این زبان چیست ؟

PHPمخفف PHP: Hypertext Preprocessor میباشد.

این یک زبان برنامه نویسی تحت وب است.

این زبان از نرم افزارهای بانک اطلاعاتی زیادی پشتیبانی میکند.

PHPیک نرم افزار متن باز است.

این نرم افزار رایگان است.

فایل های PHP چه هستند ؟

*در این فایل ها از متن،تگ های HTML و Script استفاده میشود.

*این کدها در مرورگر،به شکل HTML بارگزاری میشوند.

*این فایل ها میتوانند دارای پسوند php,php3 و phtml باشند.

MySqlچیست ؟

MySqlیک سرور بانک اطلاعاتیست.

این نرم افزار،یه نرم افزار ایده آل برای پروژه های کوچک و بزرگ است.

MySqlدارای استاندارد های SQL است.

این نرم افزار،یک نرم افزار رایگان است.

چرا PHP ؟

چون توسط سیستم عامل های زیادی پشتیبانی میشود.

PHPاز طریق وبسایت رسمی اش،به راحتی قابل دانلود است(www.php.net).

این زبان به راحتی قابل یادگیریست و به راحتی بروی سرورها نصب میشود.

چگونه شروع کنم ؟

نصب کردن Apache و MySql و Php که در قسمت بعدی به آموزش آن خواهیم پرداخت.

تهیه کردن هاستی که بتواند MySql و Php و APACHE را پشتیبانی کند.

چگونه استفاده کنیم ؟

فقط یک فایل .php را ایجاد کنید تا در آن برنامه نویسی خود را شروع کنید.

حال اگر سرور شما نتواند از PHP پشتیبانی کند،شما باید نرم افزار های زیر را در اون نصب کنید:

دانلودPHP

PHP: Downloads

دانلودMySql

MySQL :: MySQL Downloads (Generally Available)

دانلودApache

Download - The Apache HTTP Server Project

از راهنما نیز برای نصب استفاده کنید.

PHP: Installation and Configuration - Manual

چگونه شروع کنم؟

زبان برنامه نویسی,PHP مثل هر زبان دیگه این احتیاج به محیطی برای فراخوندن و پردازش اطلاعات از کتابخانه مرورگر و سیستم کاربرـه،وقتی شما شروع به نوشتن کدها میکنید،باید دقت کنید که اون در بین کدهای زیر باشه:

کد:PHP

```
<?php
PHP CODE
?>
```

نکته,PHP CODE :محل قرارگیری کدهای PHP هستش.

برای مثال،ما از دستور Echo برای چاپ یک متن روی صفحه وب استفاده میکنیم که به اینصورت استفاده میشه:

کد:PHP

```
<?php
echo "Hello World!"
?>
```

خروجی:
Hello World!

پس تا اینجا فهمیدیم که در چه بیسی برنامه نویسی کنیم و دستور چاپ متن چیه؟اما فکر کنم نیاز باشه بین کدها،یسری نوشته های یادآور بزاریم که کدی که در اون قسمت قراره استفاده بشه،برای چی هست؟به نظرتون نیازه؟نه؟چرا؟چرا دیگه نیازه!برای اینکه بدون آسیب زدن به ساختار کدها،نوشته هایی رو با هر منظوری در فایل .PHP قرار بدید،از این کد استفاده میکنیم:

کد:PHP

```
<?php
//Comment Text
?>
```

راستی فراموش نکنید برنامه نویسی تحب وب،بدون طراحی وب،اصلا رنگ و رویی نداره

متغیرها

شما الان دنبال قیمت ارز هستی،خوب،برای اینکه کارت راحت شه میای میگی:

d=دلار

p=پُند

Y=ین

اگه دلار بالا میگی D مثلا ۱۰۰ تومن افزایش یافت،مقدار تغییر میکنه،اما اصل و نماد اون مقدار ها خیر!به این شیوه به ساخت **متغیر** معروفه!

اعداد X را به خاطر دارید ؟در دبستان همیشه دشمن واقعی ما بودند،اما حالا از این اعداد میتوان در زبان PHP به نحو احسنت

استفاده کرد.در برنامه نویسی به این ها،متغیر یا Variables میگویند.برای مثال:

X=2

در اینجا، Xبرابر ۲ است،یعنی هرجا نام X بیاید سرور اطلاعات این متغیر که ۲ میباشد را دریافت میکند،چند نکته را باید در نظر بگیرید:

***متغیرها با $ تعریف میشوند.**

***متغیرها میتوانند از رشته های آلفا (A-z,1-9,_) باشند.**

***متغیرها نمیتوانند فاصله (Space) را قبول کنند.**

***متغیرها به بزرگی و کوچکی حساسیت دارند.(متغیر x با متغیر X کاملا فرق دارند)**

برای مثال میتوانیم اینگونه بگوییم:

کد:PHP

```php
$txt="Hello World!";
$x=16;
?>
```

حال برای نمایش آنها چگونه باید اقدام کرد ؟به این صورت:

