مقاله PHP
تحت نظر استاد : جناب مهندس سلیمی
تهیه کننده: امین سویزی

توابع کار با آرایه در - php

مبحث آرایه ها که در آموزش پیشین با آن آشنا شدیم، یکی از پرکاربردترین و در عین حال پیچیده ترین مباحث در بیشتر زبان های برنامه نویسی و به طور خاص php است، گستره استفاده از آرایه (Array) در php به حدی است که کم تر برنامه کاربردی را می توان یافت که در قسمتی از کدهای خود متکی بر آن نباشد، قابلیت تفکیک پذیری، تجزیه و ترکیب، تغییر چینش، دسته بندی آسان و طبقه بندی شده، دسترسی سریع و... باعث می شود که استفاده از آرایه ها در اغلب مواقع اجتناب ناپذیر به نظر برسد، به این خاطر است که در مفسر php برای مدیریت آرایه ها توابع از پیش تعریف شده زیادی در نظر گرفته شده است که هر کدام هدف و دستورالعمل خاصی دارند، از این رو اگر می خواهیم کار با آرایه ها را بیاموزیم، ناگزیر باید این توابع و نحوه کار آنها را نیز بشناسیم، البته فراوانی این توابع به حدی است که در یک مطلب نمی توان تمام آنها را خلاصه کرد، لذا در قالب چند مطلب آتی، در حد امکان به موارد عمومی می پردازیم و توابع خاص و پیچیده تر را نیز در مباحثی جداگانه مورد بررسی قرار خواهیم داد.

تابعis_array

نخستین تابعی که به آن می پردازیم، is_arrayنام دارد، is_arrayیک متغیر را به عنوان آرگومان دریافت کرده و بررسی می کند تا ببیند که آیا متغیر مورد نظر یک آرایه است)مقدار true یا 1 برگردانده می شود) یا خیر (مقدار falseیا صفر برگردانده می شود)؛ به مثال زیر توجه کنید

```
<?php
$array = array('welcome', 'to', 'php', 'array', 'learning');
$check = is_array($array);
if($check == true){
   echo 'true';
}
else{
   echo 'false';
}
?>
```

خروجی مثال بالا برابر خواهد بود باtrue ، چرا که متغیر فرضی array یک آرایه است و نتیجه بررسی تابع is_array برابر عدد 1 که معادل true در php است خواهد بود.
نکته: باستفاده از دستور print_r می توانید محتویات درون یک آرایه را به صورت قابل خواندن مشاهده کنید، به طور مثال برای آرایه بالا، به صورت زیر می توانیم این کار را انجام دهیم

```
<?php
$array = array('welcome', 'to', 'php', 'array', 'learning');
print_r($array);
?>
```

که خروجی برابر خواهد بود با مقادیر زیر

```
Array
(
[0] => welcome
[1] => to
[2] => php
[3] => array
[4] => learning
)
```

تابعexplode

اگر بخواهیم یک رشته متنی را تبدیل به آرایه کنیم، تابعexplode در php این کار را انجام می دهد، نحوه کار این تابع بسیار ساده است، یک مقدار به عنوان جدا ساز (کلید تجزیه) و یک متغیر (رشته متنی) به عنوان آرگومان دریافت کرده و در نهایت یک آرایه به ما تحویل می دهد، به طور مثال اگر متنی که می خواهیم آن را تجزیه و تبدیل به آرایه کنیم، به صورت زیر باشد:

welcome to php array learning

کلید تجزیه در اینجا فواصل خالی بین کلمات است، لذا خواهیم نوشت:

```
<?php
$string = 'welcome to php array learning';
$convert = explode(' ', $string);
print_r($convert);
?>
```

اگر اکنون از متغیر convert با استفاده از print_r خروجی بگیریم، خواهیم دید که این متغیر تبدیل به یک آرایه شده است.

```
Array
(
[0] => welcome
[1] => to
[2] => php
[3] => array
[4] => learning
)
```

نکته :تابع explode یک عدد نیز به عنوان آرگومان سوم می پذیرد، این عدد تعداد کلیدهای تجزیه ای که در تبدیل به آرایه استفاده می شوند را مشخص می کند، به طور مثال اگر بخواهیم رشته ما در3 آرایه تجزیه شود، دستور را به شکل زیر تغییر می دهیم:

```
<?php
$string = 'welcome to php array learning';
$convert = explode(' ', $string, 3);
print_r($convert);
?>
```

و خروجی برابر خواهد بود با مقادیرزیر.

```
Array
(
   [0] => welcome
   [1] => to
   [2] => php array learning
)
```

نکته: دستور print_r تنها در مورد آرایه ها کاربرد دارد.

