PHP Sorting Arrays

In this tutorial you will learn how to sort the elements or keys of an array in ascending or descending order in PHP.

PHP Functions For Sorting Arrays

In the previous chapter you've learnt the essentials of PHP arrays i.e. what arrays are, how to create them, how to view their structure, how to access their elements etc. You can do even more things with arrays like sorting the elements in any order you like.

PHP comes with a number of built-in functions designed specifically for sorting array elements in different ways like alphabetically or numerically in ascending or descending order. Here we'll explore some of these functions most commonly used for sorting arrays.

- `sort()` and `rsort()` — For sorting indexed arrays
- `asort()` and `arsort()` — For sorting associative arrays by value
- `ksort()` and `krsort()` — For sorting associative arrays by key

Sorting Indexed Arrays in Ascending Order

The `sort()` function is used for sorting the elements of the indexed array in ascending order (alphabetically for letters and numerically for numbers).

Example **Run this code »**

```php
<?php
// Define array
$colors = array("Red", "Green", "Blue", "Yellow");

// Sorting and printing array
sort($colors);
print_r($colors);
?>
```

This `print_r()` statement gives the following output:

```
Array ( [0] => Blue [1] => Green [2] => Red [3] => Yellow )
```

Similarly you can sort the numeric elements of the array in ascending order.

Example **Run this code »**

```php
<?php
// Define array
$numbers = array(1, 2, 2.5, 4, 7, 10);

// Sorting and printing array
sort($numbers);
print_r($numbers);
?>
```

This `print_r()` statement gives the following output:

```
Array ( [0] => 1 [1] => 2 [2] => 2.5 [3] => 4 [4] => 7 [5] => 10 )
```

Sorting Indexed Arrays in Descending Order

The `rsort()` function is used for sorting the elements of the indexed array in descending order (alphabetically for letters and numerically for numbers).

Example **Run this code »**

```php
<?php
// Define array
$colors = array("Red", "Green", "Blue", "Yellow");

// Sorting and printing array
rsort($colors);
print_r($colors);
?>
```

This `print_r()` statement gives the following output:

```
Array ( [0] => Yellow [1] => Red [2] => Green [3] => Blue )
```

Similarly you can sort the numeric elements of the array in descending order.

Example **Run this code »**

```php
<?php
// Define array
$numbers = array(1, 2, 2.5, 4, 7, 10);

// Sorting and printing array
rsort($numbers);
```

```php
print_r($numbers);
?>
```

This `print_r()` statement gives the following output:

```
Array ( [0] => 10 [1] => 7 [2] => 4 [3] => 2.5 [4] => 2 [5] => 1 )
```

Sorting Associative Arrays in Ascending Order By Value

The `asort()` function sorts the elements of an associative array in ascending order according to the value. It works just like `sort()`, but it preserves the association between keys and its values while sorting.

Example **Run this code »**

```php
<?php
// Define array
$age = array("Peter"=>20, "Harry"=>14, "John"=>45, "Clark"=>35);

// Sorting array by value and print
asort($age);
print_r($age);
?>
```

This `print_r()` statement gives the following output:

```
Array ( [Harry] => 14 [Peter] => 20 [Clark] => 35 [John] => 45 )
```

Sorting Associative Arrays in Descending Order By Value

The `arsort()` function sorts the elements of an associative array in descending order according to the value. It works just like `rsort()`, but it preserves the association between keys and its values while sorting.

Example **Run this code »**

```php
<?php
// Define array
$age = array("Peter"=>20, "Harry"=>14, "John"=>45, "Clark"=>35);
```

```php
// Sorting array by value and print
arsort($age);
print_r($age);
?>
```

This `print_r()` statement gives the following output:

```
Array ( [John] => 45 [Clark] => 35 [Peter] => 20 [Harry] => 14 )
```

Sorting Associative Arrays in Ascending Order By Key

The `ksort()` function sorts the elements of an associative array in ascending order by their keys. It preserves the association between keys and its values while sorting, same as `asort()` function.

Example **Run this code »**

```php
<?php
// Define array
$age = array("Peter"=>20, "Harry"=>14, "John"=>45, "Clark"=>35);

// Sorting array by key and print
ksort($age);
print_r($age);
?>
```

This `print_r()` statement gives the following output:

```
Array ( [Clark] => 35 [Harry] => 14 [John] => 45 [Peter] => 20 )
```

Sorting Associative Arrays in Descending Order By Key

The `krsort()` function sorts the elements of an associative array in descending order by their keys. It preserves the association between keys and its values while sorting, same as `arsort()` function.

Example **Run this code »**

```php
<?php
// Define array
$age = array("Peter"=>20, "Harry"=>14, "John"=>45, "Clark"=>35);
```

```
// Sorting array by key and print
krsort($age);
print_r($age);
?>
```

This `print_r()` statement gives the following output:

```
Array ( [Peter] => 20 [John] => 45 [Harry] => 14 [Clark] => 35 )
```