مرجع کامل آموزش زبان

PHP

ناشر نسخه الکترونیک
Ketabnak.com

مؤلف و گردآورنده:

محمد بشیری

مقدمه

شبکه گسترده جهانی یا به عبارتی Word Wide Web دنیای عجیبی است که تکنولوژی‌های مربوط به آن، اغلب بدون پشتیبانی کافی عرضه می شوند و کاربران این تکنولوژی ها، همه روزه با واژگان جدیدی برخورد می کنند، که باعث سردرگمی آنها می شوند.

برای نمونه می توان به رشد نرم افزارهای open source اشاره کرد. (برنامه هایی که می توان آنها را گسترش داد و یا تغییراتی در ساختار آنها ایجاد کرد.) متداولترین این برنامه ها، سیستم عامل Unix و به طور خاص Linux، می باشد. این برنامه ها، با وجود ثبات و پایداری، دارای مشکل بزرگ می باشند و آن دشوار بودن آموختن این برنامه ها می باشد. کمبود راهنماهایی که به زبان ساده، این برنامه ها را به مبتدیان آموزش دهد، باعث شده است که این دسته از نرم افزارها از جایگاه واقعی خود دور نگه داشته شوند.

در ادامه این مقاله با زبان PHP آشنا خواهیم شد. با استفاده از این مقاله شما دانش کافی برای آغاز به کار ایجاد سایت های پویا توسط PHP را کسب خواهید نمود.

تاریخچه مختصری از PHP:

فکر اولیه PHP در پاییز سال ۱۹۹۴ توسط Rasmus Lerdorf شکل گرفت . در ابتدا نگارشی از PHP در صفحه شخصی وی به کار گرفته شد تا اطلاعاتی از کسانی که روزمره او را می بینند نگاه داشته شود . اولین نگارش عمومی آن در اوایل سال ۹۵ ارائه شد و با نام " **P**ersonal **H**ome **P**age Tools " روانه بازار شد که البته شامل پارسری بسیار ساده بود که ماکروهای خاصی را می شناخت و نیز برخی کاربردهای مشترک در صفحات شخصی از قبیل شمارنده ، دفتر میهمانان و برخی از ابزارهای دیگر را به همراه داشت .

پارسر در نیمه سال ۹۵ بازنویسی شد و با نام PHP/FI " نگارش ۲ " ارائه گردید FI . نام بسته نرم افزاری دیگری از Rasmus بود که فرم های داده HTML را تفسیر می کرد . پس از آن وب مسترهای بسیاری از PHP در صفحات خود استفاده کردند . در میانه سال ۹۶ میزان استفاده کنندگان به حدود ۱۵ هزار سایت رسید . این میزان در نیمه سال ۹۷ به ۵۰ هزار سایت مختلف افزایش پیدا کرد . در این زمان PHP از حالت یک پروژه شخصی درآمد و توسط تیمی توسعه یافت . این گروه نگارش جدیدی از PHP را ارائه دادند و پارسر آن را بازنویسی نمودند و بسیاری از مشکلات اساسی آن را برطرف کردند. PHP3 به سرعت مورد استفاده قرار گرفت . هم اکنون نیز PHP4 آخرین

نگارش این محصول است که در آن از موتور اسکریپت Zend برای بدست آوردن قابلیت های بیشتر استفاده شده است .

امروزه PHP3 و PHP4 بر روی بسیاری از محصولات تجاری مانند RedHat's Stronghold WEB SERVER ارائه می گردد . هم اکنون برآورد می شود بیش از ۶ میلیون سایت از PHP استفاده کرده اند که این میزان کمی بیشتر از تمامی سایت هایی است که از سرور IIS مایکروسافت استفاده می کنند.

چرا PHP؟

گذشته از اینکه PHP یک زبان Open Source یا منبع باز است ، دلایل بسیار زیاد دیگری برای انتخاب PHP برای ایجاد محتوای محاوره ای بر روی سایت های وب وجود دارد.

- یکی از این دلایل این است که این زبان ساختار و ترکیبی بسیار شبیه زبان C دارد.
- نوع داده ها و ساختار های PHP ، به آسانی آموخته و به کار گرفته می شوند . در واقع می توان گفت PHP میداند منظور شما چیست و نوع داده های خود را بر اساس اطلاعات شما تغییر می دهد.
- نیازی به دانستن دستور خاصی برای کامپایل برنامه ندارید . برنامه شما در مرورگر اجرا می شود و لازم نیست برای شروع برنامه و نوشتن برنامه های کاربردی درباره PHP اطلاعات زیادی داشته باشید .
- PHP سرویسی از مجموعه فایلهای کتابخانه ای C را ارائه می دهد که به آسانی درون زبان قرار گرفته و با انعطاف بسیار به آن قابلیت پاسخ دهی سریع برای تغییرات در وب را می دهد.

آنچه می توانید شما با PHP انجام دهید ، با دیگر زبانها نیز قابل انجام است . اما PHP برای کار کردن در زمینه وب طراحی شده است . بنابراین کارهای مشکل و خسته کننده ای که برنامه نویسان با Perl انجام می دادند ، به آسانی با PHP قابل انجام است.

این زبان پویا وب سایت ها را قادر می سازد تا با سرعت مبهوت کننده ای گسترش یابند و این عامل یکی از دلایل عمده ای است که برای صفحات پویا و پشتیبانی پایگاه داده ها در نظر گرفته شده است . همانطور که گفته شد در حدود ۶ میلیون سایت در سراسر وب از PHP استفاده می کنند.

کدهای کوچک توکار در یک صفحه وب بسیار کارآمدند . به عنوان مثال در یک صفحه ایستا ، ممکن است شما مقدار یک متغیر را بدست آورید و سپس آن را برای ایجاد تغییرات در محتوای صفحه ، تغییر بدهید . اما در PHP مقادیر متغیر ها مستقیما در سورس صفحه یافت نمی شود . به این مثال توجه کنید:

```
<?php
$brow ser = getenn("HTTP_USER_AGENT");
?>
<p>You are using the <?php echo($brow ser);?> w eb brow ser .
</p>
```

در این مثال به جای عبارت متغیر ، نام مرورگر وب کاربر در صفحه نمایش داده خواهد شد.

پی اچ پی زبانی برای همه سیستم عامل‌ها

یکی از برترین مزایای زبان PHP سازگاری آن با اکثر سیستم عامل‌ها و نرم‌افزارهای وب سرور (مانند IIS و Apache) است. برخی از دیگر زبان‌ها و تکنولوژی‌ها مانند ASP محدود به سیستم عامل windows است و پشتیبانی از آن در دیگر سیستم عامل‌ها بسیار پرهزینه و محدود است، و برخی نیز مانند JSP مشکلاتی با برخی نرم‌افزارهای وب سرور دارد.

ساختار و امکانات پی اچ پی به شکل مستقل از سیستم عامل شکل گرفته است و این بدان معنا است که به طور مثال برنامه‌نویس می‌تواند اسکریپت خود را تحت سیستم عامل ویندوز نوشته و تست کند و سپس آنرا بدون تغییر به سیستم عامل یونیکس یا لینوکس انتقال دهد.

در PHP امکان استفاده از برخی از امکانات خاص سیستم عامل‌های مشهور نیز در نظر گرفته است که برای نمونه می‌توان از پشتیبانی از تکنولوژی DCOM و یا Windows API نام برد.

نسخه‌های جدید مفسر PHP سازگار با دیگر تکنولوژی‌های خاص وبسرورها مانند ISAPI نیز می‌باشد.

پی اچ پی رایگان و Open Source

تهیه برنامه مفسر PHP برای همه سیستم عامل‌ها رایگان است و علاقه‌مندان می‌توانند آخرین نسخه مفسر این زبان را از سایت رسمی PHP بارگذاری [1] کنند.

همچنین امکان تهیه رایگان سورس مفسر پی اچ پی نیز فراهم است ، و این مسئله علاوه بر این که در گسترش امکانات این زبان بسیار موثر بوده است، مزیتی برای شرکت‌ها و توسعه‌دهندگان برای انتخاب این زبان است چرا که پشتیبانی و اعتماد به آن را راحت‌تر کرده است.

بسیاری از ویرایشگرهای حرفه‌ای این زبان نیز یا رایگان هستند و یا با هزینه بسیار کم می‌توان آنها را تهیه کرد، در حالی که دیگر تکنولوژی‌ها، مثلاً تهیه پلاتفورم‌های جاوا هزینه هنگفتی دارد و همچنین کار حرفه‌ای با تکنولوژیNET. نیز نیاز به تهیه Visual Studio.NET و پرداخت هزینه چند صد دلاری است.

سرعت بالای تفسیر و اجرای PHP

پی اچ پی یکی از سریع‌ترین زبان‌ها در نوع خود است. تفسیر و اجرای یک اسکریپ php به طور متوسط تا سه و چهار برابر یک اسکریپ ASP است. (البته باید در نظر داشته باشیم که IIS با Cach اسکریپت‌های ASP سرعت اجرای آنها را در دفعات بعد بالا می‌برد)

همچنین در ASP استفاده زیادی از اشیا COM می‌شود که باعث کاهش سرعت و مصرف منابع سیستم می‌شود در حالی که در PHP بسیاری از امکانات و حتی برقراری ارتباط با یکی محبوب‌ترین نرم‌افزار مدیریت بانک‌های اطلاعاتی mySql به صورت توکار نهاده شده است.

شرکت Zend که تهیه کننده فعلی موتور مفسر و پشتیبانی کننده آن است، محصولات دیگری را نیز در جهت بهینه کردن سرعت اجرای PHP ارائه کرده است این محصولات با افزایش سرعت تفسیر و همچنین ذخیره کردن نتیجه تفسیر (Cash) باعث افزایش چندین برابر اجرای آن می‌شوند.

Download 1

ساختار مناسب و امکانات بالا در PHP

همان طور که در ابتدای مقاله اشاره شد، کمتری نیازی در برنامه‌نویسی تحت وب وجود دارد که در PHP امکان رفع آن نباشد. پی اچ پی شامل کتابخانه‌ای غنی از توابعی است که امکان پردازش اطلاعات فرم‌ها، کار با بانک‌های اطلاعاتی، فایل‌های متنی و باینری، فایل‌های گرافیکی، PDF، ZIP و پروتکل‌های TCP، FTP، DNS، SMTP و ... را برای برنامه‌نویس فراهم می‌کند، این را مقایسه کنید با ASP که به طور مستقل امکان Upload File، ارسال ایمیل یا کار با فایل‌های باینری را ندارد.

همچنین PHP یکی از بهترین پشتیبانی‌ها را از نرم‌افزارهای بانک اطلاعات دارد. mySql,Sql Server,mSql,dBase,Oracle,IBM DB2,PostgreSQL,InterBase و بسیاری از نرم‌افزارهای دیگر در پی اچ پی قابل استفاده هستند و البته امکان کار با ODBC و COM برا ی استفاده از بانک‌های Ms Access و دیگرمحصولات نیز هست.

قدرت زبان پی اچ پی تنها در کتابخانه توابع آن نیست، پشتیبانی بسیار خوب از برنامه‌نویسی شیءگرا (OOP)[1] و کار آسان و سریع با متغیرها از مزایای درونی این زبان است.

معایب

حال که از حسن PHP گفتیم، بد نیست اشاره‌ای نیز به برخی معایب آن داشته باشیم.

نحو (syntax) زبان PHP بسیار شبیه زبان C++ و Perl است. این اگر چه باعث استقبال از این زبان توسط برنامه‌نویسان C و یا Perl شد، اما این نحو برای بسیاری از طراحان صفحات وب چندان آسان نیست و بسیاری معتقد هستند که تکنولوژی ASP و زبان VBScript آسانتر و قابل درک‌تر است و همچنین JSP و زبان جاوا نیز به دلیل محبوبیت و ساختار قدرتمند آن مورد توجه است.

1 Object Oriented Programming

یک اشکال دیگر PHP عدم پشتیبانی خوب آن از یونیکد و به خصوص زبان فارسی است، حتی آخرین نسخه‌های این زبان نیز امکان سورت(Sort) صحیح متون فارسی را ندارد. البته این اشکال با کامپایل مجدد یا کمی برنامه‌نویسی قابل حل است.

مقدمه ای بر PHP و مقایسه آن با Perl CGI

امروزه با توجه به اینکه روز به روز بر تعداد میزبانهایی که PHP رو ساپورت می‌کنند افزوده می‌شود ، صحبت در مورد PHP و قابلیتهای آن در میان طراحان وب و برنامه نویسان زیاد است.

برای کسانی که فقط نامی از PHP شنیده اند و از تواناییها و مزیت های PHP آگاهی ندارند در این مقاله PHP رو به طور مختصر شرح شرح میدهم و آن را با اسکریپتهای CGI مقایسه میکنم.

PHP یک زبان طرف خادم (server side) می باشد و شما می‌توانید برای ساخت صفحات دینامیک وب از آن استفاده کنید. برای مثال مدیریت و ساماندهی اطلاعات دریافتی از یک form اچ.تی.ام.ال با PHP بسیار آسان است.

۱- زبان:

اگر شما با زبانهای Perl , C++ , C یا Java کار می‌کنید یاد گرفتن زبان PHP می‌تواند مانند زنگ تفریح باشد! در واقع شما خیلی سریع می‌توانید اسکریپت نویسی را با PHP شروع کنید.

متغیرها در PHP مانند PHP هستند (با پیشوند $) و انواع مختلف داده ها را می‌توانند در خود ذخیره کنند. برای مثال $whatever می‌تواند انواع داده ها شامل رشته ای ، عددی و غیره را در خود نگه دارد.

اگر مقدار $whatever یک عدد باشد شما می‌توانید مقدار آن را اینگونه افزایش دهید:

$Whatever++; یا $whatever =+1; یا $whatever=$whatever+1;

که دقیقا همان روشی است که در C، C++، Perl یا Java به کار می برید.

۲- تسهیلات توکار (Built-in facilities)

بر خلاف Perl که یک زبان همه منظوره است و شما می‌توانید تقریبا هر برنامه ای را با آن بنویسید ، PHP از ابتدا با هدف اسکریپت نویسی برای صفحات وب درست شده ، از این‌رو اسکریپت نویسی برای صفحات وب در PHP بسیار آسانتر از Perl می باشد.

برای مثال می‌خواهیم از یک فرم در یک صفحه وب ایمیلی را به آدرس خودمان ارسال کنیم. به کمک Perl شما احتمالا کدی شبیه زیر را می‌نویسید :

```
open ( MAIL,\"|/usr/sbin/sendmail -t\");
print MAIL \"To: myself\@mydomain.com\n\" ;
print MAIL \"From: visitor\@hisdomain.com\n\" ;
print MAIL \"Subject: Comments from Web Form\n\n\" ;
print MAIL $mainmessage ;
close ( MAIL ) ;
```

اما همین برنامه در PHP به شکل زیر نوشته می شود :

```
<?php
mail ( 'myself@mydomain.com', 'Comments from Web Form',
$mainmessage, 'From: visitor@hisdomain.com' );
?>
```

حتما تفاوت این دو زبان در سادگی و راحتی را متوجه شده اید!

این سادگی و روانی برای بقیه کارها هم صادق است ، مانند فرستادن یا بازیابی یک پرونده با FTP یا HTTP.

همانطور که گفته شد این سادگی از آنجا ناشی می‌شود که PHP فقط برای برنامه نویسی برای صفحات وب طراحی شده است.

تسهیلات دیگر آن در اداره کردن input های یک form می باشد ، برای مثال یک فرم مانند زیر را در نظر بگیرید:

```
<input type=text name=\"dateofbirth\">
```

شما خیلی راحت و سریع می توانید به محتویات این فرم در متغیر $dateofbirth دسترسی داشته باشید. نیازی به تجزیه و تحلیل input های فرم نیست. تمام فیلد ها در یک فرم به طور اتوماتیک به متغیرهایی تبدیل می-شوند که شما خیلی راحت می‌توانید به آنها دسترسی داشته باشید. [18]

۷ دلیل برای اینکه استفاده از PHP بهتر از ASP می باشد [8]

۱- سرعت ، سرعت ، سرعت

اولین باری که یک کد به زبان PHP نوشتم بر روی یک کامپیوتر Pentium 166Mhz بود بر روی سیستم عامل Linux و به همراه Apache Web Server . بسیار برایم جالب بود که چقدر کد های من سریع اجرا می شوند . یعنی در آن موقع با اگر شما یک Windows NT بر روی آن می توانستید سوار کنید و به فرض که IIS هم بر روی آن بالا می آمد فکر کنم اصلا وقت Serve کردن صفحات عادی html را نداشت چه برسد به اینکه بخواهد ASP را هم اجرا کند . علتش این است که Microsoft از یک Technology در اجرا کردن کدهای زبان ASP استفاده می کند که در آن هر موقع شما تصمیم به استفاده از یک عنصر خارجی مانند VBScript, MSSQL, ODBC و خیلی چیزهای دیگر که در حقیقت از Engine های خارجی استفاده می کنند دستور به آن Engine خارجی می دهد و جواب بدست آمده را بررسی و برای استفاده در اختیار ادامه برنامه می گذارد . همین رفت و برگشت و اجرا کردن Engine های خارجی باعث کند شدن سرویس دهی می شود که این را شما به خوبی می توانید در استفاده از MSSQL به طرق مختلف احساس کنید . مثلا اگر شما خود MSSQL Extensions برای استفاده از MSSQL استفاده کنید برای یک Query مشترک ۱,۸۸ ثانیه زمان تلف می شود و اگر همان را با استفاده از ODBC اجرا نمائید زمانی در حدود ۹,۵۴ ثانیه تلف می شود که این خود نشان می دهد که ASP اینها را به تنهائی اجراء نمی کند و از Engine های ویندوز استفاده می کند .

۲- استفاده بهینه از Memory

در IIS4 اگر شما در یک صفحه مثلا ۲۰ بار یک صفحه را Include کنید این صفحه ۲۰ بار در حافظه بارگذاری می شود و در حقیقت حافظه شما ۲۰ برابر زیادتر اشغال می شود . البته شنیدم که این مشکل در ویندوز ۲۰۰۰ و IIS5 حل شده است اما بازهم برای کسانی که ASP را می‌نویسند و می خواهند آنرا بر روی سرور های

Hosting که دارای سیستم عامل NT هستند اجرا کنند مشکل زا است و باعث کند شدن سیستم می شود و در Load بالا مسلما مشکل زا خواهد شد .

این مشکل به طور کلی در PHP وجود نداشته و ندارد و استفاده درست از Memory در هنگام اجرای یک کد باعث شده است که صفحات در Load بالا نیز به خوبی قابل رویت باشند .

۳- خرج اضافی ندارید !

مثلا در ASP اگر بخواهید از امکاناتی نظیر Encryption یا File Uploading یا ارسال نامه توسط کد برنامه استفاده کنید باید امکانات اضافی برای این کار خریداری کنید و نصب کنید تا این امکانات به IIS شما اضافه گردد . این در حالیست که در PHP همه اینها در هنگام Compile در نظر گرفته می شوند و همگی از امکانات Standard این زبان هستند و هیچ نصب یا خرج اضافی در کار نیست .

۴- MySQL بهترین انتخاب، بیشترین سرعت

در اینجا قصد ندارم به مقایسه MySQL و MSSQL بپردازم . اما به خاطر قدرت خارق العاده MySQL و سازگار بودن این DBMS با زبان PHP به صورتیکه PHP اتصال به MySQL را به صورت دستورات Internally پشتیبانی می کند و حتی نیاز به نصب Module اضافی برای این کار نمی باشد ، از سرعت بسیار بالائی در کار با SQL برخوردار است .

۵- نزدیک بودن Syntax به C/C++ و Java

از آنجایی که اکثر برنامه نویسان از C/C++ استفاده کرده اند و به‌خاطر محبوب بودن بی حد Java معمولا با Syntax های این دو زبان اکثرا آشنا هستند . PHP هم اکثر Syntax های خود را شبیه به این زبانها انتخاب کرده است که برای یادگیری دوباره Syntax دستورات دچار مشکل نشوید که مسلما Microsoft اصلا برایش این مسائل مشکل حساب نمی شود .

۶- رفع ایرادات ، سریع ، بی دردسر

تا حالا از Microsoft خواسته اید که ایرادی را در سیستمهای خود رفع کند ؟ مسلما اگر شرکت بزرگی مانند Boeing نباشید حرف شما خیلی خریدار ندارد یا لااقل به این زودی ها به نتیجه نمی رسید . Open Source بودن PHP این امکان را به شما می دهد که شخصا اقدام به رفع مشکل کنید و آنرا برای دستندرکاران PHP ارسال کنید و یا اینکه در Mailing List های عمومی PHP موضوع را مطرح کنید و خواهید دید که از سراسر دنیا برای رفع ایراد شما Patch ارسال می گردد .

۷- اجرا بر روی Platform های مختلف

درست است که خیلی از این ایرادات را Microsoft رفع خواهد کرد و Technology های جدیدتر ارائه خواهد کرد (چه بسا این Net. که الان آمده همه را درست کرده باشد) اما یک مشکل اساسی برای ASP وجود دارد و آن این است که ASP بدون Windows یعنی هیچ ! بدلیل اینکه ASP نصفی از کدها را توسط Engine های ویندوز اجرا می کند که در سیستم عامل های دیگر خبری از آنها نیست . لذا ASP در سیستم عاملهای دیگر همیشه دارای ضعفهای بزرگی است .

اما PHP به دلیل آنکه توسط GNU C Compiler در همه Platform ها قابل Compile شدن است و از Engine های خاص هیچ سیستم عاملی برای اجرای کدها استفاده نمی کند قابلیت اجرا بر روی تعدا زیادی از OS ها را داراست که این یک مزیت برای برنامه نویس ها محسوب می شود.

PHP چیست؟

با گسترش قابلیت‌ها و موارد استفاده از این زبان، PHP در معنای **'Hypertext Preprocessor'** به کار گرفته شد. (عبارت پیش پردازشگر (preprocessor) بدین معنی است که PHP، اطلاعات را قبل از تبدیل به زبان HTML پردازش می کند.)

مطابق سایت وب رسمی PHP، که در آدرس www.php.net قرار دارد (تصویر شماره ۱)، PHP یک زبان اسکریپت سمت سرویس دهنده (Server-side)، Cross-platform و HTML embedded می باشد.

سمت سرویس دهنده بودن PHP، بدین معناست که تمام پردازشهای این زبان بر روی سرویس دهنده (Server) انجام می گیرد. یک سرویس دهنده، در حقیقت یک کامپیوتر مخصوص می باشد که صفحات وب در آنجا نگهداری می شوند و از آنجا به مرورگر کاربران منتقل می شوند. چگونگی انجام این روند، در ادامه این قسمت، توضیح داده می شوند .

تصویر شماره ۱- نمای سایت وب رسمی PHP می باشد. این سایت که در آدرس www.php.net قرار دارد نخستین مرجع برای یافتن پاسخ سوالات شما می باشد. در این سایت راهنمایی برای آخرین نسخه این زبان ارائه شده است.

منظور از Cross-platform بودن این زبان، این است که بر روی هر سیستم و با هر سیستم عاملی از قبیل Unix و Windows NT، Macintosh و OS/2 اجرا می شود. توجه کنید که منظور از این سیستم عاملها سیستم عامل‌هایی می باشند که بر روی سرویس دهنده نصب می شوند. PHP نه تنها قابلیت اجرا بر روی هر سیستم عاملی را دارا می باشد، بلکه برای منتقل کردن برنامه های آن از یک سیستم عامل دیگر، احتیاج به تغییرات اندکی خواهید

داشت و حتی در بعضی از موارد، بدون احتیاج به هیچ تغییری می توانید، یک برنامهٔ به زبان PHP را از یک سیستم عامل به سیستم عامل دیگر منتقل کنید.

منظور از HTML embedded بودن PHP این است که دستورات این زبان در بین کدهای HTML قرار می گیرند. بنابراین برنامه نویسی به زبان PHP کمی پیچیده تر از برنامه نویسی به زبان HTML، به حساب می آید.

PHP بر خلاف زبانهای برنامه نویسی (programming Languages) یک زبان اسکریپتی (scripting Language) می باشد. به عبارت دیگر بعد از رخداد یک رویکرد (event)، اجراء می‌شوند. این رویدادها می توانند شامل ارسال یک فرم، رفتن به یک URL[1] مشخص و یا موارد دیگر باشند. متداولترین زبان اسکریپتی، زبان Java Script می باشد که معمولا برای پاسخ به رویدادهای کاربر در مرورگر وب، به کار می‌رود.

تفاوت عمدهٔ Java Script با PHP در این است که Java Script یک تکنولوژی سمت سرویس گیرنده (client-side) می باشد. زبانهایی مانند Java Script یا PHP، تفسیر شونده (interpreted) نامیده می شوند. به عبارت دیگر برای اجرا به یک مفسر مانند مرورگر وب احتیاج دارند اما زبانهای برنامه نویسی مانند C یا Java بعد از ترجمه به زبان ماشین (Compile) به خودی خود قابل اجرا می باشند.

همانطور که گفته شد جدیدترین نسخه PHP نسخه 4.0 این زبان اسکریپتی می باشد و در این مقاله به بررسی این نسخه از PHP خواهیم پرداخت. اما به دلیل جدید بودن این نسخه اکثر سرویس دهنده ها از نسخه 3.x استفاده می کنند. تفاوت این دو نسخه PHP بسیار اندک می‌باشد و تغییرات مهم، عموما در مسیر اصلاح عملکرد این زبان صورت گرفته‌اند.

به چه دلیل از PHP استفاده می کنیم؟

PHP در مقایسه با تکنولوژی‌های مشابه، سریعتر، بهتر و آسانتر است. از جمله تکنولوژی های مشابه برای طراحی یک سایت وب می‌توان به این موارد اشاره کرد:

CGI (Common Gateway Interface)، که معمولا به زبان perl نوشته می‌شوند.

ASP (Active Server Pages).

JSP (Java Server Pages).

1 URL (Uniform Resource Locator)- لغتی که در حقیقت برای آدرسهای وب به کار می رود.

Java Script به عنوان یک گزینهٔ جایگزین برای PHP در نظر گرفته نمی‌شود. زیرا بر خلاف PHP، یک تکنولوژی سمت سرویس گیرنده است. و همانند تکنولوژی هایی مانند CHT و PHP نمی‌تواند یک صفحهٔ HTML را تولید کند.

مزیتی که PHP در مقابل HTML دارد این است که HTML یک سیستم محدود به حساب می آید و توانایی ایجاد ارتباط با کاربر را ندارد، کاربران هنگامی که با صفحهٔ HTML مواجه می شوند، تنها یک صفحهٔ ساده را روبروی خود مشاهده می کنند که توانایی ایجاد واکنش به اعمال کاربر را ندارد. اما با استفاده از PHP، شما می توانید صفحاتی را ایجاد کنید که برای مثال عناصر موجود در صفحه بر اساس سیستم عامل کاربر و یا تاریخ مشاهدهٔ صفحه تنظیم شوند. همچنین PHP می تواند با فایل‌ها یا پایگاههای داده (Database) ارتباط برقرار کند و بسیاری عملیات دیگر که HTML قادر به انجام به آنها نمی باشند.

طراحان صفحات وب، از مدتها پیش به این نتیجه رسیدند که اگر در صفحات خود، فقط از کدهای HTML استفاده کنند، باید به طور مرتب آنرا تغییر دهند و اطلاعات آنرا به روز کنند. به همین علت تکنولوژی‌هایی مانند CGI از همان آغاز، طرفداران بسیاری پیدا کردند. این تکنولوژی ها به طراحان این امکان را می‌داد که برنامه هایی ایجاد کنند که به صورت دینامیک، صفحات وب را تولید کنند. همچنین در ارتباط با پایگاه داده، بدون دخالت هر انسانی، صفحات به روز آوری می شدند.

