■PHP配列関連の関数一覧

作成:2015.8.18 yoshi

	配列関連関数	機能説明	関数利用例
		配列の編集	
1	array	配列を生成する	$ary = array("a" => "orange", "b" => "banana");
2	array_pop	配列の末尾から要素を取り除く	$str = array_pop($ary);
3	array_push	一つ以上の要素を配列の末尾に追加する	array_push($ary,$str1); //$ary[ ] = $str1; と同じ操作
4	array_unshift	一つ以上の要素を配列の先頭に加える	array_unshift($ary,$str1);
5	array_shift	配列の先頭から要素を一つ取り出す	$str = array_shift($ary);
6	array_slice	配列の一部を展開する	$aryOut = array_slice($ary, 2, 2);
7	array_splice	配列の一部を削除し、他の要素で置換する	array_splice($aryFruits,5,0,array("オレンジ"));
8	array_replace	渡された配列の要素を置き換える(PHP5.3以降)	$aryOut = array_replace($ary,array(1=>"チェリー"));
9	array_replace_recursive	最初の配列の値をそれ以降の配列の同じ要素の値で再帰的に置き換える(PHP5.3以降)	$aryOut = array_replace_recursive($aryFruits,$aryFruits2);
10	array_merge	ひとつまたは複数の配列をマージする	$aryOut = array_merge($aryFruits,$aryFruits2);
11	array_merge_recursive	一つ以上の配列の要素をマージし、前の配列の最後にもう一つの配列の値を追加します。マージした後の配列を返します。二つ以上の配列を再帰的にマージする	$aryOut = array_merge_recursive($ary1, $ary2);
12	array_flip	配列のキーと値を反転する	$aryOut = array_flip($ary);
13	array_combine	一方の配列をキーとし、もう一方の配列を値として、ひとつの配列を生成する	$aryOut = array_combine($ary1,$ary2);
14	array_chunk	配列を指定数の要素で分割する	$aryOut = array_chunk($aryFruits2,2);
15	array_reverse	要素を逆順にした配列を返す	$aryOut = array_reverse($aryFruits);
16	array_keys	配列のキーすべて、あるいはその一部を返す	$aryOut = array_keys($aryFruits);
17	array_values	配列の全ての値を返す	$aryOut = array_values($aryFruits);
18	array_search	指定した値を配列で検索し、見つかった場合に対応するキーを返す	$aryOut = array_search("レモン",$aryFruits);
19	array_map	指定した配列の要素にコールバック関数を適用する	$aryOut = array_map("concate",$ary); function concate($str) { return ($str.'_key'); }
20	list	配列と同様の形式で、複数の変数への代入を行う	list($aryL[0],$aryL[1],,$aryL[3]) = $ary; //配列の添字付きに代入
21	count	変数に含まれるすべての要素、又はオブジェクトに含まれる数を数える	$int = count($ary);
22	sizeof	count のエイリアス	$int = sizeof($ary);
		配列のソート	
23	sort	配列をソートする	sort($aryFkeys);
24	rsort	配列を逆順にソートする	rsort($aryRkeys);
25	ksort	配列をキーでソートする	ksort($aryFruits);
26	krsort	配列をキーで逆順にソートする	krsort($aryFruits);
27	asort	連想キー(添字)と要素との関係を維持しつつ配列をソートする	asort($aryFruits);

