# PHP
# One dimensional (indexed) arrays
# Associative arrays

Primary source: https://www.w3schools.com/php/

**Internet Programming 2, Lesson 5**

**April 2024**

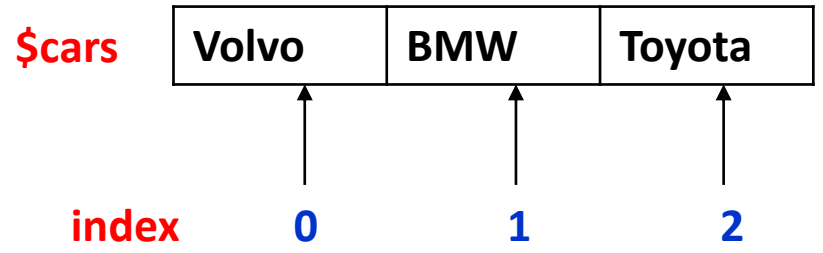
# One dimensional (indexed) arrays

Some aspects about 1D arrays:

- is viewed as: a mechanism (data structure) to store multiple values using a single name and index.
- created using a “constructor” called array(), and the argument is a comma-separated list of values
- the index “points” to an array element and represents the “address” of the element
- the index is an integer value starting at 0
- there is a function count() which returns the length of an array (number of elements)
- since the index is sequential, it is possible to step through the array elements using a for loop
- visual representation and PHP code below

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

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



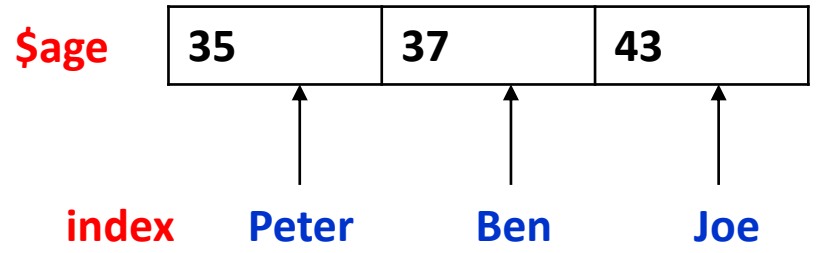
# Associative arrays

Some aspects about associative arrays:

- is viewed as: a mechanism (data structure) to store multiple values using a single name and index.
- created using a “constructor” called array(), and the argument passed to the constructor is a comma-separated list of “keys” and values
- the index “points” to an array element and represents the “address” of the element
- the index is a named constant: Associative array differ from numeric-indexed array in the sense that associative arrays use descriptive names for id keys.
- there is a function count() which returns the length of an array (number of elements)
- since the index is NOT sequential, it is NOT possible to step through the array elements using a for loop
- Need to use a foreach-loop
- visual representation and PHP code below

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

foreach($age as $x => $y) {
    echo "Key=" . $x . ", Value=" . $y;
    echo "<br>";
}
?>
```



## Associative arrays

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

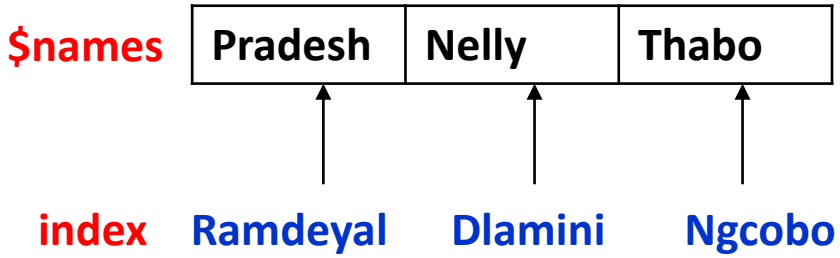
*Read the "=>" symbol as "points to" to remember which represents the index and which represents the value*

```
foreach($age as $x => $x_value) {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
}
```

*How the foreach loop works:*

At the first iteration of the loop, the "key" or "index" is stored in $x and the value (content of the array element) is stored in $x_value.
Once the loop completes an iteration and returns to the top of the loop, the next index is stored in $x and the corresponding value in $_value,
And so on…

# Associative arrays




*The above array may be implemented in PHP as follows:*

```
<?php
$names = array(“Ramdeyal"=>“Pradesh”, “Dlamini”=>“Nelly", “Ngcobo"=>“Thabo");

foreach($names as $surname => $firstname) {
    echo $firstname . “ ”. $surname;
    echo "<br>";
}
?>
```

# Multi-dimensional arrays

These are arrays that contain other nested arrays.
The advantage of multidimensional arrays is that they allow us to group related data together.

Let’s now look at a practical example that implements a php multidimensional array.

The table below shows a list of movies by category.

| Movie title | Category |
|---|---|
| Pink Panther | Comedy |
| John English | Comedy |
| Die Hard | Action |
| Expendables | Action |
| The Lord of the rings | Epic |
| Romeo and Juliet | Romance |
| See no evil hear no evil | Comedy |

The above information can be represented as a multidimensional array. The code below shows the implementation.