بنابراین سوالی که اکنون به ذهن می رسد، این است که به چه علت یک طراح صفحات وب بهتر است که از زبان PHP به جای زبان‌هایی مانند CGI و ASP و JSP، برای طراحی سایت های پویا یا دینامیک، استفاده کند؟

دلیل اول: سرعت بیشتر PHP چه در برنامه نویسی و ایجاد برنامه‌هایی به این زبان و چه در اجراء می باشد. همچنین PHP برای یادگیری بسیار ساده می‌باشد و افراد بدون نیاز به زمینه های قبلی در برنامه نویسی و تنها با مطالعهٔ همین مقاله، می توانند به زبان PHP اسکریپت نویسی کنند. در مقابل، ASP احتیاج به دانستن زبانهای VBScript و CGI (که نیازمند زبانهایی مانند Perl یا C می باشد) دارد و هر دو این زبانها، زبانهای کاملی هستند که یادگیری آنها نیز آسان نیست.

دلیل دوم این است که PHP، به صورت اختصاصی، تنها برای ایجاد صفحات دینامیک طراحی شده است. اما Perl، VBScript و یا Java اینگونه نیستند و به همین علت PHP سریعتر و ساده تر از تکنولوژی های جایگزین

کننده می‌باشد. توجه کنید که این صحبت‌ها هرگز بدین معنا نیست که PHP یک زبان کاملتر و یا یک زبان بهتری نسبت به Java، ASP، Perl و یا CGI می باشد. بلکه ما تنها پیشنهاد می کنیم که در زمینه های خاصی مانند آنچه که اشاره شد، از PHP استفاده شود.

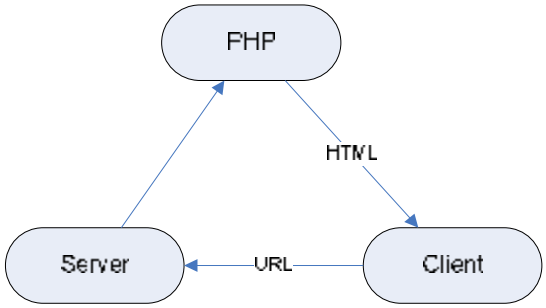
به عنوان آخرین مطلب در مورد برتریهای PHP، تنها به ذکر این مطلب بسنده می کنیم که هم‌اکنون بیشتر از چهار میلیون از سایت های وب از تکنولوژی PHP استفاده می کنند.

PHP چگونه کار می کند؟

همانگونه که اشاره شدPHP یک زبان سمت ***سرویس دهنده*** می باشد و این بدین معنی است که کدهای نوشته شده به این زبان در کامپیوتر میزبان (host) صفحات وب قرار می گیرد.

برای مثال، هنگامی که شما به سایت وب www.dmcinsights.com می‌روید،ISP (Internet Service Provider) شما، درخواست (request) شما را به سرویس‌دهنده‌ای که اطلاعات این سایت را نگهداری می کند، ارسال می‌کند.

در این هنگام سرویس‌دهنده بعد از خواندن کدهای PHP، آنها را پردازش می‌کند. برای مثال در این مورد، PHP به سرویس‌دهنده فرمان می دهد که اطلاعات یک صفحهٔ وب را به صورت برچسب‌های HTML به مرورگر شما منتقل کند. (تصویر شماره ۲)



تصویر شماره ۲: این نمودار چگونگی ارتباط بین سرویس‌گیرنده یا کاربر (client)، سرویس‌دهنده (Server) و مدل PHP را نمایش می دهد. در این حالت مدل PHP، برنامه‌ای است که سرویس‌دهنده برای افزایش کارایی آن، قرارگرفته است.

تمام تکنولوژی‌هایی سمت سرویس‌دهنده (مانند ASP) از چنین مدل طرف ثالثی (third-party) برای پردازش اطلاعات و برگرداندن نتایج به سرویس گیرنده، استفاده می‌کنند.

این حالت با هنگامی که صفحه از ابتدا با کدهای HTML طراحی شده باشد، تفاوت دارد. در حالت دوم تنها یک درخواست به سرویس‌دهنده ارسال می‌شود و سرویس‌دهنده نیز اطلاعات HTML موجود را به مرورگر کاربر منتقل می کند. (تصویر ۳)

تصویر شماره ۳- این نمودار، ارتباط مستقیم بین سرویس گیرنده و سرویس‌دهنده را در هنگام استفاده از صفحات معمولی HTML نشان می دهد. این نمودار را با نمودار ارائه شده در تصویر ۲ مقایسه کنید. اکنون می توانید متوجه شوید که چرا یک صفحهٔ HTML ساده را می‌توانید در کامپیوتر خود نیز مشاهده کنید ولی برای صفحاتی که به صورت دینامیک تولید می‌شوند، احتیاج به یک سرویس‌دهنده می‌باشد.

بنابراین، برای مرورگر کاربر تفاوتی بین home.html و home.php وجود ندارد. اما تفاوت عمده‌ای بین این دو حالت وجود دارد و آن اینست که در حالت اول صفحه به صورت دینامیک توسط سرویس‌دهنده تولید شده است و برای مثال می‌توان اطلاعات متفاوتی را در روزهای دوشنبه و یا سه شنبه ارائه داد و یا بین حالتی که کاربر قبلا صفحه را مشاهده کرده باشد و یا نکرده باشد، تفاوت قائل شد. بنابراین، هر آنچه PHP انجام می‌دهد در همان سمت سرویس‌دهنده انجام می دهد و سپس اطلاعات مناسب را به سمت سرویس‌گیرنده منتقل می‌کند.

نصب و پیکر بندی (قسمت اول)

در این بخش به بررسی نصب و پیکر بندی php بر روی دو سیستم عـــامل linux و Windows خواهیم پرداخت. اولین چیزی که باید به آن پداخت این است که ما از چه نوع سیستم عـــاملی استفاده می کنیم یعنی سیستم عاملی که ما از استفاده می کنیم قـابلیت نصب php را دارد یا نه؟

همچنین هدف این است که ما یاد بگیریم که چطـوری می توانیم برنامه هایی که به زبان php می نویسیم بر روی سیستمی که در حال کار با آن می باشیم تست و اجرا کنیم و بعد به سیستم دیگری انتقال دهیم احتمالا همـــان سرور است.

اولین کاری که باید انجام بدهیم این است که از نرم افزاری استفاده کنیم که قـابلیت تبدیل کامپیوتر ما به وب سرور را داشته باشد.

روش اول اجرای php بر روی Windows خواهد بود و بعدا نصب بر روی Linux را هم فرا خواهیم گرفت.

سه راه برای اینکه PC شما به یک وب سرور (که قابلیت پشتیبانی PHP را داشته باشد) تبدیل شود وجود دارد:

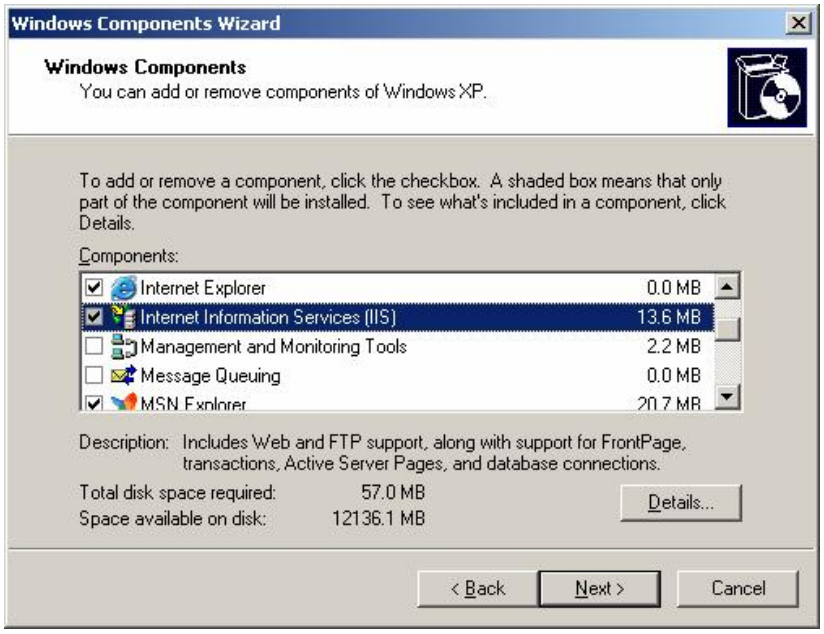
" اگر شما با ویندوزی غیر از XP یا NT یـــا ۲۰۰۰ کار می کنید باید از راه اول استفاده کنید و اگر نه باید از راه دوم استفاده کنید راه سوم را هم می توان بـــرروی تمامی ویندوز ها استفاده کرد فقط یک نکته کـــه باید روی ویندوز نسخه های XP یا NT یا 2000 - IIS رو **غیر فعال کنید** که بتوانید استفاده کنید! "

در ابتدا به توضیح راه دوم خواهیم پرداخت که روش استاندارد استفـــــاده از php در windows می باشد. ما در این روش از IIS استفاده می کنیم. IIS مخفف (**Internet Information Server**) می باشد که با کمک آن می توان سرویس هایی از قبیل www و همچنین ftp که مربوط به دریافت فایل می شود و همچنین چندین سرویس دیگر را استفـــاده کرد که البته خارج از بحث ما هست. IIS در حال حاضر در دو نسخه پرکاربرد ۴ بـــرای ویندوز NT و برای ویندوز های XP و ۲۰۰۰ وجود دارد.

روش نصب IIS:

این روش نصب IIS در ویندوز های XP و NT و 2000 تقریبـــا به یک شکل می باشد و خواهید توانست که با یاد گرفتن یکی از آنها IIS را در ویندوز هـــای مختلف نصب کنید.

برای نصب IIS ابتدا باید از منوی START گزینه Settings و در نهـــایت گزینه Control Panel را انتخاب کنید تا پنجره موسوم به کنترل پنل باز شود سپس از پنجـــــره کنترل پنل گزینه Add or Remove Programs را انتخاب کرده و آن را اجرا نمایید بعد از بـــاز شدن پنجره Add or Remove Programs از کلید های سمت چپ گـزینه Add/Remove Windows Components را انتخاب کرده و بعد از اندکی صبر پنجـــــره Windows Components Wizard باز می‌شود بعد از باز شدن از کـــادر Components گزینه Internet Information Server(IIS) را تیک بزنید کنید. به تصویر زیر دقت کنید:



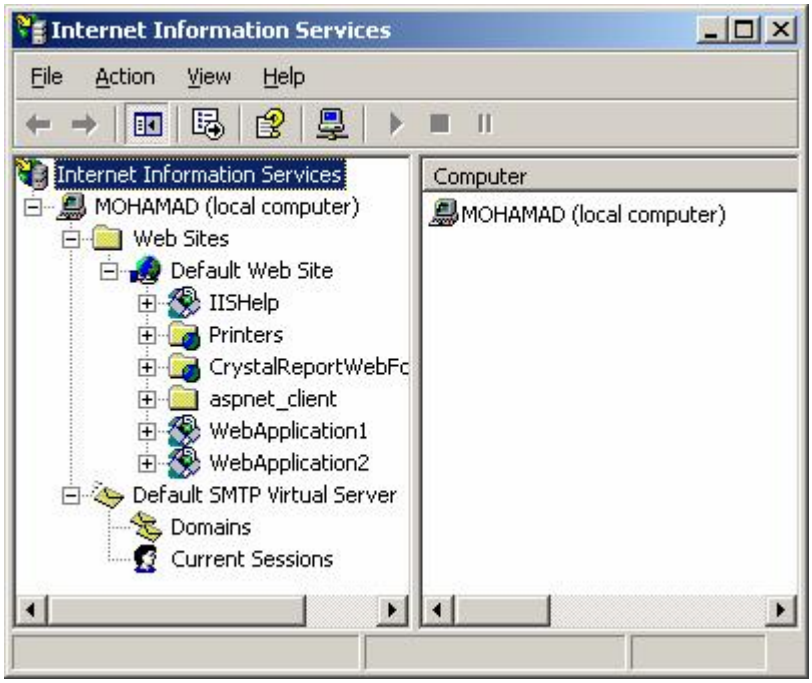
توجه : چنانچه رنگ زمینه Check Box گزینه فوق تیره بود بدین مفهوم است که زیر گروه های ایـن گزینـه غیر فعال می باشد و باید چک دار شوند برای چک دار کردن آنها باید بروی آن گزینه دوبار کلیک کرده و از پنجره ای که باز خواهد شد گزینه هایی کـــه فعال نمی باشد فعال نمایید تا کلیه سرویس های یا زیرگروه های به طور کـــامل انتخاب و نصب شود.

بعد از انتخاب گزینه مورد نظر کلید Next را فشار داده تا به مرحله بعد نصب بروید. در این مرحله گزینه های مرحله قبل مورد پردازش قرار می گیرد و کلیه تغیرات اعمـــــال می شود . چنانچه شمـــا گزینه ای را حذف (غیر فعال) کرده باشید در این قسمت از سیستم پاک خواهد شد و چنانچه گزینه ای را فعال (انتخاب) کـــــــرده باشید در این قسمت به سیستم اضافه خواهد شد.

توجه: چنانچه گزینه ای را فعال کرده باشید در این مرحلـــــه احتیاج به CD نصب ویندوز مورد نظر خواهید داشت یا اگر فایل هـــای نصبی ویندوز را بروی سیستمتان داشته باشید به آن احتیاج پیدا خواهید کرد چون باید فـــایل های مربوط به پیکربندی IIS را از CD و یا Hard Disk خوانده شود و بر روی سیستم شمـــــا کپی گردد.

بعد از اتمام این مرحله , نصب به مرحله پایانی خواهد رسید و در این قسمت شما باید دکمه Finish را فشار داده و بعد از اندکی صبر هم اکنون IIS بـــــر روی سیستم شما نصب می باشد و شما می توانید از آن استفاده کنید.

حال بعد از نصب IIS نوبت به پیکربندی IIS می‌رسد. این عمل باعث می شود که بتوانید از IIS استفاده نمایید. برای پیکر بندی IIS شما باید به Control Panel رفته و گـــزینه Administrative Tools را انتخاب کرده و از پنجره Administrative Tools گزینه Internet Information Server را انتخاب کرده و بعد از اجرای این برنامه گزینه های مربوط به پیکـــربندی IIS در پیش روی شماست و شما می توانید IIS خود را منطبق بر میل خود پیکـــربندی کنید. (چون پیکربندی IIS خارج از بحث مـــاست پس از توضیح آن خودداری خواهیم کرد). به تصویر زیر در همین ارتباط توجه نمایید.



حال که نصب IIS را فرا گرفتیم و IIS بر روی سیستم شمـــــا نصب شده است باید آن را برای استفاده از PHP آماده کنیم. برای این کـــــــار احتیاج به نصب نرم افزار PHP بر روی سیستم داریم که در ادامه روش نصب PHP را فرا خواهیم گرفت.

برای نصب PHP ابتدا باید نسخه مورد نظر PHP را تهیه کنید و ترجیحا از آخـــرین نسخه این نرم افزار استفـــاده کنید که نسخه 4.4.2 این نرم افزار می باشد می توانید از آدرس زیر دریافت کنید.

http://www.php.net/downloads.php

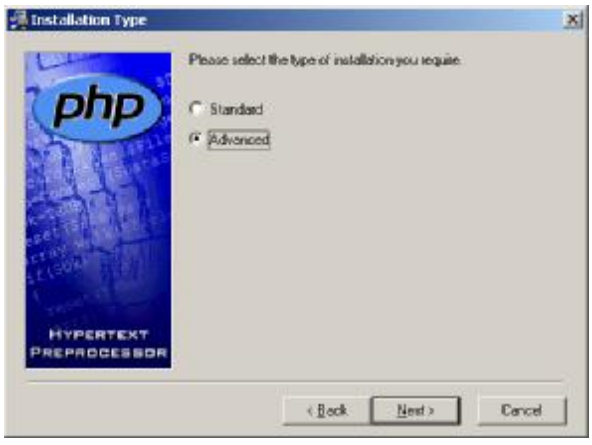
بعد از دریافت نسخه مورد نظر شما باید مراحل زیر را برای نصب دنبال کنید. ابتدا بر روی فایل اجرایی PHP کلیک کرده و آن را اجرا نمایید (معمولا فایل اجرایی PHP با نـام php-4.3.0-installer می باشد)

بعد از باز شدن پنجره installation php 4.3.0 بعد از کمی صبر پنجره Welcome همانند تصویر زیر بـــاز خواهد شد.

سپس دکمه Next را فشار دهید تا به مرحله بعدی Wizard کنتـــــــرل انتقال یابد.

بعد از فشار دکمه Next پنجره License Agreement باز خواهد شد در این پنجره باید دکمه I Agree را انتخاب کنید تا موافقت نامه PHP مورد تایید شما قرار گیرد.

بعد از تایید پنجره Installation Type را خواهید دید کـــه دارای دو گزینه Standard و Advanced می باشد که شما گزینه Advanced را چک دار کنید (البته لازم به ذکـــر است که در موقعی که شما گزینه Advanced را انتخاب می کنید تنظیمـــــات پیکربندی بیشتری نسبت به گزینه استاندارد در اختیار دارید!)



بعد از فشار دادن دکمه Next پنجـــــــره موسوم به Choose Destination Location را مشاهده خواهید کرد که در این پنجره می توانید مسیر نصب فـــــایل های PHP را مشخص کنید.

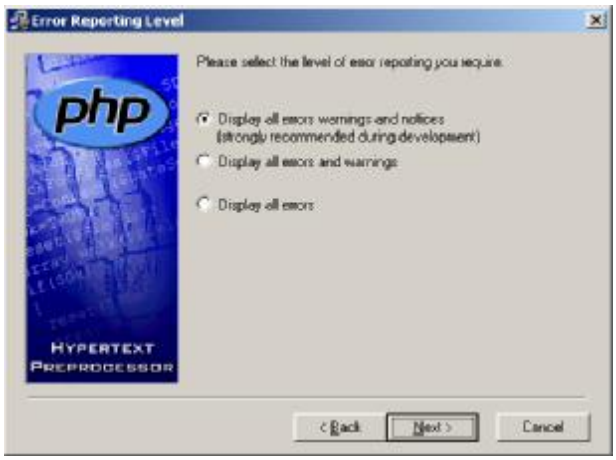
با فشار دادن دکمه Next پنجره Backup Replaced Files را مشاهـده خواهید کرد که شما در این پنجره می توانید محل قرار گیری فایل های Back up را مشخص کنید. همچنین می توانید به PHP بگویید آیا برای فایل های شما Back up تهیه کند یـــــــا خیر؟

بعد از فشار دادن دکمه Next پنجره Directory Choose Upload Temporary نمـــایش داده خواهد شد که در این اینجا محل قرار گیری فایلهای موقتی که برای اجرای برنامه های PHP به آن احتیاج دارد مشخص می شود.

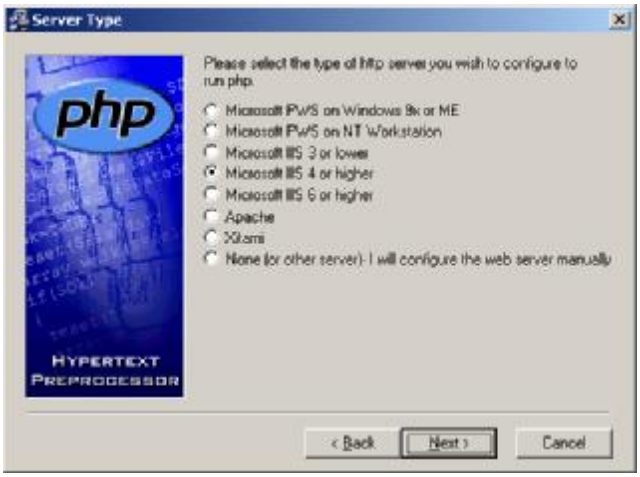
با فشار دکمه Next پنجره Choose Session Save Directory باز خواهد شد که شما می توانید محل ذخیره کردن متغییر های Session (در بخش های بعد توضیح داده خواهد شد) را مشخص کنید. بعد از فشار دکمه Next پنجره Mail Configuration باز خواهد شد که شمـــــا باید تنظیمات مربوط به Smtp

Server و ایمیل آدرس پیش فرض را وارد کنید (در صورتی که به این گزینه آشنایی ندارید می توانید تنضیمات پیش فرض را قبول کـــرده و بدون اعمال تغییرات کلید Next را فشار دهید)

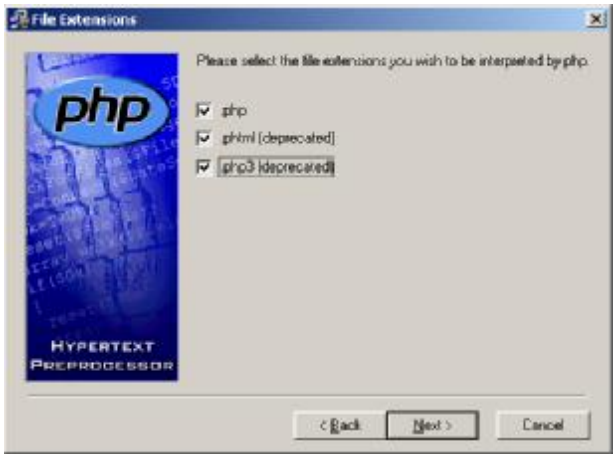
با فشار دکمه Next پنجره Error Reporting Level پدیدار خواهد شد که شمـــــا می توانید سطح گزارشـــات خطاهای احتمالی که در برنامه های به وجود می آید مشخص کنید که در اینجا شما بهتر است تنظیمات پیش فرض را قبول کرده و بـــه مرحله بعدی بروید.



سپس با فشار دکمه Next پنجره Server Type را مشاهده خواهید کـــــــــرد در این پنجره شما باید نوع Web server سیستمتان را به PHP معرفـی کنید در این جا شما باید گزینه Microsoft IIS 4 or Higher را انتخاب نمایید چون از ویندوز هـــای XP و NT و ۲۰۰۰ استفاده می کنید .



بعد از فشار دکمه Next به پنجره Extensions File خـــواهید رسید که در این قسمت شما امکان این را خواهید داشت که برای WebServer تان مشخص کنید کـــــه چه نوع فایلهای را برای اجرا اسکریپت های PHP در نظـــــر بگیرد.(در این مرحله بهتر است تمام ۳ گزینه را انتخاب کنید.)



بعد از فشار دادن کلید Next , پنجره Start Installation باز خواهد شد که از شمـــــا اجازه نصب PHP و کپی کردن فایل های را روی سیستم شما را خواستار است کـــه شما با فشار کلید Next به آن این اجازه را خواهید داد.

بعد از این کار پنجره مربوط به Installing باز خواهد شد که شما از عمل کـــــرد کپی فایل ها مطلع خواهید شد.

بعد از اتمام این مرحله چنان چه فایل "php.ini" قبـلا در دایرکتوری System32 شما وجود داشته باشد پیغامی مبنی بر اینکه این فایل قبـــلا وجود دارد و شما چنانچه مایل هستید این فایل پاک شود و نسخه جدید فـــــــایل را جایگزین کند که گزینه ok را برای تایید کلیک کنید.(توجه داشته باشید این گـزینه در صورتی نمایش داده می شود که فایل مورد نظر وجود داشته باشد)

بعد از اتمام این مراحل پنجره IIS Script tamp Node Selection را مشـــاهده خواهید کرد که شما باید در این قسمت کلید Select All را فشار داد و دکمه ok را بزنید.

در اینجا نصب PHP به پایان رسید و با پیغام تبریک و موفقیت شمـا در نصب PHP مواجه خواهید شد و با فشارداد کلید Ok آن را تایید کنید.

نصب و پیکربندی قسمت دوم

در بخش قبل نصب و پیکر بندی php را در windows های XP و ME و 2000 توضیح داده شد حال به بررسی چگونگی نصب پی اچ پی بر روی سایر ویندوزها و طریقه استفاده از آن خواهیم پرداخت. بهترین روش برای این کار استفاده از نرم افزارهایی هست که عمل یک وب سرور را شبیه سازی می کنند مثل PWS یا Easy PHP.

روش اول نصب PWS [1] یکی از محصولات شرکت Microsoft می باشد که بروی ویندوزهای غیر از XP و NT و 2000 کاربرد دارد و برای برنامه نویسان وب بسیار آشنا است. شما با کمک این نرم افزار می توانید سیستم عامل ویندوزتان را به یک وب سرور تبدیل کنید و از آن بهره لازم را ببرید. ما در این جا برای اجرای PHP از PWS کمک می گیریم پس اول باید یاد بگیریم که چطوری می توانیم یک PWS را نصب نماییم.

طریقه نصب PWS:

برای نصب باید ابتدا بروی فایل Setup.exe کلیک کرده و آن را اجرا کنیم.بعد از اجرای برنامه Setup پنجره Setup is initializing باز خواهد شد که باید کمی صبر کنید تا برنامه نصب خود را برای اجرای Wizard نصب آماده کند.

سپس پنجره Microsoft Personal Web Server Setup باز خواهد شد که اطلاعاتی در مورد نرم افزار PWS به شما می دهد و همچنین توضیحات مختصری در مورد این برنامه.

بعد از فشار دکمه Next شما می توانید به مرحله بعدی بروید که در این مرحله پنجره Microsoft Personal Server Setup Web با سر فصل End User License Agreement باز خواهد شد که در این مرحله توضیحاتی در مورد Pack برنامه داده شد و تایید نامه ای برای کپی رایت نرم افزار که با فشار دادن دکمه Accept می توانید به مرحله بعد بروید.

در این مرحله شما باید یکی از سه حالت نصب را انتخاب کنید که شما در این قسمت گزینه Typical را انتخاب نماید (دو گزینه دیگر در این مرحله گزینه Minimum برا نصب برنامه به صورت فشرده می باشد که در این گزینه از حداقل امکانات استفاده می شود و گزینه Custom برای این منظور است که کاربر بتواند خود نسبت به نصب Components های برنامه به صورت دستی اقدام نماید . گزینه Typical حالت استاندارد نصب می باشد.)

بعد از فشار دادن دکمه Typical پنجره ...Microsoft Personal Web Server Version باز خواهد شد که مسیر Root اصلی را باید در این مرحله مشخص کنید.(منظور از روت اصلی هما شاخه www می باشد که شما باید

---

1 Personal Web Server

فایل های ASP یا PHP را برای اجرا در این شاخه قرار دهید تا بتوانید آنها را از طریق مرورگر اجرا کنید.) در این مرحله شما می توانید با استفاده از گزینه Browse برای تغییر مسیر فایل اقدام کنید. دو کادر دیگری که در این قسمت غیر فعال می باشد مربوط به سرویس FTP می باشد که ما به آن احتیاج نداریم.(برای فعال کردن آنها می توانید از گزینه Custom استفاده کنید.)

بعد از تعیین مسیر Root با فشار دکمه Next به مرحله بعدی کنترل را انتقال داد تا پنجره ای با سرفصل Completing Installation باز شود در این مرحله شما از روند کپی و نصب فایل ها بر روی سیستم اطلاع پیدا خواهید کرد.

بعد از اتمام این قسمت Wizard نصب پایان یافته و PWS با تشکر کردن از شما در این پنجره برای انتخاب این نرم افزار از شما می خواهد که با فشار دکمه Finish به برنامه نصب خاتمه دهید.

بعد از فشار دکمه Finish این پنجره را خواهید دید که عمل تنظیمات را بر روی سیستم شما اعمال می کند.

اکنون PWS بر روی سیستم شما نصب شده و شما می توانید از آن استفاده کنید.

حالا برای اینکه بتوانیم از PHP استفاده کنیم باید PHP را بر روی PWS نصب کنیم

برای این کار کافیست فقط در پنجره Server Type گزینه Microsoft Pws On Windows 9x or ME را انتخاب کنید و چنانچه از ویندوز Workstation NT استفاده می کنید گزینه Microsoft PWS on NT Workstation را انتخاب کنید و دیگر در احتیاج به تغییرات در جای دیگه ای وجود ندارد.حالا شما با موفقیت PWS را نصب کردید و PHP روی آن فعال شده است .

