PHP

PHP Array Functions

PHP Array Functions [1]

Refer Part I Array handling functions also

- 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

array_push() [1]

- The array_push() function inserts one or more elements to the end of an array.
- **Tip:** You can add one value, or as many as you like.
- **Note:** Even if your array has string keys, your added elements will always have numeric keys.

Example 1

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

Parameter	Description
array	Required. Specifies an array
value1	Required. Specifies the value to add
value2	Optional. Specifies the value to add

Return Value: Returns the new number of elements in the array

array_pop() [1]

- The array_pop() function deletes the last element of an array.
- **Example 1**

Syntax

```
array_pop(array)
```

Parameter	Description
array	Required. Specifies an array

Technical Details

Return Value:	Returns the last value of *array*. If *array* is empty, or is not an array, NULL will be returned.
PHP Version:	4+

array_shift() [1]

- The array_shift() function removes the first element from an array, and returns the value of the removed element.
- **Note:** If the keys are numeric, all elements will get new keys, starting from 0 and increases by 1. **Example 1**

Syntax

```
array_shift(array)
```

Parameter	Description
array	Required. Specifies an array

Technical Details

Return Value:	Returns the value of the removed element from an array, or NULL if the array is empty
PHP Version:	4+

array_unshift() [1]

- The array_unshift() function inserts new elements to an array. The new array values will be inserted in the beginning of the array.
- **Tip:** You can add one value, or as many as you like.
- **Note:** Numeric keys will start at 0 and increase by 1. String keys will remain the same.

Example 1

Syntax

```
array_unshift(array,value1,value2,value3...)
```

Parameter	Description
array	Required. Specifying an array
value1	Required. Specifies a value to insert
value2	Optional. Specifies a value to insert
value3	Optional. Specifies a value to insert

Return Value:	Returns the new number of elements in the array
PHP Version:	4+

array_column [1]

- Example 2

The array_column() function returns the values from a single column in the input array.

Syntax

```
array_column(array,column_key,index_key);
```

Parameter	Description
array	Required. Specifies the multi-dimensional array (record-set) to use
column_key	Required. An integer key or a string key name of the column of values to return. This parameter can also be NULL to return complete arrays (useful together with index_key to re-index the array)
index_key	Optional. The column to use as the index/keys for the returned array

array_combine [1]

- The array_combine() function creates an array by using the elements from one "keys" array and one "values" array.
- **Note:** Both arrays must have equal number of elements!
- **Example 3**

array_combine [1]

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

Parameter	Description
keys	Required. Array of keys
values	Required. Array of values

Technical Details

Return Value:	Returns the combined array. FALSE if number of elements does not match
PHP Version:	5+

array_count_values() [1]

- **Example 3**

Syntax

```
array_count_values(array)
```

Parameter	Description
array	Required. Specifying the array to count values of

Technical Details

Return Value:	Returns an associative array, where the keys are the original array's values, and the values are the number of occurrences
PHP Version:	4+

array_diff() [1]

- The array_diff() function compares **the values** of two (or more) arrays, and returns the differences.
- This function compares the values of two (or more) arrays, and return an array that contains the entries from *array1* that are not present in *array2* or *array3*, etc.
- **Example 4**

array_diff() [1]

Syntax

```
array_diff(array1,array2,array3...);
```

Parameter	Description
array1	Required. The array to compare from
array2	Required. An array to compare against
array3,...	Optional. More arrays to compare against

Technical Details

Return Value:	Returns an array containing the entries from *array1* that are not present in any of the other arrays
PHP Version:	4.0.1+

array_diff_assoc() [1]

- The array_diff_assoc() function compares **the keys and values** of two (or more) arrays, and returns the differences.
- This function compares the keys and values of two (or more) arrays, and return an array that contains the entries from *array1* that are not present in *array2* or *array3*, etc.
- **Example 4**

array_diff_assoc() [1]

Syntax

```
array_diff_assoc(array1,array2,array3...);
```

Parameter	Description
array1	Required. The array to compare from
array2	Required. An array to compare against
array3,...	Optional. More arrays to compare against

Technical Details

Return Value:	Returns an array containing the entries from *array1* that are not present in any of the other arrays
PHP Version:	4.3+

array_diff_key() [1]

- The array_diff_key() function compares **the keys** of two (or more) arrays, and returns the differences.
- This function compares the keys of two (or more) arrays, and return an array that contains the entries from *array1* that are not present in *array2* or *array3*, etc.
- **Example 4**

array_diff_key() [1]

Syntax

```
array_diff_key(array1,array2,array3...);
```

Parameter	Description
array1	Required. The array to compare from
array2	Required. An array to compare against
array3,...	Optional. More arrays to compare against

Technical Details

Return Value:	Returns an array containing the entries from *array1* that are not present in any of the other arrays
PHP Version:	5.1+

array_fill() [1]

- **Example 5**

The array_fill() function fills an array with values.

