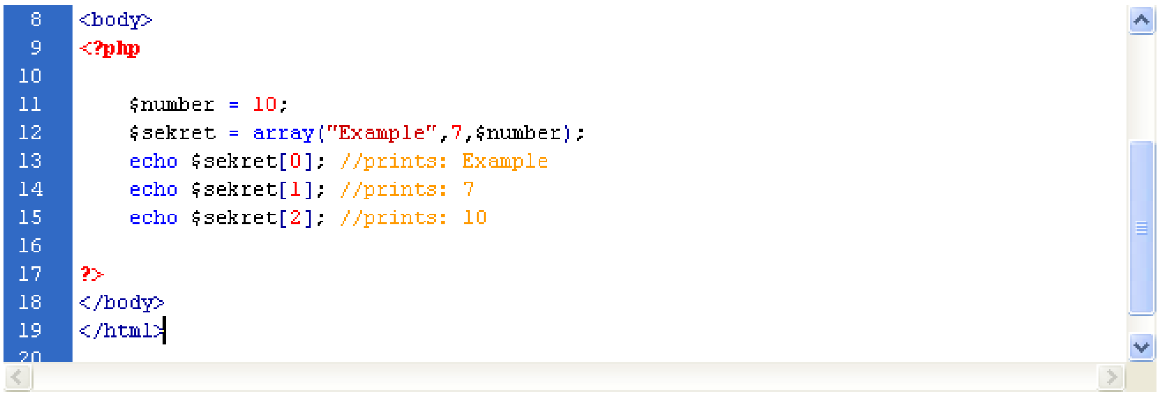
Example 1:



```
<body>
<?
    $names = array("ali","abu","bakar");

?>
</body>
</html>
```

You can see the variable $names that used as array that access by its element. This is how when a string has been initialized into the array. Because array by default is empty and to be cleared every array is start from 0. If we declared the array with the size of 9 actually the max for it is only 8, because all array is start from 0. Next using the same technique, we see in the second example.

Example 2:

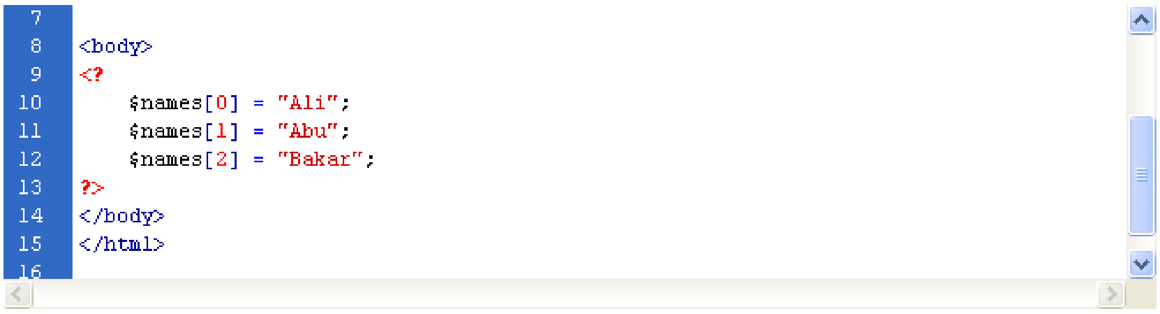


```
<body>
<?php

    $number = 10;
    $sekret = array("Example",7,$number);
    echo $sekret[0]; //prints: Example
    echo $sekret[1]; //prints: 7
    echo $sekret[2]; //prints: 10

?>
</body>
</html>
```

As you can see, elements in array can be any type scalar of data (string, number, variable) and so on. So, here the advantage when using array. You can initialize any type data into it as long it follows the rule using the array.

The second way stored the data in numeric ID can be done manually. As shown in the example below:

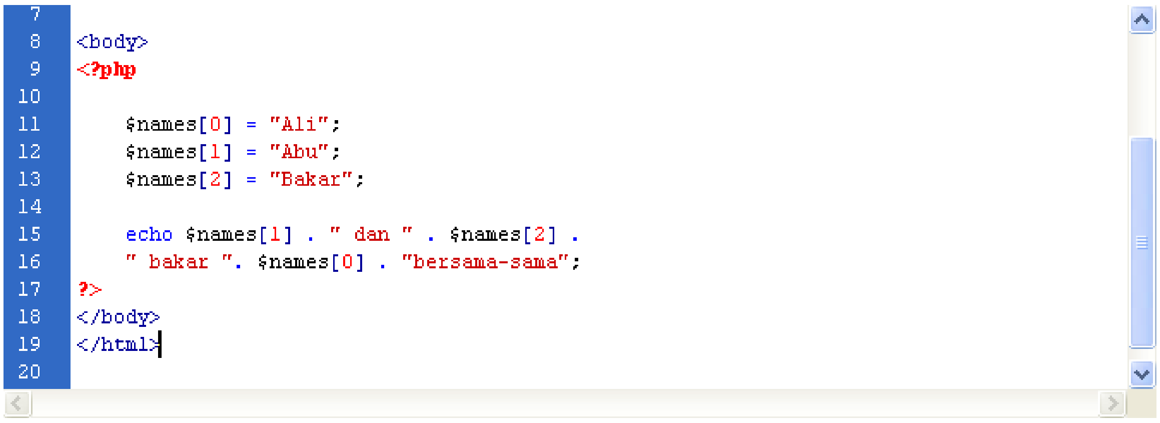
Example 1:



```

<body>
<?
    $names[0] = "Ali";
    $names[1] = "Abu";
    $names[2] = "Bakar";
?>
</body>
</html>
```

