PHP 5 Array Functions

The following section contains a list of useful PHP array functions.

PHP Array Functions

The following array functions are the part of the PHP core so you can use these functions within your script without any further installation

Function	Description
`array()`	Create an array
`array_change_key_case()`	Changes the case of all keys in an array (either lowercase or uppercase)
`array_chunk()`	Split an array into chunks of arrays
`array_column()`	Return the values from a single column in the input array
`array_combine()`	Creates an array by using one array for keys and another for its values
`array_count_values()`	Counts all the values of an array
`array_diff()`	Compare arrays values, and returns the differences
`array_diff_assoc()`	Compare arrays keys and values, and returns the differences
`array_diff_key()`	Compare arrays keys, and returns the differences
`array_diff_uassoc()`	Compare arrays keys and values, using a user-defined key comparison function, and returns the differences
`array_diff_ukey()`	Compare array keys, using a user-defined key comparison function, and returns the differences
`array_fill()`	Fill an array with values
`array_fill_keys()`	Fill an array with values, specifying keys
`array_filter()`	Filters elements of an array using a user-defined function
`array_flip()`	Flips or Exchanges all keys with their associated values in an array
`array_intersect()`	Compare arrays values, and returns the matches
`array_intersect_assoc()`	Compare arrays keys and values, and returns the matches
`array_intersect_key()`	Compare arrays keys, and returns the matches
`array_intersect_uassoc()`	Compare arrays keys and values, using a user-defined key comparison function, and returns the matches

Function	Description
`array_intersect_ukey()`	Compare arrays keys, using a user-defined key comparison function, and returns the matches
`array_keys()`	Return all the keys or a subset of the keys of an array
`array_key_exists()`	Checks if the specified key exists in the array
`array_map()`	Sends the elements of the given arrays to a user-defined function, which may use it to returns new values
`array_merge()`	Merges one or more arrays into one array
`array_merge_recursive()`	Merges one or more arrays into one array recursively
`array_multisort()`	Sorts multiple or multi-dimensional arrays
`array_pad()`	Inserts a specified number of items, with a specified value, to an array
`array_pop()`	Removes the last element of an array, and returns the value of the removed element
`array_product()`	Calculates the product of the values in an array
`array_push()`	Inserts one or more elements to the end of an array
`array_rand()`	Returns one or more random keys from an array
`array_reduce()`	Reduce the array to a single value by using a user-defined callback function
`array_replace()`	Replaces the values of the first array with the values from following arrays
`array_replace_recursive()`	Replaces the values of the first array with the values from following arrays recursively
`array_reverse()`	Return an array with elements in reverse order
`array_search()`	Searches an array for a given value and returns the corresponding key if successful
`array_shift()`	Removes the first element from an array, and returns the value of the removed element
`array_slice()`	Extract a slice from an array
`array_splice()`	Remove a portion of the array and replace it with something else
`array_sum()`	Calculate the sum of values in an array
`array_udiff()`	Compares only arrays values by using a user-defined comparison callback function, and returns the differences
`array_udiff_assoc()`	Compares arrays values by using a user-defined comparison callback function, with additional keys comparison using an internal (or built-in) function, and returns the differences
`array_udiff_uassoc()`	Compares arrays keys and values by using two separate user-defined comparison callback functions, and returns the differences

Function	Description
`array_uintersect()`	Compares only arrays values by using a user-defined comparison callback function, and returns the matches
`array_uintersect_assoc()`	Compares arrays values by using a user-defined comparison callback function, while uses an internal (or built-in) function for comparing the key, and returns the matches
`array_uintersect_uassoc()`	Compares arrays keys and values by using two separate user-defined comparison callback functions, and returns the matches
`array_unique()`	Removes duplicate values from an array
`array_unshift()`	Adds one or more elements to the beginning of an array
`array_values()`	Return all the values of an array
`array_walk()`	Applies a user-defined function to each element of an array
`array_walk_recursive()`	Applies a user-defined function recursively to each element of an array
`asort()`	Sort an associative array by value, in ascending order
`arsort()`	Sort an associative array by value, in reverse or descending order
`compact()`	Create array containing variables and their values
`count()`	Count all elements in an array
`current()`	Return the current element in an array
`each()`	Return the current key and value pair from an array and advance the array cursor
`end()`	Sets the internal pointer of an array to its last element
`extract()`	Import variables into the current symbol table from an array
`in_array()`	Checks if a value exists in an array
`key_exists()`	Checks if the specified key exists in the array. Alias of `array_key_exists()`
`key()`	Fetches a key from an array
`ksort()`	Sort an associative array by key, in ascending order
`krsort()`	Sort an associative array by key, in reverse or descending order
`list()`	Assign variables as if they were an array
`natcasesort()`	Sort an array using a case insensitive "natural order" algorithm
`natsort()`	Sort an array using a "natural order" algorithm
`next()`	Advance the internal array pointer of an array
`pos()`	Return the current element in an array. Alias of `current()`
`prev()`	Rewind the internal array pointer

Function	Description
`range()`	Create an array containing a range of elements
`reset()`	Set the internal pointer of an array to its first element
`rsort()`	Sort an array in reverse or descending order
`shuffle()`	Shuffle an array
`sizeof()`	Count all elements in an array. Alias of `count()`
`sort()`	Sort an array in ascending order
`uasort()`	Sort an array using a user-defined comparison function and maintain index association
`uksort()`	Sort an array by keys using a user-defined comparison function
`usort()`	Sort an array by values using a user-defined comparison function