Array:

An array is a spl. var., which can hold more than 1 value at a time with similar type.

2 ways to declare an array:

Using array() func.

Syntax:

```
$array-name = array("val-1", "val-2", "val-n");
```

Eg:

```
<?php
    $cars = array("Volvo", "BMW", "Toyota");
    echo "Car brands are". $cars[0]. ",". $cars[1]. "and".
        $cars[2];
?>
```

O/P: car brands are volvo, BMW and Toyota

Using index value:

The array values can be directly assigned using the index values without using array() func.

Syntax:

```
$array-name[0] = "val1 Volvo";
$array-name[1] = "val2";
$array-name[n] = "valn";
```

Eg:

```
<?php
    $cars[0] = "Volvo";
    $cars[1] = "BMW";
    $cars[2] = "Toyota";
    echo "Car brands:". $cars[0]. ",". $cars[1]. "and". $cars[2];
?>
```

Types of Array:

In Php, there are 3 types of arrays:

* Indexed arrays - Arrays with a numeric index
* Associative arrays - " " named keys
* Multidimensional arrays - " containing 1 / more arrays.

Indexed arrays:

~~There are~~ There are 2 ways to create indexed arrays:

using array() func.

The index values are assigned automatically when array() func. is used. Index val. always starts from 0.

Eg: $fruits = array("Apple", "Orange", "Mango");

using index val.

The index values are assigned manually to set the array val.

Eg: $fruits[0] = "Apple";
$fruits[1] = "Orange";
$fruits[2] = "Mango";

Loop thro' an Indexed array:

To loop thro' & print all the val. of an indexed array, for loop could be used. For eg:

```
<?php
    $fruits = array("Apple", "Orange", "Mango");
    $arrlength = count($cars); // counts the no. of array val.
    for($i = 0; $i < $arrlength; $i++)
    {
        echo $fruits[$i];
        echo "<br>";
    }
?>
```

O/P: Apple
Orange
Mango

Associative Arrays:

* ~~Arr~~. Associative arrays use named keys that we assign to them.
* There are two ways to declare an associative array.

Eg:
```
<?php
    $age = array("Angel" => "35",
                 "Bharathi" => "30",
                 "Priya" => "20");
    echo "Bharathi is ". $age['Bharathi']. "years old";
?>
```

O/P: Bharathi is 30 years old.

In associative arrays, instead of index value the named keys are used to assign the array values.

Eg:
```
<?php
$age['Angel'] = "35";
$age['Bharathi'] = "30";
$age['Priya'] = "20";
echo "Angel is ". $age['Angel']. "years old";
```

O/P: Angel is 35 years old.

Loop thro' an Associative array:

To loop thro' & print all the val. of an associative array, foreach loop could be used.

Eg:
```
<?php
$age = array ("Angel" => "35", "Bharathi" => "30", "Priya" => "20");
foreach ($age as $i => $i_value)
{
    echo "Key = ". $i. ", Value = ". $i_value;
    echo "<br>";
}
?>
```

O/P: Key = Angel, Value = 35

Key = Bharathi, Value = 30

Key = Priya, Value = 20

Multidimensional Arrays:

* A multidimensional array is an array containing 1/ more arrays.

* Php supports multidimensional arrays that are two, three, four / more levels deep.

Eg:
```
<?php
$marks = array ("A" => array ("Tamil" = 75,
                              "Eng" = 55,
                              "Maths" = 90),
                "B" => array ("Tamil" = 90,
                              "Eng" = 95,
                              "Maths" = 92));
echo "The marks of A in Maths :", $marks['A']['M'];
?>
```

O/P: The marks of A in Maths : 90

Date() func.

date() func. is used to format a date & / a time.

Syntax:

```
date (format, timestamp)
```

format - specifies format of timestamp

timestamp - (Optional) specifies a timestamp. Default is current date & time.

Seq. of ch. denoting the date & / time at which certain event occured.

Get a date:

* The format para. in date() func. specifies how to format the date.

* Some ch. that are commonly used for dates are:

d - rep. day of the month (01 to 31)

m - rep. a month (01 to 12)

Y - rep. a year (in 4 digits)

l - rep. day of the week

/ , . , - → inserted bet. ch. for tal formatting

Eg:
```
<?php
    echo "Today is". date ("Y/m/d"). "<br>";
    echo date ("d.m.Y"). "<br>";
    echo date ("d-m-Y"). "<br>";
    echo date ("l");
?>
```

O/P: 2021/03/16
16.03.2021

Automatic copyright yr.

date() func. is used to automatically update the copyright yr. on the website.

