PHP Array Functions

PHP Array Functions allow you to interact with and manipulate arrays in various ways. PHP arrays are essential for storing, managing, and operating on sets of variables.

PHP supports simple and multi-dimensional arrays and may be either user created or created by another function.

Installation

There is no installation needed to use PHP array functions; they are part of the PHP core and comes alongwith standard PHP installation.

Runtime Configuration

This extension has no configuration directives defined in php.ini.

> Explore our latest online courses and learn new skills at your own pace. Enroll and become a certified expert to boost your career.

PHP Array Functions

Following table lists down all the functions related to PHP Array. Here column version indicates the earliest version of PHP that supports the function.

Sr.No	Function & Description	Version
1	**array()** Create an array	4.2.0
2	**array_change_key_case()** Returns an array with all keys in lowercase or uppercase	4.2.0
3	**array_chunk()** Splits an array into chunks of arrays	4.2.0
3	**array_column()** Return the values from a single column in the input array	5.5.0
4	**array_combine()**	5

	Creates an array by using one array for keys and another for its values	5
5	**array_count_values()** Returns an array with the number of occurrences for each value	4
6	**array_diff()** Compares array values, and returns the differences	4
7	**array_diff_assoc()** Compares array keys and values, and returns the differences	4
8	**array_diff_key()** Compares array keys, and returns the differences	5
9	**array_diff_uassoc()** Compares array keys and values, with an additional user-made function check, and returns the differences	5
10	**array_diff_ukey()** Compares array keys, with an additional user-made function check, and returns the differences	5
11	**array_fill()** Fills an array with values	4
12	**array_fill_keys()** Fill an array with values, specifying keys	5
13	**array_filter()** Filters elements of an array using a user-made function	4
14	**array_flip()** Exchanges all keys with their associated values in an array	4
15	**array_intersect()** Compares array values, and returns the matches	4
16	**array_intersect_assoc()** Compares array keys and values, and returns the matches	4
17	**array_intersect_key()** Compares array keys, and returns the matches	5
18	**array_intersect_uassoc()** Compares array keys and values, with an additional user-made function check, and returns the matches	5
19	**array_intersect_ukey()**	5

	Compares array keys, with an additional user-made function check, and returns the matches	
20	**array_key_exists()** Checks if the specified key exists in the array	4
21	**array_keys()** Returns all the keys of an array	4
22	**array_map()** Sends each value of an array to a user-made function, which returns new values	4
23	**array_merge()** Merges one or more arrays into one array	4
24	**array_merge_recursive()** Merges one or more arrays into one array	4
25	**array_multisort()** Sorts multiple or multi-dimensional arrays	4
26	**array_pad()** Inserts a specified number of items, with a specified value, to an array	4
27	**array_pop()** Deletes the last element of an array	4
28	**array_product()** Calculates the product of the values in an array	5
29	**array_push()** Inserts one or more elements to the end of an array	4
30	**array_rand()** Returns one or more random keys from an array	4
31	**array_reduce()** Returns an array as a string, using a user-defined function	4
32	**array_reverse()** Returns an array in the reverse order	4
33	**array_search()** Searches an array for a given value and returns the key	4
34	**array_shift()**	4

	Removes the first element from an array, and returns the value of the removed element	
35	**array_slice()** Returns selected parts of an array	4
36	**array_splice()** Removes and replaces specified elements of an array	4
37	**array_sum()** Returns the sum of the values in an array	4
38	**array_udiff()** Compares array values in a user-made function and returns an array	5
39	**array_udiff_assoc()** Compares array keys, and compares array values in a user-made function, and returns an array	5
40	**array_udiff_uassoc()** Compares array keys and array values in user-made functions, and returns an array	5
41	**array_uintersect()** Compares array values in a user-made function and returns an array	5
42	**array_uintersect_assoc()** Compares array keys, and compares array values in a user-made function, and returns an array	5
43	**array_uintersect_uassoc()** Compares array keys and array values in user-made functions, and returns an array	5
44	**array_unique()** Removes duplicate values from an array	4
45	**array_unshift()** Adds one or more elements to the beginning of an array	4
46	**array_values()** Returns all the values of an array	4
47	**array_walk()** Applies a user function to every member of an array	3
48	**array_walk_recursive()** Applies a user function recursively to every member of an array	5

49	**arsort()** Sorts an array in reverse order and maintain index association	3
50	**asort()** Sorts an array and maintain index association	3
51	**compact()** Create array containing variables and their values	4
52	**count()** Counts elements in an array, or properties in an object	3
53	**current()** Returns the current element in an array	3
54	**each()** Returns the current key and value pair from an array	3
55	**end()** Sets the internal pointer of an array to its last element	3
56	**extract()** Imports variables into the current symbol table from an array	3
57	**in_array()** Checks if a specified value exists in an array	4
58	**key()** Fetches a key from an array	3
59	**krsort()** Sorts an array by key in reverse order	3
60	**ksort()** Sorts an array by key	3
61	**list()** Assigns variables as if they were an array	3
62	**natcasesort()** Sorts an array using a case insensitive "natural order" algorithm	4
63	**natsort()** Sorts an array using a "natural order" algorithm	4
64	**next()** Advance the internal array pointer of an array	3

65	**pos()** Alias of current()	3
66	**prev()** Rewinds the internal array pointer	3
67	**range()** Creates an array containing a range of elements	3
68	**reset()** Sets the internal pointer of an array to its first element	3
69	**rsort()** Sorts an array in reverse order	3
70	**shuffle()** Shuffles an array	3
71	**sizeof()** Alias of count()	3
72	**sort()** Sorts an array	3
73	**uasort()** Sorts an array with a user-defined function and maintain index association	3
74	**uksort()** Sorts an array by keys using a user-defined function	3
75	**usort()** Sorts an array by values using a user-defined function	3

PHP Array Constants

Sr.No	Constant & Description
1	**CASE_LOWER** Used with array_change_key_case() to convert array keys to lower case
2	**CASE_UPPER** Used with array_change_key_case() to convert array keys to upper case
3	**SORT_ASC** Used with array_multisort() to sort in ascending order

4	**SORT_DESC** Used with array_multisort() to sort in descending order
5	**SORT_REGULAR** Used to compare items normally
6	**SORT_NUMERIC** Used to compare items numerically
7	**SORT_STRING** Used to compare items as strings
8	**SORT_LOCALE_STRING** Used to compare items as strings, based on the current locale
9	**COUNT_NORMAL**
10	**COUNT_RECURSIVE**
11	**EXTR_OVERWRITE**
12	**EXTR_SKIP**
13	**EXTR_PREFIX_SAME**
14	**EXTR_PREFIX_ALL**
15	**EXTR_PREFIX_INVALID**
16	**EXTR_PREFIX_IF_EXISTS**
17	**EXTR_IF_EXISTS**
18	**EXTR_REFS**