آموزش php

گردآورنده: عباس احمدوند

انواع عملگر ها درPHP

Operator یا عملگر کارکتریست که نشاندهنده عملیات خاصی بر روی یکسری مقادیر استمانند عملگر + که نشاندهنده عملیات جمع است . در PHP چهار دسته عملگر پر کاربرد وجود دارد که به ترتیب آنها را بررسی می کنیم(شایان ذکر است که عملگر های دیگری نیز غیر از آنچه که در زیر گفته شده وجود دارد که به علت کمی استفاده فعلا از ذکر آن خود داری می کنیم):

۱- عملگر های حسابی (Arithmatic Operators)
این عملگر ها برای محاسبه عبارات ریاضی بکار می روند و بر روی عملوند های عددی استفاده می‌شوند . در جدول زیر می توانید این عملگر ها را مشاهده کنید:

Operator	Description	Example	Result
+	Addition	x=2 x+2	 4
-	Subtraction	x=2 5-x	 3
*	Multiplication	x=2 x*3	 6
/	Division	x=2 8/x 9/x	 4 4.25
%	Modulus	x=2 9%2	 1
++	Increment	x=2 ++x x++	 3 3
--	Decrement	x=2 x-- --x	 1 1

تمامی عملگر های بالا به استثناء دو مورد آخر باینری هستند یعنی دارای دو عملوند می باشند

۲- عملگرهای انتسابی (Assignment Operators)
این عملگرهای برای انتساب مقدار به متغیر ها استفاده می شوند. این عملگر ها همه باینری هستند(یعنی بر روی دو عملوند عمل می کنند) و همیشه مقدار سمت راست خود را به متغیر سمت چپ نسبت می دهند

Operator	Example	Is the same as
=	x=y	x=y

x=x+y	x+=y	+=
x=x-y	x-=y	-=
x=x*y	x*=y	*=
x=x/y	x/=y	/=
x=x%y	x%=y	%=
x=x.y	x.=y	.=

۳- عملگرهای مقایسه ای(Comparison Operators)

این عملگر ها که همگی باینری هستند برای‌مقایسه دو مقدار به کار می روند و دارای خروجی هستند. یعنی به نوعی شرط را در دل خود دارند. خروجی این عملگر هاtrue (هر جا که شرط برقرار باشد) وfalse (هر جا که شرط برقرار نباشد) می باشد .

Example	Description	Name	Operator
5==8 returns false 5=='5' returns true 5==5 returns true	true : اگر بعد از تبدیل نوع دو طرف دارای یک مقدار باشند	Equal	==
5==8 returns false 5=='5' returns false 5==5 returns true	true : اگر دو طرف هم از نظر نوع و هم از نظر مقدار یکسان باشند	Identical	===
5!=8 returns true 5!='5' returns false 5!=5 returns false	true : اگر دو طرف بعد از تبدیل نوع نامساوی باشند	Not Equal to	!=
	true : اگر دو طرف بعد از تبدیل نوع نامساوی باشند	Not Equal to	<>
5!==8 returns true	true : اگر دو طرف نوع یا مقدار متفاوتی داشته باشند	Not Identical	!==
5<8 return	true : اگر عملوند سمت چپ از عملوند سمت راست کوچکتر باشد	Less Than	<

			true 7<4 return false 5<5 return false
>	Greater Than	true : اگر عملوند سمت چپ از عملوند سمت راست بزرگ تر باشد	5>8 return false 7>4 return true 5>5 return false
<=	Less Than or equal to	true : اگر عملوند سمت چپ از عملوند سمت راست کوچکتر یا با آن مساوی باشد	5<=8 return true 5<=5 return true
>=	Greater Than or equal to	true : اگر عملوند سمت چپ از عملوند سمت راست بزرگ تر یا با آن مساوی باشد	5>=8 return false 5>=5 return true

۴- عملگر های منطقی(Logical Operators)

عملگرهای منطقی که دوتای اول آنها(&& و ||) باینری و آخری (!) یکانی است طبق قوانین منطق گزاره ای رفتار می کنند. این عملگر ها برای ساخت شرط در جملات شرطی(Conditional Statement) و حلقه (Loops)ها استفاده می شوند .