28	arsort	連想キー(添字)と要素との関係を維持しつつ配列を逆順にソートする	arsort($aryFruits);
29	natsort	自然順アルゴリズムで配列をソートする	natsort($aryFiles);
30	natcasesort	大文字小文字を区別しない"自然順"アルゴリズムを用いて配列をソートする	natcasesort($aryFiles);
31	array_multisort	複数の多次元配列をソート。連想配列キーは不変、数値添字は再度振り直し	array_multisort($ary1, SORT_DESC, $ary2);
32	usort	ユーザー定義の比較関数を使用して、配列を値(値が小数点のときは内部的に integer にキャスト)でソートする。ユーザー定義の比較関数は、最初の引数が二番目の引数より大きい場合は正の数を、等しい場合はゼロを、小さい場合は負の数を返す必要あり。	usort($ary, "cmp"); function cmp($a,$b) { return ($a == $b)? 0 : (($a < $b)? -1 : 1); }
33	uksort	ユーザー定義の比較関数を用いて、キーで配列をソートする。ユーザー定義の比較関数は、usort と同じ。	uksort($ary, "cmp"); function cmp($a,$b) { return ($a == $b)? 0 : (($a < $b)? -1 : 1); }
34	uasort	ユーザー定義の比較関数で連想配列をソートし、連想インデックスを保持する。ユーザー定義の比較関数は、usort と同じ。	uasort($ary, "cmp"); function cmp($a,$b) { return ($a == $b)? 0 : (($a < $b)? -1 : 1); }
配列のステータス情報			
35	array_key_exists	指定したキーまたは添字が配列にあるかどうかを調べる	if(array_key_exists('apple',$aryFruits)) { … }
36	key_exists	array_key_exists のエイリアス	if(key_exists('apple',$aryFruits)) { … }
37	in_array	配列に値があるかチェックする	if(in_array('りんご',$aryFruits)) { … }
配列のループ制御			
38	each	配列から現在のキーと値のペアを返して、カーソルを進める	while (list($key, $value) = each($aryFruits)) { … }
39	key	現在の配列位置における連想配列要素のキーを返す。ポインタ移動なし。	$idx = key($aryFruits);
40	reset	配列の内部ポインタを先頭の要素にセットする	reset($aryFruits);
41	current	配列内の現在の要素を返す	current($aryFruits);
42	pos	current のエイリアス	pos($aryFruits);
43	end	配列の内部ポインタを最終要素にセットする	end($aryFruits);
44	next	内部配列ポインタを進める	next($aryFruits);
45	prev	内部の配列ポインタをひとつ前に戻す	prev($aryFruits);
配列の要素編集			
46	array_unique	配列から重複した値を削除する	$aryOut = array_unique($ary);
47	array_fill	配列を指定した値で埋める。(開始添字、挿入要素数、値)の指定	$aryOut = array_fill(2, 4, 'リンゴ');
48	array_fill_keys	キーとなる配列を指定して、配列を値で埋める	$aryOut = array_fill_keys($aryKey, 'リンゴ');
49	array_reduce	配列の各要素にコールバック関数を用いて反復処理し、配列を一つの値に減らす	$int = array_reduce($ary, "call_func", 10); function call_func($carry, $item) { return $carry+$item; }
50	array_pad	配列要素の指定数で不足時、追加要素を指定値で埋める	$aryOut = array_pad($ary,12,'null');
51	array_filter	コールバック関数を使用して、配列の要素をフィルタリングする	$aryOut = array_filter($ary,"odd"); function odd($str){return($str & 1);}
52	array_column	配列の指定キー(カラム)の値を返す(オプションで返す配列のキーカラムの指定)… DBのレコード処理に適す(PHP5.5以降)	$aryOut = array_column($ary,'name','id');
53	array_count_values	配列の値をキーとし、その値の出現回数を値とした連想配列を返します	$aryOut = array_count_values($ary);