```
<?php
$movies =array(     "comedy" => array("Pink Panther", "John English", "See no evil hear no evil"),
                    "action" => array("Die Hard", "Expendables"),
                    "epic" => array("The Lord of the rings"),
                    "Romance" => array("Romeo and Juliet")
                    );
?>
```

Another way to define the same array is as follows

```
<?php
$film=array(
        "comedy" => array(
                0 => "Pink Panther",
                1 => "john English",
                2 => "See no evil hear no evil"
                ),
        "action" => array (
                0 => "Die Hard",
                1 => "Expendables"
                ),
        "epic" => array (
                0 => "The Lord of the rings"
                ),
        "Romance" => array
                (
                0 => "Romeo and Juliet"
                )
);
echo $film["comedy"][0];
?>
```

# PHP - Sort Functions For Arrays

Review these from the w3schools site or other web resources.

sort() - sort arrays in ascending order

rsort() - sort arrays in descending order

asort() - sort associative arrays in ascending order, according to the value

ksort() - sort associative arrays in ascending order, according to the key

arsort() - sort associative arrays in descending order, according to the value

krsort() - sort associative arrays in descending order, according to the key

Also get familiar with the Complete PHP Array Reference in the w3schools site

The reference contains a brief description, and examples of use, for each function!

# Examples

Create a php script (save as arrays1.php) and write the code to perform the following:
Create an array called $squares and populate the first 10 elements of the array with the square of it's index value.
Display the index and the actual value in a neat table.

```
<?php

echo "<h3 style ='color:red'>Diaplay the index and value of each element of the Squares array:</h3>";
for ($i = 0; $i <10; $i++){
        $squares[$i] = $i*$i;
}
echo '<table align="CENTER" cellpadding="2" border="2"><tr><th>Index</th><th>Value</th></tr>';

for ($i = 0; $i <10; $i++){
        echo "<tr><td>" . $i ."</td><td>" . $squares[$i]. "</td></tr>";

}
echo'</table>';
?>
```

Create a php script (save as arrays2.php) and write the code to perform the following:
Create an associative array called $users. The index of the array represents the surname and the contents (value) represents the first name. Populate the first 5 elements of the array, making use of the array function. Now write the code to display only the first names in a neat table.

```
<?php

$users = array("Ramdeyal" => "Pradesh",
                                        "Zuma" => "Jacob",
                                        "Dlamini" => "Nelly",
                                        "Duma" => "Thabo",
                                        "Singh" => "Arvish",);

echo "<h3 style ='color:red'>Diaplay the first name (value) of each element of the users array:</h3>";
$len = count($users);

echo '<table align="CENTER" cellpadding="2" border="2"><tr><th>Value</th></tr>';

foreach ($users as $key => $value){
        echo "<tr><td>" . $value. "</td></tr>";
}
echo'</table>';
?>
```

Same as question 5 (save as arrays3.php), but display both the first name and surname for each element of the array.

```
<?php

$users = array("Ramdeyal" => "Pradesh",
                                        "Zuma" => "Jacob",
                                        "Dlamini" => "Nelly",
                                        "Duma" => "Thabo",
                                        "Singh" => "Arvish",);

echo "<h3 style ='color:red'>Diaplay the first name (value) and surname (index) of each element of the users array:</h3>";
echo '<table align="CENTER" cellpadding="2" border="2"><tr><th>Value (First Name)</th><th>Index(Surname)</th></tr>';

foreach ($users as $key => $value){
        echo "<tr><td>" . $value ."</td><td>" . $key. "</td></tr>";

}
echo'</table>';
?>
```

## Summary

Arrays are special variables with the capacity to store multi values.

Arrays are flexibility and can be easily stretched to accommodate more values

Numerically-Indexed arrays use numbers for the array keys

PHP Associative array use descriptive names for array keys

Multidimensional arrays contain other arrays inside them.

The count function is used to get the number of items that have been stored in an array

The is_array function is used to determine whether a variable is a valid array or not.

Other array functions include sort, ksort, assort etc.

***Exercise***

1. Write a PHP Script to:

   Create a five-element 1D array containing the first names of 5 of your friends; and another array to contain their corresponding surnames

   Display each friends Full Name but rotate the colors of the displayed test between red and blue. Display this one full name per line in the webpage.

2. Write a PHP program to perform the following tasks: save the solution as ***passwords.php***

   2.1. Create an associative array called $passwords;

   2.2. populate the above array with 5 randomly-generated and md5-encrypted passwords – use the randomly-generated password as the key (index) and the encrypted form of the password as the content of the array elements – see below for rules pertaining to password formulation.

   2.3. Display the actual password and it's encrypted form in a neat table on the web page.

   Generate the passwords as follows: [write a function to generate the password]

   - Minimum length is 3 and maximum length is 8 characters;
   - Passwords may contain lowercase letters and/or the digits 0 to 9 only;
   - Passwords cannot start with a number.

3. Write a PHP script to perform the following tasks: save the solution as ***accessControl.php***

   3.1. Create an associative array called $users;

   3.2. populate the array with a minimum of 5 elements as follows:

   | userID | Password |
   |---|---|
   | Ramdeyal | ram123 |
   | Dlamini | dla123 |
   | Zuma | zum123 |

   …and so on. Remember to encrypt the password using the md5() function.

   3.3. provide a log-in form for the user, and control access as follows: if the userId does not exist, display the message "user unknown – please check your userid before trying again" and allow the user to try to login again. If the password is correct, Display "Access granted" and donot display the login for again. If the password is incorrect, allow 3 chances for the user to login. Once 3 incorrect passwords have been submitted, block the user and display "Access blocked: please consult with your administrator".