Eg:
```
&copy; 2010- <?php echo date ("Y"); ?>
```

O/P: © 2010 - 2021

Get a time:

The format para. in date() func. specifies how to format the time.

Some ch. that are commonly used for time are:

H - 24-hr format of an hr (00 to 23)

h - 12-hr format of an hr with leading zeros (01 to 12)

i - min. with leading zeros (00 to 59)

s - sec. " " " " "

a - lowercase am or pm

Eg:
```
<?php
    echo "The time is". date ("h:i:sa");
?>
```

O/P: 08:37:20 am

Array Functions:

1. array_change_key_case()

This func. changes all keys in an array to lowercase or uppercase.

Syntax:

```
array_change_key_case (array, case)
```

array - Specifies the array to use

case - Optional. Possible val.

- CASE_LOWER - Default. changes the keys to lowercase.
- CASE_UPPER - Changes the keys to uppercase.

Eg:
```
<?php
$age = array ("Anu"=>"35", "Vini"=>"20", "Abi"=>"30");
print_r (array_change_key_case ($age, CASE_UPPER));
?>
```

O/P: Array ([ANU] => 35 [VINI] => 20 [ABI] => 30)

2. array_chunk()

Split an array into chunks of new arrays.

Syntax:

```
array_chunk (array, size, preserve_key)
```

array - Specifies the array to use

size - An integer spe that specifies the size of each chunk.

preserve_key - Optional. Possible val.

- true - preserves the keys
- fake - Default. Reindexes the chunk numerically.

Eg:
```
<?php
$fruit = array ("Apple", "Grapes", "Mango", "Orange");
print_r (array_chunk ($fruit, 2));
?>
```

O/P: Array ([0] => Array ([0] => Apple [1] => Grapes)
[1] => Array ([0] => Mango [1] => Orange))

Eg:
```
<?php
$fruit = array ("Apple", "Grapes", "Mango", "Orange");
print_r (array_chunk ($fruit, 2, true));
?>
```

O/P: Array ([0] => Array ([0] => Apple [1] => Grapes)
[1] => Array ([2] => Mango [3] => Orange))

3. array_combine()

* creates an array by using the elements from one "keys" array & 1 "values" array.

* Both arrays must have = no. of ele.

Syntax:

```
array_combine (keys, values)
```

keys: array of keys; values: array of val.

Eg:
```
<?php
$name = array ("Anu", "Rani", "Riya");
$age = array ("20", "25", "27");
print_r (array_combine ($name, $age));
?>
```

O/P: Array ([Anu] => 20 [Rani] => 25 [Riya] => 27)

4. array_count_values()

counts all the values of an array.

Syntax:

```
array_count_values(array)
```

Eg: <?php
$a = array("A", "Cat", "Dog", "A", "Dog");
print_r(array_count_values($a));
?>

O/P: Array ([A] => 2 [Cat] => 1 [Dog] => 2)

5. array_merge()

* merges 1 / more arrays into 1 array.

* If 2 / more array ele. have the same key, the last one overrides the others.

Syntax:

```
array_merge(array1, array2, array3...)
```

Eg: <?php
$a1 = array("red", "blue");
$a2 = array("green", "yellow");
print_r(array_merge($a1, $a2));
?>

O/P: Array ([0] => red [1] => blue [2] => green [3] => yellow)

6. array_product()

calculates & returns the product of an array.

Syntax:

```
array_product(array)
```

Eg: <?php
$a = array(5, 5, 10);
echo (array_product($a));
?>

O/P: 250

7. array_pop()

deletes the last ele. of an array.

Syntax:

```
array_pop(array)
```

Eg: <?php
$a = array("red", "green", "blue");
array_pop($a);
print_r($a);
?>

O/P: Array ([0] => red [1] => green)

8. array_push()

* Insert 1 / more ele. to the end of an array.
* Even if the array has str. keys, the ted ele. will always have numeric keys.

Syntax:

array_push (array, value1, value2, ...)

Eg:

```
<?php
$a = array ("red", "green");
array_push ($a, "blue", "yellow");
print_r($a);
$b = array ("a" => "red", "b" => "green");
array_push ($a, "blue", "yellow");
print_r($b);
?>
```

O/P: Array ([0] => red [1] => green [2] => blue [3] => yellow)

Array ([a] => red [b] => green [0] => blue [1] => yellow)

9. array_reverse()

returns an array in the reverse order.