کد:PHP

```php
echo $txt
?>
```

متغیرهای رشته ای

خوب اول باید گفت که رشته چیست ؟ رشته شامل حروف و کلمات و اعداد و فاصله ها و علامت ها و ..... میشود.ساخت رشته،دقیقا همانند ساخت متغیر است که در جلسه پیشین به آن اشاره کردیم.فقط باید مقادیر این متغیرها یکی از عناصر نامبرده باشد.مثلا:

کد:PHP

```php
$txt="Hello World";
echo $txt;
?>
```

خروجی این کد برابر است با:

Hello World

حال اگر بخواییم دو مقدار رشته ای را در کنار هم قرار دهیم از "." استفاده میکنیم،مثلا:

کد:PHP
```
$txt1="Hello World!";
$txt2="What a nice day!";
echo $txt1 . " " . $txt2;
?>
```

شمارش رشته های یک متغیر

برای اینکار کافی است تا به شکل زیر عمل کنید و روبروی دستور echo متن زیر را قرار دهید:

کد:PHP
```
echo strlen("Hello world!");
?>
```

فهمیدن مقدار رشته ها تا رشته مورد نظر

کد:PHP
```
echo strpos("Hello world!","world");
?>
```

چند متغیر در قالب یک متغیر

از فرمول Arrays میتوان برای ساخت چند متغیر در قالب یک متغیر استفاده کرد،برای مثال:

```
$cars[0]="Saab";
$cars[1]="Volvo";
$cars[2]="BMW";
$cars[3]="Toyota";
```

میتوانید از فرمول بالا برای این عمل استفاده کنید.برای خروجی گرفتن از این فرمول:

```
$cars[0]="Saab";
$cars[1]="Volvo";
$cars[2]="BMW";
$cars[3]="Toyota";
```

```
echo $cars[0] . " and " . $cars[1] . " are Swedish cars.";
?>
```

که خروجی برابر است با:

Saab and Volvo are Swedish cars.

مثالی دیگر:

```
$ages['Peter'] = "32";
$ages['Quagmire'] = "30";
$ages['Joe'] = "34";

echo "Peter is " . $ages['Peter'] . " years old.";
?>
```

خروجی:

Peter is 32 years old.

برای رشته های طولانی تر میتوانید از این کد استفاده کنید:

```
Array
)
[Griffin] => Array
)
[0] => Peter
[1] => Lois
[2] => Megan
(
[Quagmire] => Array
)
[0] => Glenn
(
[Brown] => Array
)
[0] => Cleveland
```

[1] => Loretta
[2] => Junior
)

)

عملگرها در php

در php همانند تمامی زبان های برنامه نویسی این قابلیت وجود دارد که با استفاده عملگرها از جمله عملگرهای ریاضی بتوان معادلات و عملیات ها متفاوتی از جمله عملیات های ریاضی و منطقی را به خوبی انجام داد php انواع متعددی از عملگرها را پشتیبانی میکند که در این قسمت در قالب چندین جدول این عملگرها را برای شما معرفی خواهیم کرد و مثالی اسان برای شما میزنیم از عملگرها میتوان در شرط ها متغیر ها توابع و بسیاری جاهای دیگر استفاده کرد.

عملگرهای ریاضی:

عملگر	مفهموم	مثال	نتیجه
+	جمع	$a = 2; Echo $a + 2;	4
-	تفریق	$a = 2; Echo $a – 1;	1
*	ضرب	$a = 2; Echo $a * 3;	6
/	تقسیم	$a = 2; Echo $a / 2;	1
++	مقدار به علاوه یک	$a = 2; $a++; Echo $a;	3
--	مقدار به علاوه دو	$a = 2; $a--; Echo $a;	1

عملگرهای انتصابی:

برابر است با	مثال	عملگر
$a = $b	$a = $b	=
$a = $a + $b	$a += $b	+=
$a = $a - $b	$a -= $b	-=
$a = $a * $b	$a *= $b	*=
$a = $a / $b	$a /= $b	/=
$a = $a.$b	$a .= $b	.=
$a = $a % $b	$a %= $b	%=

عملگرهای مقایسه:

آموزش و نتیجه	مفهموم	عملگر
5 == 2 returns false	مساوی است با	==
5 =! 2 returns true	نامساوی است با	!=
5 <> 2 returns true	نامساوی است با	<>
5 < 2 returns false	بزرگتر از	>
5 > 2 returns true	کوچکتر از	<
5 =< 2 returns false	بزرگتر یا مساوی از	>=
5 => 2 returns true	کوچکتر یا مساوی از	<=

عملگر های منطقی:

آموزش و نتیجه	مفهموم	عملگر
$a=6; $b=3; ($a < 10 && $b > 1) returns true	و	&&
$a=6; $b=3; ($a < 10 || $b > 1) returns false	یا	||
$a=6; $b=3; !(x==y) returns true	نامساوی	!