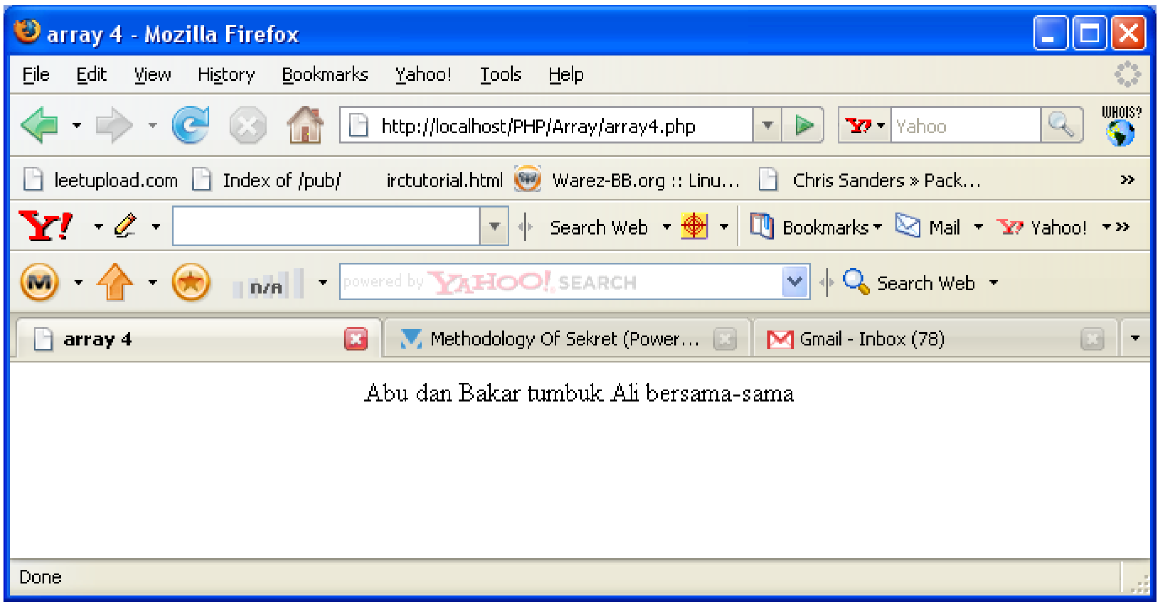
As we look the example above, every element is initialized using the manual method. “peter” was initialize to Array number 1 a.k.a names[0] and so on . So that’s the example how it done using manual method.

The ID key can also be used in script. Refer to the  example below.



```

<body>
<?php

    $names[0] = "Ali";
    $names[1] = "Abu";
    $names[2] = "Bakar";

    echo $names[1] . " dan " . $names[2] .
    " bakar ". $names[0] . "bersama-sama";
?>
</body>
</html>

```

Output :



```
Abu dan Bakar tumbuk Ali bersama-sama
```

## Associative array

Basic concept for associative array is quite similar with numeric array. The difference between those arrays is the way how to initialize ID key. Each ID key is associated with value. When storing data about specific named values, a numerical array is not a best way to do it.

By using associative array, value can be assign as keys, and assign value to them. Refer to example 1 and 2; there are 2 methods to initiate this array.

```
<?php

        $ages = array("Ali"=>32 , "Ahmad"=>25, "Abu"=> 28);

?>
```

Example 1: First method to initiate associative array

```
<?php

        $ages["Ali"] = 32;
        $ages["Ahmad"] = 25;
        $ages["Abu"] = 28;

?>
```

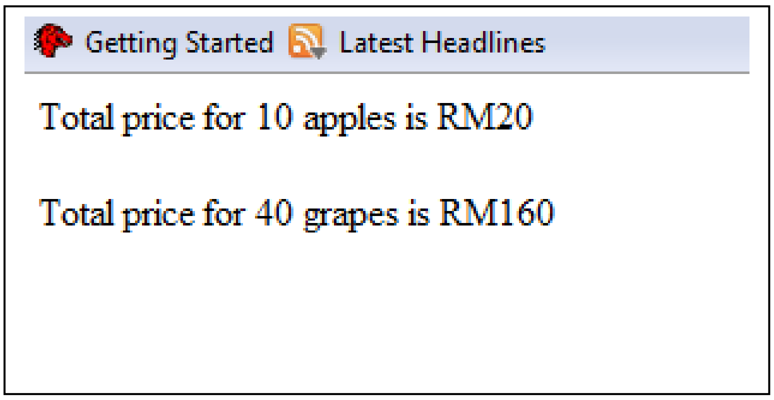
Example 2: Second method to initiate associative array

Refer to example 3, the application sample code to assign array using first and second methods, and output for this type of array.

Sample code:

```
<?php

  //first method
  $fruit = array("Apple" => 2, "Pineapple" => 3, "Orange" => 1, "Grape"=>4);

  //second method
  $quantity["Apple"] = 10;
  $quantity["Pineapple"] = 20;
  $quantity["Orange"] = 30;
  $quantity["Grape"] = 40;

  //application or operation
  $price["Apple"] = $fruit["Apple"] * $quantity["Apple"];

  $price["Grape"] = $fruit["Grape"] * $quantity["Grape"];

  //output to screen
  echo "Total price for $quantity['Apple'] apples is RM$price['Apple']<br><br>";

  echo "Total price for $quantity['Grape'] grapes is RM$price['Grape']<br><br>";

?>
```

Output :



```
Total price for 10 apples is RM20

Total price for 40 grapes is RM160
```