تابعimplode

در کنار تابع explode تابع دیگری به نام implode وجود دارد که دقیقا کار آن برعکس explode است، implodeمقادیر موجود در کلیدهای آرایه را به وسیله یک کلید ترکیب، تبدیل به یک رشته می کند، به مثال زیر توجه کنید

```
<?php
$array = array('welcome', 'to', 'php', 'array', 'learning');
$convert = implode(' ', $array);
echo $convert;
?>
```

ملاحظه می کنید که در اینجا ما از یک فضای خالی به عنوان کلید ترکیب استفاده کرده ایم، نتیجه مثال بالا به صورت زیر خواهد بود

```
welcome to php array learning
```

که در واقع آرایه ما تبدیل به یک رشته متنی شده است

تابعarray_change_key_case

اگر آرایه ما به جای اعداد دارای کلیدهای متنی باشد و بخواهیم حروف کوچک کلیدها را تبدیل به حروف بزرگ یا برعکس حروف بزرگ کلیدها را به حروف کوچک تبدیل کنیم، از تابعarray_change_key_case استفاده می کنیم، البته این تابع بیشتر برای مقادیر به زبان انگلیسی کاربرد دارد

تابع array_change_key_caseدو آرگومان می پذیرد، آرگومان اول همان آرایه ای است که قصد بررسی آن را داریم (input)، آرگومان دوم می تواند دو مقدار (CASE_UPPERتبدیل به حروف بزرگ) یا CASE_LOWER تبدیل به حروف کوچک، حالت پیش فرض) باشد(case) ، به عنوان مثال آرایه زیر را در نظر بگیرید

```
<?php
$array = array('ali' => 3, 'pedram' => 6, 'maryam' => 5);
$change = array_change_key_case($array, CASE_UPPER);
print_r($change);
?>
```

خروجی مثال بالا به شکل زیر خواهد بود

```
Array
(
[ALI] => 3
[PEDRAM] => 6
[MARYAM] => 5
)
```

تابعarray_chunk

اگر بخواهیم یک آرایه را به چند آرایه تجزیه کنیم، تابع array_chunk می تواند به کار گرفته شود، این تابع سه آرگومان می پذیرد، آرگومان اول همان آرایه ای است که روی آن کار می کنیم(input) ، آرگومان دوم تعداد کلید و مقدار برای آرایه های زیر مجموعه است(size) و آرگومان سوم نحوه افزایش شماره کلیدها را (به صورت true یا false)در آرایه های زیر مجموعه نشان می دهد)حالت پیش فرض false است)که به آن preserve_keys گفته می شود، به مثال زیر توجه کنید

```
<?php
$array = array('a', 'b', 'c', 'd' , 'e');
$chunk = array_chunk($array, 2, false);
print_r($chunk);
?>
```

خروجی مثال بالا به صورت زیر خواهد بود

```
Array
(
```

```
    [0] => Array
        (
            [0] => a
            [1] => b
        )

    [1] => Array
        (
            [0] => c
            [1] => d
        )

    [2] => Array
        (
            [0] => e
        )

)
```

ملاحظه می کنید که آرایه های زیرمجموعه با دو کلید و مقدار تفکیک شده اند، البته چون تعداد کلید های ما پنج عدد بود، آرایه زیرمجموعه سوم تنها یک کلید و مقدار دارد.
اگر مقادیر preserve_keys برابر true باشد، خروجی مثال ما به صورت زیر خواهد بود

```
Array
(
    [0] => Array
        (
            [0] => a
            [1] => b
        )

    [1] => Array
        (
            [2] => c
            [3] => d
        )

    [2] => Array
        (
            [4] => e
        )

)
```