یک نکته

کوچک در مورد عملگرهای ریاضی وجود دارد گاها ممکن است پیش آید مثلا شما بخواهید جمع چند عدد یا متغیر را با جمع چند عدد یا متغیر دیگر ضرب یا جمع یا تفریق یا تقسیم کنید در این زمان باید شما دامنه هر قسمت معادله را با پرانتز از هم جدا کنید تا نتیجه مورد نظر شما به وجود آید ، مثال:

کد:PHP

```
<?php
$a = 12;
$b = (12 * 5 - (41 - $a)) * ($a - 1);
echo $b;
?>
```

دستورهای شرطی در php

شرط ها در php وظیفه چک کردن مقادیر خاصی را با استفاده از عملگرهای متفاوت دارند ، به طور مثال زمانی که شما میخواهید یک فرم ورود در برنامه خودتان ایجاد کنید نیاز دارید در قسمت های مختلف از شرط ها استفاده کنید. شکل کلی یک شرط را در زیر مثال میزنیم:

کد:PHP

```
if  (دستورات شرط)   {
محتویات شرط در صورت برقرار بودن دستورات شرط;
}
```

همان طور که در مثال اولیه میبینید دستور شرط با کلمه if اغاز و سپس دستورات در داخل پرانتز بعد از کلمه if قرار گرفته و بعد از آن کدهای اجرایی که مشروط به صحیح بودن دستورات داخل پرانتز نوشته میشود . حال یک مثال:

کد:PHP

```
<?php
$num1 = 5;
$num2 = 1;
if($num1 + $num2 == 10) {
echo "$num1 + $num2 is equal to 10";
}
?>
```

مثال اول ما در شرط چک کردیم اگر جمع متغیر اول و دوم مساوی ۱۰ بود مقدار داخلی شرط را نمایش دهد ، حال ما اگر بخواهیم درصورت نادرست بودن شرط مقدار دیگری را نشان دهیم میتوان از فرمان else پس از پایان شرط استفاده کرد .نکته قابل ذکر این است که در فرمان else هیچ دستوری نوشته نمیشود ، برای مثال:

کد:PHP

```
<?php
$num1 = 5;
$num2 = 1;
if($num1 + $num2 == 10) {
echo "$num1 + $num2 is equal to 10";
}
else {
echo "$num1 + $num2 is less than 10";
}
?>
```

ما در شرط بالا که کامل تر شده همان شرط قبلی هست این قسمت را اضافه کردیم که با فرمان else در صورت برقرار نشدن شرط تکه کده داخل else نشان داده شود یعنی اگر متغیر اول و دوم جمعشان مساوی ۱۰ شد مقدار داخلی else نمایان شود. باید توجه داشت که دستور else تنها درصورتی قابل پذیرش و کارگیری است که در ادامه یک شرط باشد و در صورت استفاده در جای نامناسب با خطای مفسر برخواهیم خورد.

حال زمانی هست که شرط ما چندین بار مجبور به چک کردن ورودی باشد تا خروجی مناسب را ایجاد کند به طور مثال زمانی که ما نیاز داشته باشیم چک کنیم دو عدد با هم مساوی اند یا عدد اول از دومی بزرگتر باشد یا عدد اولی کوچیکتر از عدد دومی باشد ما باید چندین مرحله برای رسیدن به خروجی مناسب شرط را چک کنیم. در این زمان از فرمان elseif استفاده مینماییم . این فرمان همانند فرمان ifاست باید دستورات مورد نیاز را در پرانتزی در جلوی خط فرمان نوشت سپس سایر کد ها را ادامه داد ،

مثال این بخش:

کد:PHP

```
<?php
$num1 = 5;
$num2 = 1;
if($num1 + $num2 == 10) {
echo "$num1 + $num2 is equal to 10";
}
elseif($num1 + $num2 > 10) {
echo "$num1 + $num2 is greater than 10";
}
else {
echo "$num1 + $num2 is less than 10";
}
?>
```

همانطور که دیدید ما همانطور که در بالا گفیم در شرط بالا شرط را چک کردیم که دو عدد با هم مساوی اند یا دو عدد بزرگتر از ده هستند یا کوچکتر از ده هستند و با توجه به حالت بازگشتی دستور مقدار مناسب چاپ شود. توجه کنید دستور elseif همواره باید در ادامه if نوشته شود و به تنهایی کاربرد ندارد و همچنین نمیتواند زیر فرمان else قرار بگیرد .

فرمان سوئیچ در php

Switchرا میتوان به نوعی گفت یک دستور شرطی محسوب میشود با این تفاوت که از دستور سوئیچ در زمان هایی استفاده میشود که یک ورودی داشته باشیم بخواهیم تعداد زیادی بلوک را آنالیز و مقدار را مشخص کنیم ، شکل کلی فرمان سوئیچ به شکل زیر است

کد:PHP

```
switch (ورودی) {

case شرط یک:
 نمایش محتویات به شرط صحیح شدن ورودی و شرط یک;
 break;

case شرط دو:
 نمایش محتویات به شرط صحیح شدن ورودی و شرط دو;
 break;

default:
 نمایش محتویات پیش فرض در صورتی که هیچ کدام از قسمت ها صحیح نشد;
}
```