نرم افزار Easy PHP:

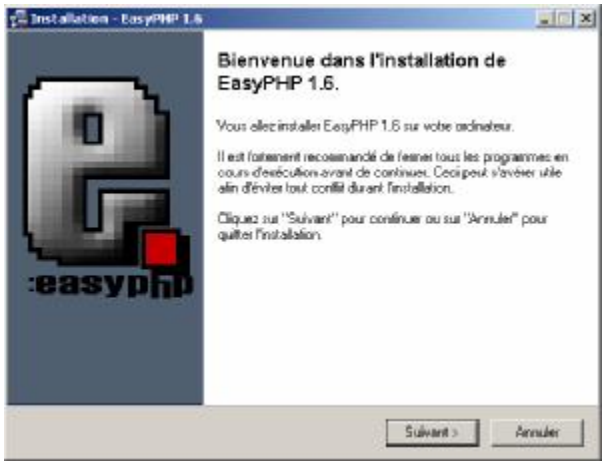
در این قسمت می خواهیم در مورد چگونگی استفاد از نرم افزار Easy PHP صحبت کنیم. در این قسمت مرحله سوم یا آخرین مرحله نصب و پیکربندی PHP را یاد خواهید گرفت. در ابتدا کمی درباره اینکه Easy PHP چیست و چکار می کند توضیح خواهیم داد. این نرم افزار یک شبیه ساز وب سرور هست که می تواند بروی کامپیوتر شما بدون نیاز به IIS و PWS برنامه‌های PHP را با استفاده از کاوشگر اینترنت اجرا کند.

همچنین این نرم افزار امکان استفاده از بانک اطلاعاتی مورد استفاده در PHP را به شما می‌دهد , در آینده بیشتر در مورد بانکهای اطلاعاتی صحبت می کنیم! برای نصب و پیکربندی Easy PHP ابتدا باید آنرا از آدرس زیر دریافت کنید و بعد مراحل زیر را برای نصب طی کنید.

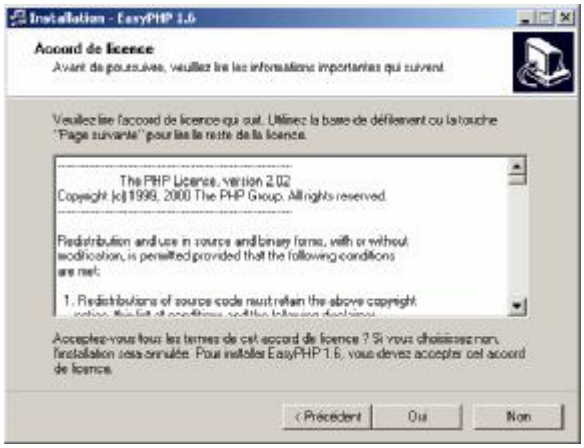
http://easyphp.abbal.com/depot/easyphp1-6_setup.exe

با کلیک کردن روی فایل اجرایی "easyphp1-6_setup" می توانید Wizard نصب را اجرا کنید. با اجرای فایل نصب پیغامی را مشاهده خواهید کرد که در آن از شما برای نصب نرم افزار Easy PHP اجازه کسب می کند که شما با زدن دکمه YES کادر را تایید کرده و کار نصب را ادامه می دهید.

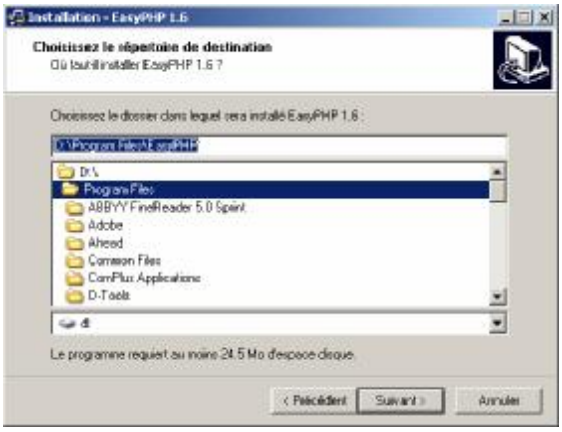
سپس پنجره‌ای باز خواهد شد که به شما اطلاعاتی در مورد نرم افزار Easy PHP می‌دهد و شما می توانید با زدن دکمه Suivant به مرحله بعد بروید.



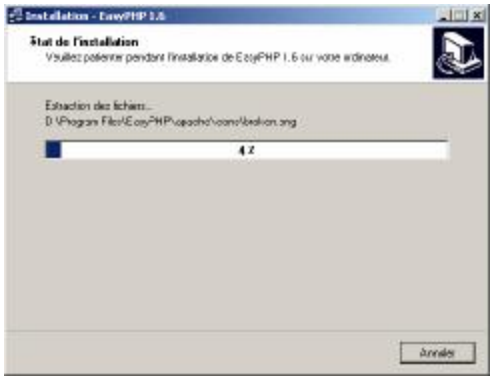
سپس پنجره Accord de License باز خواهد شد که شما با فشار دکمه Oui می توانید به مرحله بعدی بروید.



در این مرحله از Wizard نصب مسیری که فایل های Easy PHP قراراست در آنجا کپی شوند را به شما نشان خواهد داد که شما می توانید این مسیر نصب را عوض کنید و با فشار دکمه Suivant> به کار خود ادامه دهید.



در مرحله بعد محلی که برای قرار گرفتن میانبر های Easy PHP در Programes را مشخص می کند که شما می تونید با فشار دکمه Suivant> به Wizard نصب ادامه دهید و به مرحله بعدی بروید. در این مرحله از شما برای کپی کردن فایلهای Easy PHP اجازه می خواهد که شما با فشار دکمه Installer این کادر را تایید می کنید. حال شما شاهد کپی شدن فایلها در مسیر تعیین شده هستید و باید اندکی صبر کنید تا عمل کپی انجام شود.



بعد از اتمام کپی فایل از شما می خواهد که سیستم را دوباره راه اندازی کنید که شما با فشار دکمه Terminer اجازه این کار را به برنامه خواهید داد.

حالا بعد از دوباره راه اندازی سیستم در قسمت System Try سیستم شمایل Easy PHP نمایش داده خواهد شد و شما هم اکنون می توانید با استفاده از مرورگر خود برنامه های PHP را اجرا نمایید.

آغاز کار با PHP

برای شروع کار با PHP چه نیاز داریم؟

PHP برای اجرا نیاز به یک Web-Server دارد. ساده ترین Web-Server برای شما احتمالا IIS خواهد بود. (در بخش قبل طریقه نصب را بررسی کردیم).

طبیعتا برای مشاهده نتیجه اجرای فایلهای PHP نیاز به یک مرورگر وب (Web Browser) دارید که مسلما رایجترین آن Internet Explorer است. برای درست کردن هر فایل PHP هم نیاز به یک ویرایشگر ساده متنی دارید (مثلا Wordpad یا Notepad) توجه کنید که از ویرایشگرهای حرفه ای مانند Microsoft Word نمی‌توانید استفاده کنید زیرا این ویرایشگرها از کاراکترهای پنهانی فراوانی استفاده می کنند که در هنگام ذخیره فایل متنی این کاراکترها هم ذخیره خواهند شد که باعث عدم اجرای دستورات PHP می شوند. با هم یک کد ساده PHP بنویسیم.

حالا می خواهیم اولین کد PHP را با هم درست کنیم. ویرایشگر متن را باز کنید (مثلا Word pad) و در آن چنین بنویسید:

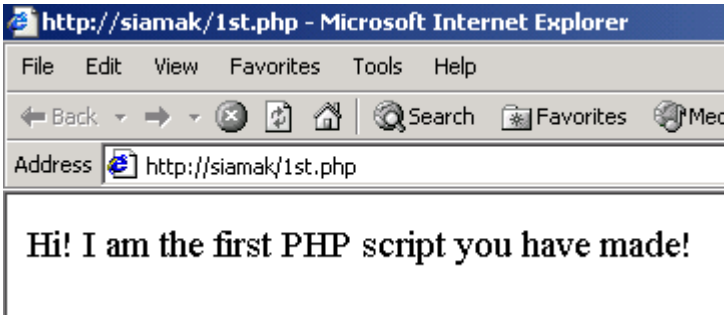
```
<?php
echo (“Hi! I am the first PHP script you have made!”);
?>
```

حالا فایلتان را با عنوان 1st.php در Web Server Local Folder ذخیره نمایید (اگر از IIS استفاده می کنید این پوشه در درایوی که Windows را در آن نصب کرده اید، با نام Inetpub وجود دارد. در داخل آن فولدر دیگری به نام wwwroot وجود دارد که باید فایلتان را در آن بریزید.)

حالا مرورگر وب خود را باز کنید (مثلا Internet Explorer) و در قسمت آدرس آن ابتدا http:// و سپس نام کامپیوتر خود و یا آدرس IP آن را تایپ نمایید (نام کامیوتر را در Control panel/System/Network Identification و IP را در صورتیکه تعیین کرده باشید در قسمت Command Prompt و با اجرای دستور ipconfig می توانید مشاهده کنید) و پس از یک Backslash (/) نام فایل (1st.php) را تایپ نموده و Enter را بزنید. به‌عنوان مثال نام کامپیوتر من simak است. پس باید در قسمت آدرس مرورگر آدرس زیر را بنویسم:

http://simak/1st.php یا http://localhost/1st.php

اگر همه چیز را تا اینجا درست انجام داده باشید، متن زیر در مرورگرتان پدیدار می شود:



همانطور که احتمالا متوجه شدید، باید کد PHP خود را در داخل تگ <?php ?> قرار دهید. البته می توانید از روشهای دیگر نیز استفاده کنید اما متداولترین روش همین است.

دستور ()echo در PHP وظیفه چاپ در خروجی را بر عهده دارد.

می توانستیم همین کد را با کمک تگهای HTML و به‌صورت زیر نیز بنویسیم:

```
<html>
 <body>
  <?php
     echo ("Hi! I am the first PHP script you have made!");
  ?>
 </body>
</html>
```

توضیحات برنامه نویس در حین برنامه نویسی (Comments):

مانند تمام زبانهای برنامه نویسی دیگر، می توانید توضیحات خود را برای آسان کردن رجوع های بعدی در PHP داشته باشید. به این منظور می توانید از // یا # برای بازداشتن PHP از انجام پردازش بر روی متن روبروی آن

استفاده کنید. اگر می خواهید بیش از یک خط را زیر پوشش Comment خود قرار دهید، آن را در بین علامتهای /* */ قرار دهید.

```
<?php
// This line will not be parsed in PHP
# This line will not be parsed in PHP like the line above
/* Line number1: These 3 lines will not either!
Line number2
Line number3: End of comment */
?>
```

متغیرها

در فصل قبل با چگونگی ارسال متن های ساده و HTML به مرورگر، توسط PHP، آشنا شدید. اما شما برای این کار لزوما احتیاج به استفاده از PHP، نداشتید. بنابراین برای استفادهٔ صحیح از PHP، باید چگونگی استفاده از تابع **()print** به همراه ویژگیهای دیگر PHP ، آشنا شوید.

برای تبدیل صفحات ساده و ثابت به برنامه های دینامیک و سایت های جذاب، در ابتدا، شما احتیاج به این خواهید داشت که بتوانید اطلاعات را در اختیار بگیرید. متغیرها، همان ابزاری هستند که شما با استفاده از آنها می-توانید، اطلاعات را در اختیار بگیرید و آنها را در دسترس خود قرار دهید. متغیرها یکی از مهمترین ابزارها و مفاهیم هر زبان برنامه نویسی، محسوب می شوند.

متغیرها به شما اجازه می‌دهند که داده ها را به طور موقت ذخیره کنید و یا آنها را تغییر دهید. از مسائلی که در مورد متغیرها باید مورد توجه قرار گیرند می‌توان به درک مفهوم متغیرها، انواع مختلف آنها که توسط زبان برنامه نویسی پشتیبانی می‌شوند، و چگونگی استفاده از آنها، اشاره کرد. در این فصل مفاهیم پایه متغیرها در PHP بررسی می‌شوند و در آینده به طور مشخص، کاربردهای مختلف انواع متغیر توضیح داده می‌شوند.

در زبان PHP باید قبل از هر متغیر یک علامت $ قرار دهید. در واقع PHP از روی علامت $ تشخیص می دهد که متغیرهای برنامه شما کدامند.

اگر تاکنون با زبانهای برنامه نویسی مانند C، C++، Pascal و مانند اینها کار می کرده اید، احتمالا انتظار دارید انواع مختلف متغیرها اعم از Integer، Real، Float و غیره را برایتان معرفی کنم. اما احتمالا خوشحال خواهید شد که بدانید **نوع متغیرها برای PHP اهمیت ندارد**. کافیست متغیری را نامگذاری کنید و سپس هر مقدار، از هر نوع و به هر اندازه که می خواهید در آن قرار دهید. سوالی که پیش خواهد آمد احتمالا "نحوه عمل PHP با این متغیرهای همه منظوره" است.

قبل از اینکه به مثال برسیم باید توجه شما را به دو نکته جلب کنم:

۱- متغیرها در PHP نسبت به کوچکی و بزرگی حساسیت دارند (Case Sensitive). یعنی به عنوان مثال متغیرهای my_var و My_Var و MY_VAR از نظر زبان PHP با هم متفاوت هستند.

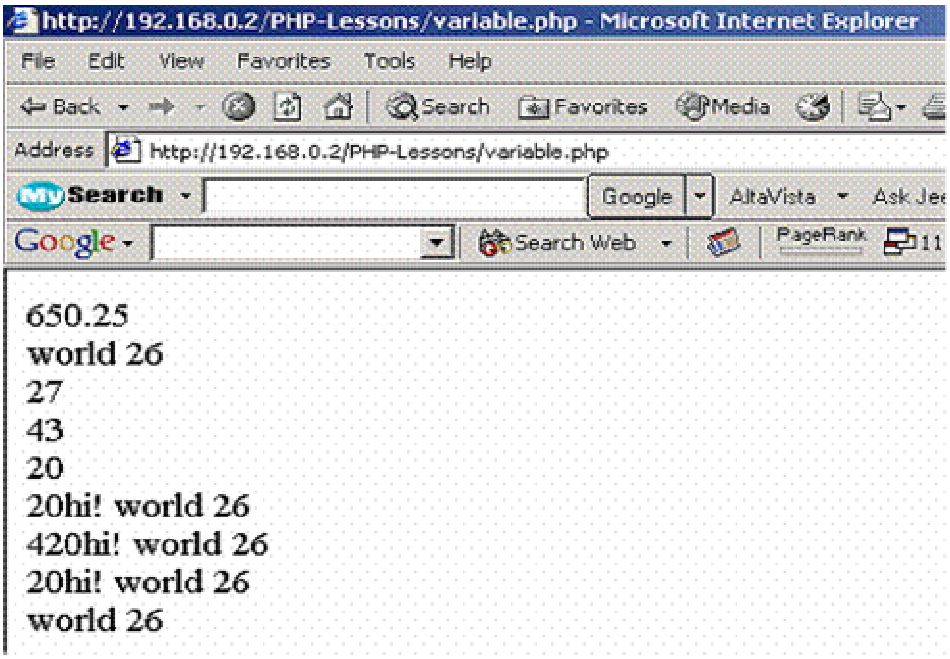
۲- متغیرها می توانند نامهایی با حروف کوچک و بزرگ انگلیسی و همینطور Underscore (خط فاصله پایین) را اخذ نمایند. استفاده از اعداد به شرطی‌که اولین حرف متغیر نباشند نیز مجاز است. (یعنی مثلا $s12 و $S_1 به عنوان نام متغیر مجاز است اما $1s2 مجاز نمی باشد.) به شما به عنوان یک برنامه نویس که به تازگی شروع به کار با PHP نموده است پیشنهاد می کنم که از استفاده از Underscore ها نیز در ابتدای نام متغیرهای خود اجتناب ورزید. بعدها خواهیم دید که بسیاری از متغیرهایی که PHP خود از آنها استفاده می کند دارای Underscore در ابتدایشان می باشد.

حالا به مثال زیر توجه کنید:

```
<?php
$a=4;
$b=23;
$c="20hi! ";
$d="world 26";
$D=650.25; //It is different from $d
echo($D);//650.25
echo("<br>");
echo($d);//world 26
echo("<br>");
echo($a+$b); //27
echo("<br>");
echo($b+$c);//43
echo("<br>");
echo($c+$d);//20
echo("<br>");
echo($c.$d);//20hi! world 26
echo("<br>");
echo($a.$c.$d);//420hi! world 26
echo("<br>");
$c.=$d;
```

```
echo($c);//20hi! world 26
echo("<br>");
echo($d);//world 26
?>
```

خروجی برنامه بصورت زیر خواهد بود:



همانطور که از مثال بالا هم متوجه می شوید، در صورتیکه عددی را بصورت عادی و بدون علامت نقل قول (" ") به یک متغیر نسبت می دهیم، PHP می تواند با آن هم بصورت عدد و هم بصورت رشته برخورد نماید. در صورتیکه یک مقدار را در داخل " " قرار دهیم، اگر در ابتدای آن یک عدد وجود داشته باشد، PHP می تواند در مقابل عملگرهای ریاضی با آن عددها همچون اعداد عادی برخورد نماید و در مقابل عملگرهای رشته ای به عنوان رشته.

انواع متغیرها

در این مقاله سه دستهٔ مختلف از انواع متغیرها بررسی می‌شوند: اعداد (numbers)، رشته ها (strings) و آرایه‌ها (Arrays).

دسته اعداد، شامل دو نوع متغیر: اعداد صحیح (integers) و اعداد اعشاری (Floating-point) (همچنین اعداد اعشاری با دقت مضاعف (double precision floating point) و یا (doubles)) می شوند. اما از آنجا که تفاوت چندانی در چگونگی به کار گرفتن این دو نوع متغیر، وجود ندارند، ما این دو را در یک دسته قرار داده‌ایم.

PHP همچنین دارای یک نوع متغیر به نام شیء (object) می باشد. که در فصول آتی به بررسی کامال آن خواهیم پرداخت.

اعداد

همانگونه که ذکر شد، برای آسانتر شدن آموزش، دو نوع متغیر اعداد صحیح و اعداد اعشاری در دستهٔ اعداد قرار داده شده‌اند. در این قسمت به صورت مختصر به ذکر تفاوتهای بین این دو می پردازیم.

اعداد به صورت اعشاری(همراه با ممیز) و یا اعداد کسری ار نوع متغیرهای اعشاری محسوب می‌شوند. برای مثال 1.0 در PHP یک عدد اعشاری به حساب می آید. توجه کنید که در PHP اعداد، به صورت کسری ذخیره نمی-شوند بلکه به معادل اعشاری خود تبدیل شده و سپس ذخیره می‌شوند. برای مثال عبارت 1/4 به صورت 0.25 ذخیره و نوشته می‌شود.

مثالهایی از اعداد صحیح معتبر عبارتند از:

```
1 1972 -1
```

و مثال‌هایی از اعداد اعشاری معتبر عبارتند از:

```
1.0 19.72 -1.0
```

همچنین مثال‌هایی که در دسته اعداد قرار نمی‌گیرند عبارتند از:

```
11/4 1972a 02.23.72
```

در مورد مثال اول باید ذکر کرد که نمایش اعداد به این صورت در PHP اشتباه می‌باشد و باید به جای آن از 1.25 استفاده شود. در مثال دوم، عدد به همراه یک کاراکتر غیر عددی نشان داده شده است. در مثال سوم، دو علامت ممیز در یک عبارت آورده شده است.

رشته ها

یک متغیر از نوع رشته ای (string) از ترکیب هر نوع کاراکتری (حروف، اعداد، علائم و جای خالی) می تواند ساخته شود. اما این کاراکترها باید در داخل یکی از علامت‌های (' ') quotation و یا ("") Double Qutation قرار گیرند.

مثال‌هایی از داده‌های رشته‌ای مجاز عبارتند از:

"Hello, world!"

"Hello, First Name!"

"1 1/4"

"Hello World! How are you today?"

"02.23.72"

"1972"

به مثال آخر توجه کنید. که اگر یک عدد نیز در داخل علامت‌های نقل قول قرار گیرد، به عنوان یک دادهٔ رشته‌ای در نظر گرفته می‌شود. به عبارت دیگر این دادهٔ رشته‌ای از ترکیب کاراکترهای عددی، تشکیل شده است.

مثال‌هایی از داده‌های رشته‌ای غیرمجاز عبارتند از :

Hello ,World !

"I said, "how are you?""

در مثال اول، از آنجایی که عبارت، داخل یکی از علامت‌های مخصوص داده‌های رشته ای قرار نگرفته است. به عنوان یک دادهٔ رشته‌ای در نظر گرفته نمی‌شود. در مثال دوم، PHP بعد از رسیدن به علامت نقل قول دوم، دادهٔ رشته‌ای را تمام شده فرض می‌کند. بنابراین ادامهٔ عبارت باعث ایجاد اشکال می‌شود.

اما برای رفع این مشکل چه باید کرد؟ همانند فصل ۱ که در تابع print() قبل از علامتهای quotation از backslash(\) استفاده کردیم. در این مورد نیز برای اینکه، PHP علامت‌های quotation داخل عبارت را به صورت کاراکتر آن در نظر بگیرد (و نه علامتی برای آغاز یا ختم یک داده رشته‌ای)، مثال دوم را به این صورت اصلاح می کنیم.

"I said,\"How are you?\""

بنابراین هر چند که گفته شد در دادهٔ رشته‌ای هر ترکیبی از کاراکترها به کار می‌رود. باید توجه داشته باشید که در مورد کاراکترهای ویژه، باید دقت خاصی اعمال شود. کاراکترهای ویژهٔ دیگری نیز وجود دارند که در هنگام استفاده از آنها در یک دادهٔ رشته‌ای باید یک علامت (\) Backslash قبل از آنها قرار دهیم. این کاراکترها عبارتند از:

Apostrophe، (') single quotation mark ، (\) Backslash و علامت ($) dollar.

توجه: مزیت استفاده از double quotes به جای Single quotes در این است که در حالت دوم، اگر متغیری داخل دادهٔ رشته‌ای خود به کار ببرید، نام متغیر به عنوان جزئی از داده در نظر گرفته می‌شود، و نه مقدار آن متغیر جایگزین نام آن نمی‌شود. برای مثال نتیجهٔ عبارت زیر در متن ارسالی:

print 'Hello,$FirstName!';

به صورت Hello,$FirstName می باشد. اما اگر فرض کنیم که از قبل به $FirstName مقدار Larry نسبت داده شده باشد). نتیجهٔ عبارت زیر:

Print "Hello,$FirstName!";

به صورت Hello,Larry! می‌باشد.

کوتیشن یا دابل کوتیشن؛ کی و کجا استفاده کنیم: [14][15]

آیا شما هم جز اون دسته از افرادی هستید که هنگام کدنویسی PHP، همواره از دابل کوتیشن برای مشخص کردن یک رشته کاراکتر استفاده می کنید؟!

به نظر می رسه اکثر مثالها و نمونه هایی که حتی در راهنمای PHP وجود داره تمایل زیادی به استفاده از دابل کوت رو نشون میده. اگرچه راه دومی هم وجود داره که به مراتب بهتر از قبلی هم هست و اون استفاده از کوتیشن یا (Single Quote) به جای Double Quoteهاست. برای مثال به جای:

```
<?php
   echo "Visit http://www.phpmystery.com";
?>
```

میتوان از:

```
<?php
   echo 'Visit http://www.phpmystery.com';
?>
```

استفاده کرد. خوب حتما این تو ذهنتون شکل گرفته که «چه فرقی داره؟»

پاسخ اینه که در مثال بالا هیچ تفاوتی وجود نداره اما به مثال زیر توجه کنید:

میخواهیم یه تکه کد HTML رو داخل یک متغیر ذخیره کنیم؛ اگر از دابل کوتیشن استفاده کنیم:

```
<?php
   echo "<table border=\"1\" cellspacing=\"0\" cellpadding=\"0\">";
?>
```

و اگر از کوتیشن استفاده کنیم.

```
<?php
   echo '<table border="1" cellspacing="0" cellpadding="0">';
?>
```

همونطور که می بینید اگر رشته کاراکتر، حاوی کاراکتر (") باشه باید اون رو با کاراکتر بک اسلش (\) اسکیپ کنیم تا Parse Error پیش نیاد ولی در مثال دوم می بیینم که هر چه داخل (') باشه، عینا و حرف به حرف نوشته میشه.

دلیل بعدی اینکه راه دوم پرفورمنس بالاتری داره! استفاده از کوتیشن همواره سریعتر از دابل کوتیشن هست و در برخی موارد این تفاوت چند صد درصد میشه!

با نگاهی به Benchmark زیر فکر کنم همه چیز روشن بشه.

double (") vs. single (') quotes
Is a there a difference in using double (") and single (') quotes for strings. Call 1'000x
+ 101 % 1: single (') quotes. Just an empty string: $tmp[] = ''; Total time: 3[ms]
+ 100 % 2: double (") quotes. Just an empty string: $tmp[] = ""; Total time: 3[ms]
+ 111 % 3: single (') quotes. 20 bytes Text : $tmp[] = 'aaaaaaaaaaaaaaaaaaaa'; Total time: 3[ms]
+ 118 % 4: double (") quotes. 20 bytes Text : $tmp[] = "aaaaaaaaaaaaaaaaaaaa"; Total time: 3[ms]
+ 115 % 5: single (') quotes. 20 bytes Text and 3x a $: $tmp[] = 'aa $ aaaa $ aaaa $ a'; Total time: 3[ms]
+ 461 % 6: double (") quotes. 20 bytes Text and 3x a $: $tmp[] = "aa $ aaaa $ aaaa $ a"; Total time: 13[ms]
+ 113 % 7: double (") quotes. 20 bytes Text and 3x a \$: $tmp[] = "aa \$ aaaa \$ aaaa \$ a"; Total time: 3[ms]

نتیجه: توی رشته کاراکترهایی که در " قرار میدید، از کاراکتر $ به تنهایی استفاده نکنید، مگر اینکه بخواهید مقدار یک متغیر را جانشین کنید و اگر هم خواستید این کار رو انجام بدید بهتره اون رو با یک بک اسلش اسکیپ کنید.

وقتی از کاراکترهای خاصی نظیر \n یا \r و \t که برای فرمت بندی رشته کاراکتر بکار میره، استفاده می کنید، اگر اونها رو داخل ' قرار بدید اثر خودشون رو از دست میدن و باید حتما داخل " قرار بگیرند.

```
<?php
   echo 'check out http://www.phpmystery.com' . "\n\r" . 'to discover unknowns about PHP';
?>
```

با این تفاسیر میشه گفت:

۱» اگر متغیری داخل رشته کاراکتر نیست که بخواهید مقدارش جانشین بشه از کوتیشن استفاده کنید.

۲» اگر متغیری داخل رشته کاراکتر باشه افزودن {} باعث سرعت عملکرد میشه.(*)

```
<?php
$sql = "select * from `mytable` where `status` = '{$status}' order by name"
?>
```

۳» بجای دستور Print از Echo استفاده کنید.

(*) دو جا هست که آکولادها به کار ما میان؛ اولا وقتی که بخواهیم یک سری کاراکتر درست چسبیده به نام متغیر رو نمایش بدیم و مورد بعدی وقتی عبارتی که میخواهیم درون رشته کاراکتر جای بدیم یک متغیر ساده نباشه(مثلا یک آرایه دوبعدی یا خروجی متد یک آبجکت یا Property اون آبجکت باشه) برای مثال:

```
<?php
$sport1 = 'volley';
$plan1 = "I will play $sport1ball in the summertime";
$plan2 = "I will play {$sport1}ball in the summertime";
?>
```

مشکل از اونجا ناشی میشه که مفسر وقتی داخل یه رشته کاراکتر به $ برخورد کنه، تا رسیدن به فاصله خالی شروع به جمع آوری کاراکترها میکنه و نام متغیر رو تشخیص میده و اگر قبلا مقدار گرفته بود، مقدارش رو جایگزین میکنه وگرنه مقدار NULL یا هیچ رو به جای نام متغیر قرار میده. همونطور که فهمیدید خروجی مثال بالا به قرار زیره:

```
<?php
// $plan1 = 'I will play  in the summertime';
// $plan2 = 'I will play volleyball in the summertime'
?>
```

نکته: ترکیب n\، برای مثال در تابع print، باعث ایجاد خط جدید می‌شود. بنابراین مشاهده می‌کنید که در این حالت خاص، علامت backslash، باعث در نظر گرفتن n به صورت یک کاراکتر معمولی نشد. از موارد خاص دیگر می توان به ترکیب r\ بازگشت خطی[1] و t\ (برای قرار دادن یک tab) اشاره کرد.