Syntax:

array_reverse (array, preserve)

Optional. specifies if the func. shld preserve the keys of array / not.

Possible val.

- true
- false

Eg:

```
<?php
$a = array ("a" => "Red", "b" => "Blue", "c" => "Green");
print_r (array_reverse ($a));
print_r (array_reverse ($a, true));
?>
```

O/P: Array ([c] => Green [b] => Blue [c] => Red)

Array ([c] => Green [b] => Blue [c] => Red)

Eg:

```
<?php
$a = array ("Red", "Green", "Blue");
print_r (array_reverse ($a));
print_r (array_reverse ($a, true));
?>
```

O/P: Array ([0] => Blue [1] => Green [2] => Red)

Array ([2] => Blue [1] => Green [0] => Red)

10. count()

returns the no. of ele. in an array.

Syntax:

count (array, mode)

Optional. Possible val.

- 0 - Default. Does not count all ele. of multidimensional arrays.
- 1 - Count the array recursively (counts all the ele. of multidimensional arrays)

Eg:

```
<?php
    $fruit = array ("Apple", "Orange", "Mango");
    echo count ($fruit);
?>
```

O/P: 3

15/3/21 Form

* A document that containing black fields, that the user can fill the data or user can select the data usually the data will store in the database.

* a form is an HTML tag that contains GUI (Graphical user interface) items such as input box, check boxes radio buttons etc.,

The form is defined using the <form> ... </form> tags and GUI items are defined using form elements such as input

Php form Handling.

Php - A simple HTML form:-

```
<html>
<form action = "Welcome.php" method = "post">
Name: <input type = "text" name = "name"> <br>
E-mail: <input type = "text" name = "email"> <br>
<input type = "submit">
</form>
</body>
</html>
```

* The example given above displays a simple HTML form with two input fields and a submit button.

* The Php superglobals $_GET and $_POST are used to collect form-data.

* When the user fills out the form above and clicks the submit button, the form data is sent for processing to a php file named "Welcome.php" the form data is sent with the HTTP post method.

the welcome.php looks like this.

```
<html>
<body>
Welcome <?php echo $_POST["name"];
?><br>
Your email address is: <?php echo $_POST["email"];
?>
</body>
</html>
```

O/P:

Welcome John
Your email address is
John doe@example.com

Accessing /validating the php form:-

* Validation means check the input submitted by the user. There are two types of validation are available in php. They are as follows
* client side validation
* server side validation

client side validation:- Validation is performed on the client machine web browsers.

server side validation - After submitted by data, the data has sent to a server and perform validation checks in server machine.

Validation rules for some fields

Field	Validation Rules
Name	
Email	
Website	
Radio	
checkBox	Must be checked atleast one
Drop down menu	Must be selected atleast one

Insert php code into an HTML file.

→ If we want to insert php code into an HTML file we write the php anywhere you want within the <?php and ?> tags also.

Ex

```
<html>
<title> HTML with php </title>
<body>
<h1> My Example </h1>
<?php
   // our php code
?>
<b> Here is some more HTML </b>
<?php
   // more php code
?>
</body>
</html>
```

Hidden fields to save state:

* We can use a hidden field to keep track of some data. A hidden field behaves exactly the same as a text field, except that the user cannot see it unless she views the HTML source of the document that contains it.

=> A hidden field let web developers includes data that cannot be seen or modified by users when a form is submitted.

=> A hidden field often stores what database record that needs to be updated when the form is submitted

Syntax

<input type="hidden">

< HTML <input> type attribute.

Ex

<input type="hidden" id="custId name="customer Value="348">

How to make a redirect in php?

* Redirection from one page to another in php is commonly achieved by.

Using Header Function in php:-

=> The header() function is an inbuilt function in php which is used to send the raw HTTP header to the client.

Syntax:

header($header, $replace, $http_response_code) die()

Parameters:-

This function accept 3 parameters

1) $header - This parameter is used to hold the header string

2) $replace - This parameter is used to hold the replace parameter which indicate the header should replace a previous similar header, or add a second header of same type. This is optional parameter.

3) $http_response_code: This parameter hold the HTTP response code

Program:

```
<?php
header("Location: http://www.jjj.org");
exit;
?>
```

Important note

The die() (or) exit() function after header is mandatory. If it is not put after header the script may continue resulting unexpected behavior.