همانطور که میبینید ما مقدار ورودی را در پرانتز کنار فرمان switch قرار دادیم سپس شروط را در در قالب کیس های در داخل فرمان سوئیچ قرار میدهیم در صورتی که شرط و مقدار ورودی یکسان شوند آن قسمت سوئیچ اجرا و از مابقی برنامه خارج شود در فرمان بالا از break; برای خروج از ادامه برنامه در صورت صحیح شدن شرط استفاده کردیم اگر این فرمان را در کد سوئیچ قرار ندهیم با وجود اینکه

شرط اجرا شده با این حال ادامه شرط نیز چک میشود و در اخر نیز با فرمان :default مقدار پیش فرض را قرار میدهیم این دستور در سوئیچ حکم else در فرمان if را دارد. یک مثال در این مورد:

کد:PHP

```
<?php
switch ($x) {
case 2:
 echo "Number 2";
 break;
case 3:
 echo "Number 3";
 break;
case 1:
 echo "Number 1";
 break;
default:
 echo "No number between 1 and 3";
}
?>
```

به صورت ساده اگر بخواهیم کد بالا را شرح دهیم باید بگوییم ما متغیر x را در برنامه وارد کردیم و سه کیس را برای چک کردن وارد و یک مقدار پیش فرض را در صورت ست نشدن برنامه قرار دادیم در هر کیس یک عدد را بین یک تا سه قرار دادیم اگر متغیر xبا یکی از اعداد کیس ها یکسان شود پیغامی را نمایش دهد در غیر این صورت اگر عدد ما بین یک تا سه نبود خروجی پیش فرض را نمایش دهد ، البته قرار دادن مقدار پیش فرض هیچ الزامی ندارد و میتوان بعد از آخرین دستور break; با } دستور سوئیچ را میبندیم .

آرایه ها در php

آرایه ها یکی از مهم ترین مباحث در php هستند ولی متاسفانه چون به درستی به تازه کارها فهمانده نمیشود از این رو به دلیل مبهم نمایان بودن مسئله همیشه تازه کارها از ارایه ها فراری هستند درحالی که نادانسته بسیار با آرایه ها سرکار خواهند داشت ، نمیخواهم تعریف معمول و گنگ را برای آرایه ها به شما بگویم همین کافیست بگویم آرایه ها در قالب متغیرهای چندین بعدی به ما ارائه میشوند که در آینده و جلسات بعدی نمونه های زیادی از آنها را خواهیم دید استفاده از آرایه ها بسیار خوب و کارآمد است .

ما میبایست برای یادگیری بهتر آرایه ها آن را به سه مبحث مختلف تقسیم کنیم که هر یک را برای شما توضیح میدهم.

۱. ***آرایه های عددی*** : به آرایه هایی گفته میشود که با شاخص عددی فراخوانی میشوند
۲. ***آرایه های انجمنی*** : به آرایه هایی گفته میشود که برای هر عنصر آن را نام گذاری و با کلید نام آرایه را فراخوانی میکنیم.
۳. ***آرایه های چند بعدی*** : به آرایه هایی گفته میشوند که از زیر مجموعه شدن چندین آرایه به دست می آیند.

به طور مثال ما اگر بخواهیم نام ده محصول را از شرکتی در قالب ذخیره کردن در متغیر ها بریزیم مجبوریم ده بار و در ده متغیر متفاوت محصولات را وارد کنیم . قطعا این کار اصلا اصولی و مناسب نیست و باعث گنگ شدن حجیم شدن و بسیاری مسائل دیگر میشود به

طور مثال:

کد:PHP

```php
<?php
$item1 = 'test1';
$item2 = 'test2';
$item3 = 'test3';
/*...*/
$item10 = 'test10';
?>
```

همان طور که میبینید این کار اصلا اصولی و جالب نیست حال در اینجا ما همین کار را با یک ارایه انجام میدهیم:

کد:PHP

```php
<?php
$item = array('test1', 'test2', 'test3' /*...*/,'test10');
?>
```

در مثال بالا ما از یک آرایه عددی برای هدفمان استفاده کردیم حال برای توضیح هر سه نوع آرایه ای که گفته ایم را بخش بخش توضیح خواهیم داد.

۱-آرایه های عددی **:** در بالای این بخش یک آرایه عددی را مثال زدیم در آرایه های عددی بعد از فرمان array در داخل پرانتز عناصر آرایه را وارد کرده و با کاما (,) آنها را از هم جدا میکنیم ما یک آرایه عددی را ساختیم اما حال اگر بخواهیم این از این آرایه استفاده کنیم باید چه کرد؟

قطعا با دستور echo نمیتوانیم یک آرایه را بدون آدرس دهی درست نشان دهیم همان طور که گفته شده آرایه ها در قالب متغیر های چند بعدی به ارائه میشود اگر بخواهیم یک ارایه را با فرمان echo بدون آدرس دهی درست و تنها با قرار دادن نام متغیر آرایه قرار دهیم چیزی برایمان جز کلمه Array نمایان نخواهد گردید. البته میتوان با دستور print_r بدنه یک آرایه را به طور کامل نمایش داد حال کمی تمرین کنیم و مسائل بیشتری درمورد آرایه های عددی بیاموزیم:

شمای آرایه:

کد:PHP

```php
<?php
$item = array('test1', 'test2', 'test3', 'test4', 'test5', 'test6', 't
est7', 'test8', 'test9', 'test10');
?>
```

حالا با فرمان echo یک امتحان بکنیم:

کد:PHP

```php
<?php
echo $item;
?>
```

نتیجه ای جز کلمه Array به ما داده نشد درست است؟ چیزی غیر این هم توقع نمیرفت چون ما آدرس دهی درستی برای نمایش آرایه نداده ایم ، خب حالا بیاییم با فرمان print_r بدنه آرایه مان را نمایش دهیم:

کد:PHP

```php
print_r($item);
```

حال کد را دوباره در مرورگر بارگزاری نمایید.چیزی شبیه به کدی که ما نوشتیم در مرورگر نشان داده میشود البته نه کاملا متشابه !!! همانطور که میبینید پشت هر آیتم یک عدد گذاشته شده است که این عدد از ۰ به عنوان کلید ID اولین عنصر آرایه قرار گرفته با هر عنصر جدید یکی به آن افزوده میشود ، یک موضوع دیگر این است که برای اینکه فرمان print_r به درستی اجرا شود باید نام آرایه را در پرانتزی جلوی فرمان بنویسیم دلیل این موضوع را در مبحث توابع خواهیم گفت در این تنها به صورت یک تذکر به شما برای دانستن میگوییم.

بسیار خب حال ما آرایه را ساختیم و میخواهیم از آن استفاده کنیم باید چه کار انجام داد؟ ما به همین دلیل گفتیم اول با فرمان print_rیک بار بدنه آرایه مان را ببینیم آن چیزی که ما دیدیم شکل تفسیر شده آرایه اولیه مان بود که در مرورگر نشان داد همانطور گفته شد برای هر عنصر یک کلید ID عددی که از صفر شروع میشود داده شد که به عنوان وسیله ای برای شناسایی مکان هر عنصر به آن نیاز است به دلیل همین موضوع است که به این نوع آرایه ما آرایه عددی میگوییم .حال ببینم چطور باید یک عنصر از آرایه عددی را فراخوانی کرد ، برای این کار باید در کنار نام متغیر یک براکت را باز کرد و ID عنصر مورد نظر را در آن نوشت مثال(این کد را با کد قبلی که به برنامتون اضافه کردید جایگزین کنید.

کد:PHP

```php
echo $item[0];
```

اگر فرمان بالا را در زیر آرایه بنویسیم اولین عنصر آرایه که ID شماره صفر به آن تعلق گرفته است نمایش داده میشود ما میتوانیم تا عدد ۹ را در براکت قرار دهیم چون عدد ID ها از ۰ شروع میشوند زمانی که به ID 9 برسیم به دهمین خانه یا آخرین عنصر آرایه ای که ساختیم رسیده ایم و بعد از آن هر عددی وارد نماییم به هیچ نتیجه ای دست نخواهیم یافت مگر آنکه آن کلید ID در آرایه موجود باشد.

یک نکته دیگر برای آرایه ها مطرح است این است که میتوان آرایه ها را به شکل متغیری هم نوشت یعنی جدا جدا به طور مثال:

کد:PHP

```php
<?php
$item = array('test1', 'test1', 'test1');
?>
```

و

کد:PHP

```php
<?php
$item[] = 'test1';
$item[] = 'test2';
$item[] = 'test3';
?>
```

با هم هیچ تفاوتی ندارند ! البته باید بدانیم که در روش دوم با استفاده از براکت [] باید متغیر را آماده کرد برای افزودن عنصر جدید به آرایه در غیر این صورت متغیر از نوع معمولی در مفسر php شناخته میشود که در مبحث متغیر ها به آن اشاره کردیم و گفتیم که متغیر هم نام در زیر یک متغیر مقدار داخلی متغیر دوم به عنوان مقدار داخلی تمامی متغیر های قبلی قرار میگیرد!
و اما یک نکته دیگر : در آرایه ها عددی ما میتوانیم خودمان کلید اولیه را با عدد تایین کنیم یعنی میتوانیم فرمان دهیم آرایه بجای اینکه از عدد ۰ کلید بخورد از عدد ۱۰ کلید بخورد : مثال:

کد:PHP

```
<?php
$item = array(10 => 'test1', 'test1', 'test1');
?>
```

و

کد:PHP

```
<?php
$item[10] = 'test1';
$item[] = 'test2';
$item[] = 'test3';
?>
```

همانطور که میبنید برای انجام این کار ما مقدار عددی اولین عنصر را خودمان وارد میکنیم و بقیه عنصرها کاری نداریم با این کار مقدار اولیه کلید ها را میزنیم میتوانیم کد های بالا را با فرمان:

کد:PHP

```
print_r($item);
```

آن را تست کنیم این هم از توضیح این بخش از آرایه ها.در آخر کد کامل:

کد:PHP

```
<?php
$item = array(10 => 'test1', 'test2', 'test3', 'test4', 'test5', 'test
6', 'test7', 'test8', 'test9', 'test10');
print_r($item);
echo "<p>{$item[13]} after in {$item[12]} and {$item[12]} before in {$
item[11]}</p>";
?>
```

۲- آرایه های انجمنی : تفاوت آرایه های انجمنی با آرایه های عددی در این است که آرایه های انجمنی ما برای هر عنصر آرایه یک کلید ID مشخص توسط خودمان میسازیم یعنی هر عنصر از آرایه یک کلید مشخص شده که میتواند یک رشته یا عدد باشد مشخص میشود چون ما آرایه را در آرایه های عددی به کاملی توضیح دادیم دیگر نیاز توضیحات بیش از حد نیست اگر شما آرایه های عددی را به خوبی متوجه شده باشید با مثال های که در مورد آرایه های انجمنی میزنیم به صورت کامل تفاوت ها را خواهید فهمید ، شکل الگوی کلی یک آرایه انجمنی به این حالت است:
به طور مثال اگر ما بخواهیم سن چند کاربر مشخص شده را در داخل یک آرایه بریزیم:

کد:PHP

```
<?php
$age = array(user1 => 17 , user2 => 19 , 'user 3' => 17);
print_r($age);
```

```
echo "<br>".$age[user1];
?>
```

یا

کد:PHP

```
<?php
$age[user1] = 17;
$age[user2] = 19;
$age['user 3'] = 17;
print_r($age);
echo "<br>".$age[user1];
?>
```

این دو مثال هر دو یکی هستند اما یک نکته را من در این مثال ها گذاشتم تا علاوه بر آموزش یافتن نکات ریز را هم بفهمید. من کلید 'user 3' را داخل کوتیشن ' قرار دادم ولی سایر کلید ها را بدون کوتیشن نوشتم دلیل این کار این است که در این کلید از فاصله استفاده شده در صورتی که در چنین کلیدهای از کوتیشن یا دبل کوتیشن استفاده نشود **Parse error** مواجه خواهیم شد برای همین تمامی کلید ها یا عنصر های رشته ای چند قسمتی خود را در داخل کوتیشن یا دبل کوتیشن بگذارند البته برای تازه کارها پیشنهاد میشود تمامی کلید ها و عنصر های رشته ایشان را در کوتیشن یا دبل کوتیشن قرار دهند تا مشکلی پیش نیاید همچنین در هنگام فراخوانی هم این موضوع صدق میکند . برای این بخش توضیح بیشتری نمیدهیم.

۳-آرایه های چند بعدی : حال ما آشنایی کامل را با آرایه ها داریم ولی یک مسئله وجود دارد حال اگر بخواهیم چندین اطلاعات را در آرایه ها بریزیم چه؟ مثلا بخواهیم با دادن نام یک کاربری نام،نام خانوادگی،سن، و ...را داشته باشیم چه؟ در این وضعیت ما باید از آرایه های چند بعدی استفاده کنیم در این نوع آرایه ما ممکن است برای رسیدن به هدفمان چندین آرایه را زیر مجموعه هم بکنیم. مثال آسان:

کد:PHP

```
<?php
$user = array(
user1 => array(
name => ali,
family => ahmadi,
age => 17
),
user2 => array(
name => mohammad,
family => parsi,
age => 19
),
'User 3' => array(
name => 'ahmad ali',
family => abasi,
age => 17
)
);
```

```
print_r($user);
?>
```

این کد را ذخیره و در مرورگر بارگزاری کنید (به دلیل اینکه شاید حالت نمایش گنگ به نظر بیاید source صفحه را نگاه کنید نه خود صفحه را) اینجا شاید کمی گنگ شوید ما در اینجا هر عنصر آرایه که با کلید یک کاربر ایجاد میشود را یک آرایه در نظر گرفتیم یعنی هر عنصر آرایمان خود یک آرایه هست در اینجا ما برای اینکه اطلاعات را نمایش دهیم باید با توجه به مکان قرار گیری هر عنصر آن را بارگزاری کنیم یعنی اگر فرمان:

کد:PHP

```
echo $user[user1];
```

همان نتیجه ای را خواهیم گرفت که برای یک آرایه آدرس داده نشده به ما میدهد یعنی تنها کلمه Array برای ما نشان داده میشود برای اینکه نتیجه درست داده شود ما باید ادرس دقیق عنصرمان را به متغیر آرایه بدهیم خب حالا در زیر آرایه print_r($user);را حذف و به جای آن بنویسید:

کد:PHP

```
print_r($user[user1]);
```

شاهد خواهید بود که برای ما آرایه زیر مجموعه شده را نشان میدهد خب برای نمایش باید کلید عنصر آرایه مان را به ادامه متغیر آرایه اضافه کنیم یعنی حالا فرمان زیر را در فایلتان افزوده و بارگزاری کنید:

echo $user[user1][name];