آرایه‌ها

از آنجایی که آرایه ها کمی پیچیده‌تر از داده‌های عددی و رشته‌ای به حساب می‌آیند، در این قسمت تنها مختصری در مورد آنها توضیح داده می‌شود و سپس در بخش‌های آتی توضیح داده خواهد شد و همچنین استفاده از آرایه، به صورت کاملتری شرح داده می‌شوند. برخلاف داده‌های عددی و رشته‌ای که تنها می‌توانند دارای یک ارزش و یا مقدار باشند، آرایه ها می‌توانند حاوی لیستی از مقادیر باشند. بنابراین شما می‌توانید مقادیر مختلف عددی و یا رشته‌ای را داخل یک آرایه قرار دهید. همچنین آرایه‌ها، خود، می‌توانند شامل لیستی از آرایه‌ها باشند.

نکته: آرایه‌های استاندارد در PHP، از مقادیر داده‌ای و یا عددی تشکیل می‌شوند.(این آرایه‌های شماره گذاری شده (indexed) و یا برداری (vector) نیز معروف هستند)، و این همان نامی است که perl، آرایه‌هایی که خود از آرایه تشکیل شده باشد، به نامهای، آرایه‌های hash[2]، associative[3] و multi_dimensional[4] شناخته می‌شوند. در PHP به هر دو دسته (یک یا چند بعدی) لفظ "آرایه" اطلاق می‌شود.

«««« بحث مربوط به آرایه ها در فصول آتی به طور کامل بررسی شده است»»»»

نسبت دادن مقادیر به متغیرها:

1 Carriage return: یک کاراکتر کنترلی که به کامپیوتر یا چاپگر می گوید به ابتدای خط فعلی برگردد. این که مانند کلید بازگشت (return) در ماشینهای تحریر بوده، اما به طور اتوماتیک باعث رفتن به ابتدای خط بعدی نمی‌شود. (برگرفته از کتاب " فرهنگ تشریعی اصطلاحات کامپیوتری میکروسافت: مترجم:مجید سماوی، انتشارات ناقوس، ص ۷۰)

2 به معنای خرد کردن، ریز کردن و یا مخلوط کردن

3 به معنای مشارکتی یا انجمنی

4 به معنای چند بعدی

در PHP شما، به اعلان (declare) متغیرها احتیاج ندارید. همچنین نوع یک متغیر در هنگام عمل انتساب مشخص می‌شود. در PHP برای نسبت دادن یک مقدار به یک متغیر و ذخیرهٔ آن از علامت مساوی (=) استفاده می-کنید. در این هنگام این علامت با نام عملگر انتساب (assignment operator) خوانده می‌شود.

برای مثال به نمونه‌های زیر توجه کنید:

```
$number=1;

$floating-number=1.2;

$string="Hello,World!";
```

در PHP نیز همانند Java Script، نوع متغیر در طول برنامه، می‌تواند تغییر کند. برای مثال بعد از اجراء دستورات زیر، اگر بخواهیم مقدار $variable را چاپ کنیم، نتیجه به صورت Greeting خواهد بود.

```
$Variable=1;

$Variable="Greeting";
```

متغیرهای از پیش تعریف شده:

متغیرهای از پیش تعریف شده (Predefined Variables) انواع خاصی از متغیرها هستند که در یکی از این برنامه‌های کاربردی سرویس دهندهٔ وب (Web server Application) مانند آپاچی، سیستم عاملهای سرویس دهندهٔ وب (Web Server Operating System) (مانند Windows NT و یا Solaris) و یا خود مدل PHP . در دو دستهٔ اول، این متغیرها به متغیرهای محیطی (Variables environment) معروفند. متغیرهای از پیش تعریف شده در سرویس دهنده‌های مختلف، ممکن است دارای تفاوتهایی باشند.

دو دلیل برای آشنایی شما با مفهوم متغیرهای از پیش تعریف شده، وجود دارد: دلیل اول اینست که این متغیرها در برنامه‌نویسی شما کاربرد خواهند داشت و دلیل دیگر آنکه با شناخت این متغیرها، شما دیگر، به صورت تصادفی نام یک متغیر را هم نام با این متغیرها، انتخاب نمی‌کنید.

نمونه هایی از متغیرهای محیطی سرویس دهنده عبارتند از: HOSTNAME (نامی که سرویس‌دهنده به خود نسبت داده است). و OSTYPE (سیستم عاملی که بر روی سرویس‌دهندهٔ در حال اجرا می باشد).

نمونه هایی از متغیرهای محیطی Apache عبارتند از: DOCUMENT_ROOT (مکان ذخیرهٔ فایلها بر روی سرویس دهنده) و HTTP_USER_AGENT (جزئیاتی در مورد مرورگر و platform کاربر ارائه می‌دهد).

متغیر PHP-SELF پرکاربردترین متغیر PHP می باشد که نام صفحهٔ جاری را در خود ذخیره کرده است.

در این بخش خواهیم خواند : [8]

- نحوه ساخت ، بارگذاری و اجرای یک فایل PHP
- چگونه از PHP و HTML در یک صفحه استفاده کنیم
- چگونه کدها را با استفاده از کامنتهای برنامه نویسی قابل فهم تر کنیم

خوب برای شروع یک TextEditor رو باز کنید. php هم مانند html مبتنی بر متن ساده است بنابراین از هر ویرایشگری می‌توان برای ساخت فایل php استفاده کرد.(قبلا اشاره شد که از ورد استفاده نکنید) مثلا Notepad خود را باز کنید. خطوط زیر را وارد کنید و فایل را به هر اسمی با پسوندphp. ذخیره کنید. مثلا first**.php**

Code:

```
1: <?
2: print "Hello Web!";
3: ?>
```

در صورتیکه php را روی کامپیوتر خود نصب کرده اید ، فایل رو در وب سرور خود کپی کنید و آن را از طریق browser صدا کنید. و یا اینکه فایل رو در یک سایت با دسترسی PHP بارگذاری نمایید و آدرس آن را در browser تایپ کنید.

پس از اجرای فایل جمله Hello Web! رو بر روی صفحه خواهید دید.

در صورتیکه کدهای خود، یعنی هما چیزی که در ادیتور تایپ کردید رو روی صفحه دیدید. این بدان معنی است که وب سرور شما فایل php رو اجرا نکرده (یا php بر روی آن نصب نیست و یا پسوند فایل رو به درستی انتخاب نکرده اید.)

خوب حالا که صفحه اول php خود را upload کرده اید کمی در آن دقت کنید.

در شروع و در اولین خط کد

Code:

```
<?
```

رو داریم. همیشه شروع یک کد php باید با همین تگ ها باشد. در غیر اینصورت سیستم با کدها مثل html رفتار می‌کند و بدون هیچگونه عملیاتی همان متن کد را نمایش خواهد داد.

و در انتهای کد PHP نیز حتما باید عبارت یا علامت

Code:

```
?>
```

را وارد کنید که به server می فهماند که کد php اینجا تمام شده است و از این به بعد با کدهای html روبرو است.

در انتهای هر خط از کدها باید از

Code:

```
;
```

استفاده شود. در غیر اینصورت سرور کدها را در یک خط و پشت سر هم تشخیص می دهد. پس شما می توانستید کد بالا را در یک خط و به صورت

Code:

```
<? Print "Hello Word!" ; ?>
```

نیز بنویسید.

دستور Print :

این دستور در واقع نمایش دهنده است و هر چیزی که به آن بدهید را بر روی صفحه نمایش می دهد. شما می توانید مقدار یک متغیر را نیز با استفاده از Print نمایش دهید.

ترکیب PHP و HTML در یک صفحه :

کدهای زیر را در notepad وارد نمایید و ذخیره و upload کنید.

Code:

```
1: <html>
2: <head>
```

```
3: <title>Listing 3.2 A PHP script including HTML</title>
4: </head>
5: <body>
6: <b>
7: <?
8: print "hello world";
9: ?>
10: </b>
11: </body>
12: </html>
```

خوب همانطور که می بینید کدهای HTML به راحتی می توانند در کنار کدهای PHP کار کنند. در واقع سرور قبل از رسیدن به کد

Code:

```
<?
```

همه کدها رو html فرض کرده و کدهای در داخل

Code:

```
<?
```

و

Code:

```
?>
```

را به عنوان کد php می شناسد و بر روی آنها عملیات انجام می دهد.

گذاشتن Comment در میان کدها

comment در واقع جملاتی است که نویسنده برنامه در میان کدها می نویسد تا توضیحی باشد بر کد نوشته شده. به این صورت که اگه دفعه بعد خود نویسنده و یا کس دیگری کدها رو دید ، متوجه بشود که هر دستور برای چه چیزی نوشته شده است. (توصیه می کنم همیشه از comment استفاده کنید)

نحوه قراردادن commentهم به این صورت است که در اول خط از تگ

Code:

```
//
```

و یا

Code:

```
#
```

استفاده کنید.

مثلا

Code:

```
// This is a comment
```

یا

Code:

```
# in yek comment ast
```

در این بخش می خوانیم :

- درباره متغیرها (متغیر چیست و چگونه از آن استفاده کنیم)
- چگونه یک متغیر را تعریف کنیم و به مقدار آن دسترسی پیدا کنیم
- برخی از علمگرهای متداول
- چگونگی تعریف و استفاده از مقادیر ثابت

متغیرها:

متغیر نگهدارنده ویژه ایست برای مقادیر. هر متغیر دارای یک نام است که با علامت $ در اول آن مشخص می شود. نام یک متغیر می تواند شامل حروف ، اعداد و _ باشد. نام یک متغیر نمی تواند شامل space و یا کارکترهای غیر حرفی باشد.

کدهای زیر چند متغیر را تعریف می کنید :

Code:

```
$a;
$a_longish_variable_name;
$2453;
$sleepyZZZZ
```

توجه داشته باشید که **;** در انتهای هر خط جزونام متغیر می باشد و در واقع نشان دهنده پایان جمله کد PHP است.

برای مقدار دادن به متغیر کافیست که آن را مساوی با مقدارش قرار دهید. به طور معمول شما در یک دستور php متغیر را تعریف می کنید و به آن مقدار می دهید. مانند کدهای زیر :

Code:

```
$num1 = 8;
$num2 = 23;
```

وقتی که شما به یک متغیر مقدار دادید می توانید دقیقا مانند یک کاراکتر با آن رفتار کنید. به طور مثال :

Code:

```
print $num1;
```

دقیقا برابر با دستور

Code:

```
print 8;
```

می باشد.

نوع داده داخل متغیر DATA TYPE :

انواع مختلف اطلاعات در یک متغیر می تواند ذخیره شود که در طول برنامه می توانید رفتارهای متفاوتی با آن نمابید.

برخی زبانهای برنامه نویسی شما را وادار می کنند که در ابتدا و در موقع تعریف متغیر نوع آن را نیز مشخص نمایید. ولی در PHP لزومی به این کار نیست و نوع اولین مقداری که وارد متغیر شود ، به عنوان نوع متغیر شناخته می شود.

Type - Example - Description
number Integer - 5 - A whole
Double - 3.234 - A floating-point number
collection of characters String - "hello" - A
false Boolean - true - One of the special values true or

آرایه و OBJECT:

شما می توانید از دستور gettype() برای مشاهده نوع یک متغیر استفاده کنید. به عنوان مثال :

Code:

```
1: <html>
2: <head>
3: <title>Listing 4.3 Testing the type of a variable</title>
4: </head>
5: <body>
6: <?php
7: $testing = 5;
8: print gettype( $testing ); // integer
9: print "<br>";
```

```
10: $testing = "five";
11: print gettype( $testing ); // string
12: print("<br>");
13: $testing = 5.0;
14: print gettype( $testing ); // double
15: print("<br>");
16: $testing = true;
17: print gettype( $testing ); // boolean
18: print "<br>";
19: ?>
20: </body>
21: </html>
```

کدهای بالا در خروجی جملات زیر را نشان خواهد داد :

Code:

```
integer
string
double
boolean
```

INTEGER یک عدد صحیح می باشد. به کلام ساده یک عدد بدون ممیز می باشد. STRING یک سری کاراکتر می باشد. وقتی در PHP با STRING کار می کنید باید حتما اطراف آن از " و یا ' استفاده شود. DOUBLE یک عددی است که ممیز نیز دارد. BOOLEAN یا TRUE است و یا FALSE.

تغییر با استفاده از دستور settype() :

در PHP با استفاده از دستور settype() شما می توانید نوع یک متغیر را تغییر دهید. برای این کار باید نام متغیر و نوع متغیر که می خواهید به آن تغییر یابد را در بین پرانتز و با فاصله یک کاما در بینشان مشخص نمایید.

به عنوان مثال :

Code:

```
1: <html>
```

```
2: <head>
3: <title>Listing 4.5 Changing the type of a variable with settype()</title>
4: </head>
5: <body>
6: <?php
7: $undecided = 3.14;
8: print gettype( $undecided ); // double
9: print " -- $undecided<br>"; // 3.14
10: settype( $undecided, string );
11: print gettype( $undecided ); // string
12: print " -- $undecided<br>"; // 3.14
13: settype( $undecided, integer );
14: print gettype( $undecided ); // integer
15: print " -- $undecided<br>"; // 3
16: settype( $undecided, double );
17: print gettype( $undecided ); // double
18: print " -- $undecided<br>"; // 3.0
19: settype( $undecided, boolean );
20: print gettype( $undecided ); // boolean
21: print " -- $undecided<br>"; // 1
22: ?>
23: </body>
24: </html>
```

در هر دفعه تغییر متغیر ما با استفاده از دستور() GETTYPE نوع متغیر را چاپ می کنیم که از تغییر آن مطمئن شویم.

همانطور که می بینید در خط ۷ مقدار متغیر ۳,۱۴ است و به صورت DOUBLE و در خط ۱۰ به STRING تبدیل می شود و در خط ۱۳ به INTEGER تغییر می کند و به ۳ تبدیل می شود. (یعنی رند می شود) و به همین صورت.....

تغییر نوع داده بدون اینکه اصل متغیر تغییر کند :

با قرار دادن نام نوع داده Data Type در داخل پرانتز و قبل از نام متغیر یک کپی از متغیر با نوع داده جدید بدون تغییر دادن متغیر اصلی ایجاد می کند.

به عنوان مثال :

Code:

```
1: <html>
2: <head>
3: <title>Listing 4.6 Casting a variable</title>
4: </head>
5: <body>
6: <?php
7: $undecided = 3.14;
8: $holder = ( double ) $undecided;
9: print gettype( $holder ) ; // double
10: print " -- $holder<br>"; // 3.14
11: $holder = ( string ) $undecided;
12: print gettype( $holder ); // string
13: print " -- $holder<br>"; // 3.14
14: $holder = ( integer ) $undecided;
15: print gettype( $holder ); // integer
16: print " -- $holder<br>"; // 3
17: $holder = ( double ) $undecided;
18: print gettype( $holder ); // double
19: print " -- $holder<br>"; // 3.14
20: $holder = ( boolean ) $undecided;
21: print gettype( $holder ); // boolean
22: print " -- $holder<br>"; // 1
23: ?>
18: print gettype( $holder ); // double
19: print " -- $holder<br>"; // 3.14
20: $holder = ( boolean ) $undecided;
21: print gettype( $holder ); // boolean
22: print " -- $holder<br>"; // 1
23: ?>
```

در کد بالا هیچ وقت نوع متغیر اصلی را تغییر ندادیم بلکه مثلا در خط ۱۴ در متغیر $holder مقدار Integer شده متغیر اصلی یعنی مقدار ۳ را قرار دادیم در خط ۱۶ آن را چاپ کردیم.

توجه: همانند زبانهایی مانند C در اینجا هم می توانیم از عملگرهای پیشوندی مانند $a+=$b یا $a++ استفاده نماییم.

عملگرها :

در درس های قبلی یاد گرفتیم که مقدار به متغیر بدهیم و data type متغیر ها رو تغییر بدهیم. یک زبان برنامه نویسی تا وقتی که نتونیم به وسیله اون بر روی متغیرها عملیات ریاضی انجام بدیم در واقع به درد نمی خوره. عملگرها سمبول هایی هستند که به وسیله اون می تونیم با استفاده از چند مقدار ، مقدار جدیدی رو تولید کنیم.

یک عملگر به عنوان مثال همون + است.

۴ + ۵ = ۹

در اینجا ما از عملگر + استفاده کردیم تا با استفاده از دو مقدار ۴ و ۵ مقدار جدید ۹ را تولید کنیم.

عملگر **مقدار ده** یا همون (=) کارش اینه که مقدار سمت راست خودش رو توی متغیر سمت چپ می ریزه.

PHP Code:

```
print ( $name = "matt" );
```

دستور بالا کلمه matt رو چاپ می کند و همچنین متغیر name رو مساوی matt قرار می دهد.

عملگرهای ریاضی :

عملگرهای ریاضی در PHP طبق جدول زیر می باشند.

مثال	نام	عملگر
$a + $b	جمع	+
$a - $b	تفریق	-
$a * $b	ضرب	*
$a / $b	تقسیم	/
$a % $b	باقیمانده	٪

عملگر	نام	مثال	جواب
+	جمع	5+3	8
-	تفریق	10-3	7
/	تقسیم	10/2	5
*	ضرب	2*10	20
%	باقیمانده	10%3	1

عملگر پیوند دهنده یا همان (.) :

این عملگر وظیفه پیوند دادن متغیر های متنی رو دارد.

بعنوان مثال :

PHP Code:

```
"hello"." world"
returns
"hello world"
```

در php یک سری عملگرهای دیگه ای برای مقدار دهی دارد.

=+ عملگری است که با استفاده از اون متغیر با خودش جمع می شود.

مثلا

PHP Code:

```
$x = 4;
$x += 4; // $x now equals 8
```

همچنین =- و =/ نیز می توان استفاده نمود.

PHP Code:

```
$x = 4;
$x -= 4; // $x now equals 1
```

عملگرهای مقایسه ای:

عملگرهای مقایسه ای بر روی متغیرها اعمال می شود و مقایسه می کند و در صورت درست بودن True و در صورت غلط بودن False بر می گرداند.

مثلا

PHP Code:

```
$x < 5
```

اگر مقدار x مثلا ۳ باشد این عملگر True رو برمی گرداند.

مثلا == مقدار سمت راست و سمت چپ رو چک می کند. اگه مثلا ما x رو ۴ قرار داده باشیم.

PHP Code:

```
$x == 5
```

مقدار False بر می گرداند.

=! :چک می کند که مقدار سمت راست و چپ برابر نباشند و اگر x همان ۴ باشد :

PHP Code:

```
$x =! 5
```

True را بر می گرداند.

=== چک می کند که مقدار چپ و راست برابر باشند و همچنین نوع دیتا آنها Data type آنها یکی باشد.

علامت بزرگتر و کوچکتر و بزرگتر مساوی و کوچکتر مساوی نیز به همین صورت.

عملگرهای منطقی :

اولین عملگر منطقی همان عملگر or (یا) می باشد. دیگر نشانه این عملگر || می باشد.

مثلا

PHP Code:

```
true || false
```

مقدار True را بر می گرداند.

&& فقط وقتی True برمی گرداند که هر دو طرف True باشند.

مثلا

PHP Code:

```
( $x > 2 ) && ( $x < 15 )
```

هنگامی مقدار True برمی گرداند که x بزرگتر از ۲ و کوچکتر از ۱۵ باشد.

عملگر	نام	وقتی True می شود که.....	مثال	جواب
||	یا	چپ یا راست True باشد	true || false	true
or	یا	چپ یا راست True باشد	true || false	true
xor	**Xor**	**چپ یا راست True باشد ولی هر دو تا True نباشد**	**true || false**	**false**
&&	و	چپ و راست هر دو True باشد	true && false	false
and	و	چپ و راست هر دو True باشد	true && false	false
!	نه	درست نباشد	! true	false

در جدول بالا قسمت Bold شده تنها عملگری است که کمی جالب است.

عملگر ++ و --

عملگر ++ یک عدد به متغیر اضافه می کند و -- یک متغیر از آن کم می کند.

مثلا

PHP Code:

```
$x++;
```

یکی به $x اضافه می کند.

به عنوان مثال

PHP Code:

```
$x = 3;
$x++ < 4; // true
```

مقدار فوق True است.

در مثال بالا همونطور که می بینید ابتدا عمل مقایسه انجام شده و بعد جمع صورت گرفته.

حال :

PHP Code:

```
$x = 3;
++$x < 4; // false
```

مقدار فوق False می باشد یعنی ابتدا جمع صورت گرفته و بعد مقایسه شده است.

ترتیب اجرای عملگرها :

شاید همه شما این را بدانید ولی در php ترتیب اجرای عملگرها به صورت زیر می باشند :

PHP Code:

```
++ – – (cast)
/ *%
+ –
< <= => >
== === !=
&&
||
= += – = /= *=%= .=
and
xor
or
```

یعنی مثلا در

PHP Code:

```
4 + 5 * 2
```

ابتدا ۲ در ۵ ضرب می شود و بعد با ۴ جمع می شود.

یعنی جواب مقدار فوق ۱۴ می باشد.

البته شما با گذاشتن پرانتز می توانید php را مجبور کنید که به صورت دلخواه شما عمل کند. مثلا

PHP Code:

```
(4+5) * 2
```

مقدار ۱۸ را می دهد.

ساختارهای کنترلی در PHP

در درسهای قبلی همواره کدها در یک جهت حرکت می کردند. در واقع خط به خط کدهای ما اجرا می شد و جلو می رفت. این روش جایی برای کدنویسی منعطف نمی گذارد.

در این فصل می خوانیم :

- ***چگونه یک کد را وقتی اجرا کنیم که یک عبارت True باشد. CLAUSE IF***
- ***چگونه قسمت دیگری از کد اجرا شود وقتی که همان عبارت False شود. ELSE***
- ***چگونه از دستور switch استفاده کنیم.***
- ***چگونه یک قسمت از کد را چند مربطه اجرا کنیم . while***
- ***چگونه از FOR برای اجرای حلقه استفاده کنیم.***
- ***چگونه یک حلقه FOR را قطع کنیم.***

اغلب script ها نیاز به تغییر خروجی در شرایط مختلف دارند. با استفاده از IF شما می توانید خروجی کدهای خودتون را در php با توجه به یک سری شرایط تغییر دهید.

IF :

IF عبارت داخل پرانتز جلوی خود را کنترل می کند و درصورتی که Trueباشد آن قسمت از کد را اجر می نماید.

PHP Code:

```
if ( expression )
{
// code to execute if the expression evaluates to true
}
```

فرمت کلی IF به صورت بالا است. expression عبارتی است که باید کنترل شود. و عبارات بین { و } کدی است که در صورت true بودن عبارت If باید اجرا شود.

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 5.1</title>
4: </head>
5: <body>
6: <?php
7: $mood = "happy";
8: if ( $mood == "happy" )
9: {
10: print "Hooray, I'm in a good mood";
11: }
12: ?>
13: </body>
14: </html>
```

در خط ۸ کد بالا ما از == استفاده کردیم تا کنترل کنیم که mood مقدار happy را در خود دارد یا خیر.

از { و } فقط وقتی استفاده می کنیم که کدهای ما بیشتر از یک خط باشند.

کد بالا رو به صورت زیر نیز می توان نوشت :

PHP Code:

```
if ( $mood == "happy" )
print "Hooray, I'm in a good mood";
```

در این حالت Hooray, I'm in a good mood چاپ می شود. اگر ما متغیر mood رو "sad" قرار دهیم دیگر چیزی چاپ نخواهد شد.

استفاده از ELSE در IF:

فرمت کلی آن به صورت زیر است :

PHP Code:

```
if ( expression )
{
// code to execute if the expression evaluates to true
}
```

```
else
{
// code to execute in all other cases
}
```

قسمت بعد از else فقط وقتی اجرا می شود که عبارت داخل if ، برابر False باشد.

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 5.2</title>
4: </head>
5: <body>
6: <?php
7: $mood = "sad";
8: if ( $mood == "happy" )
9: {
10: print "Hooray, I'm in a good mood";
11: }
12: else
13: {
14: print "Not happy but $mood";
15: }
16: ?>
17: </body>
18: </html>
```

در مثال بالا متغیر mood مقدار "sad" دارد و در واقع با "happy" برابر نیست پس قسمت داخل IF اجرا نمی شود و فقط قسمتی که داخل else می باشد اجرا خواهد شد. خروجی دستور بالا به صورت

Code:

```
Not happy but sad
```

خواهد بود.

استفاده از ElseIf :

ElseIf مجددا یک عبارت دیگر را اجرا می نماید و در صورت True بودن دستورات داخلش اجرا خواهد شد.

فرمت کلی به صورت زیر است :

PHP Code:

```
if ( expression )
{
// code to execute if the expression evaluates to true
}
elseif ( another expression )
{
// code to execute if the previous expression failed
// and this one evaluates to true
else
{
// code to execute in all other cases
}
```

اگر عبارت IF درست نباشد تکه اول کد نادیده گرفته می شود و نوبت عبارت Esleif می رسد اگر این عبارت درست باشد قسمت دوم کدها اجرا می شود. دستور else در نهایت وقتی اجرا می شود که هیچ کدام از عبارات IF و Elseif درست نباشند.

شما به هر تعداد که می خواهید می توانید esleif در کدتان بگذارید. و در نهایت Else دلخواه است و می تواند در کد وجود نداشته باشد.

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 5.3</title>
4: </head>
5: <body>
6: <?php
7: $mood = "sad";
8: if ( $mood == "happy" )
9: {
10: print "Hooray, I'm in a good mood";
11: }
```

```
12: elseif ( $mood == "sad" )
13: {
14: print "Awww. Don't be down!";
15: }
16: else
17: {
18: print "Neither happy nor sad but $mood";
19: }
20: ?>
21: </body>
22: </html>
```

در مثال بالا mood مقدار sad دارد. این مقدار با Happy برابر نیست پس قسمت اول کدها نادیده گرفته می شود. Elseif متغیر mood را با مقدار sad مقایسه می کند که True است و کدهای قسمت دوم اجرا می شوند.

دستور Switch :

این دستور روش دیگری برای تغییر مسیر حرکت اجرا شدن کدهاست. دستور switch فقط یک عبارت رو چک می کند و می تواند این عبارت را با مقادیر متفاوتی مقایسه کند و فقط کدی را اجرا کند که مقدار مورد نظر در آن True شود.

فرمت کلی به صورت زیر است :

PHP Code:

```
switch ( expression )
{
case result1:
// execute this if expression results in result1
break;
case result2:
// execute this if expression results in result2
break;
default:
// execute this if no break statement
// has been encountered hitherto
```

```
}
```

عبارت داخل دستور switch (منظور همان expression داخل پرانتز جلوی آن است) معمولا یک متغیر است. در کد های داخل switch (منظور قسمت بین { و } می باشد) شما case های مختلفی را می بینید که مقدار متغیر switch با همه این case ها مقایسه می شود و وقتی که مقدار ها با هم برابر بود کد مربوطه اجرا می شود. گذاشتن قسمت default اختیاری است. در صورتی که متغیر با هیچ کدام از مقادیر caseها برابر نباشد ، دستورات داخل default اجرا خواهد شد.

احتیاط : دقت کنید که کلمه break در انتهای هر case را حتما بگذارید. در غیر اینصورت کد بعد از اجرا کردن case مورد نظر به راه خود ادامه می دهد و به مقدار default می رسد و آن را نیز اجرا می کند و این در اکثر مواقع چیزی نیست که ما دنبالش باشیم. دستور break در واقع از کل کدهای case خارج شده و به انتهای دستور switch می رود.

مثال :

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 5.4</title>
4: </head>
5: <body>
6: <?php
7: $mood = "sad";
8: switch ( $mood )
9: {
10: case "happy":
11: print "Hooray, I'm in a good mood";
12: break;
13: case "sad":
14: print "Awww. Don't be down!";
15: break;
16: default:
```

```
17: print "Neither happy nor sad but $mood";
18: }
19: ?>
20: </body>
21: </html>
```

در مثال بالا ، در ابتدا mood مقدار sad را دارد و وقتی که در خط ۸ Php وارد switch می شود ابتدا mood را با happy مقایسه می کند (خط ۱۰) و به دلیل اینکه True نمی شود به خط ۱۳ می رود در آنجا چون mood=sad است کد خط ۱۱ و سپس خط ۱۲ را اجرا می کند. در خط ۱۲ با دیدن دستور break به خط ۱۸ می رود.