54	array_change_key_case	配列のすべてのキーの大文字小文字を変更する	$aryOut = array_change_key_case($ary,CASE_UPPER);
55	array_walk	配列の全ての要素にユーザー定義の関数を適用する	array_walk($aryOut,'edit','web'); function edit(&$value,$key,$prefix){ $value="$prefix.$key.$value"; }
56	array_walk_recursive	配列の全ての要素に、ユーザー関数を再帰的に適用する	array_walk_recursive($aryOut,'edit','web'); function edit(&$value,$key,$prefix){ $value="$prefix.$key.$value"; }
57	range	ある範囲の整数を有する配列を作成する	$ary = range(1,10);
58	shuffle	配列をシャッフルする	shuffle($ary);
59	array_rand	配列から一つ以上の要素をランダムに取得する	$aryOut = array_rand($aryFruits,3);
60	compact	**変数から配列生成**:変数名と配列の値から配列を生成する	$zip = "228"; $pref = "神奈川県"; $city = "横浜市"; $aryVars = array("pref","city"); $aryOut = compact("zip",$aryVars);
61	extract	**配列から可変変数の生成**:配列からキーに対応する変数を生成。キーについて変数名として有効か、衝突しないかの設定	$aryVKey=array('color'=>"レインボー",'size'=>"大型"); $size="中"; extract($aryVKey, EXTR_PREFIX_SAME,"pre");
配列の比較計算編集			
62	array_diff	配列の差を計算する(array1に存在しないものだけを返す)	$aryOut = array_diff($ary1, $ary2);
63	array_diff_assoc	対象配列の添字を含めて配列の差を計算する(多次元配列の一次元のみ)	$aryOut = array_diff_assoc($ary1, $ary2);
64	array_diff_key	対象配列のキーを基準にして配列の差を計算する	$aryOut = array_diff_key($ary1, $ary2);
65	array_diff_uassoc	ユーザーが指定したコールバック関数を利用し、追加された添字の確認を含めて配列の差を計算する	$aryOut = array_diff_uassoc($ary1, $ary2, 'diff_cmp'); function diff_cmp($a, $b){ return ($a == $b)? '0' : (($a > $b)? '1' : '-1');}
66	array_diff_ukey	キー比較を基準にし、コールバック関数を用いて配列の差を計算する (注)連想配列のみのキー比較でないと正しい比較ができない	$aryOut = array_diff_ukey($ary1, $ary2, 'diff_cmp'); function diff_cmp($a, $b){ return ($a == $b)? '0' : (($a > $b)? '1' : '-1');}
67	array_intersect	配列の共通項を計算して、連想配列を返す	$aryOut = array_intersect($ary1, $ary2);
68	array_intersect_assoc	追加された添字の確認も含めて配列の共通項を確認する	$aryOut = array_intersect_assoc($ary1, $ary2);
69	array_intersect_key	キー比較を基準にして配列の共通項を計算して、連想配列を返す	$aryOut = array_intersect_key($ary1, $ary2);
70	array_intersect_uassoc	追加された添字の確認も含め、コールバック関数で配列のキー比較で共通項を確認する。Array1の値のうち、全ての引数に存在するもののみを返す	$aryOut = array_intersect_uassoc($ary1, $ary2, 'diff_cmp'); function diff_cmp($a, $b){ return ($a == $b)? '0' : (($a > $b)? '1' : '-1');}
71	array_intersect_ukey	キー比較を基準にし、コールバック関数を用いて 配列の共通項を計算して、連想配列を返す	$aryOut = array_intersect_ukey($ary1, $ary2, 'diff_cmp'); function diff_cmp($a, $b){ return ($a == $b)? '0' : (($a > $b)? '1' : '-1');}
72	array_udiff	データの比較にコールバック関数を用い、配列の差を計算する	$aryOut = array_udiff($ary1, $ary2, 'value_cmp'); function value_cmp($a, $b){return ($a == $b)? '0' : (($a > $b)? '1' : '-1');}
73	array_udiff_assoc	データの比較にコールバック関数を用い、追加された添字の確認を含めて配列の差を計算する	$aryOut = array_udiff_assoc($ary1, $ary2, 'value_cmp'); function value_cmp($a, $b){return ($a == $b)? '0' : (($a > $b)? '1' : '-1');}
74	array_udiff_uassoc	データと添字の比較にコールバック関数を用い、追加された添字の確認を含めて配列の差を計算する(キーが比較に使用される)	$aryOut = array_udiff_uassoc($ary1, $ary2, 'value_cmp', 'key_cmp'); function value_cmp($a, $b){return ($a == $b)? '0' : (($a > $b)? '1' : '-1');} function key_cmp($a, $b){return ($a == $b)? '0' : (($a > $b)? '1' : '-1');}
75	array_uintersect	データの比較にコールバック関数を用い、配列の共通項を計算する	$aryOut = array_uintersect($ary1, $ary2, 'value_cmp'); function value_cmp($a, $b){return ($a == $b)? '0' : (($a > $b)? '1' : '-1');}
76	array_uintersect_assoc	データの比較にコールバック関数を用い、追加された添字の確認も含めて配列の共通項を計算する	$aryOut = array_uintersect_assoc($ary1, $ary2, 'value_cmp'); function value_cmp($a, $b){return ($a == $b)? '0' : (($a > $b)? '1' : '-1');}

77	array_uintersect_uassoc	データと添字の比較に個別のコールバック関数を用い、追加された添字の確認も含めて配列の共通項を計算する	$aryOut = array_uintersect_uassoc($ary1, $ary2, 'value_cmp', 'key_cmp'); function value_cmp($a, $b){return ($a == $b)? '0' : (($a > $b)? '1' : '-1');} function key_cmp($a, $b){return ($a == $b)? '0' : (($a > $b)? '1' : '-1');}
78	array_product	配列の値の積を計算する(整数または float として返す)	$iSum = array_product($ary);
79	array_sum	配列の中の値の合計を計算する(整数または float として返す)	$iSum = array_sum($ary);