Operator	Description	Is the same as
&&(AND)	true : اگر هر دو عملوند هم زمان ارزشtrue داشته باشند (برابر با جدول درستی رابطه عطف AND- در منطق گزاره ای)	x=4 y=9 (x<5 && y>7) returns true
||(OR)	true : اگر حداقل یکی از دو عملوند ارزشtrue داشته باشند (برابر با جدول درستی رابطه فصلOR- در منطق گزاره ای)	x=4 y=9 (x<3 || y>7) returns true
!(NOT)	true : اگر تنها عملوند آن دارای ارزشfalse باشد(برابر با جدول درستی رابطه نقیضNOT- در منطق گزاره ای)	x=4 !(x<3) returns true

شایان ذکر است که درphp نیز مانند بسیاری زبان های دیگرfalse هم ارز ۰ (صفر) است .

آرایه چیست ؟
آرایه عبارتست از یک سری خانه های به هم پیوسته حافظه که همه آنها هم جنس هستند و با یک نام به آنها رجوع می شود این تعریف یک تعریف کلی در اکثر زبانهای برنامه نویسی است. البته در php شرط هم جنس بودن وجود ندارد.
مثلا فرض کنید بخواهید یک برنامه بنویسید که درآن قرار است اطلاعات یکسری از دانشجویان یک کلاس را در یکسری متغیر قرار دهیم و پردازشی روی آن انجام شود چه کار باید انجام دهیم ؟ آیا به ازاء هر دانشجو یک متغیر بگیریم ؟ آیا اصلا تعداد دانشجویان را از قبل می دانیم ؟ اگر تعداد آنها مرتب تغییر کند و پویا باشدچه ؟ اصلا فرض کنیم تعداد را هم بدانیم اگر تعداد آنها زیاد بود چه کنیم مثلا در مورد یک دانشگاه که ممکن است ده هزار نفر دانشجو داشته باشد به تعداد این متغیر ها فکر کرده اید ؟ ضمن اینکه احتمالا پردازش همه آنها شامل عملیات یکسانیست ولی با تعریف یک متغیر برای هر دانشجو عملا از کامپیوتر هیچ بهره ای نبردیم . چرا که آن عملیات را برای هر ده هزار نفر باید تکرار کنیم (یعنی کد را کپی و پیست کنیم) . حال لگر برای هر دانشجو بخواهیم فقره های اطلاعاتی مختلفی از جمله شماره دانشجویی ، نام ، نام خانوادگی و.. را ذخیره کنیم شرایط به مراتب بد تر هم می شود .
فلسفه وجودی آرایه همینست که قرار است یکسری پردازش مشخص بر روی تعدادی(احتمالا زیاد) شیء هم جنس انجام شود . یعنی برای تک تک اشیاء عملیات یکسان خواهد بود. حال بهینه ترین راه برای دستیابی به این اشیاء استفاده از ساختمان داده ایست که همه را تحت یک نام ذخیره کرده و بعد با یک آفست یا شماره یا هر مکانیزم دیگری بتوان به آنها دسترسی داشت
برای تعریف آرایه در PHP می توان از دستور زیر استفاده کرد:

```
<?php
$array = array();
?>
```

با این کار (استفاده از دستور array) یک آرایه ایجاد کرده ایم. حتی می توانیم به هنگام ایجاد آرایه مقادیر خانه های آنرا نیز مشخص کنیم . ولی نکته ای که در همین ابتدا باید به آن اشاره کنم اینست که آرایه در PHP بسیار منعطف تر از آرایه در زبان های دیگری مانند c است . چرا که هم این متد را می توانید بدون پارامتر بکار برید و هم با پارامتر. همچنین در تعداد پارامتر ها محدودیتی وجود ندارد می توانید تا دلتان بخواهد پارامتر به این متد بدهید و لزومی هم ندارد که تعداد خانه های آرایه از قبل مشخص باشد و حتی می توانید بعد از تعریف آرایه خانه ای به آن بیفزائید . به مثال زیر توجه کنید :

```
<?php
$cars = array("BMW" , "Audi" , "Benz");
$countries = new array("Iran" , "Germany" , "Egypt" , "The U.S");
?>
```