به همین ترتیب ما بایدبه اعضای هر آرایه زیر مجموعه کلید عنصر آن را به ادامه نام متغیر آرایه اضافه کنیم به این نوع آرایه ما ارایه های چند بعدی میگویند . ارایه های چند بعدی را نیز میتوان مثل سایر آرایه به شکل متغییری نیز نوشت به طور مثال:

کد:PHP

```
<?php
$user[user1][name] = ali;
$user[user1][family] = ahmadi;
$user[user1][age] = 17;

$user[user2][name] = mohammad;
$user[user2][family] = parsi;
$user[user2][age] = 19;

$user['user 3'][name] = 'ahmad ali';
$user['user 3'][family] = abasi;
$user['user 3'][age] = 17;

print_r($user);
?>
```

این در مورد مبحث آرایه ها بود که گفتیم برای آرایه ها توابع بسیار زیادی وجود دارد که کاربرد آن ها را برای ما بیشتر میکند.

کار حلقه ها در PHP انجام کارهای تکراری هست شما حلقه ها را در آینده بسیار زیاد خواهید دید به طور مثال زمانی که شما به سایتی میروید لیستی از لینک ها ، پست ها ، دسته ها ، صفحه بندی ها و ... را میبینید این ها همگی با کمک حلقه ها ساخته شده اند ما در اینجا به شما انواع حلقه ها را به صورت مبتدی آموزش میدهیم و در یک کتاب جدا به صورت حرفه ای روش کار حلقه ها را در سایت های پویا و دارای بانک اطلاعاتی شرح میدهیم.
به طور کلی ما میتوانیم حلقه ها را به چهار دست تقسیم کنیم:

۱. ***While***
۲. ***Do...while***
۳. ***For***
۴. ***Foreach***

کار تمامی این حلقه ها تقریبا مشابه است و تنها حالت نوشتاری آن تفاوت دارد که یک به یک آن ها را توضیح خواهیم داد

۱- ***While***: در این حلقه یک شرط بررسی میشود در صورتی که شرط برقرار باشد محتویات حلقه اجرا و دوباره چک میشود مادامی که شرط حلقه صحیح باشد حلقه تکرار خواهد شد. شکل کلی دستور:

کد:PHP

```
while  (شرط حلقه)  {
  کدهای داخلی حلقه در صورت برقرار بودن شرط;
  }
```

فکر نمیکنم نیازی به توضیح باشد یک مثال در این مورد برای شما میزنم:

کد:PHP

```
<?php
$i=1;
while($i<=5( {
echo "The number is " . $i . "<br>";
$i++;
}
?>
```

ما در این حلقه یک متغیر عددی به نام i را تعریف کردیم سپس دستور حلقه را نوشته و شرط حلقه را با عملگر کوچکتر مساوی بر این گذاشتیم که مادامی که متغیر i از عدد ۵ کوچکتر یا مساوی باشد حلقه تکرار شود و در اخر حلقه قبل از بسته شدن کارلی براکت با عملگر ++ یا به روایتی با گام حلقه مقدار متغیر i را یکی رو به بالا افزایش دادیم. این توضیح ساده یک حلقه بود .

- نکته : در صورتی که در آخر حلقه گام حلقه یا عملگر ++ را برای افزایش مقدار شرط ندهید و حلقه را در مرورگر نمایش دهید حلقه هرگز پایان نمیابد

۲- ***Do…While*** : این حلقه شباهت زیادی با حلقه while دارد با این تفاوت که شرط حلقه در پایان آن است ، این حلقه حداقل یک بار نمایش داده میشود ولی در سایر شرط ها به دلیل اینکه شرط حلقه در ابتدا چک میشود در صورت ست نبودن آن حلقه به هیچ وجه اجرا نمیشود ، اما شکل کلی دستور:

کد:PHP

```
do{
 کدهای داخلی حلقه در صورت برقرار بودن شرط;
 }
while  (شرط حلقه);
```

مثالی برای این حلقه :

کد:PHP

```
<?php
$i=1;
do{
  $i++;
   echo "The number is " . $i . "<br>";
 }
while($i<=5);
?>
```

۳- ***For***: این حلقه که مرسوم ترین نوع حلقه است بدین شکل است که مقدار اولیه ، شرط و گام حلقه همگی در داخل پرانتز نوشته و با علامت سیمیکلن از هم جدا میشوند کار کلی کار این حلقه نیز مانند حلقه های دیگر است:

کد:PHP

```
for  (مقدار اولیه;  شرط;  گام حلقه)  {
 کدهای داخلی حلقه در صورت برقرار بودن شرط;
 }
```

مثال این حلقه :

کد:PHP

```
<?php
for ($i=1; $i<=5; $i++) {
  echo "The number is " . $i . "<br>";
```

```
}
?>
```

کاملا مشخص هست ما در پرانتز مقدار اولیه را ۱ دادیم سپس شرط گذاشتیم که مادامی که مقدار اولیه از کوچکتر یا مساوی باشد حلقه ادامه یابد و در مرحله آخر گام حلقه را گذاشتیم.