Syntax

```
array_fill(index,number,value);
```

Parameter	Description
index	Required. The first index of the returned array
number	Required. Specifies the number of elements to insert
value	Required. Specifies the value to use for filling the array

Technical Details

Return Value:	Returns the filled array

array_merge() [1]

- The array_merge() function merges one or more arrays into one array.
- **Tip:** You can assign one array to the function, or as many as you like.
- **Note:** If two or more array elements have the same key, the last one overrides the others.
- **Note:** If you assign only one array to the array_merge() function, and the keys are integers, the function returns a new array with integer keys starting at 0 and increases by 1 for each value.
- **Tip:** The difference between this function and the array_merge_recursive() function is when two or more array elements have the same key. Instead of override the keys, the array_merge_recursive() function makes the value as an array.
- **Example 6**

array_merge() [1]

Syntax

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

Parameter	Description
array1	Required. Specifies an array
array2	Optional. Specifies an array
array3,...	Optional. Specifies an array

array_merge_recursive() [1]

- The array_merge_recursive() function merges one or more arrays into one array.
- The difference between this function and the array_merge() function is when two or more array elements have the same key. Instead of override the keys, the array_merge_recursive() function makes the value as an array.
- Note: If you assign only one array to the array_merge_recursive() function, it will behave exactly the same as the array_merge() function.
- **Example 6**

array_merge_recursive() [1]

Syntax

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

Parameter	Description
array1	Required. Specifies an array
array2	Optional. Specifies an array
array3,...	Optional. Specifies an array

array_replace() [1]

- The array_replace() function replaces the values of the first array with the values from following arrays.
- **Tip:** You can assign one array to the function, or as many as you like.
- If a key from array1 exists in array2, values from array1 will be replaced by the values from array2. If the key only exists in array1, it will be left as it is.
- If a key exist in array2 and not in array1, it will be created in array1.
- If multiple arrays are used, values from later arrays will overwrite the previous ones.
- **Tip:** Use array_replace_recursive() to replace the values of array1 with the values from following arrays recursively.
- **Example 7**

array_replace() [1]

Syntax

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

Parameter	Description
array1	Required. Specifies an array
array2	Optional. Specifies an array which will replace the values of *array1*
array3,...	Optional. Specifies more arrays to replace the values of *array1* and *array2, etc.* Values from later arrays will overwrite the previous ones.

array_product () [1]

- **Example 9**

The array_product() function calculates and returns the product of an array.

Syntax

```
array_product(array)
```

Parameter	Description
array	Required. Specifies an array

Technical Details

Return Value:	Returns the product as an integer or float

array_reverse() [1]

- **Example 9**

The array_reverse() function returns an array in the reverse order.

Syntax

```
array_reverse(array,preserve)
```

Parameter	Description
array	Required. Specifies an array
preserve	Optional. Specifies if the function should preserve the keys of the array or not. Possible values: • true • false

array_slice() [1]

- The array_slice() function returns selected parts of an array.
- **Note:** If the array have string keys, the returned array will always preserve the keys
- **Example 11**

array_slice() [1]

```
array_slice(array,start,length,preserve)
```

Parameter	Description
array	Required. Specifies an array
start	Required. Numeric value. Specifies where the function will start the slice. 0 = the first element. If this value is set to a negative number, the function will start slicing that far from the last element. -2 means start at the second last element of the array.
length	Optional. Numeric value. Specifies the length of the returned array. If this value is set to a negative number, the function will stop slicing that far from the last element. If this value is not set, the function will return all elements, starting from the position set by the start-parameter.
preserve	Optional. Specifies if the function should preserve or reset the keys. Possible values: • true - Preserve keys • false - Default. Reset keys

array_splice() [1]

- The array_splice() function removes selected elements from an array and replaces it with new elements. The function also returns an array with the removed elements.
- Tip: If the function does not remove any elements (length=0), the replaced array will be inserted from the position of the start parameter.
- Note: The keys in the replaced array are not preserved.
- **Example 10**

array_splice() [1]

Syntax

```
array_splice(array,start,length,array)
```

Parameter	Description
array	Required. Specifies an array
start	Required. Numeric value. Specifies where the function will start removing elements. 0 = the first element. If this value is set to a negative number, the function will start that far from the last element. -2 means start at the second last element of the array.
length	Optional. Numeric value. Specifies how many elements will be removed, and also length of the returned array. If this value is set to a negative number, the function will stop that far from the last element. If this value is not set, the function will remove all elements, starting from the position set by the start-parameter.
array	Optional. Specifies an array with the elements that will be inserted to the original array. If it's only one element, it can be a string, and does not have to be an array.

array_unique() [1]

- The array_unique() function removes duplicate values from an array. If two or more array values are the same, the first appearance will be kept and the other will be removed.
- **Note:** The returned array will keep the **first** array item's key type.
- **Example 12**

array_unique() [1]

Syntax

```
array_unique(array)
```

Parameter	Description
array	Required. Specifying an array

in_array() [1]

- The in_array() function searches an array for a specific value.
- **Note:** If the search parameter is a string and the type parameter is set to TRUE, the search is case-sensitive.
- **Example 12**

in_array() [1]

Syntax

```
in_array(search,array,type)
```

Parameter	Description
search	Required. Specifies the what to search for
array	Required. Specifies the array to search
type	Optional. If this parameter is set to TRUE, the in_array() function searches for the search-string and specific type in the array.

list() [1]

- The list() function is used to assign values to a list of variables in one operation.
- **Note:** This function only works on numerical arrays.
- **Example 12**

Syntax

```
list(var1,var2...)
```

Parameter	Description
var1	Required. The first variable to assign a value to
var2,...	Optional. More variables to assign values to

array_change_key_case() [1]

- **Example 13**

The array_change_key_case() function changes all keys in an array to lowercase or uppercase.

Syntax

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

Parameter	Description
array	Required. Specifies the array to use
case	Optional. Possible values: • CASE_LOWER - Default value. Changes the keys to lowercase • CASE_UPPER - Changes the keys to uppercase

References

1. https://www.w3schools.com/php/

Thank you….