تابعarray_combine

تابع array_combineبرای ترکیب دو آرایه و ایجاد یک آرایه جدید کاربرد دارد، عملکرد این تابع به این شکل است که مقادیر آرایه اول را (آرگومان اول) به عنوان کلید و مقادیر آرایه دوم(آرگومان دوم) را به عنوان مقدار آرایه نهایی در نظر می گیرد، یعنی تنها با استفاده از مقادیر (value) دو آرایه را به یک آرایه تبدیل می کند، به مثال زیر توجه کنید

```
<?php
$array1 = array('a', 'b', 'c', 'd' , 'e');
$array2 = array('55', '23', '43', '12' , '98');
$combine = array_combine($array1, $array2);
```

```
print_r($combine);
?>
```
خروجی مثال بالا به شکل زیر خواهد بود.
```
Array
(
[a] => 55
[b] => 23
[c] => 43
[d] => 12
[e] => 98
)
```
یادآوری: دقت کنید که کلید ها در آرایه های بالا در واقع به صورت عددی و پنهان هستند، یعنی به فرض آرایه اول در واقع به شکل زیر است.
```
$array1 = array(0 => 'a', 1 => 'b', 2 => 'c', 3 => 'd' , 4 => 'e');
```

تابعarray_count_values

تابع array_count_valuesتعداد فراوانی تکرار هر مقدار (value) از یک آرایه را محاسبه کرده و با استفاده از آن مقدار، یک آرایه جدید می سازد، ساختار آرایه جدید به این صورت است که مقادیر به عنوان کلید های آرایه جدید و تعداد دفعات تکرار آنها به عنوان مقادیر آرایه جدید به کار می روند، به مثال زیر توجه کنید

```
<?php
$array = array('a', 'b', 'a', 'd' , 'b');
$count_value = array_count_values($array);
print_r($count_value);
?>
```

خروجی مثال بالا به شکل زیر خواهد بود.

```
Array
(
   [a] => 2
   [b] => 2
   [d] => 1
)
```

تابعarray_diff_assoc

تابع array_diff_assocبرای مقایسه یک آرایه با یک یا چند آرایه دیگر کاربرد دارد، نتیجه و خروجی این تابع، یک آرایه جدید با مقادیری است که در تابع اصلی وجود داشته، ولی در تابع یاتوابع مقایسه شده وجود نداشته است، به مثال زیر توجه کنید.

```
<?php
$array1 = array('Tehran', 'Fars', 'Markazi', 'Gilan' , 'Esfahan');
$array2 = array('Tehran', 'Qazvin', 'Markazi', 'Gilan' , 'Khouzestan');
$array_diff = array_diff_assoc($array1, $array2);
print_r($array_diff);
?>
```
خروجی مثال بالا به صورت زیر خواهد بود
```
Array
(
   [1] => Fars
```

```
    [4] => Esfahan
)
```
نکته :ترتیب کلیدها و مقادیر بین آرایه هایی که مقایسه می شوند در این تابع اهمیت دارد، یعنی به فرض اگر مقدار Tehranدر آرایه اول با کلید صفر مشخص شود، در آرایه دوم هم باید در کلید صفر وجود داشته باشد، در غیر این صورت تابع array_diff_assoc آن را نیز به عنوان یک تفاوت بین آرایه ها به خروجی ارسال می کند

تابعarray_diff_key

تابع array_diff_key مشابه تابع array_diff_assoc عمل می کند، با این تفاوت که در اینجا به جای مقادیر، اگر کلید ها بین دو یا چند آرایه مقایسه شده متفاوت باشند، مقادیر متفاوت آرایه اول به خروجی ارسال می شوند، به مثال زیر توجه کنید

```
<?php
$array1 = array('Tehran' => 12, 'Fars' => 45, 'Markazi' => 33, 'Gilan' => 47 , 'Esfahan' => 51);
$array2 = array('Tehran' => 16, 'Qazvin' => 32, 'Markazi' => 39, 'Gilan' => 48, 'Khouzestan' => 51);
$array_diff = array_diff_key($array1, $array2);
print_r($array_diff);
?>
```
خروجی مثال بالا به صورت زیر خواهد بود
```
Array
(
    [Fars] => 45
    [Esfahan] => 51
)
```
ملاحظه می کنید با اینکه مقادیر در آرایه ها با هم متفاوت هستند، اما تنها مواردی به خروجی ارسال شده اند که کلیدهایشان متفاوت بوده.

