The GET Method:

The GET method sends the encoded user information appended to the page request. The page and the encoded information are separated by the? character.

The GET method produces a long string that appears in your server logs, in the browser's Location: box.

The GET method is restricted to send upto 1024 characters only.

Never use GET method if you have password or other sensitive information to be sent to the server.

GET can't be used to send binary data, like images or word documents, to the server

```
<html>
<body>
<form action = "<?php $_PHP_SELF ?>" method = "GET">
Name: <input type = "text" name = "name" />
Age: <input type = "text" name = "age" />
<input type = "submit" />
</form>
</body>
</html>
<?php
if( $_GET["name"] || $_GET["age"] ) {
echo "Welcome ". $_GET['name']. "<br />";
echo "You are ". $_GET['age']. " years old.";
exit();
}
?>
```

The POST Method:

The POST method does not have any restriction on data size to be sent.

The POST method can be used to send ASCII as well as binary data.

The data sent by POST method goes through HTTP header so security depends on HTTP protocol. By using Secure HTTP you can make sure that your information is secure.

The PHP provides **$_POST** associative array to access all the sent information using POST method.

```
<?php
if( $_POST["name"] || $_POST["age"] ) {
if (preg_match("/[^A-Za-z'-]/",$_POST['name'] )) {
die ("invalid name and name should be alpha");
} echo "Welcome ". $_POST['name']. "<br />";
echo "You are ". $_POST['age']. " years old.";
exit();
}
?>
<html>
<body>
<form action = "<?php $_PHP_SELF ?>" method = "POST">
Name: <input type = "text" name = "name" />
Age: <input type = "text" name = "age" />
<input type = "submit" />
</form>
</body>
</html>
```

Basic PHP String Functions

Strlen(string) displays the length of any string

Strrev() is used for reversing a string. You can use this function to get the reverse version of any string.

Strpos() enables searching particular text within a string. It works simply by matching the specific text in a string

Str_replace(string to be replaced,text,string) is a built-in function, basically used for replacing specific text within a string.

Ucwords() is used to convert first alphabet of every word into uppercase.

Strtoupper() is used to convert a whole string to uppercase

Strtolower() is used to convert a string into lowercase

ARRAYS IN PHP

An array is a data structure that stores one or more similar type of values in a single value. For example, if you want to store 100 numbers then instead of defining 100 variables it's easy to define an array of 100 length.

There are three different kind of arrays and each array value is accessed using an ID c which is called array index.

Numeric array − An array with a numeric index. Values are stored and accessed in linear fashion.

Associative array − An array with strings as index. This store element values in association with key values rather than in a strict linear index order.

Multidimensional array − An array containing one or more arrays and values are accessed using multiple indices

Numeric Array:

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

Example

Following is the example showing how to create and access numeric arrays. Here we have used array() function to create array.

```
<html>

<body>

<?php

/* First method to create array. */

$numbers = array( 1, 2, 3, 4, 5);

foreach( $numbers as $value ) {

echo "Value is $value <br />";

}

/* 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 />";

}

?>
```

</body>

</html>

Output:

Value is 1

Value is 2

Value is 3

Value is 4

Value is 5

Value is one

Value is two

Value is three

Value is four

Value is five

Associative Arrays:

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.

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

Example:

<html>

<body>

<?php

/* First method to associate create array. */

$salaries = array("Rahul" => 2000, "qadir" => 1000, "zara" => 500);

echo "Salary of Rahul is ". $salaries['Rahul'] . "
";

echo "Salary of qadir is ". $salaries['qadir']. "
";

echo "Salary of zara is ". $salaries['zara']. "
";

/* Second method to create array. */

$salaries['Rahul'] = "high";

$salaries['qadir'] = "medium";

```
$salaries['zara'] = "low";
echo "Salary of Rahul is ". $salaries['Rahul'] . "<br />";
echo "Salary of qadir is ". $salaries['qadir']. "<br />";
echo "Salary of zara is ". $salaries['zara']. "<br />";
?>
</body>
</html>
```

Output:

Salary of Rahul is 2000

Salary of qadir is 1000

Salary of zara is 500

Salary of Rahul is high

Salary of qadir is medium

Salary of zara is low

Multidimensional Arrays:

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

```
<html>
<body>
<?php
$marks = array(
"Rahul" => array (
"physics" => 35,
"maths" => 30,
"chemistry" => 39
),
"qadir" => array (
"physics" => 30,
"maths" => 32,
```

```
"chemistry" => 29
),
"zara" => array (
"physics" => 31,
"maths" => 22,
"chemistry" => 39
)
);
/* Accessing multi-dimensional array values */
echo "Marks for Rahul in physics : " ;
echo $marks['Rahul']['physics'] . "<br />";
echo "Marks for qadir in maths : ";
echo $marks['qadir']['maths'] . "<br />";
echo "Marks for zara in chemistry : " ;
echo $marks['zara']['chemistry'] . "<br />";
?>
</body>
</html>
```

Output:

Marks for Rahul in physics : 35

Marks for qadir in maths : 32

Marks for zara in chemistry : 39