همانطور که می بینید در خط اول ، آرایه ای با ۳ عضو و در خط دوم آرایه ای با ۴ عضو تعریف شده است .
در **زبان PHP** سه نوع **آرایه** وجود دارد :
۱- آرایه های عددی (Numeric Array)
۲- آرایه های انجمنی (Associative Array)
۳- آرایه های چند بعدی (Multidimensional Array)

۱- آرایه های عددی در PHP
آرایه عددی یا **Numeric Array** آرایه ایست که در اکثر زبان های برنامه نویسی وجوددارد . در این آرایه ، به کل آرایه یک نام می دهیم بدین ترتیب هر یک از خانه های آرایه یک شماره ایندکس می گیرد و با آن شماره خاص می توانیم به یک خانه خاص آرایه دست بیابیم. به مثال زیر توجه کنید :

```
<?php
$cars = array("BMW" , "Audi" , "Benz");
$countries = new array("Iran" , "Germany" , "Egypt" , "The U.S");
echo "Our cars are : ".$cars[0]." , ".$cars[1]." , ".$cars[2];
$cars[3] = "Opel";
?>
```

همانطور که می بینید به خانه های آرایه با یک شماره که به نام آرایه اضافه شده است دسترسی پیدا کردیم این شماره **اندیس آرایه** یا **ایندکس (index)** یا **آفست** نام دارد . اندیس آرایه از شماره صفر شروع می شود و همانطور که می بینید در خط آخر این برنامه یک خانه به آرایه اضافه شده است به همین سادگی! بدون آنکه از قبل و در هنگام تعریف آرایه فکرش را کرده باشیم . البته پیمایش آرایه یا دستیابی به خانه های آرایه معمولا به این شکل انجام نمی شود بلکه با استفاده از دستور foreach (حلقه foreach) این کار قابل انجام است . اگر میخواهید راجع به دستور foreach بیشتر بدانید، مطلب حلقه های تکرار while و for از همین سایت را مطالعه فرمائید

۲- آرایه های انجمنی (Associative Array)
آرایه انجمنی آرایه ایست که در آن به ازاء هر مقدار(هر خانه) یک فیلد بنام **کلید** نیز در نظر گرفته می شود که این کلید باید در بین کلید های خانه های دیگر منحصر به فرد و یکتا باشد. برای دستیابی به خانه های یک آرایه انجمنی ، دیگر از اندیس عددی استفاده نمی شود بلکه از فیلد کلید به عنوان رشته کلیدی دستیابی استفاده می شود. به مثال زیر توجه کنید :

```
<?php
$ages =  array("Ali" => 21 , "Ahmad" => 24 , "Akbar" => 19);
echo $ages["Ali"];
?>
```

این کد یک آرایه انجمنی با سه عضو ایجاد می کند و بعد به هنگام استفاده از کلید هایی که به هنگام تعریف داده شده برای دستیابی به عضو مربوطه استفاده می شود در اینجا باید حواستان باشد که این کلید مانند سایر موارد دیگر در PHP به حروف بزرگ و کوچک حساس است (Case Sensitive).
آرایه های انجمنی نقش عمده و مهمی در PHP بازی می کنند . اکثر کارها در PHP از دریافت پارامترهای ارسالی از صفحات وب گرفته تا واکشی اطلاعات از دیتابیس همه و همه از آرایه های انجمنی استفاده می کنند آرایه هایی مانند $_GET ، $_POST، $_SESSION، $_COOKIE و ... نمونه هایی از آرایه های انجمنی هستند.