۴- ***Foreach***: این حلقه برای آرایه ها به کار میرود ، با توجه تنظیماتی که این حلقه دارد بسیار برای آرایه ها مناسب است برای اینکه یک آرایه را تکرار بخواهید بکنید و محتویات آن را نشان دهید میتوانید از این حلقه استفاده کنید . شکل کلی دستوری این حلقه به این شکل است:

کد:PHP

```
foreach ($array as $value)  {
 کدهای داخلی حلقه در صورت برقرار بودن;
 }
```

توضیحی نمیخواهد فقط در قسمت شرط حلقه نام متغیر آرایه را نوشته و با فرمان as نوع حلقه را مشخص کنید به طول مثال برای حلقه های عددی از حلقه متشابه حلقه بالا استفاده میکنیم و برای حلقه های انجمنی از شکل کی زیر استفاده میکنیم:

کد:PHP

```
foreach ($array as $key => $value)  {
 کدهای داخلی حلقه در صورت برقرار بودن;
 }
```

البته باید توجه داشت متغیر های بعد از as هستند به عنوان نام جدید ID و عنصر متغیر آرایه قرار میگیرد اگر متوجه نشدید به مثالها توجه کنید

مثالی برای حلقه آرایه های عددی

کد:PHP

```
<?php
$x=array("one","two","three");
foreach ($x as $value) {
echo $value ."<br>";
}
?>
```

و مثالی برای آرایه های انجمنی:

کد:PHP

```
<?php
$x=array("one" => 1,"two" => 2,"three" =>3);
foreach ($x as $key => $value) {
echo $key." => ".$value ."<br>";
}
?>
```

در این حلقه تعداد چرخش بر اساس تعداد عنصر ها می باشد با امتحان کردن حلقه بالا به خوبی متوجه خواهید شد چگونه یک حلقه **foreach** کار میکند .

کامنت ها در php

در php مانند بیشتر زبان های برنامه نویسی مطرح این قابلیت وجود دارد که در بین کدها کامنت ها نوشته شوند ، کامنت ها در اصل هیچ اثری در برنامه ندارند و صرفا به جهت اینکه اطلاعات مطالب و یا موضوعاتی مرتبط قسمتی از برنامه وارد شود نوشته میشود . کامنت ها در پروژه های بزرگ یا پروژه هایی که قرار است توسط چند برنامه نویس نگاشته یا ویرایش شوند بسیار کارآمد هستند زیرا با استفاده از کامنت ها میتوان اطلاعاتی در مورد قسمت های مهم برنامه ذکر کرد که در سهل کردن کار و ویرایش پروژه بسیار تاثیر گذار است در php سه روش معرفی شده برای کامنت گذاری وجود دارد که تمامی آن ها را برای شما ذکر میکنیم.

استفاده از # و // در برنامه ها

در بین تگ های پی اچ پی هرکجا که از این دو علامت استفاده شود تمامی محتویات آن خط از برنامه بعد از علامت کامنت به کامنت تبدیل میشود(حتی اگر وظیفه انجام کاری را داشته باشند آنها بی اثر میشوند) مثال:

کد:PHP

```
<?php
print "This is a test <br>"; # show on page => "This is a test"

// echo "This is a test <br> ";
?>
```

در مثال بالا ما در خط اول توضیحاتی از وظیفه تابع print همان کار echo را توضیح دادیم و در خط بعد تابعی که نوشته بودیم را به شکل یک کامنت بیان کردیم. توجه داشته باشید که هیچ تفاوتی بین // و # نیست و کار هردو وظیفه کامنت کردن و بی اثر نمودن محتویات بعدی خود را در خطی که هستند دارند.

اما یک روش دیگر برای کامنت گذاری در برنامه ها هست که برای کامنت هایی است که بیش از یک خط هستند یا اینکه باید کامنت در وسط یک قسمت در یک خط قرار گیرند برای اینکه محدوده این نوع کامنت مشخص شود از دو تگ آغازین و پایانی استفاده میشود که در مثال زیر با آن آشنا میشوید:

کد:PHP

```
<?php
/*
print "This is a test <br>";
//show on page => "This is a test"

echo "This is a test <br>";
#show on page => "This is a test"
*/
?>
```

در کد بالا تمامی نوشته ها به شکل کامنت در آمده اند ما برای این کار از دو تگ /* .... */ استفاده کردیم اگر شما این دو تگ را حذف کنید کدها داخل آنها اجرا میشوند که باز هم در خطوط میانی یک در میان کامنت های خطی در آن گذاشته ایم.

همیشه اگر میخواهید یک پروژه بزرگ را شروع کنید به شما پیشنهاد میکنم از کامنت ها برای راحتی کار خودتان هم که شده

استفاده کنید البته لازم نیست مانند مثال های بالا کار هایی که echo و print را توضیح دهید هرجا که دیدید شاید در آینده برایتان مبهم باشد یا نیاز به توضیح داشته باشد را تنها در قالب کامنت ها تشریح کنید تا اگر بعد از ساخت برنامه به هر دلیلی مجبور به ویرایش قسمت خاصی باشید دچار گنگی و مبهمی نباشید .