تابعarray_diff_uassoc

تابع array_diff_uassocنیز با هدف مقایسه دو یا چند آرایه و به دست آوردن خروجی از مقادیر متفاوت آنها به صوت یک آرایه جدید مورد استفاده قرار می گیرد، تنها تفاوت در اینجا این است که خود تابع از یک تابع دیگر که توسط کاربر تعریف می شود، به عنوان آرگومان سوم استفاده می کند، این تابع(تابع کاربر) باید طوری نوشته شود که یکی از اعداد 1-، صفر یا 1 را برگرداند، به مثال زیر توجه کنید

```
<?php
function value_compare($a, $b){
    if ($a === $b){
        return 0;
    }
    return ($a > $b)? 1:-1;
}
$array1 = array("lang" => "fa", "page" => "index", "date" => "2012", "10", "03", "10:40");
$array2 = array("lang" => "fa", "page" => "index", "date" => "2013", "16", "03");
$result = array_diff_uassoc($array1, $array2, "value_compare");
print_r($result);
?>
```
خروجی مثال بالا به شکل زیر خواهد بود
```
Array
(
[date] => 2012
```

```
    [0] => 10
    [2] => 10:40
)
```

ملاحظه می کنید که آرایه خروجی ما، از عناصری که در آرایه اول وجود دارد ولی در آرایه دوم وجود ندارد تشکیل شده است، کلید ها نیز از آرایه اول استخراج شده اند

نکته 1: در تابعی که توسط کاربر تعریف می شود (در اینجا تابع value_compare)، برای هر کلید و مقدار آرایه ها مقایسه ای صورت می گیرد، اگر دو مقدار مساوی باشند، عدد صفر برگردانده می شود و این یعنی آن مورد نباید در خروجی باشد، در غیر این صورت عدد 1 یا 1- برگردانده می شود و مقادیر مربوطه به خروجی ارسال می شود

نکته 2: در قسمت دوم تابع value_compare از شیوه مختصر نویسی دستورات شرطی (if) و (else در php استفاده شده است، در اینجا علامت ? بعد از پرانتز مقایسه، نقش if و علامت : نقش else را دارد.

تابع array_diff_ukey

تابع array_diff_ukey نیز کارکردی مشابه array_diff_uassoc دارد، با این تفاوت که در اینجا به جای مقادیر، از کلیدها برای بررسی دو یا چند آرایه استفاده می شود، در اینجا نیز از یک تابع جانبی که توسط کاربر تعریف می شود استفاده شده و خروجی آن بر مبنای اعداد صفر، 1- یا 1 برگردانده می شود، به مثال زیر توجه کنید

```
<?php
function key_compare($a, $b){
   if ($a === $b){
      return 0;
   }
   return ($a > $b)? 1:-1;
}
$array1 = array("lang" => "fa", "page" => "index", "date" => "2012", "10", "03", "10:40");
$array2 = array("lang" => "fa", "page" => "index", "date" => "2013", "16", "03");
$result = array_diff_ukey($array1, $array2, "key_compare");
print_r($result);
?>
```

خروجی مثال بالا به شکل زیر خواهد بود

```
Array
(
   [2] => 10:40
)
```

اتفاقی که در کد بالا می افتد این است که چون مقدار فرضی 10:40 دارای کلید 2 است و چنین کلیدی در آرایه دوم وجود ندارد، به عنوان خروجی برگردانده می شود، در واقع اگر از آرایه‌ها به صورت مستقل با دستور print_r خروجی بگیرید، به طور ملموس تری با نحوه کارکرد این تابع آشنا خواهید شد

تابع array_diff

تابع array_diff نیز از خانواده توابع مقایسه ای در مورد آرایه ها است، این تابع مواردی را به عنوان خروجی انتخاب می کند که مقادیر دو کلی همسان، متفاوت باشند، یا مقادیر همسان و کلیدها متفاوت باشند و اگر یک کلید در آرایه اول وجود داشته باشد ولی آن کلید در آرایه دوم وجود نداشته باشد نیز به عنوان خروجی برگردانده می شود، به مثال زیر توجه کنید

```
<?php
$array1 = array(0 => "a", 1 => "b", 2 => "c", 3 => "b");
$array2 = array(0 => "a", 1 => "b", 2 => "f");
$result = array_diff($array1, $array2);
print_r($result);
?>
```

خروجی مثال بالا به صورت زیر خواهد بود

```
Array
(
[2] => c
)
```

نکته :همان طور که ملاحظه می کنید، این تابع مقادیر تکراری را مجددا بررسی نمی کند، لذا چون مقدار b در کلید 1 بررسی شده و تفاوتی وجود نداشته است، در کلید3 از آرایه اول مجددا بررسی نمی شود، اما اگر مقادیر b در کلید 1 از آرایه اول با مقادیر کلید 1 از آرایه دوم متفاوت بود، هم کلید 1 و هم کلید 3 آرایه اول به خروجی ارسال می شد