۳- آرایه های چند بعدی (Multidimensional Array) :
این نوع آرایه ، آرایه ایست که هر عضو آن می تواند خود آرایه باشد در حقیقت این نوع آرایه همانست که به آن آرایه ای از آرایه ها می گویند . حال این آرایه می تواند عددی باشد ، می تواند انجمنی باشد در هر دو نوع آرایه چند بعدی وجود دارد. حتی بعد های مختلف می توانند انواع مختلف داشته باشند مثلا بعد اول انجمنی باشد و بعد دوم عددی مانند مثال زیر:

```
<?php
$families = array(
"Alavi"=>array("Ali" , "Ahmad" , "Akbar") ,
"Akbari=>array("Hasan" , "Mahdi"),
"Ahmadi"=>array("Zahra" , "Majid" , "Maryam" , "Meisam")
);
echo "Second Member of Alavi family is : ".$families['Alavi'][1];
?>
```

این آرایه دارای بعد اول انجمنی و بعد دوم عددیست. در حقیقت در سطح اول یک آرایه انجمنی داریم که دارای سه عضو است: خانواده های Alavi، Akbari و Ahmadi . و در سطح دوم هر یک از این خانواه های اعضای خود را دارند که در داخل هر خانواده با شماره (ایندکس) به آنها دست می یابیم بنابراین سطح دوم عددیست.
همانطور که از مثال فوق نیز پیداست یکی از تفاوت های عمده ای که **آرایه در PHP** با آرایه در سایر زبانها دارد اینست که در آرایه های چند بعدی PHP لازم نیست تعداد اعضای بعد های دوم به بعد با هم برابر باشند مانند مثال فوق که یک آرایه دو بعدیست که اولین عضو بعد اول دارای ۳ عضو در بعد دوم و دومین عضو بعد اول (خانواده Akbari) دارای دو عضو در بعد دوم است .

تابع چیست ؟
آیا تا به حال شده که از انجام یک کار تکراری در برنامه نویسی خسته شده باشید و پیش خود بگوئید ای کاش یکبار این کار را به کامپیوتر می گفتم و هر موقع دلم میخواست از او می خواستم که آن کار را انجام دهد ؟ اگر جواب مثبت است **تابع** همان چیزیست که شما آرزوی آنرا داشتید . (اگر هم جواب منفی است به این معنیست که شما هنوز در اصول پایه ای برنامه نویسی مشکل دارید بهتر است به عقب برگردید و اصول برنامه نویسی ساخت یافته را دقیق تر و با تمرین بیشتری یاد بگیرید)
تابع در حقیقت قطعه کدیست که یکبار آنرا می نویسید و برای آن یک اسم انتخاب می کنید و هربار احساس نیاز کردید یکی که داخل تابع نوشته اید باید اجرا شود می توانید فقط با صدا زدن نام تابع (خواهیم دید) آن تابع و در حقیقت دستورات داخل آنرا اجرا کنید . کد زیر شکل کلی تعریف تابع است:

```
<?php
function <function_name>(<parameters_list>)
{
//Do Somethings
}
?>
```

برای اجرا و اصطلاحا فراخوانی تابع فوق به شکل زیر عمل می کنیم:

```
<?php
<function_name>(<arguments_list>);
?>
```

نام تابع چه به هنگام تعریف و چه به هنگام فراخوانی حداقل با یک زوج پرانتز باز و بسته همراهی می شود. این زوج پرانتز برای نگهداری لیست پارامتر ها یا آرگومان های تابع است که در ادامه معنی و لزوم وجود آنها را خواهیم دید . فرض کنید بخواهیم تابعی بنویسیم که فقط وظیفه چاپ یک پیغام را دارد و هیچ آرگومانی دریافت نمی کند تعریف و فراخوانی این تابع به شکل زیر خواهد بود :

```
<?php
function printName()
{
echo "Mohammad Abdollahi";
}
echo " Your Name is ";
printName();
?>
```

این تابع وظیفه نوشتن نام را بر عهده دارد و تا وقتی که فراخوانی نشود اجرا نخواهد شد. در حقیقت نوشتن یک قطعه کد داخل بلوک تابع ضمانت نی کند که به هنگام لود صفحه و تا وقتی که تابع فراخوانی نشود آن قطعه کد اجرا نگردد . بنابراین پس از

اجرای خط اول برنامه پیام Your Name is چاپ شده و سپس در خط بعدی با فراخوانی تابع قطعه کد داخل تابع اجرا شده و عبارت Mohammad Abdollahi چاپ می شود .
همانطور که شما نیز متوجه شدید اینگونه استفاده از تابع کارایی آنرا کاهش میدهد (بلکه موجب افزایش کد نویسی نیز می شود) . از آنجا که فلسفه ایجاد تابع تقسیم یک کار به بخش های کوچک تر است باید تابع را به گونه ای نوشت که کارها را زیادتر نکند و اصطلاحا بار اضافی وارد ننماید. مثلا فرض کنید برنامه ای نوشته اید که در یک سازمان در حال کار است. می خواهید هر فرد در سازمان پس از ورود به سیستم با پیغام فوق مواجه شود اگر به شکل بالا کدنویسی کنید باید برای چاپ نام هر فرد داخل سازمان یک تابع بنویسید بنابراین برای سازمانی با۲۰۰۰ نفر پرسنل نیاز به ۲۰۰۰ تابع فقط برای چاپ نام هایشان دارید!!!! برای فائق آمدن بر این مشکل از پارامترهای تابع استفاده می کنیم . در حقیقت با این کار ، **روال و منطق کار را از داده ها جدا می کنیم** . به مثال زیر که تغییر یافته مثال بالا است توجه کنید:

```
<?php
function printName($userName)
{
echo $userName;
}
$name = "Mohammad Abdollahi";
echo " Your Name is ";
printName($name);
?>
```

در مثال فوق تابعی تعریف کرده ایم که یک **پارامتر** بنام userName دریافت کرده و آنرا چاپ می کند . به هنگام فراخوانی ، برای این پارامتر تابع یک **آرگومان** بنام name ارسال کرده ایم . در اینجا چون مثال ما صرفا جهت آموزش استفاده از پارامتر است متغیر name را دستی مقدار دهی کرده ایم ولی مثلا می تواند از دیتابیس پر شود . متغیر name که به تابع ارسال می شود در بدنه تابع نام userName به خود می گیرد . در حقیقت از این متغیر یک کپی در حافظه گرفته می شود و به آن در بدنه تابعuserName گفته می شود و دیگر با متغیر اصلی ارسال شده کاری نداریم . حال برای چاپ نام هر نفر کافیست فقط نام وی را به این تابع ارسال کنیم و خود تابع را فقط یکبار می نویسیم . نکته مهم در مطالعه PHP اینست که هر جا به عبارت پارامتر برخورد کردید منظور متغیریست که به هنگام تعریف تابع درسرآیند (Header) تابع معرفی شده است و هر گاه بنام آرگومان برخورد کردید منظور متغیر یا عبارتیست که به هنگام **فراخوانی تابع** استفاده شده است.

بازگشت مقدار از تابع با استفاده از دستورreturn :

اکثر اوقات لازم است که تابع وظیفه اي که بر عهده دارد را انجام داده و نتیجه آنرا به صورت خروجي به نقطه اي از برنامه که تابع را فراخواني کرده بازگرداند . این مهم با استفاده از **دستور return** قابل انجام است . بدین شکل که مقدار بازگشتي از تابع را جلوي **دستور return** مي نویسیم . هر تابع مي تواند يك يا چند **return** داشته باشد ولي فقط يكي از آنها اجرا خواهد شد . چرا که با اجراي اولين **دستور return** اجراي تابع خاتمه يافته و كنترل اجراي برنامه به كد فراخواني كننده باز مي گردد به مثال زير دقت كنيد :

```
<?php
function add($num1 , $num2)
{
$sum = $num1 + $num2;
return $sum;
}
echo add(1,2);
?>
```