روش جالب عملگر شرطی علامت سوال (؟) :

من اسم این روش رو IF یک خطی گذاشتم ! در این روش شما بدون نیاز به نوشتن دستورات IF می توانید تابع شرطی ایجاد کنید که در آن عبارتی چک شود و در صورت درست بودن یک سری کد اجرا شود و در صورت اشتباه بودن کد دیگر.

PHP Code:

```
( expression )?returned_if_expression_is_true:returned_if_expression_is
_false;
```

اگر عبارت داخل پرانتز ما True بود کد بعد از علامت سوال اجرا می شود و در غیر اینصورت کد بعد از : اجرا خواهد شد.

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 5.5</title>
4: </head>
5: <body>
6: <?php
7: $mood = "sad";
8: $text = ( $mood=="happy" )?"Hooray, I'm in a good mood":"Not happy but
```

```
$mood";
9: print "$text";
10: ?>
11: </body>
12: </html>
```

در مثال بالا در خط ۸ از این روش استفاده شده است. در اینجا متغیر moodبا مقدار happy مقایسه می شود اگر درست بود خواهد نوشتHooray, Im in good mood و در غیر اینصورت (که اینجا اینطور نیست) می نویسد Not Happy But $mood که چون اینجا $mood مقدار Sad دارد خروجی می شود Not Happy But Sad. نوشتن و خواندن کد اینطوری کمی سخت است ولی اگر شما فقط یک شرایط رو بخواهید چک کنید و علاقه به نوشتن کدهای فشرده دارید این روش بسیار خوبی است.

حلقه ها :

تا به حال روشهایی را دیدید که به وسیله آن کد می تواند بین اینکه کدام کد را اجرا کند انتخاب داشته باشد. همچنین کد می تواند تصمیم بگیرد که چند دفعه یک قسمت را اجرا کند.

حلقه های برای این ایجاد شده اند که به شما اجازه دهند یک عملیات را چند مرتبه اجرا نمایید. تقریبا بدون استثناء ، همه حلقه ها آنقدر اجرا می شوند تا اینکه یک شرایطی (که از توسط شما مشخص می شود) اتفاق بیافتد و یا اینکه شما شخصا دستور قطع و خروج از loop را بدهید.

while :

PHP Code:

```
while ( expression )
{
// do something
```

```
}
```

تا وقتی که عبارت while (منظور همان expression داخل پرانتز جلوی آن است) True باشد کد داخل while پشت سر هم اجرا می شود. معمولا شما در داخل حلقه کاری می کنید که عبارت مرتباً تغییر نماید و یک جا False شود در غیر اینصورت حلقه شما تا بینهایت اجرا می شود.

مثال زیر حلقه ای ایجاد می کند و مضارب ۲ را نمایش می دهد.

PHP Code:

```
Listing 5.6: A while Statement
1: <html>
2: <head>
3: <title>Listing 5.6</title>
4: </head>
5: <body>
6: <?php
7: $counter = 1;
8: while ( $counter <= 12 )
9: {
10: print "$counter times 2 is ".($counter*2)."<br>";
11: $counter++;
12: }
13: ?>
14: </body>
15: </html>
```

در خط ۷ ما مقدار counter را ۱ گذاشتیم. در خط ۸ حقله ای تشکیل دادم که تا وقتی اجرا می شود که counter از ۱۲ کوچکتر و یا مساوی آن باشد. در خط ۱۰ خروجی برنامه را می نویسیم و در خط ۱۱ به counter دو عدد اضافه می کنیم. روش ++ را در درس قبلی خواندیم.

پس خروجی برنامه می شود. ۲ – ۴ – ۸ – ۱۰ – ۱۲

اگر شما در خط ۱۱ فراموش می کردید که counter را زیاد کنید این حلقه تا بینهایت اجرا می شد چون هیچگاه counter زیاد نمی شد و هیچوقت از ۱۲ بیشتر نمی شد.

حلقه do و while :

این حلقه شبیه حلقه while است فقط سر و ته شده است ! بزرگترین فرق آن این است که ابتدا کدها اجرا می شوند و بعد درستی یا نادرستی عبارت چک می شود.

PHP Code:

```
do {
// code to be executed
}
while ( expression );
```

توجه : آخر عبارت while در خط آخر حتما باید ; گذاشته شود.

این متد وقتی خیلی به درد می خورد که شما بخواهید کد شما حداقل یکبار اجرا شود حتی اگر شرایط عبارت while اتفاق نیافتاده باشد.

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 5.7</title>
4: </head>
5: <body>
6: <?php
7: $num = 1;
8: do
9: {
10: print "Execution number: $num<br>\n";
11: $num++;
12: }
13: while ( $num > 200 && $num < 400 );
14: ?>
```

```
15: </body>
16: </html>
```

در مثال بالا حقله do....while کنترل می کند که num بزرگتر از ۲۰۰ و کوچتر از ۴۰۰ باشد. چون ما num رو در خط ۷ مساوی یک قراردادیم پس عبارت while درست نیست و False می باشد

ولی در هر حال خروجی کد حداقل یک خط است که نوشته می شود: Execution numnber۱.

حقله For :

شما هیچ چیزی از استفاده از For به دست نمی آورید که نتوانید با While آن را ایجاد کنید. ولی در هر حال در اغلب مواقع استفاده از For کدهای زیباتر و بهتری نسبت به while ایجاد می‌کند.

فرمت کلی به صورت زیر است :

PHP Code:

```
for ( variable assignment; test expression; variable increment )
{
// code to be executed
}
```

هر عبارت داخل for باید حتما با ; از هم جدا شوند. معمولا ، عبارت اول یک متغیر شمارنده ایجاد می کند ، و در عبارت دوم عبارت کنترلی برای loop است ، و قسمت سوم اضافه کردن و کم نمودن متغیر را ایجاد می کند.

مثال :

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 5.8</title>
4: </head>
5: <body>
6: <?php
7: for ( $counter=1; $counter<=12; $counter++ )
```

```
8: {
9: print "$counter times 2 is ".($counter*2)."<br>";
10: }
11: ?>
12: </body>
13: </html>
```

در خط ۷ بالا می توانید ببینید ، در قسمت اول counter را مساوی ۱ نمودیم و در عبارت وسط محدود کردیم که حلقه تا وقتی کار کند که counter از ۱۲ کوچکتر و یا مساوی آن باشد. در قسمت سوم هم عبارتی است که هر بار که کد اجرا شود یک عدد به counter اضافه نماید.

مثال بالا و مثال قبلی هر دو یک خروجی را می دهند فقط کد for کمی جمع و جور تر از while است. به علت اینکه شروع ، پایان و شرایط حلقه در همان خط اول در For مشخص است ، با یک نگاه به این نوع حلقه کل حقله دستمان می آید.

خروج از حلقه با استفاده از دستور **break** :

در هر دو روش ایجاد حلقه دستوری برای پایان دادن حلقه وجود دارد. گاهی اوقات نیاز است که وقتی کد به شرایط خاصی رسید حلقه پایان یابد.

مثلا :

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 5.9</title>
4: </head>
5: <body>
6: <?php
7: for ( $counter=1; $counter <= 10, $counter++ )
8: {
9: $temp = 4000/$counter;
10: print "4000 divided by $counter is... $temp<br>";
11: }
```

```
12: ?>
13: </body>
14: </html>
```

در کد بالا ما عدد ۴۰۰۰ رو بر counter که از ۱ تا ۱۰ متغیر است تقسیم می کنیم و خروجی را چاپ می کنیم. تا ایجا کد بدون نقص به نظر می رسد. ولی اگه مثلا counter از ورودی کاربر گرفته شود و کاربر مثلا عددی منفی بزند یا صفر وارد نماید و یا اینکه یک کلمه به جای عدد وارد کند. در این صورت ما باید حلقه رو قطع کنیم چون می دونیم که تقسیم کردن یک عدد بر صفر ایجاد خطا در php می کند.

مثلا فرض کنید از ۴- شروع کنیم تا ۱۰ این وسط صفر هم جزو مقادیر counter خواهد شد.

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 5.10</title>
4: </head>
5: <body>
6: <?php
7: $counter = – 4;
8: for ( ; $counter <= 10; $counter++ )
9: {
10: if ( $counter == 0 )
11: break;
12: $temp = 4000/$counter;
13: print "4000 divided by $counter is... $temp<br>";
14: }
15: ?>
16: </body>
17: </html>
```

در کد بالا همونطور که می بینید در خط ۱۰ و ۱۱ تعریف کردیم که اگه counter صفر شد حلقه قطع شود. بنابراین کد بعد از رسیدن به صفر و به break می رسد حلقه را قطع می نماید و به خط ۱۴ می رود.

نکته جالب در کد بالا در خط ۷ است که ما counter را خارج از حلقه تعریف نمودیم. بنابراین در خط ۸ دیگه counter رو تعریف نکردیم و جاش رو خالی گذاشتیم.

شما هر کدام از قسمتهای for رو می تونید خالی بگذارید ولی ; ها رو حتما باید بگذارید.

استفاده از دستور continue :

خوب ، حالا فرض کنید که ما در کد بالا نمی خواهیم که وقتی به صفر رسید کد قطع شود و فقط می خواهیم که حلقه برای مقدار صفر اجرا نشود ولی باقی مقادیر اجرا شود. در این حالت از دستور continue استفاده می کنیم.

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 5.11</title>
4: </head>
5: <body>
6: <?php
7: $counter = – 4;
8: for ( ; $counter <= 10; $counter++ )
9: {
10: if ( $counter == 0 )
11: continue;
12: $temp = 4000/$counter;
13: print "4000 divided by $counter is... $temp<br>";
14: }
15: ?>
16: </body>
17: </html>
```

در کد بالا در خط ۱۰ و ۱۱ وقتی مقدار counter صفر شود ، حلقه قطع می شود و دوباره کد بر می گردد به خط ۸ ولی فقط مقدار صفر رو اجرا نخواهد کرد.

حلقه های تو در تو :

در php این قابلیت رو دارید که در داخل یک حقله یک یا چند حلقه دیگه رو نیز بیارید. ولی باید توجه داشته باشید که هر حلقه ای که در یک حلقه دیگر استفاده می کنید باید در همان حلقه تمام شود.

مثلا :

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 5.12</title>
4: </head>
5: <body>
6: <?php
7: print "<table border="1">\n";
8: for ( $y=1; $y<=12; $y++ )
9: {
10: print "<tr>\n";
11: for ( $x=1; $x<=12; $x++ )
12: {
13: print "\t<td>";
14: print ($x*$y);
15: print "</td>\n";
16: }
17: print "</tr>\n";
18: }
19: print "</table>";
20: ?>
21: </body>
22: </html>
```

توابع (Functions)

توابع قلب یک کد درست طراحی شده است و باعث می شوند کدها خوانا تر شوند و بتوان دوباره از آنها استفاده نمود. هیچ پروژه بزرگی بدون استفاده از تابع نمی تواند انجام شود.

در این فصل می خوانیم :

- چگونه یک تابع را معرفی کنیم و از آن استفاده کنیم.
- چگونه مقدایر به تابع ارسال کنیم و از آنها مقادیر را بازخوانی کنیم.
- چگونه توابع با صورت داینامیک استفاده کنیم.
- چگونه به متغیرهای Global در توابع دسترسی پیدا کنیم.
- چگونه به یک تابع حافظه دهیم.

تابع چیست ؟

شما می توانید تابع را یک ماشین در نظر بگیرید. یک ماشین مواد اولیه را از شما می گیرد و بر روی آنها عملیات از پیش تعیین شده را انجام می دهد و در نهایت به شما محصولی را می دهد. تابع مقادیر را از شما دریافت می کند ، بر روی آنها عملیات انجام می دهد و در نهایت کاری که می خواهید با آن انجام می دهد و نتیجه را برای شما بر می گرداند.

اگر شما نیاز به درست کردن یک کیک داشته باشید مسلما خودتان آن را درست می کنید. ولی اگه ۱۰۰۰ کیک بخواهید درست کنید مطمئنا ماشینی طراحی می کنید که برای شما کیک درست کند. در موقع نوشتن تابع هم همیشه باید این مورد مدنظرتان باشد که طوری تابع را بنویسید که بتوان از آن بارها استفاده کرد. تابع در خود کدهایی را جای می دهد که شما هر وقت به آن نیاز دارید آن تابع را صدا می کنید مقادیر اولیه را به آن می دهید و تابع جواب را برای شما برمی گرداند.

فراخوانی تابع :

دو مدل تابع وجود دارد. اولی توابعی هستند که درون خود php هستند و دیگری توابعی است که شما می نویسید.

یکی از ابتدایی ترین توابعی که در خود php هستند تابع print است.

PHP Code:

```
print("Hello Web");
```

در جلو تمامی توابع حتما باید () پرانتزها باشند ، البته print یک استثنا است که بدون پرانتز هم کار می کند

PHP Code:

```
print(("Hello Web");
and
print "Hello Web";
```

هر دو دستور بالا یک خروجی را می دهند ولی این مورد فقط در دستور print عملی است. در مثال بالا ما تابع print را صدا کردیم و مقدار "hello word" رو برای اون فرستادیم. حالا تابع وارد عمل می شود و این جمله را چاپ می کند. تابع شامل دو بخش است. اولی نام تابع Print در اینجا و دیگری مقادیری که برای تابع می فرستیم argument همان که در داخل پرانتز جلوی تابع آمده است. برخی توابع نیاز به چند Argument دارند که آنها را با کاما ، جدا می کنیم. مثلا :

PHP Code:

```
some_function( $an_argument, $another_argument );
```

بسیاری از توابع اطلاعاتی برای شما بر می گردانند در راستای عملی که انجام می دهند. مثلا در صورت درست بودن یا نبودن True یا Falseبر می گردانند.

ABS() مثلا ، یک عدد را می گیرد و قدر مطلق آن را بر می گرداند.

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 6.1</title>
4: </head>
5: <body>
6: <?php
7: $num = – 321;
```

```
8: $newnum = abs( $num );
9: print $newnum;
10: // prints "321"
11: ?>
12: </body>
13: </html>
```

در این مثال ما عدد ۳۲۱- را به $num دادیم. این مقدار را به تابع abs فرستادیم در آنجا محاسبات لازم انجام شد و جواب برگردانده شد که ما آنرا در داخل $newnum ریختیم و آن را چاپ کردیم. البته ما می توانستیم کد را کمی جمع و جور تر بنویسیم و مستقیما عدد را به abs بدهیم و همانجا چاپ کنیم.

PHP Code:

```
print( abs( − 321 ) );
```

این یک خط کد همان کاری را می کند که در مثال قبل انجام دادیم.

قوانین استفاده از توابعی که خودمان می نویسیم هم به همین شکل است.

تعریف یک تابع :

شما می توانید تابع را با استفاده از دستور Function تعریف نمایید.

PHP Code:

```
function some_function( $argument1, $argument2 )
{
// function code here
}
```

نام تابع درست بعد از دستور Function می آید و بلافاصله بعد از آن پرانتزها قرار می گیرند. اگر تابع شما Argument احتیاج دارد ، شما باید متغیرهای مورد نیاز را (که به وسیله کاما از هم جدا شده اند) را داخل پرانتز بنویسید. اگر تابع شما به Argument احتیاجی ندارد داخل پرانتز چیزی ننویسید.

PHP Code:

```
1: <html>
2: <head>
```

```
3: <title>Listing 6.2</title>
4: </head>
5: <body>
6: <?php
7: function bighello()
8: {
9: print "<h1>HELLO!</h1>";
10: }
11: bighello();
12: ?>
13: </body>
14: </html>
```

در خط ۷ کد بالا ما تابع bighello را تعریف کردیم. مشخص است عملیاتی که این تابع انجام می دهد این است که کلمه Hello! را بین کدهای H1 چاپ خواهد نمود. ما تابع bighello را بدون Argument تعریف کردیم ، به همین دلیل داخل پرانتز چیزی ننوشتیم.

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 6.3</title>
4: </head>
5: <body>
6: <?php
7: function printBR( $txt )
8: {
9: print ("$txt<br>\n");
10: }
11: printBR("This is a line");
12: printBR("This is a new line");
13: printBR("This is yet another line");
14: ?>
15: </body>
16: </html>
```

در مثال بالا ما تابع printBR را با Argument تعریف می کنیم. حالا در خطوط ۱۱ و۱۲ و ۱۳ سه مقدار متفاوت را به تابع می فرستیم و مثلا سه خط چاپ شده در خروجی خواهیم داشت. $txt همانطور که می بینید در خط ۷ تعریف شده است. موقعی که در خطوط ۱۱ و ۱۲ و۱۳ ما تابع را صدا می کنیم $txt هر دفعه مقداری که برایش فرستاده شده است را به خود می گیرد و در خط ۹ آن را چاپ می کند.

هر مقدار که بخواهیم می توانیم این تابع را اجرا کنیم و خروجی بگیریم.

توجه داشته باشید که اگر تابعی Argument نیاز داشته باشد ، موقع صدا کردن تابع باید حتما مقدار برای آن بفرستیم.

نکته در صورتیکه تابع را به این صورت تعریف کنید :

PHP Code:

```
function printBR( $txt = "nothing")
```

در این حالت $txt به صورت default مقدار "nothing" را دارد. یعنی اگر ما موقع صدا کردن تابع مقداری برای Argument نفرستیم تابع خودش مقدار Default را به $txt می دهد ولی اگر ما مقدار بفرستیم $txt برابر با مقدار فرستاده شده می باشد.

مثلا :

[php]

PrintBR();
PrintBr("Hello");

PHP Code:

در خط اول کد بالا چاپ خواهد شد Nothing و در خط بعد چاپ می شود . Hello مقدار Default در برخی توابع بسیار کارا هستند .

[b]بازگرداندن مقادیر از توابع تعریف شده توسط کاربر

شما می توانید از داخل تابع با استفاده از دستور Return مقداری را برگردانید. دستور return عملیات تابع را متوقف می نماید و مقدار گفته شده را بر می گرداند.

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 6.4</title>
4: </head>
5: <body>
6: <?php
7: function addNums( $firstnum, $secondnum;
8: {
9: $result = $firstnum + $secondnum )
10: return $result;
11: }
12: print addNums(3,5);
13: // will print "8"
14: ?>
15: </body>
16: </html>
```

کد بالا عدد ۸ را در خروجی چاپ می کند. عددهای ۳ و ۵ در $firstnum **and** $secondnum ذخیره شده-اند و بعد با هم جمع شدند. و جواب آنها در $result ذخیره شد و سپس در خط ۱۰ آن مقدار برگردانده شده است.

با دستور Return شما می توانید هر چیزی را برگردانید مثلا :

PHP Code:

```
function addNums( $firstnum, $secondnum )
{
return ( $firstnum + $secondnum );
}
```

تابع بالا نیز دقیقا همان کاری را می کند که تابع قبلی می نمود.

حتی می توانید به این صورت نیز از return استفاده کنید.

PHP Code:

```
return 4;
```

می توانید نتیجه یک عملیات را برگردانید :

PHP Code:

```
return ( $a/$b );
```

یا حتی مقداری از یک تابع دیگر را برگردانید :

PHP Code:

```
return ( another_function( $an_argument ) );
```

صدا کردن یک Function به صورت داینامیک :

این امکان وجود داره که شما اسم تابع یک String یا یک متغیر بگذارید. و برای صدا کردنش از اون استفاده کنید. مثلا :

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 6.5</title>
4: </head>
5: <body>
6: <?php
7: function sayHello()
8: {
9: print "hello<br>";
10: }
11: $function_holder = "sayHello";
12: $function_holder();
13: ?>
14: </body>
15: </html>
```

در مثال بالا در خط ۷ تابع با اسم Sayhello تعریف شده و در خط ۱۱ function_holder یک متغیری تعریف شده با مقدار sayHello حالا می توان از function_holder با اضافه پرانتزها برای صدا کردن تابع استفاده کرد.

شاید این سوال پیش بیاد که چرا ما باید همچین چیزی رو لازم داشته باشیم. در مثال فوق عملا ما فقط کار خودمون رو زیادتر کردیم. ولی در واقع در برخی مواقع لازم داریم که جریان کد رو با توجه به مولفه های داخل url یا شرایط برنامه عوض کنیم.

یعنی مثلا در شرایطی یک function اجرا شود و در شرایط دیگه function دیگری.

متغیر ها در داخل تابع :

مهم : متغیرهایی که داخل یک تابع تعریف می شوند ، فقط در داخل همون تابع قابل دسترسی هستند. یعنی اون متغیر ها بیرون تابع یا در تابع های دیگر در دسترس نیستند. در پروژهای بزرگ این امکان خیلی به شما کمک می کند

چون شما می توانید از اسم های تکراری برای متغیرهایتان در تابع های مختلف استفاده کنید بدون اینکه دخالتی در یکدیگر داشته باشند.

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 6.6</title>
4: </head>
5: <body>
6: <?php
7: function test()
8: {
9: $testvariable = "this is a test variable";
10: }
11: print "test variable: $testvariable<br>";
12: ?>
13: </body>
14: </html>
```

در مثال بالا خروجی چیزی رو چاپ نخواهد نمود. چون $testvariable که در خط ۱۱ برای چاپ خوانده می شود قبلا تعریف نشده است و $testvariable خط ۹ فقط در داخل function قابل دسترسی هستند.

استفاده از متغیر به صورت Global (یعنی داخل و خارج تابع ها) :

به صورت Default متغیرهای تعریف شده بیرون یک تابع ،داخل تابع در دسترس نیست.

مثلا در مثال زیر

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 6.7</title>
4: </head>
5: <body>
6: <?php
7: $life = 42;
8: function meaningOfLife()
9: {
10: print "The meaning of life is $life<br>";
11: }
12: meaningOfLife();
13: ?>
14: </body>
15: </html>
```

خروجی خالی چاپ می شود و مقدار $life را چاپ نخواهد کرد. در برخی موارد ما نیاز به استفاده از متغیرهای بیرون تابع داخل یک تابع داریم. برای این کار کافیه که از دستور Global استفاده کنیم.

به طور مثال می توانید برای این منظور کد بالا را به صورت زیر بازنویسی کنید :

PHP Code:

```
1: <html>
2: <head>
```

```
3: <title>Listing 6.8</title>
4: </head>
5: <body>
6: <?php
7: $life = 42;
8: function meaningOfLife()
9: {
10: global $life;
        11: print "The meaning of life is $life<br>";
12: }
13: meaningOfLife();
14: ?>
15: </body>
16: </html>
```

در خط ۱۰ از دستور global $life; استفاده کردیم. در این حالت مقدار $life که بیرون تابع و در خط ۷ تعریف شده در داخل تابع قابل دسترس می شود و خروجی این کد The meaning of life is 42 چاپ خواهد شد.

شما باید برای هر متغیری که می خواهید از آن در تابع استفاده کنید از این دستور استفاده کنید. و همچنین هر تابعی که می خواهید از متغیری خارج از آن تابع استفاده کند باید از این دستور استفاده شود.

مهم : دقت کنید که اگه $life داخل تابع تغییر دهید مقدار $life در کل برنامه عوض می شود.

جی دی (GD)

مقدمه

پی اچ پی فقط به دادن خروجی متن محدود نیست شما میتونید عکس های متعددی با فرمت های متنوع با پی اچ پی خروجی داشته باشید مثل XPM , WBMP . JPG , GIF , PNG و البته چندین فرمت دیگر که باید پی اچ پی رو با اون سازگاری بدین.

پی اچ پی میتونه عکس رو مستقیم به سوی مرور گر هدایت کنه. البته پی اچ پی به صورت پیش فرض از ساخت عکس پشتیبانی نمیکنه که باید آن را نصب کنید (GD) البته این کتابخانه (library) روی بیشتر سرور ها به صورت پیش فرض نصب هست ...

اما اگر نصب نبود می‌توانید این را از http://www.boutell.com/gd/ دانلود و نصب کنید و یا به مدیر سرور خود بگویید که آن را نصب کند ...

در جیدی پشتیبانی از PNG در نسخه ۱,۶ به بعد به وجود آمد و پشتیبانی از GIF در نسخه ۲,۰,۲۸- به وجود آمد.

یک مثال :

PHP Code:

```php
<?php

header("Content-type: image/png");
$string = $_GET['text'];
$im     = imagecreatefrompng("images/button1.png");
$orange = imagecolorallocate($im, 220, 210, 60);
$px     = (imagesx($im) - 7.5 * strlen($string)) / 2;
imagestring($im, 3, $px, 9, $string, $orange);
imagepng($im);
imagedestroy($im);

?>
```

برای استفاده از مثال بالا در صفحات خود از این دستور استفاده کنید :

HTML Code:

```
<img src="button.php?text=text">
```

حالا این مثال چیکار میکنه ؟

مثال بالا (button.php) اول فایل images/button1.png رو باز میکنه و بعد متغیر TEXT رو روش مینویسه (به صورت یک لایه جدا)

مثلا اگر شما دکمه های بسیاری و مانند هم با متن متفاوت دارید می‌توانید با استفاده از این اسکریپت در حافظه صرفه جویی کنید و دیگر همه دکمه ها را در سایت آپلود نکنید و به عبارتی دکمه به صورت دینامیکی (dynamically) ساخته می‌شود.

در پایین لیست و نحوه استفاده تمامی توابع gd نوشته شده :

gd_info : دادن اطلاعات دربارهٔ نسخه و دیگر چیزهای GD نصب شده . این تابع خروجی از نسخه جی دی و فرمت های پشتیبانی شده توسط جی دی و ... را به شما می‌دهد :

PHP Code:

```
<?php
print_r(gd_info());
?>
```

مثال بالا خروجی زیر را به‌دنبال داره (برای همه یکسان نیست)

Code:

```
Array
(
    [GD Version] => bundled (2.0.28 compatible)
    [FreeType Support] => 1
    [FreeType Linkage] => with freetype
    [T1Lib Support] =>
    [GIF Read Support] => 1
    [GIF Create Support] => 1
    [JPG Support] => 1
    [PNG Support] => 1
    [WBMP Support] => 1
```

```
    [XBM Support] => 1
    [JIS-mapped Japanese Font Support] =>
)
```

getimagesize : گرفتن اندازه عکس (ابعاد)

با استفاده از این تابع میتوان اندازه عکس رو به دست آورد مثلا ۴۸*۵۲ این تابع از GIF, JPG, PNG, JPX, JB2, JPC, XBM, or WBMP ,SWF, SWC, PSD, TIFF, BMP, IFF, JP2 پشتیبانی می‌کند .

مثلا :

PHP Code:

```
<?php
list($width, $height, $type, $attr) = getimagesize("img/flag.jpg");
echo "<img src=\"img/flag.jpg\" $attr alt=\"getimagesize() example\" />";
?>
```

در نسخه ۴,۰,۵ به بعد از URL هم پشتیبانی شد :

PHP Code:

```
<?php
$size = getimagesize("http://www.example.com/gifs/logo.gif");
?>
```

image_type_to_extension:گرفتن پسوند عکس برای image type()

روش استفاده :

PHP Code:

```
image_type_to_extension ( int imagetype [, bool include_dot] )
```

image_type_to_mime_type : گرفتن mime type یک عکس

mime type در شناسوندن نوع عکس (مثلا PNG) به مرورگر کاربرد دارد البته در جاهای دیگر هم کاربرد دارد .

مثال :

PHP Code:

```php
<?php
header("Content-type: " . image_type_to_mime_type(IMAGETYPE_PNG));
?>
```

این تابع میتواند حاوی یکی از خروجی های زیر باشد :

خروجی نوع عکس

image/gif IMAGETYPE_GIF
IMAGETYPE_JPEG image/jpeg
IMAGETYPE_PNG image/png
IMAGETYPE_SWF application/x-shockwave-flash
IMAGETYPE_PSD image/psd
IMAGETYPE_BMP image/bmp
IMAGETYPE_TIFF_II(intel byte order) image/tiff
IMAGETYPE_TIFF_MM (motorola byte order) image/tiff
application/octet-stream IMAGETYPE_JPC
IMAGETYPE_JP2 image/jp2
application/octet-stream IMAGETYPE_JPX
IMAGETYPE_JB2 application/octet-stream
IMAGETYPE_SWC application/x-shockwave-flash
IMAGETYPE_IFF image/iff
IMAGETYPE_WBMP image/vnd.wap.wbmp
IMAGETYPE_XBM image/xbm

توجه : این تابع نیازی به کتابخانه جی دی ندارد .

image2wbmp: دادن خروجی به یک فایل یا مرورگر

مثال :

PHP Code:

```php
<?php
$file = 'php.png';
$image = imagecreatefrompng($file);
header('Content-type: ' . image_type_to_mime_type(IMAGETYPE_WBMP));
image2wbmp($image); // output the stream directly
?>
```

PHP Code:

```
<?PHP if($os !=="FREE"){ die("Please install a free os "); } ?>
```

آرایه ها در PHP

دوست دارم قبل از اینکه شروع به یادگیری درس جدید بکنیم یه مثال با فانکشن(تابع) رو نگاه کنید .

PHP Code:

```
<?php
function fonts($txt,$siz)
{
print "<font size=$siz >$txt";
}
fonts("salam",5);
fonts("khoobi",10);
 ?>
```

خیلی ساده و کاربردی !

خوب رسیدیم به Array یا همون آرایه ها : آرایه ها در واقع مانند یک ظرف هستن که میتونیم چندین مقدار رو توشون قرار بدیم و سپس از توی ظرف مقادیرمون رو یا مقدار مورد نظرمون رو برداریم ویکی از پرکاربرد ترین دستورات در همه زبانها بشمار میاد. در ضمن ترتیب آرایه ها از صفر شروع میشه.

برای مثال :

index value Array

Mehdi ۰

Asef ۱

tsotodeh ۲

knowhow ۳

Piter1356 ۴

بزارید چندین مقدار رو در یک مقدار دیگر ذخیره کنیم :

PHP Code:

```
$users=array("Mehdi","Asef","tsotodeh","knowhow","Piter1356");
```

در اینجا اگه بخواهیم مقدار Asef رو چاپ کنیم کافیه بنویسیم :

PHP Code:

```
Print "$users[1]";
```

و برای اضافه کردن آرایه جدید میتونیم از دستور زیر استفاده کنیم

PHP Code:

```
$users[]="azemati";
```

و برای صدا کردن فقط اندیس یا ایندکسش رو صدا بزنیم .

در ضمن میتونیم خودمون هم ایندکس گذاری کنیم یعنی بجای ۰ – ۱ - ۲ – ۳ - .. .. خودمون حروف بزاریم

برای مثال به کد زیر توجه کنید :

PHP Code:

```
$user = array (name=>"Asef",job=>"Programming",age=>24,
"skill"=>"everyThing");
```

خوب حالا به‌راحتی هرکدوم رو که بخواهیم میتونیم صدا بزنیم :

PHP Code:

```
Print"$user[name]";
Or
Print"$user[job]";
```

همونطور که دیدید در قسمت تعریف کردن ارایه ها حروف را در داخل گیومه قرار میدیم و قرار دادن اعداد اجباری نیست و هم میتونن داخل گیومه قرار بگیرن هم نگیرن .

همچنین میتونیم متغییر های داخل آرایمون رو مقدار دهی کنیم برای مثال :

PHP Code:

```
$user[name]="azemati";
$user[job]="webmaster";
```

و برای تعریف کردن چندین ارایه با چندین مقدار به این صورت عمل میکنیم :

PHP Code:

```
$user = array (array(name=>"Asef",job=>"Programming",age=>24,
"skill"=>"everyThing"),array(name=>"mehdi",job=>"Programming",age=>18,
"skill"=>"noThing"),array(name=>"daftarkhatereh",job=>"Programming",age=>24,
"skill"=>"everyThing")    );
```

و برای صدا کردن مقداری خاص به این شکل عمل میکنیم :

PHP Code:

```
print $user[0][job];
//Print "Programming"
```

برای پی بردن به تعداد مقادیر یک آرایه می‌توانید از دستور print $user[count]; استفاده کنید . ولی دقت کنید که در دستور کانت مقدار ایندکس ما از صفر شروع نمیشه بلکه از **یک** شروع میشه برای مثال برای دسترسی به مقداری با استفاده از کانت به این صورت عمل میکنیم :

PHP Code:

```
<?php
$users=array("a","b","c","d","e");
print $users[count($users)- 1];
//Print 5
?>
```

که در این کد اخرین مقدار یعنی حرف ای را چاپ میکنه !

ارایه ها را به شکل های گوناگونی میتوان استفاده کرد و کاربرد بسیار زیادی دارند

دستور دیگری که میخواهیم اشنا بشیم دستور Array_merge() هستش که با مثالی اشنا میشیم :

PHP Code:

```
<?php
$first = array("a", "b", "c");
$second = array(1,2,3);
$third = array_merge( $first, $second );
foreach ( $third as $val )
{
```

```
print "$val<BR>";
}
?>
```

در این مثال این کد این متغییر های اول و دوم را با هم ترکیب میکنه و در متغییر سوم قرار میده اما دستور foreach مثل دستور for عمل میکنه با این تفاوت که در اینجا میاد متغییر سوم رو در متغییر جدیدی میزاره و اون رو چاپ میکنه شکل کلی این دستور به این صورت هستش :

PHP Code:

```
foreach( $array as $temp )
{
//...
}
```

برای مرتب کردن یک ارایه از دستور sort استفاده میکنیم مانند مثال

PHP Code:

```
<?php
$an_array = array("x","a","f","c");
sort( $an_array);
foreach ( $an_array as $var )
{
print "$var<BR>";
}
?>
```

بدیهی است که با اعداد هم میتوانیم همچین کاری رو بکنیم.

همونطور که دیدید با دستور ;Sort() میتونیم مقادیر یک ارایه رو مرتب کنیم. در مثال بالا در خروجی حروف به صورت مرتب شده نمایش داده میشوند .

دستور ;assort() هم داریم که بر اساس مقادیر آرایه ها آنها را مرتب میکنه باز به مثالی دیگر توجه کنید :

PHP Code:

```
<?php
$first = array("first"=>5,"second"=>2,"third"=>1);
asort( $first );
foreach ( $first as $key => $val )
{
```

```
print "$key = $val<BR>";
}
?>
```

که در خروجی این چنین چاپ میشود :

third = 1
second = 2
first = 5

در دستور بالا همونطور که دید بر حسب مقادیر مرتب شد ولی اگر بخواهیم بر حسب ایندکسی که میدهم مرتب بشه میتونیم از دستور ksort(); استفاده کنیم :

PHP Code:

```
<?php
$first = array("x"=>5,"a"=>2,"f"=>1);
ksort( $first );
foreach ( $first as $key => $val )
{
print "$key = $val<BR>";
}
?>
```

که در خروجی چاپ میکنه

a = 2
f = 1
x = 5

کلاسها در PHP

در این فصل می خواهم به بررسی یکی از زیباترین، و در عین حال خطرناکترین مباحث برنامه نویسی بپردازم، این مبحث از این جهت خطرناک هست که اگر شما اصول اولیه را یاد بگیرید و این نوع برنامه نویسی بر شما تاثیر بگذارد از آن پس دیگر به همه چیز به چشم یک شیء نگاه خواهید کرد و تمام روشهای برنامه نویسی گذشته خود را به کنار خواهید گذاشت. در این مقاله اصلا قصد ندارم که شروع کنم به آوردن مثالهایی از Object در دنیای واقعی و فرض می کنم که شما مثالهایی مثل رنگ ماشین و یا تلفن را بلد باشید! در حالت کلی یک شیء شامل یک سری **متغیرها** و **توابع** می باشد که درون یک قالب کلی به نام کلاس قرار دارند، به متغیرهای درون کلاسها Properties و به توابع موجود در آن Method گفته می شود.

آبجکت ها :

آبجکت چیست ؟ مجموعه‌ای از متغیرها و توابع است که از یک الگوی خاص به نام کلاس ساخته شده اند . اما کلاس ها چی هستن ؟ فرض کنید ما یک شرکت داریم که این شرکت از بخش های مختلفی تشکیل شده است حال ما در هربخش احتیاج داریم که هر ماه یک گزارش مالی بگیریم ! ما اینجا دو کار میتونیم بکنیم هم میتونیم برای هربخش چند نفر بزاریم و آنها گزارش مالی را تهیه کنند و هر ماه تحویل دهند در این صورت در هربخش شلوغی و همچنین کارمند بیشتری نیازمندیم ! راه دیگر اینست که یک قسمت به عنوان اتاق گزارش مالی درست کنیم و هر بخش داده های خود را به این قسمت بدهیم و گزارش مالی خود را دریافت کنیم در این روش هم بخش ها منظم تر خواهند بود و هم دیگر احتیاج به کارمند اضافی نداریم .

در اینجا آبجکتها نقش کارمند در بخش گزارش مالی را بر عهده دارند فکر میکنم مفهوم کلاس را درک کرده باشید. .

بزارید شکل کلی یک کلاس را برایتان نشان دهم:

PHP Code:

```
Class First_class
{
// properties
```

```
این شکل کلی از یک کلاس هستش//
// methods
}
```

در مثال بالا به این موارد توجه کنید:

ساخته شدن یک کلاس توسط کلمه کلیدی class صورت می گیرد.

شما در هر جای کلاس قادر به تعریف متغیرهای کلاس یا properties هستید اما بهتر است که آنها را در ابتدای class تعریف کنید.

بعد از آن به معرفی توابع کلاس یا methods بپردازید.

تمام این موارد بین دو {} تعریف کلاس صورت می گیرند.

پروپرتی ها:

آبجکتها به متغییر های خاصی دسترسی دارند که به آنها پروپرتی می‌گویند این پروپرتی ها میتونند در هرجای بدنه کلاس باشند اما برای اینکه کد مون مرتب باشه بهتره که در بالای کلاس تعریف بشن. بگزارید با مثالی دیگر بیشتر توضیح بدم :

PHP Code:

```
Class f_class {
var $name="mehdi";
}
$obj1=new f_class();
$obj2=new f_class();
$obj1->name="Ali";
print "$obj1->name<br>";
print"$obj2->name<br>";
```

دیدیم که برای اختصاص دادن یک کلاس به یک متغییر اینگونه عمل کردیم:

PHP Code:

```
$obj1=new f_class();
```

علامت <- به شما اجازه میده تا به متغییر های درون یک کلاس دسترسی داشته باشید و اونها رو تغییر بدید همونطور که در کد میبینید ما در خط ششم متغییر name در آبجکت یک رو مساوی علی قرار دادیم که باعث عوض شدن متغییر میشه همچنین برای چاپ خروجی نیز به همین صورت عمل کردیم ولی بدون علامت مساوی

PHP Code:

```
Print  " $obj1->name ";
```

متد ها :

متدها در واقع توابعی هستند که داخل یک کلاس وجود دارند بزارید با یک مثال واضح تر بیان کنم :

PHP Code:

```
class f_class
   {
var $name;
function sayHello()
 {
Print "Hello World";
}
    }
$obj1=new f_class();
$obj1->sayHello();
// Hello World  میشود  چاپ
```

همونطور که میبینید یک متد خیلی شبیه به تابع معمولی هستش با این تفاوت که متد همیشه داخل کلاس تعریف میشه در ضمن شما میتونید با علامت <- یک متد ابجکت را صدا بزنید.

مهمتر اینکه متدها به اعضای متغییر های یک کلاس دسترسی دارند

شما همین الان دیدید که چطوری به یک پروپرتی از خارج یک آبجکت دسترسی پیدا کنیم اما چطوری یک آبجکت میتونه خودشو به اصطلاح Return کنه :

PHP Code:

```
class f_class
   {
```

```
var $name="mehdi";
function sayHello()
 {
Print "Hello My names $this->name<br>";
}
    }
$obj1=new f_class();
$obj1->sayHello();
// Hello My names mehdi   چاپ  میکنه
```

یک عبارت مخصوص رو بکار بردیم به اسم $this تا کلاس به آبجکت کنونی Return بشه شما با ترکیب این عبارت با علامت -> میتونید داخل یک کلاس به هر پروپرتی و متدی که بخواهید دسترسی داشته باشید حالا اگه بخواهیم به پروپرتی name در همه آبجکت های کلاسمون مقدار خاصی بدیم میتونیم به این صورت عمل کنیم :

PHP Code:

```
class f_class    {
var $name="mehdi";
function setName($n){
$this->name=$n;     }
function sayHello()
 {
Print "Hello My names $this->name<br>";
}
    }
$obj1=new f_class();
$obj1->setName("Ali");
$obj1->sayHello();
// Hello My names Ali   چاپ  میکنه
```

همونطور که دیدید اومدیم یه تابع تعریف کردیم که اسم رو بتونیم همه جا تغییر بدیم واز دستورthis برای عوض کردنش داخل کلاس استفاده کردیم.

در ابتدا اسم ما مهدی بود ولی بعد از اینکه تابع عوض کردن اسم رو بکار بردیم متد صدا زده شد و اسم تغییر کرد.

کد بالا رو میتونستیم بصورت ساده تر وکمی پیچیده تر هم بنویسیم :

PHP Code:

```
class first_class {
 var $name;
 function first_class( $n="mehdi" ) {
 $this->name = $n;
 }
 function sayHello() {
 print "hello my name is $this->name<BR>";
 }
 }
 $obj1 = new first_class("´Ali");
 $obj2 = new first_class("Asef");
 $obj1->sayHello();
 // hello my name is Ali  چاپ  میکنه

 $obj2->sayHello();
 // hello my name is Asef  چاپ  میکنه
```

همونطور که دیدید اومدیم و یه متد یا تابع با همون اسم کلاس خودمون ساختیم و مقدار دیفالتش رو روی مهدی گزاشتیم که اگر هیچی وارد نشد این عبارت چاپ بشه.

حالا در پایین در خط دهم و یازدهم هنگامی که ابجکت یک و ابجکت دو رو به کلاس ها اختصاص میدیم همون موقع هم مقدار اسم رو عوض میکنیم و مشکلی پیش نمیاد کلاسمون هنگامی که متد صدا زده میشه خود به خود صدا زده میشه.

حال بزارید با عبارت دیگری به اسم extends اشنا بشیم این عبارت به این معنی ارث بردن است به این مثال توجه کنید :

PHP Code:

```
class first_class{
 var $name = "mehdi";
 function first_class( $n ) {
```

```
$this->name = $n;
}
function sayHello(){
print "Hello my name is $this->name<br>";
}
}
class second_class extends first_class {
}
$test = new second_class("son of mehdi");
$test->sayHello();
// outputs "Hello my name is son of mehdi"
```

همونطور که میبینید ما کلاس دو رو تنظیم کردیم تا از کلاس یک ارث بری کنه وهمه متد های کلاس یک را خواهد داشت.

ما میتونیم اینجا داخل کلاس دو یک تابع با اسم sayHello بسازیم و بگیم عبارت I dont know my name رو چاپ کنه در اینصورت اگه حتی در خط ۱۲ ام ما یک عبارت برای تعویض اسم بدیم تاثیری نمیکنه و همون عبارت I dont know my name رو چاپ میکنه راه دیگری هم هست و اون چاپ هردو کلاس باهم هست :

PHP Code:

```
class first_class {
var $name = "harry";
function first_class( $n ) {
$this->name = $n;
}
function sayHello() {
print "Hello my name is $this->name<br>";
}
}
class second_class extends first_class  {
function sayHello() {
print "I'm not going to tell you my name -- ";
first_class::sayHello();
}
}
```

```
$test = new second_class("son of harry");
$test->sayHello();
// "I'm not going to tell you my name -- Hello my name is son of harry" چاپ میکنه
```

همونطور که میبینید دستور (متد:: وارث) ما میتونیم هر متدی رو که ما تغییرش دادیم دوباره صدا بزنیم چون در کلاس دو ما متد sayHello رو تغییر دادیم با این دستور اونو دوباره برگردوندیم .

آموزش برنامه نویسی شیء گرا با PHP

کلاسها (Classes) :

یک کلاس تعریف یا نمای یک نوع خاص داده است و کلاسها به عنوان روشی برای حالت دادن به تمام انواع متفاوت اشیاء و سیستم شما عمل می‌کنند . هنگامی که می‌خواهیم یک شیء جدید را تعریف کنیم ، ابتدا از کلمه کلیدی class برای تعریف آن ، پیش از استفاده از آن در اسکریپت‌های PHP خود استفاده می‌کنیم . تفاوت واضح یک کلاس و یک شیء این است که کلاس‌ها اشیایی را تعریف می‌کنند که در برنامه‌هایمان به کار می‌بریم . پیش از آن که درباره روش ساخت یک کلاس صحبت کنیم ، می‌خواهیم شروع به تأمل کنید که یک کلاس نمایشی از یک ایده است . مهم است که هنگامی که کلاس‌های خودتان را طراحی می‌کنید . آنها یک هدف را دنبال کنند و تمامی رفتاری را که از آن ایده انتظار می‌رفت را فراهم کنند .

یک کلاس در PHP حاوی سه کامپوننت اصلی است : members (اعضاء) که به آنها به عنوان داده یا صفت اشاره می‌شود) متدها ، و Consructor یک عضو members تکه‌ای از داده است که شیء در بر دارد . اشیاء می‌توانند هر تعداد عضو داشته باشند . برای مثال ، اگر قرار باشد اتومبیل را با استفاده از یک کلاس طراحی کنیم ، یک چرخ در حال دوران یا گیربکس باید به عنوان یک عضو از ماشین تعریف شوند .

متدها سرویس‌هایی هستند که شیء برای سرویس گیرنده‌هایش فراهم می‌کند که از اعضا داخلی آن استفاده می‌کنند و آنها را دستکاری می‌کنند . برای مثال ، اگر قرار باشد اتومبیل را با استفاده از یک کلاس طراحی کنیم ، یک چرخ در حال دوران یا گیربکس باید به عنوان یک عضو از ماشین تعریف شوند .

متدها سرویس‌هایی هستند که شیء برای سرویس گیرنده‌هایش فراهم می‌کند که از اعضا داخلی آن استفاده می‌کنند و آنها را دستکاری می‌کنند . برای مثال ، یک کلاس car می‌تواند یک متد را برای روشن کردن وسیله نقله و استفاده از چرخ در حال دوران در داخل آن فراهم کند .

یک Constructor متد خاصی است که شیء را درون وضعیت آماده آن معرفی می‌نند . تنها یک Constructor برای یک شیء در PHP می‌تواند موجود باشد . در یک کلاس car ، افزودن بدنه ، موتور ، لاستیک‌ها ، گیربکس ، صندلی و غیره بر روی car با هم متفاوتند . هنگامی که سرویس گیرنده‌ها می‌خواهند از متدهای روی یک شیء استفاده کنند ، Constructor اطمینان می‌دهد که هر متد عملیات را با موفقیت به انجام خواهد رساند و نتیجه مورد انتظار را برخواهد گرداند . برای مثال ، برای روشن کردن رادیو درون اتومبیل شما ، باید یک رادیو نصب شده باشد . در این نمونه ، Constructor مسئول اطمینان بخشیدن از این موضوع است که رادیو پیش از استفاده نصب شده است. به غیر از مقدار دهی شیء به یک وضعیت آماده معتبر ، تفاوت اساسی دیگر این است که یک Constructor هیچ مقدار برگشتی صریحی ندارد . تمامی Constructor ها یک متغیر جدیداً اختصاص یافته را برای استفاده در برنامه شما بر می‌گردانند .

در نتیجه ، برگرداندن یک مقدار در Constructor کلاس ، غیرقانونی است . درباره استفاده از اشیاء در برنامه‌های شما ، در بخش بعدی بیشتر صحبت خواهیم کرد . اطمینان یافتن از طراحی مناسب اشیاء و Constructor های آنها ، مسئله‌ای است که توسعه دهندگان زیادی اغلب با آن روبرو می‌شوند . هنگامی که کلاس برنامه‌نویسان را وادار می‌کند که اعضای شیء را پیش از استفاده از متدهای خود تنظیم کنند یا هنگامی که کلاس برنامه‌نویس را وادار می‌کند تا از ترتیب خاصی ، هنگام فراخوانی متدهای شیء پیروی کند ، کد گیج کننده و مبهمی را ایجاد می‌کند . از OPP به این منظور استفاده می‌کنیم که به طور کلی از بروز چنین مسئله‌ای جلوگیری کنیم . اگر کلاس مهندسی شده است تا از Constructor های خود برای معرفی بخشهای کلیدی کلاس استفاده نکند ، اشکال از طراحی ضعیف ناشی می‌گردد . دوباره در همان تله گرفتار نشوید .

کلاسی که خوب طراحی شده باشد برنامه‌نویسی ، اشکال‌زدایی و نگهداری زیادی را حذف می‌کند .

بیایید نگاهی به دستور زبان کلی برای Class در PHP بیندازیم ، که استفاده از این سه نوع کامپوننت را نشان می‌دهد :

PHP Code:

```
class ClassName [extends  ParentclassName]
{
    var $member1;
    var $member2;
```

```
    …
    var $memberN;
    // Constructor
    function Class Name()
    {
    }
    function method1()
    {
    }
…
    function method2()
    {
    }
    function methodN()
    {
    }
}
```

همان‌گونه که می‌بینید ، یک کلاس چیزی نیست جز یک مجموعه از اعضای تعریف شده (متغیرها) و متدها (توابع) . اعضا می‌توانند یا انواع داده اولیه نظیر integer ها و رشته‌ها یا انواع پیچیده‌تری نظیر آرایه‌ها یا اشیاء دیگر باشند . از آنجایی که PHP از شما انتظار ندارد که انواع را تعریف کنید ، فقط می‌توانید متغیرهایتان را در بالای کلاس ، به صورتی که در بالا نشان داده شد ، نام ببرید .

با PHP می‌توانید متغیرهایی را در تابع خود ایجاد کنید ؛ آنها به همان خوبی که مورد انتظارتان است کا رخواهند کرد . اگرچه ، تمرین خوبی نخواهد بود اگر این کار را انجام دهید . این به آن خاطر است که وقتی برنامه‌نویسان دیگر به کلاس شما نگاه می‌کنند ، فوراً تمامی اعضای آن را پیش از نگاه کردن به پیاده‌سازی توابع بشناسند .

متدها به سادگی تمام سرویس‌هایی هستند که این کلاس تضمین می‌کند تا برای سرویس گیرنده‌هایش فراهم کند . سرویس گیرنده‌ها می‌توانند برنامه‌های دیگر ، پروژه‌های دیگر و غیره باشند.

در این قسمت به ساخت یک کلاس بسیار ساده می پردازیم. همچنین بیشتر در مورد شیء ها صحبت خواهیم کرد.

بیایید کد یک کلاس Car را ایجاد کنیم . در این مثال شروع به تعریف کلاس خود می‌کنیم . این کار را با استفاده از کلمه کلیدی Class در خط دوم می‌کنیم . تمرین مهندسی نرم‌افزار خوبی است تا حرف اول تمامی نامهای کلاسها را برای تشخیص آنها از متغیرها یا توابع با حروف بزرگ بنویسیم .

برنامه‌نویسان این کار را برای سالها در زبانهای گوناگون دیگر انجام داده‌اند . شناسایی Constructor در میان متدهای متنوع دیگر در کلاس ساده است . همچنین عادت خوبی است که نام فایل‌هایتان را با نام کلاس نامگذاری کنید . نظیر Car.php یک فایل تنها باید شامل یک کلاس باشد . اگر چند کلاس دارید که به یکدیگر مرتبط هستند ، نظیر مجموعه کلاسهای انواع داده اصلی ، باید آنها را درون یک زیرادایرکتوری تحت برنامه کاربردی اصلی خود قرار دهید . اگر روی یک پروژه عظیم کار می‌کنید ، این تمرین ضروری است .

با بزرگتر شدن سیستم‌ها ، ضروری خواهد بود که از یک ساختار دایرکتوری درخت مانند ، برای نگهداری تمامی کلاسهایی که در برنامه کاربردی وب شما بکار می‌روند ، استفاده کنید . شما باید از () include_onceیا require_once() برای اضافه کردن کلاسها به فایل‌های سورس خود در هنگام نیاز به آنها استفاده کنید .

PHP Code:

```
<? Php
// Car.php
class Car
{
```

در یک مدل بی‌نهایت ساده از یک اتومبیل ، کلاس شامل موتور و نمایش کلیدی برای روشن کردن اتومبیل است . یک اتومبیل واقعی باید یک بدنه ، یک پدال گاز و یک ترمز و یک چرخ ، گیربکس و غیره داشته باشد ، اما این تنها برای نمایش است :

PHP Code:

```
var $engine ;
var $requiredkey;
```

اتومبیل ما همچنین یک constructor دارد که موتور آن را تنظیم می‌کند و کلیدی دارد که اتومبیل را روشن می‌کند . اگر این عناصر اتومبیل را شناسایی نمی‌کردیم ، هر فراخوانی start() و stop() از کار می‌ایستاد و خطاهایی را بر می‌گرداند . چنانکه قبلاً ذکر کردیم ، وظیفه constructor شناسایی تمامی عناصر شیء ، جهت کسب اطمینان از امکان استفاده از تمامی سرویس‌ها در هنگام نیاز است .

توجه داشته‌باشید که اگر می‌خواهید به یک عضو کلاس رجوع کنید ، باید یک کلمه کلیدی$this-> را در ابتدای نام عضو قرار دهید . این ارتباط از جاوا یا C++ متفاوت است که در آنها اختیاری است . این بدلیل کارایی ضعیف PHP ، سه سطح namespace وجود دارند که متغیرها در آن مرتب می‌شوند . (یک namespace اصولاً مجموعه‌ای از نام متغیرها است).

پایین‌ترین سطح namespace برای متغیرهای محلی درون توابع یا متدها بکار می‌رود . هر متغیر ایجاد شده در این سطح به namespace محلی اضافه شده است . Namespace بعدی حاوی تمامی اعضای یک شیء است. بالاترین سطح namespace برای متغیرهای عمومی بکار می‌رود . کلمهه کلیدی $this به PHP می‌گوید که متغیر را از namespace شیء می‌خواهید (سطح وسط) . اگر فراموش کنید که کلمه کلیدی$this را لحاظ کنید ، یک متغیر کاملاً جدید را در namespace محلی ایجاد خواهید کرد . از آنجایی که این به یک متغیر کاملاً متفاوت از آنچه که قرار بود رجوع می‌کند ، چند خطای منطقی که اشکال زدایی آنها دشوار است را خواهید داشت .

اطمینان پیدا نماید که گزارش خطا را فعال می‌کنید ، که در فصل بعدی مورد بحث قرار گرفته است ، و چند assertion را برای محافظت از این خطای رایج در هنگام توسعه کلاسهایتان اضافه کنید .

متد start() اتومبیل را با استفاده از key برای کاربر روشن خواهد کرد . اگر key صحیح باشد ، آبجکت اتومبیل به موتور خواهد گفت تا شروع به کار کند :

PHP Code:

```
// Constructor
function Car()
{
$this-> requiredkey();
$this->engine= new Engine() ;
}
function start ($Key)
{
if ($key->equals($this->requiredKey)) {
$this->engine->start();
return true ;
}
return false ;
}
```

متد stop() ساختاری مشابه متد start() دارد . این متد بررسی می‌کند تا ببیند آیا موتور روشن است یا خیر ، و اگر روشن باشد ، اتومبیل را متوقف خواهد کرد . توجه کنید که چک کردن موتور برای اطمینان از روشن بودن آن می‌توانست در تابع stop() آبجکت engine صورت گیرد ، تا ما راحتی از فکر کردن درباره آن باز دارد . از خودتان سؤال خواهید کرد که منطق(logic) در کجا بکار خواهد رفت . این ، پایه‌های توسعه معماری خوب و موفق است :

PHP Code:

```
function stop ()
{
if ($this->engine->isRunning()) {
$this->engine->stop() ;
}
}
// … Several other methods such as moving and turning , and so on .
}
?>
```

حال اجازه دهید ببینیم که چگونه می‌توانیم از این آبجکت در برنامه‌هایمان استفاده کنیم .

آبجکت‌ها (Objects):

یک شیء در برنامه‌ها نمونه‌ای از یک کلاس است . دلیل این که یک نمونه خوانده می‌شود این است که می‌توانیم چندین شیء را ایجاد کنیم (یا نمونه‌هایی) که از یک کلاس باشند . همان‌طور که اتومبیل‌های متعددی از یک کلاس می‌توانند در جاده‌ها باشند برای ایجاد دو اتومبیل جدید ، تمام آن چیزی که نیاز خواهیم داشت ، اجرای این خطوط کد در برنامه ما است :

PHP Code:

```
<? Php
$car1=new Car();
$car2=new Car();
```

از کلمه کلیدی new برای ساخت نمونه جدیدی از کلاس استفاده می‌کنیم ، که ایجاد یک شیء جدید است . هنگامی که شیء یا یک نمونه کلاس را ایجاد می‌کنیم ، می‌گوییم که شیء برای اولین بار نمونه‌سازی (instantiate) شده است . مرجع شیء تازه نمونه‌سازی شده به ترتیب درون متغیرهای $car1 و $car2 قرار می‌گیرد . حال دو شیء داریم که برای استفاده در دسترس هستند . اگر می‌خواستیم ده اتومبیل ایجاد کنیم ، از آرایه‌ای از اشیاء مانند این استفاده می‌کردیم :

PHP Code:

```
$cars = array() ;
for($I=0;$I<10;$I++) {
$cars[$I]=new Car ();
}
```

اگر بخواهیم یک اتومبیل را روشن کنیم ، متد start() آن را به صورت زیر فراخوانی می‌نماییم :

PHP Code:

```
$carHasStarted = $car1->start($myKey);
if ($carHasStarted) echo(“Car has started.”);
```

و اگر خواستیم اتومبیل را متوقف کنیم ، به صورت زیر عمل می‌نماییم :

PHP Code:

```
$car1->stop();
?>
```

متوجه شدید که این شیء دارای یک واسط ساده برای استفاده است . شما مجبور نیستید بدانید که واسط چگونه توسعه یافته است . به عنوان یک برنامه‌نویس ، تنها چیزی که باید بدانید ، سرویس‌هایی هستند که توسط یک شیء فراهم می‌گردند . این برنامه می‌توانست به خوبی یک اتومبیل فیزیکی را برای روشن شدن و متوقف شدن بسازد ، اما پیچیدگی این متدها و جزئیات اعضای آن به کلی ناشناخته هستند . این ایده ایجاد اشیاء قابل استفاده به آسانی ، ما را به بخش بعدی هدایت می :ند که Encapsulation (کپسوله سازی) نام دارد .

در این قسمت درمورد Factory Methods (متد های کارخانه) و Encapsulation (کپسوله سازی) صحبت خواهیم کرد:

متدهای کارخانه :

گاهی بهتر است از یک شیء بخواهید تا یک شیء جدید را برای شما ایجاد کند تا این که خودتان اپراتور جدیدی را صدا بزنید . این کلاسها ، کارخانه (factory) نام می‌گیرند و متدهایی که این اشیاء را ایجاد می‌کنند ، متدهای کارخانه نام دارند . کلمه کارخانه ، ریشه در استعاره‌ای از سهولت تولید دارد . برای مثال ، یک کارخانه موتورسازی که مالک آن جنرال موتور است و تولیدکننده موتورهای اتومبیل است ، بسیار شبیه به یک کارخانه شیء است که اشیایی از نوع خاص را تولید می‌کند . بدون تعمیق در جزئیات مدلهای آبجکتی پیچیده ، بیایید ببینیم چگونه می‌توانیم از کارخانه‌های شیء در برخی قسمتهای توسعه برنامه کاربردی وب استفاده کنیم . در این‌جا چند مثال داریم :

- ممکن است بخواهید یک Form Control Factory را ایجاد کنید که عناصر فرمی متنوعی را ایجاد می‌کند (نظیر فیلدهای متنی ، گروههای رادیویی ، دکمه‌های submit و غیره) تا روی یک فرم HTML نظیر آن چه که در کتابخانه eXtreme PHP پیاده‌سازی شده است (یک کتابخانه open source که در آدرس http://www.extremephp.org/ قراردارد) قرار دهید .

• ممکن است بخواهید یک کارخانه را برای واردن کردن سطرهای جدید به داخل جدول پایگاه داده و بازگرداندن آبجکت دسترسی داده مناسب برای آن سطر خاص ، ایجاد کنید .

حال باید ببینیم چگونه می‌توان یک کلاس کارخانه و متدهای مربوط به آن را با ایجاد TextField و آبجکت‌هایSubmitButton)از (PHP Extreme درون کلاس Form Control Factory ایجاد کرد .

در اینجا دو فایل را وارد می‌کنیم که در نظر می‌گیریم از پیش ساخته شده‌اند . فایل TextField.php حاوی کد مربوط به کلاس TextField و SubmitButton.php حاوی کد مربوط به کلاس SubmitButton است . چنانکه خواهید دید ، این دو فایل نیازمند یک نام و یک مقدار برای انتقال به Constructor های خود ، هنگام ایجاد نمونه‌های جدید هستند :

PHP Code:

```
<?php
include_once(“./Text Field.php”);
include_once(“./SubmitButton.php”);
```

هنگام توسعه کلاسهای کارخانه ، تمرین خوبی خواهد بود اگر کلمه “Factory” را به انتهای نام کلاس اضافه کنید کلمه “Factory”تبدیل به یک قرارداد مشترک در دنیای شیء گرایی شده است و به برنامه‌نویسان دیگر کمک خواهد کرد تا از این واژه شناسی مشترک تشخیص دهند که کلاس چه کاری را انجام می‌دهد :

PHP Code:

```
// FormControl Factory.php
class FormControlFactory
{
```

این اولین متد کارخانه ما ، یعنی createTextField() می‌باشد و به سادگی یک نمونه جدید از کلاس TextField را با انتقالname $ و $value تأمین شده توسط سرویس گیرنده ، ایجاد می‌کند :

PHP Code:

```
function createTextField($name,$value)
{
return new TextField($name,$value) ;
}
```

متد createSubmitButton() به همان روش تعریف شده است . همچنین یک قرارداد مشترک برای اتصال کلمه “create” به ابتدای متد کارخانه برای مشخص کردن شیء جدیدی است که بر می‌گرداند . این یک واژه سازی

مشترک را در برنامه کاربردی شما ایجاد خواهد کرد و فهم کد شما و سطح قابل ردیابی شدن (traceability) آن را افزایش خواهد داد :

PHP Code:

```
function createSubmitButton($name,$value)
{
return new SubmitButton ($name,$value) ;
}
}
```

حال بجای این که آبجکت‌های TextField و SubmitButton را با استفاده از اپراتور جدید برای اولین مرتبه معرفی کنیم ، می‌توانیم از Form Control Factory برای انجام این کار استفاده کنیم :

PHP Code:

```
$ Form Control Factory =new Form Control Factory();
$firstNameFild=
$ Form Control Factory -> createTextField('firstname', 'Ken');
$lastNameFild=
$ Form Control Factory -> createTextField('lastname', 'Egervai');
$ SubmitButton=
$ Form Control Factory -> create SubmitButton('submit', 'Submit Name');
?>
```

در اینجا نمونه جدیدی از Form Control Factory را ایجاد می‌کنیم و سه کلاس جدید را با استفاده از متدهای کارخانه آن ایجاد می‌کنیم . دو فراخوانی اول createTextField() ، فیلدهای متنی را ایجاد می‌کند که یک نام و نام‌خانوادگی را ذخیره می‌کنند . فراخوانی بعدی یک دکمه submit با عنوان "Submit Name" را ایجاد می‌کند در این نقطه ، برنامه کاربردی ما می‌تواند هرکاری که نیاز دارد با این آبجکت‌های جدید انجام دهد . اهمیت در معنی برنامه کاربردی نیست ، بلکه در ساختار و مفهوم آن چه که متدهای کارخانه هستند و چگونگی استفاده از آنها در برنامه‌های وب نهفته است .

کلاسهای کارخانه تنها محدود به ایجاد متدها نیستند . شما می‌توانید متدهای دیگری را اضافه کنید که با مدل کارخانه نزدیک می‌باشند نظیر متدهای find که به دنبال آبجکت‌های کارخانه می‌گردند و آنها را باز می‌گردانند و

متدهایی را حذف می‌کنند که می‌توانند آبجکت‌هایی را در کارخانه اوراق کنند. این پیاده‌سازیها به پروژه مربوط می‌شوند و برعهده شما به عنوان طراح برنامه کاربردی هستند. حال بیایید توجه خود را به اصول encapsulation و پنهان‌سازی اطلاعات معطوف کنیم .

Encapsulation (کپسوله‌سازی):

هنگامی که مسکن سردرد خود را مصرف می‌کنید ، احتمالاً از محتویات آن آگاه نیستید . تمام آنچه که مد نظر شماست ، توانایی آن در برطرف کردن سردرد شماست . این مسئله زمانی صادق است که برنامه‌نویسان از آبجکت‌های ارائه شده برای آنها استفاده می‌کنند . هنگامی که به استفاده از آبجکت Car خود پرداختیم ، چیزی درباره گیربکس ، سیستم اگزوز یا موتور وسیله نقلیه نمی‌دانستیم . تمام آن چه که لازم داشتیم ، پیچاندن کلید و روشن کردن اتومبیل بود . هنگام طراحی آبجکت‌ها هدف همین خواهد بود .

تمامی داده‌های مرکب و منطق را درون آبجکت جمع کنید و برای کاربران تنها سرویس‌های معنی‌داری را که آنها انتظار دارند با آبجکت به تعامل بپردازند را فراهم نمایید . در اینجا کپسوله نمودن داده مرکب و جزئیات منطقی درون آبجکت را می‌بینیم . اگر این کار به صورت مناسب انجام گیرد ، فایده مخفی کردن اطلاعات را خواهیم دید ، که بعداً مورد بررسی قرار خواهد گرفت .

چنانکه پیشتر ذکر کردیم ، برای کاربران کلاس اهمیت دارد تا از اعضای داده‌ای درون کلاس کاملاً بی‌اطلاع باشند .

اگرچه این مسئله در PHP کاملاً معتبر است ، که اعضای آبجکتی که برای اولین بار نمونه‌سازی شده است را در هر زمانی تغییر دهیم ، اما انجام این کار به عنوان یک عادت بد در نظر گرفته می‌شود .

در اینجا مثالی داریم که چند رویداد ناگوار را نشان می‌دهد که در صورتی که اعضای آبجکت را بدون رفتن به واسط آبجکت تغییر دهیم ، اتفاق خواهند افتاد . در این مثال در نظر می‌گیریم که متدی برای تنظیم سرعت اتومبیل وجود دارد که نام آن setSpeed است است ($speed) ، که در صورتی که آن را روی بیش از ۲۰۰km/h تنظیم کنید و یا وقتی سرعت کمتر از ۰km/h باشد با شکست مواجه خواهد شد . همچنینی باید تصور کنیم که constructor (سازنده) ما کلید و موتور را برای روشن کردن اتومبیل شناسایی نمی‌کند :

PHP Code:

```
myKey =new Key('Key of my Porsche');
$car = new Car();
$car->engine = new Engine();

$car->speed=400;
$car->start($myKey);

$car->engine =0;
$car->stop();
```

خطاهای زیادی در این کد وجود دارند که یا در تفسیر دچار مشکل خواهند شد و یا حتی بدتر ، کارخواهند کرد ، اما در رفتار مناسب با شکست مواجه خواهند شد . در سه خط اول ، در تنظیم عضو $requiredkey آبجکت $car خود شکست خوردیم ، چرا که این کار توسط constructorها انجام نشده بود .

کلید لازم نخواهد بود ، مگر این که بخواهیم واقعاً اتومبیل را روشن کنیم ، پس هیچ خطایی از این جا نتیجه نمی‌شود . پس همه چیز پس از چند خط اول خوب به نظر می‌رسد . به طور جداگانه نگاهی به خطی می‌اندازیم که آبجکت Engine را ساختیم . اگر به جای آن می‌نوشتیم :

$car->engine = new Engine(); چه می‌شد ؟

("E" بزرگ را در کلمه Engine مورد توجه قرار دهید.) اتومبیل از روشن شدن باز می‌ایستاد ، زیرا موتور نیز شناسایی نمی‌شد . حتماً خواهید توانست این اشکالات را به آسانی برطرف کنید ، اما آنها نباید در اولین وهله رخ دهند . سپس سعی می‌کنیم اتومبیل را روشن کنیم :

PHP Code:

```
$car->speed=400;//should have been $car ->setSpeed(400); to cause // a failure
$car->start($myKey);
```

هنگامی که اتومبیل روشن می‌شود به جلو رانده می‌شود و سرعتش به ۴۰۰ km/h خواهد رسید . این می‌تواند باعث تصادف و کشته شدن مردم در جاده (یا خارج از آن) شود . این یقیناً آن چیزی نخواهد بود که ما می‌خواهیم . اتومبیل از کجا می‌داند که به چه کلیدی برای روشن کردن اتومبیل نیاز دارد ؟ باید کلید ساخته شده را با متغیری که حتی وجود ندارد مقایسه کند (و نتیجه‌ای برابر ۰ در برداشته باشد) و در نهایت در روشن کردن اتومبیل با شکست

مواجه خواهد شد . مقایسه‌ای نظیر این مستقیماً به چک کردن مفسر مربوط می‌شود ، چرا که $key ورودی است که برابری را چک می‌کند و نه عضو را . برای مالک اتومبیل خجالت‌آور خواهد بود اگر که یک وسیله نقلیه جدید بخرد و تازه بفهمد که کلید ارائه شده توسط فروشنده هرگز کار نکرده است . بیایید ببینیم که چه اتفاقی خواهد افتاد اگر بخواهیم هنگامی که engine را روی ۰ تنظیم کرده‌ایم ، اتومبیل را خاموش کنیم :

PHP Code:

```
$car->engine=0
$car->stop();
```

هنگامی که متد ()stop فراخوانده می‌شود . با یک خطای زمان اجرا مواجه خواهیم شد ، چرا که آبجکت Engine با اینکه حتی وجود ندارد ، آن را وادار به پذیرفتن مقدار integer کرده‌ایم . چنانکه می‌بینید ، تنظیم اعضا از خارج کلاس احتمالاً می‌توانست مشکلات فراوانی را به بار بیاورد . در دنیایی که چندین برنامه‌نویس را در حال کارکردن بر روی یک پروژه دارید ، باید انتظار داشته باشید که بقیه بتوانند کد شما را بخوانند و احتمالاً از آن استفاده کنند .

چه درسهایی از این مثال آموختیم ؟ استفاده از اعضا در خارج از آبجکت (object violations) می‌تواند :

- عدم اطمینان از اینکه سرویس‌هایی که به وسیله آبجکت فراهم شده‌اند ، به گونه‌ای که از آنها انتظار داریم رفتار نمایند .
- عدم جامعیت (integrity) اعضای داده آبجکت (یا وضعیت آبجکت) در یکی از دو روش زیر :
- تخطی از تعیین وضعیت ابتدایی اعضاء
- ایجاد واسطهایی پیچیده‌تری نسبت به آن چه که شما واقعاً نیاز دارید .
- مسئولیت سنگین‌تری روی دوش برنامه‌نویسان بگذارد تا بیشت درباره آبجکت و روشی که داده با سرویس‌ها کار می‌کند ، فکر کنند .

• هنگامی که زمان استفاده مجدد از آبجکت فرا می‌رسد ، ممکن است مجبور باشید اعضا را مجدداً تغییر دهید . گاهی به غیر از فراموشکاری ، خطاهای جدیدی در پروژه بعدی ایجاد خواهید کرد . این دقیقاً چیزی است که می‌خواهیم از آن دوری گزینیم .

طراحی کلاس بگونه‌ای که برای انجام هر آن چه که می‌خواهید با آبجکت انجام دهید مناسب باشد ، راه عملی خوبی خواهد بود . هرگز به اعضای خارج از کلاس دستیابی پیدا نکنید و همیشه کلاس‌های خود را به طور مناسب در کپسول قرار دهید ، تا فوائد مخفی کردن اطلاعات را حاصل کنید . برخی زبانها قابلیت غیر مجاز ساختن دسترسی به اعضا را به طور کلی و بوسیله خصوصی ساختن (یا محافظت کردن) آنها تنها برای کلاس ارائه می‌دهند . در حال حاضر PHP از این امکان بهره‌مند نیست ، اما پیروی از عادتهای خوب کدنویسی ، بدون شک مفید خواهد بود .

فرم ها در PHP

کار کردن با فرم ها:

خوب در این قسمت میخواهیم ببینیم چطوری با استفاده از فرمها اطلاعات رو بین صفحات ارسال کنیم.

خوب یه مثال ساده ولی پرکاربرد :

PHP Code:

```
#this page name is a.php
<form action="b.php" method="get">
<input type="text" name="user">
<input type="submit" name="btn">
</form>
```

قصد داریم اطلاعات یک تکست باکس رو که در این صفحه قرار داره توسط صفحه دوم به اسم b.php بخونیم پس اسم این صفحه رو که هیچ کد پی اچ پی توش استفاده نکردیم میزاریم a.php و ذخیره میکنیم. حالا میرسیم به صفحه دوم این کد رو تو صفحه دوم قرار بدید .

```
php?>
"print "$user
<?
```

خیلی ساده و به این صورت اطلاعات رو میگیریم یوزر اسم تکست باکسی هست که تو فرم اول قرار داشت. بقیه کد ها هم اچ تی ام هست و نیازی به توضیح نیست فقط در مورد get بگم که این دستور متغییر ها رو موقع ارسال تو ادرس نشون داده میشه و شاید بعضی جاها زیاد راه مطمئنی نباشه ولی اگه بجای این عبارت از post استفاده کنیم دیگه در ادرس نشون داده نمیشه .

ترکیب کردن PHP با HTML :

به این کد نگاه کنید :

PHP Code:

```
<form action="<?php print $PHP_SELF?>" method="POST">

name: <input type="text" name="user">

</form>
```

وقتی این فرم رو اجرا کنیم فرم همواره خودشو صدا میزنه چون ما از عبارت $PHP_SELF استفاده کرده ایم

توجه کنید که ما هیچ دکمه ای رو صفحه قرار ندادیم و در بیشتر مرورگرها با زدن کلید اینتر میتونیم فرم رو اجرا کنیم .

البته در کد بالا هیچ اطلاعاتی رو بیرون نمیده ولی در کد زیر میخواهیم یه بازی ساده بنویسیم تا بیشتر متوجه بشیم :

PHP Code:

```
<?php
$num_guess=42;
$msg=" ";
if (! Isset($guess))
{
$msg="welcome To This Little Game";
}
elseif ($guess>$num_guess)
{
$msg="number $guess is  Big! Try Smaller number.";
}
elseif($guess<$num_$guess)
{
$msg="Number $guess is Small ! Try Big Number";
}
else // vagti mosavi bashe
{
$msg="Well Done You Win";
}
?>
```

توضیحات : با دستور ! که اشنا هستید همون دستور نقض ، اما دستور isset این دستور چک میکنه ببینه اصلا اطلاعاتی وارد شده یا نه ما در خط اول چک میکنیم ببینم (اگه نه اطلاعاتی وارد شده) انگاه پیغام خوش آمد گویی رو پخش کن این کد موقعی میتونه اتفاق بیفته که برا اولین بار صفحه رو باز کرده یا هیچ اطلاعاتی وارد نکرده . حالا میرسیم به قسمت اچ تی ام ال این کد رو زیر کد بالایی قرار بدید

PHP Code:

```
<h1>
<?php print $msg ?>
</h1>
<form action="<?php print $PHP_SELF?>" method="POST">
Guess Number : <input type="text" name="guess">
</form>
```

خوب این کد نیز اطلاعات رو از کاربر میگیره و صفحه رو دوباره اجرا میکنه و پیغام مناسب رو پخش میکنه .

میخوایم با یه دستور جدید و کاربردی اشنا بشیم فرض کنیم بعد از اینکه کاربر در بازی بالا برنده شد میخواهیم او را به یه صفحه جدید بفرستیم یعنی ریدایرکت کنیم میتونیم از دستور زیر استفاده کنیم این دستور رو بجای پیغامی بزارید که وقتی کاربر برنده میشه انتخاب میشه یعنی بعد از else قرار بدید :

PHP Code:

```
header("Location: page.html");
exit;
```

با دستور exit هم از داخل این کد بیرون میایم و به صفحه بعد منتقل میشیم .

نوبتی هم باشه نوبت کار با فایل هاست . (البته کار با فایلها در آینده و در فصلی جداگانه بررسی شده است)

PHP Code:

```
<form enctype="multipart/form-data" action="<?print $PHP_SELF?>" method="POST">
<input type="hidden" name="MAX_FILE_SIZE" value="51200">
<input type="file" name="fupload"><br>
<input type="submit" value="upload!">
</form>
```

خوب ما با استفاده از دستور "enctype="multipart/form-data درواقع میایم همون فایل دیالوگ رو باز میکنیم.

در قسمت بعد ما یک متغییر مخفی ایجاد میکنیم و سایز فایل رو تعیین میکنیم و توش قرار میدیم که در اینجا ۵۱۲۰۰ انتخاب کردیم

حالا با دو خط بعد یه تکست باکس و یه دکمه میزاریم . ما اینجا به کاربر اجازه میدیم تا ۵۰ کیلوبایت فایل اپلود کنه .

البته کد بالا جنبه نمایش داشت و عملا کاربرد زیادی نداره قبل از اینکه نمونه قابل اجرا رو امتحان کنیم بزارید لیستی از متغییر های فایل اپلود رو براتون بگم.

مثال توضیح نام متغییر

fupload$ مسیر فایل اپلود شده tmp/php3d3ef/

fuploadname$ اسم فایل اپلود شده Test.gif

fuploadsize$ حجم بر حسب بایت ۵۱۲۰۰

fupload type$ نوع فایل اپلودی Image/gif

حالا مثال ، در این مثال ما اطلاعات فایل اپلئدی رو نمایش میدیم

PHP Code:

```
1: <html>
2: <head>
3: <title>Listing 9.15 A file upload script</title>
4: </head>
5: <?php
6: $file_dir = "/home/mehdi/htdocs/uploads";
7: $file_url ="http:// www.safary.com/mehdi/uploads";
8: if ( isset( $fupload ) )
9: {
10: print "path: $fupload<br>\n";
11: print "name: $fupload_name<br>\n";
12: print "size: $fupload_size bytes<br>\n";
13: print "type: $fupload_type<p>\n\n";
14: if ( $fupload_type == "image/gif" )
```

```
15: {
16: copy ( $fupload, "$file_dir/$fupload_name") or die ("Couldn't copy");
17:
18: print "<img src=\"$file_url/$fupload_name\"><p>\n\n";
19: }
20: }
21: ?>
22: <body>
23: <form enctype="multipart/form-data" action="<?php print $PHP_SELF?>"
method="POST">
24: <input type="hidden" name="MAX_FILE_SIZE" value="51200">
25: <input type="file" name="fupload"><br>
x<input type="submit" value="Send file!">
27: </form>
28: </body>
29: </html>
```

فکر نمیکنم نیازی به توضیح باشه بجز این خط $fupload, "$file_dir/$fupload_name" این دستور فایل مورد نظر رو روی سرور کپی میکنه و اگه اشکالی پیش اومد عبارت copy Couldn't رو چاپ میکنه در اینجا ما پس از اینکه از کاربر فایل رو درخواست کردیم اطلاعات مربوط به اون فایل رو نمایش میدیم که ما در اینجا فرض کردیم که کاربر یک عکس را اپلود کرده است .

دیتابیس های فایلی

کار با دیتابیس ها (فایل):

اگه شما به دیتابیس هاس اس کیو ال مثل مای اس کیو ال و اوراکل دسترسی ندارید میتونید از دیتابیس های DBM استفاده کنید حتی اگه شما اگه نداشته باشینش پی اچ پی اون رو برای شما شبیه سازی میکنه و به شما اجازه میده داده هاتون رو توش ذخیره کنید و بازیابی کنید شون .

در ضمن شما نیاز به استفاده از دستورات اس کی ال ندارید چون بسیار راحت و انعطاف پذیر هستن .

در این بخش یاد میگیریم چطور :

- یک دیتابیس DBM رو باز کنیم
- چگونه داده را وارد کنیم
- چگونه داده را فراخوانی کنیم
- چگونه تغییر ایجاد کنیم یا داده هارا پاک کنیم
- چگونه داده های پیچیده ای را وارد دیتا بیس کنیم

باز کردن دیتا بیس :

با استفاده از دستور dbmopen() شما میتونید دیتا بیس را باز کنید و این تا بع دو آرگومان میگیره یکی مسیر دیتابیسمون و دیگری نوع دسترسی به دیتا بیس که نوع دسترسی ها به این شکل هست:

توضیح نوع دسترسی

- در حالت فقط خواندی r
- برای خواندن و نوشتن w
- ساختن دیتابیس و اگر وجود دارد برای خواندن و نوشتن c
- ساخت یک دیتابیس جدید n

با استفاده از دستور زیر اگر دیتابیسی با این نام وجود نداشته باشد باز میشود :

PHP Code:

```
$db = dbmopen( "./products", "c" ) or die( "Couldn't open DBM" );
```

نکته : دقت کنید که ما از دستور die برای زمانی استفاده کردیم که قادر به ایجاد دیتابیس نباشیم .

وقتی کارمون با دیتا بیس تموم شد با استفاده از دستور زیر دیتابیس رو میبندیم .

PHP Code:

```
dbmclose($db);
```

اگه شما دیتابیس رو نبندید قسمت های دیگری از برنامه‌مون که به دیتابیس نیاز دارند مجبور هستن تا صبر کنند و این اصلا خوب نیست .

اضافه کردن اطلاعات به دیتابیس :

به مثال زیر توجه کنید :

PHP Code:

```
<?php
$db=dbmopen("./users","c") or die("Could not open DBM");
dbminsert($db,"PersianTools.com","30000");
dbminsert($db,"IranVig.com","6000");
dbminsert($db,"CodeVig.com","100");
dbmclose($db);
?>
```

و اینگونه اطلاعات رو اضافه میکنیم و حالا برای تغییر اطلاعات به این شکل عمل میکنیم :

PHP Code:

```
<?php
$db=dbmopen("./users","c") or die("Could not open DBM");
dbmreplace($db,"PersianTools.com","464420");
dbmreplace ($db,"IranVig.com","62351");
dbmreplace ($db,"CodeVig.com","6463");
dbmclose($db);
?>
```

حالا که ما تونستیم اطلاعاتمون رو وارد دیتابیس کنیم احتیاج داریم تا اونهارو بتونیم بخونیم پس به این صورت عمل میکنیم :

PHP Code:

```
$users=dbmfetch($db,"PersianTools.com");
```

خیلی ساده بزارید این اطلاعات رو در قالب یک جدول فراخوانی کنیم :

PHP Code:

```
<table border=1 cellpadding ="5">
<tr>
<td align="center"> <b>product</b></td>
<td align="center"> <b>price</b> </td>
</tr>
<?php
$db = dbmopen( "./users", "c" ) or die( "Couldn't open DBM" );
$key = dbmfirstkey( $db );
while ( $key != "" )
{
$value = dbmfetch( $db, $key );
print "<tr><td align = \"left\"> $key </td>";
print "<td align = \"right\"> \$$value </td></tr>";
$key = dbmnextkey( $db, $key );
}
dbmclose( $db);
?>
</table>
```

و به این صورت تمام اطلاعات رو با استفاده از یک حلقه فراخوانی میکنیم

حالا شاید سوال پیش بیاد که چجوری بفهمیم دادمون قبلا تو دیتا بیس وارد شده یا نه ؟؟

PHP Code:

```
if ( dbmexists( $db, "PersianTools.com" ) )
print dbmfetch( $db, " PersianTools.com " );
```

برای پاک کردن یک مقدار از دستور

PHP Code:

```
dbmdelete($db,"PersianTools.com");
```

استفاده میکنیم .

اضافه کردن یک مقدار پیچیده به دیتابیس :

PHP Code:

```
$array = array( 1, 2, 3, 4 );
$db = dbmopen( "./ test", "c" ) or die("Couldn't open test DBM");
dbminsert( $db, "arraytest", $array );
print gettype( dbmfetch( $db, "arraytest" ) );
```

خوب باشه یه مثال کامل :

PHP Code:

```
<?php
$products = array("PersianTools.com" => array( users=>"2200",stats=>"1500", color=>"blue" ),
 "Iranvig.com" => array( users=>"2213",stats=>"1234",color=>"blue" )
);
$db = dbmopen( "./test", "c" ) or die("Couldn't open products DBM");
while ( list ( $key, $value ) = each ( $products ) )
dbmreplace( $db, $key, serialize( $value ) );
dbmclose( $db );
?>
</table>
```

بزارید یک مثال کلی بزنیم تا بیشتر آشنا بشیم ما در این مثال اطلاعات دیتابیسمون رو داخل یه جدول میکشیم و کنار هر داده یه گزینه میزاریم تا با انتخاب اون بتونه پاک کنه داده رو و همینطور یک تکست برای ویرایش میزاریم و این بخش رو تموم میکنیم .

PHP Code:

```
<?php
 $dbh = dbmopen( "./users", "c" ) or die("Couldn't open test DBM");

 if ( isset ( $delete ) )
 {
 while ( list ( $key, $val ) = each ( $delete ) )
 {
 unset( $prices[$val]);
 dbmdelete( $dbh, $val );
 }
```

```
}

if ( isset ( $prices ) )
{
while ( list ( $key, $val ) = each ( $stats ) )
dbmreplace( $dbh, $key, $val );
}

if ( ! empty( $name_add ) && ! empty( $stats_add ) )
dbminsert( $dbh, "$name_add", "$stats_add" );
?>

<html>
<body>
<form action="<? print $PHP_SELF; ?>" action="POST">
<table border="1">
<tr>
<td>delete</td>
<td>Users</td>
<td>stats</td>
</tr>
<?php
$key = dbmfirstkey( $dbh );
while ( $key != "" )
{
$price = dbmfetch( $dbh, $key );
print "<tr><td><input type='checkbox' name=\"delete[]\" ";
print "value=\"$key\">$/td>";
print "<td>$key</td>";
print "<td> <input type=\"text\" name=\"prices[$key]\" ";
print "value=\"$stat\"> </td></tr>";
$key = dbmnextkey( $dbh, $key );
}

dbmclose( $dbh );
?>
```

```
<tr>
<td> </td>
<td><input type="text" name="name_add"></td>
<td><input type="text" name="stats_add"></td>
</tr>

<tr>
<td colspan=3 align="right">
<input type="submit" value="amend">
</td>
</tr>
</table>
</form>
```

فکر میکنم این سوال براتون پیش بیاد که چه موقع باید از این نوع ذخیره داده استفاده کنیم : خوب باید بگم زمانی که داده های کمی برا ذخیره کردن دارید میتونید از این روش استفاده کنید که خیلی روش ساده ای هست و بدرد بخور .

نمایش چند صفحه ای اطلاعات

جستجو در پایگاه داده ها:

در این بخش از MySQL استفاده شده اما این روش را می توان با هر DataBase بکار برد .

جزء LIMIT :

جزء LIMIT در عبارت Select کلید حل مشکل ماست ، به وسیله این جزء است که ما می توانیم سطرهای مورد نظر خود از DataBase را در Query خود داشته باشیم .

LIMIT به دو شکل به کار می رود با یک آرگومان و یا با دو آرگومان ،این آرگومانها عدد هستند ، اگر LIMIT با یک آرگومان استفاده شود تعداد جواب به آن عدد محدود می شود :

```
SELECT * FROM table LIMIT 5
```

عبارت Select در این حالت ۵ سطر اول Table را در جواب بر می گرداند ،اما اگر LIMIT با دو آرگومان استفاده شود آرگومان اول مشخص کننده سطر آغازین است و آرگومان دوم تعداد سطرهایی را که باید برگردانده شوند را مشخص می کند ، البته باید به این نکته توجه کرد که در SQL سطرها از صفر (۰) شروع می شوند نه از یک (۱) .

```
SELECT * FROM table LIMIT 5, 5
```

این عبارت SQL سطرهای ۶ تا ۱۰ از DataBase را برمی گرداند .

کار با جزء LIMIT مشکل نیست ولی برای اینکه به هدف مطرح شده در این مقاله نیل کنیم لازم است که چگونگی عملکرد آن را درک کنید .

گام اول متغیر $start:

در اینجا می دانیم که از چه چیزی باید برای بدست آوردن سطرهای مورد نظر خود از DataBase استفاده استفاده کنیم . اما سوال این است که چگونه متوجه می شویم که از کجا باید LIMIT کنیم یا به عبارت دیگر از کدام سطر باید شروع کنیم ؟ پاسخ خیلی روشن است ما از یک متغیر به نام $start استفده می کنیم و این متغیر را از صفحه ای به صفحه دیگر می فرستیم .

حال که روش کار مشخص شده شروع به نوشتن کد در PHP می کنیم :

```
<?php
  $query = "SELECT * FROM table LIMIT \" . $start . \", 10\";
 ?>
```

این شکل query ماست و این 10 query سطر ازtable را از جایی که مساوی مقدار start باشد را select می کند . اما برای مشخص کردن مقدار start دو راه وجود دارد(در هنگامی که اولین ۱۰ سطر را بازیافت می کنیم) یا آن را در URL می نویسیم و یا با شرطی چک می کنیم که اگر قبلا مقداری به آن داده نشده ما مقدار آن را صفر می کنیم لازم است برای این کار از کدی شبیه به کد زیر استفاده می کنیم :

```
<?php
  if(!isset($start)) $start = 0;
  ?>
```

حالا وقتی کسی صفحه ما را مشاهده کند مقدار $start به صورت خودکار صفر خواهد شد .

UPLOAD کردن MySQL:

ابتدا توسط mysqldump اطلاعات مربوطه را در یک فایل txt ذخیره سازی میکنیم

```
Shell > mysqldump --opt DBname TBname > (c:\data.txt or /home/user/data.txt)
```

نکته:

۱ – برای ذخیره سازی تمامی table های یک DB باید اسم database مورد نظر را بدون اسم table ها ذکر کنید

۲ - برای ذخیره سازی چندین database باید اسم DB ها را پشت سر هم ذکر کنید

Shell > mysqldump --opt --databases DB1 DB2 > (c:\data.txt or /home/user/data.txt)

۳ - شما می‌توانید تمامیdatabase ها را بصورت زیر ذخیره سازی کنید

Shell > mysqldump --opt --all-databases > (c:\data.txt or /home/user/data.txt)

حال اطلاعات مربوط به table مورد نظر را در یک فایل txt ذخیره سازی کرده ایم انرا بر روی host مورد نظر FTP میکنیم.

سپس با یک دستور ساده اطلاعات را وارد MySQL میکنیم

Shell > mysql database < data.sql

نکته :

برای استفاده از username & password جهت استفاده از MySQL باید بصورت زیر عمل کنید

Shell > Mysql –h host -u user -p database < backup-file.sql

نکته :

چنانچه شما برای انجام این کار بر روی host دسترسی به Shell ندارید می‌توانید از این راه استفاده کنید.از یک PHP استفاده می‌کنید بصورت زیر :

```
<?php
shell_exec(“Mysql –h host -u user -p database < backup-file.sql”);
?>
```

SQLite پیشنهاد سبک وزن:

این درسته که MYSQL و بقیه RDBMS ها برای برنامه های بزرگ و چند کاربره مفید هستند ولی خوب اگر شما می خواین یک سایت کم ترافیک و کوچیک درست کنین، شاید SQLite بهترین راه حل باشه. چون سرعتش نسبت به MYSQL 2تا ۳ برابر بیشتره که خوب مزیت بزرگیه.

کار با SQLite در PHP و بوسیله دستور PEAR آسان است. در نصب PEAR بطور پیش فرض SQLite وجود نداره ولی خیلی راحت Pear می تونه پکیج SQLite را download کنه سپس کامپایل کنه و در نهایت نصبش کنه.

```
[root@zaemis www]# pear download SQLite
File SQLite-1.0.2.tgz downloaded (362412 bytes)
[root@zaemis www]# pear install SQLite-1.0.2.tgz
```

الان SQLite در شاخه DB از PEAR قابل دسترسی است.

```
<?php
require_once "DB.php\";
require_once \"DB/SQLite.php\";
$db = new DB_sqlite();
?>
```

برای کاربران windows یک فایل dll کامپایل شده وجود دارد که شما می تونید اونو از این آدرس download کنید.

SQLite binary for Windows: snaps.php.net/win32/PECL_STABLE/php_sqlite.dll

حالا باید به فایل php.ini اینها رو اضافه کنیم (البته اگر شما از php version 5 استفاده می کنید نیازی به این تغییرات نیست).

```
; load the SQLite extension (UNIX)
extension=sqlite.so
; windows users will want to use this version instead
; extension=php_sqlite.dll
```

اتصال به دیتابیس:

اتصال به SQLite با MySQL یه مقدار فرق داره. چون MySQL برای محیط های چند کاربره طراحی شده در حالیکه SQLite برای برنامه های کاربردی تک کاربره طراحی شده که نیازی به ID و password هم برای وصل شدن نداره.

در عوض باید از یک آرایه برای ایجاد یک DSN (Data Source Name) استفاده کرد. DSN اشاره گری است برای مشخص کردن محل قرارگیری فایلهای دیتابیس. بعد این DSN رو به متد connect() پاس می کنیم.

```
<?php
$DSN = array(
"database\" => getcwd() . \"/dbase/mydbase.db\",
\"mode\"=> 0644
);
$db->connect($DSN);
?>
```

اگر شما به دیتابیسی متصل شوید که وجود خارجی نداشته باشد یک دیتابیس جدید با permission های ذکر شده جاوی mode ایجاد می شود.

متد disconnect() برای قطع کردن اتصال استفاده می شود.

```
<?php
$db->disconnect();
```

```
?>
```

پرس و جوها:

متد query() برای این منظور استفاده می شود. مثل MySQL پرس و جوها می تونن برای ایجاد جداول و وارد کردن دیتا استفاده شوند.

```
<?php
$query =
"INSERT INTO guestbook (fname, lname, email, comments)
VALUES ('John', 'Smith', 'jsmith@example.org', 'Great Website!')\"
;
$result = $db->query();
?>
```

متد fetchRow() می تواند برای نشان دادن نتیجه پرس و جو استفاده شود که معادل متد mysql_fetch_row هست.

```
<?php
$query = "SELECT * FROM guestbook\";
$result = $db->query($query);
while (list($fname, $lname, $email, $comment) = $db->fetchRow(
$result))
{
echo \"<p><a href=\\"mailto:$email"
>$fname $lname</a> said:<br />\";
echo nl2br(htmlspecialchars(stripslashes($comment))) .
\"</p>\";
}
```

```
?>
```

نوشتن کد:

در اینجا یک نمونه اسکریپت که با استفاده از SQLite نوشته شده داریم:

```
<?php
require_once "DB.php\";
require_once \"DB/SQLite.php\";
$db = new DB_sqlite();
$DSN = array(\"database\" => getcwd() . \"/dbase/mydbase.db\");
$db->connect($DSN);
$query = \"SELECT * FROM guestbook\";
$result = $db->query($query);
while (list($fname, $lname, $email, $comment) = $db->fetchRow(
$result))
{
echo \"<p><a href=\\"mailto:$email"
>$fname $lname</a> said:<br />\";
echo nl2br(htmlspecialchars(stripslashes($comment))) .
\"</p>\";
}
$db->disconnect();
?>
```

دستوراتی از SQL که پشتیبانی نمی شوند

هر برنامه دیتابیسی تفسیر خودش رو از SQL92 استاندارد داره، SQLite هم مستثنا نیست. بعضی از اینها ناشی از مدل طراحی embedded این دیتابیس دارند (مثل فقدان REVOKE و GRANT) بعضی هاشون هم جدا لج آورند (مثل نادیده گرفتن data type ها)

در این آدرس می تونید لیستی از دستورات SQL رو که توسط SQLite پشتیبانی می شوند را ببینید:

(http://www.sqlite.org/lang.html) http://www.sqlite.org/lang.html

اینجا هم لیست دستوراتی است که فعلا توسط SQLite پشتیبانی نمی شوند.

http://www.sqlite.org/omitted.html) http://www.sqlite.org/omitted.html
http://www.sqlite.org/cvstrac/wiki?p=UnsupportedSql
http://www.sqlite.org/cvstrac/wiki?p=UnsupportedSql

SQLite: A Lightweight Alternative
by Timothy Boronczyk
The Swiss army knife of data storage
Wez Furlong

provided by Shilan

شروعی ساده با MySQL پایگاه داده:

مقدمه:

در این بخش سعی شده است که اصول و مبانی کار با MySQL به صورتی ساده و قابل فهم برای کسانی که تازه شروع به کار با پایگاه‌های داده را کرده‌اند، گفته شود.

PHP به صورت از پیش تعریف شده، اکثر پایگاه‌های داده را پشتیبانی می‌کند: dBase, FilePro, mSQL, MySQL,Oracle, PostgreSQL,Sybase.اگر شما بخواهید از پایگاه های داده‌ای که به صورت از پیش تعریف شده پشتیبانی نمی شوند، استفاده کنید (مانند Access و یا SQL Server) باید از توابع PHP ODBC (Open DataBase Connectivity) به همراه راه‌اندازهای پایگاه داده (ODBC Drivers) استفاده کنید.

MySQL یکی از انواع سرویس دهنده‌های پایگاه داده است که با وجود اینکه نسبت به سایرDBMS های موجود زیاد قدرتمند نمیباشد، اما به علت قیمت و عملکرد مناسب آن، مورد استفاده گسترده قرار می‌گیرد.

طریقه اتصال به MySQL:

قبل از شروع هر کاری باید به MySQL متصل شویم. برای این کار از دستور

```
<?
 $Conn=mysql_connect($Location,$Username,$Password);
?>
```

استفاده می‌کنیم. تابع mysql_connect دارای سه آرگومان می‌باشد. آرگومان اول ، میزبان (Host) را مشخص می‌کند. دو آرگومان بعدی، نام کاربری و کد رمز را برای اتصال به پایگاه داده، ارسال می‌کنند. در صورتی که اتصال با شکست مواجه شود، پیغام خطایی درbrowser نشان داده می‌شود.

با استفاده از دستور

```
<? mysql_select_db($database)?>
```

بانک اطلاعاتی مورد نظر انتخاب خواهد شد و از این پس کارهای شما همه بر روی این بانک انجام خواهد گرفت. در صورتی که بانک مورد نظر پیدا نشد، پیغام خطایی نشان داده می‌شود.

ایجاد یک بانک اطلاعاتی در MySQL :

```
<? if (mysql_creat_db($database,$Conn)
{ print(“The database,$database,was { successfully created!<BR>n”);
else
{ print(" The database,$database, could not be created! <BR>\n”);}
?>
```

پس از اتصال به MySQL شما می‌توانید با استفاده از دستور

```
<?
mysql_creat_db($database,$Conn);
?>
```

یک بانک اطلاعاتی جدید ایجاد کنید. این دستور شامل دو آرگومان : نام بانک اطلاعاتی و نام لینک مورد نظربه پایگاه داده می‌باشد. در صورتی که عمل ایجاد با موفقیت انجام شده باشد، پیغامی مبنی بر این موضوع نمایش داده می‌شود و در غیر اینصورت، به کاربر اعلام می‌شود که عملیات نتیجه نداشته است.

انجام کارهای مختلف بر روی بانک اطلاعاتی در MySQL :

برای انجام کارهای مختلف بر روی بانک اطلاعاتی ازدستورات SQL استفاده می‌کنیم. روند استخراج اطلاعات از یک پایگاه داده به این صورت است که ابتدا یک query به زبان SQL نوشته می‌شود وسپس این query با استفاده ازدستور:

```
mysql_db_query($database,$Query);
```

عملیات لازم را بر روی بانک اطلاعاتی انتخاب شده، انجام می‌دهد.فرم کلی به صورت زیر است:

```
<?$Query=”text for the query goes here in SQL format";
if (mysql_db_query($database,$Query))
}
print( “The query was successfully executed!<BR>\n”);
{
else
}
print(“The query could not be executed!<BR>\n”);
{ ?>
```

ایجاد یک جدول :

برای ایجاد یک جدول ، این query را می نویسیم:

```
<?
 $Query=”CREATE table $Tablename(id INT PRIMARY KEY ,
 Firstname char(12), Lastname char(15))”;
 ?>
```

با استفاده ازاین query، جدول جدیدی ایجاد می‌شود که شامل یک فیلد به نام id (که key Primaryاست) و فیلدی به نام Firstname از نوع کاراکتری با طول ۱۲ و همچنین فیلدی به نام Lastname با طول ۱۵ می باشد.

ارسال داده‌ها به بانک اطلاعاتی:

برای ارسال داده‌ها به بانک اطلاعاتی این q uery را می نویسیم:

```
<?$Query=“INSERT into $Tablename(‘value1’, ‘value2’, ‘value3’,…)”;?>
```

توجه داشته باشید که تعداد مقادیری که در این q uery مشخص می‌شوند، باید با تعداد ستونهای جدول و همچنین نوع آن برابر باشد. در غیراینصورت query کارنمی‌کند.

بازیابی اطلاعات و نمایش آن:

به قطعه کد زیر توجه کنید:

```
<?
 $query = "SELECT * FROM $Tablename“;
 $result = mysql_query($query);
 $numrows = mysql_num_rows($result);
 while($row = mysql_fetch_array($result))
```

```
{
echo \"You have $numrows user(s) in the database”;
echo \"ID number : $row[id]”;
echo \"firstname: $row[firstname]”;
echo \"Lastname: $row[lastname]”;
}
?>
```

در خط اول، queryای نوشته شده است که تمام رکوردهای موجود درTable را انتخاب می‌کند و سپس در خط دوم، این query برروی بانک اطلاعاتی اجرا می‌شود و در خط سوم تعداد رکوردهای موجود درTable ، در متغییر numrows ذخیره می‌گردد و سپس با استفاده از حلقهٔ while (که تعداد loop آن به اندازه رکوردهای Table است) رکوردها، خوانده شده و نمایش داده می‌شود.

تغییر اطلاعات موجود در بانک اطلاعاتی:

برای این کار کافی است که از queryای استفاده کنیم که در آن دستورUPDATE به کار برده شده باشد. بطور مثال:

```
<?$Query=“UPDATE $Tablename SET firstname=’mina’ WHERE
 firstname=’maryam’”; ?>
```

این query ، اسامی موجود درTable مورد نظر را در صورتی که برابر با mina باشد، به maryam تغییر می‌دهد.

نکته: بعد از اتمام کار، به وسیله دستور

```
<?mysql_close($Conn);?>
```

اتصال بهMySQL را قطع کنید. این دستور شامل یک آرگومان است، که همان متغیراشاره گر،به MySQL می‌باشد.

بیشترین اشتباهات و خطاهای به وجود آمده، مربوط به نحوه نوشتن queryها می باشد. چنانچه در دستورات PHP ، هیچ نوع اشکال منطقی‌ای مشاهده نکردید، q uery مورد نظر را بررسی کنید. به احتمال زیاد مشکل حل خواهد شد.

شما می توانید به جای دستور <?mysql_db_query( );> از دستور <?mysql_query( );?> استفاده کنید.

لبته هر کدام از این دستورات، کاربردهای خاص خود را دارند که توصیه می‌شود برای آگاهی،h elp مربوطه را در PHP ، مطالعه بفرمایید.

توجه داشته باشید که باز و بسته شدن‌های متعدد Database موجب به وجود آمدن حفره‌هایی می‌شود که امنیت اطلاعات را به خطر می‌اندازد (برای آگاهی بیشتر، می‌توانید از منابعی در مورد Security، استفاده کنید.) لذا توصیه می‌شود با دسته‌بندی مناسب عملیات مورد نیاز روی D atabase،تعداد دفعات باز و بسته شدن D atabase را به حداقل برسانید.

یک نکته ساده و در عین حال جالب(مخصوص مبتدی‌ها):

یک Database و یا یک Table را نمی‌شود بیش از یک باربه وجود آورد.!! بنابراین در صورتی که هنگام به وجود آوردن Database و یا Tableای ، مدام با Error برخورد می‌کنید و هیچ نوع توجیه‌ای برای آن نمی‌یابید، پیشنهاد می‌شود که با بررسی Mysql_Front از وجود نداشتن Database و یا Tableمورد نظر مطمئن شوید.

(شیلان جوانمردی Sh_javanmardi@srtnet.net)

کار با فایلها در PHP

شما میتونید یک کد رو در یک فایل دیگر بنویسد و هرجا لازم شد در یک کد دیگر که در فایل دیگری هم قرار داره اونو صدا بزنید تابع ;Include() این امکان رو به شما میده و باعث سهولت و کم حجم شدن صفحات میشه . این تابع فقط به یک آرگومان نیاز داره و اون مسیر فایل پی اچ پی هست که میخوایم به صفحمون پیوند بدیم. اجازه دهید با یک مثال راحتتر بیان کنم : فرض کنید ما در فایل a.php یک کد داریم که یک پیغام رو چاپ میکنه حالا ما میخوایم همین دستور در فایل اول رو در فایل b.php بدون نوشتن دوباره دستور و با دستور اینکلود بنویسیم کافیه بنویسیم:

PHP Code:

```
        // file name is b.php
<?php
include("a.php");
?>
//will Print message in a.php
```

البته چون در اینجا ما در فایل اولمون یک خط دستور داریم فرقی نمیکنه ولی اگه چندین خط و چندین دستور باشه کار مارو خیلی راحت میکنه چون فقط با یه خط کد میتونیم همون دستورات رو صدا بزنیم .

حتی میتونیم یک مقدار رو با دستور اینکلود صدا بزنیم یا دستور اینکلود رو با توجه به یه شرط صدا بزنیم.

یک مثال :

PHP Code:

```
<?php
//this file name is a.php
$ret=(4+4);
return $ret;
?>
```

PHP Code:

```
<?php
//this file name is b.php
$flag=true;
```

```
if($flag) {
$resualt=include("a.php");
print " The Sum Of (4+4) Is $resualt";
}
?>
```

همونطور که دیدید میتونیم طوری تعریف کنیم که اگه شرط درست نبود اصلا دستور اینکلود اجرا نشه و در خط بعدی مقدار فایل a.php رو توی یه متغییر دیگه مینوسیسم و چاپ میکنیم .

خوب بهتر بود قبل از اینکه بقیه آموزش رو بگم اینارو میدونستید.

کار با فایل ها :

اولین تابعی که میخوایم بررسی کنیم تابع چک کردن وجود یک فایل هست !

به این مثال توجه کنید :

PHP Code:

```
<?php
if (file_exists("a.php"))
print "The File Exists";
?>
```

حتی میتونید با تابع دیگری بفهمید مسیر داده شده یک فایل هست یا یک دایرکتوری

PHP Code:

```
//Check if it's a file
<?php
if(is_file("a.php")
print"yes this is file";
?>
/*----------------------------*/
//Check if Current Path is a dir
<?php
if(is_dir("/tmp"))
print"/tmp is valid";
?>
```

تابع دیگری که وجود داره توابع is_executeable , is_writeable , is_readable هستن که چک میکنن ببینن فایل مورد نظر قابل خواندن و یا نوشتن ویا اجرا شدن هست یا مسیر داده شده معتبر هست یا نه و یک مقدار از نوع بولین برمیگردونه .

تابع دیگری نیز وجود دارد که سایز یک فایل رو برمیگردونه خیلی ساده

Print filesize("a.php");

این تابع سایز فایل شما رو برحسب بایت نمایش میده.

تابع دیگری که میخوایم بررسی کنیم تابع ; ()fileatime می‌باشد که اخرین باری که یک فایل دسترسی پیدا کرد رو به ما بر میگردونه ما در مثال زیر میخوایم بدونیم فایل a.php در چه تاریخ و زمانی برای اخرین بار دسترسی پیدا کرده است :

PHP Code:

```
<?php
$lasttime=fileatime("a.php");
print "The File last time accessed  in ".date("D d M Y  g:i A",$lasttime).".";
// Will Print Sat 14 jan 2006 10:30 Pm
?>
```

تابع;()filemtime نیز مشابه ; ()fileatime هستش با این تفاوت که تاریخ و زمان اخرین باری که فایل ویرایش شد رو برمیگردونه . تابع ; ()filectime نیز وجود داره که در سیستم های یونیک تاریخ تغییر یا ویرایش فایل رو برمیگردونه ولی در پلت فرم های دیگه تاریخ بوجود آمدن فایل رو برمیگردونه .

توابع کاربردی تر :

تابع ;("touch("file-path.txt در صورتی که فایلی با این نام وجود نداشته باشد این فایل رو ایجاد میکنه ولی اگه وجود داشته باشه کاری نمیکنه و فقط تاریخ ویرایش فایل تغییر پیدا میکنه و فایل از بین نمیره با تابع ;("unlink("file-path.txt میتونید یک فایل رو پاک کنید.

نکته : در سیستم های یونیکس برای اینکه یک فایل را پاک یا ویرایش یا دست یابی پیدا کنیم لازم است که دسترسی به فایل رو داده باشید .

بازکردن فایل قبل از خواندن و نوشتن :

قبل از اینکه بتونید یک فایل رو بخونید یا محتوایش رو عوض کنید به این احتیاج دارید که اون فایل رو باز کنید. شما با این دستور میتونید یک فایل رو برای خواندن اماده کنید.

f=fopen("file.txt",'r$');

و با این دستور میتونید فایل رو برای نوشتن اماده کنید

f=fopen("file.txt",'w$');

و برای اضافه کردن اطلاعات به یک فایل باید از این دستور استفاده کنید (Append).

f=fopen("file.txt",'a$');

بهتره قبل از اینکه اقدام به ویرایش یا باز کردن یک فایل کنید اون رو امتحان کنید ببینید اجازه باز شدن یا ویرایش شدن رو داره ؟

PHP Code:

```
If ($fp=fopen("file.txt",'w'))
{
// codehaie marboot be viraiesh file
}
```

یا میتونید بجای کد بالا اینگونه عمل کنید:

PHP Code:

```
($fp=fopen("file.txt",'w')) or die("Could Not open file");
```

اگه دستور فوق مقدار درست رو برگردونه پیغام Could Not open file نشون داده نمیشه در غیر اینصورت نشون داده میشه .

همون طور که متوجه شدید هر عملیاتی که بخوایم بر سر فایل اجرا کنیم باید داخل :

fopen();

Code //

fclose();

انجام بدیم.

پی اچ پی امکانات زیادی رو برای خوندن یک فایل در اختیار ما میزاره بعنوان مثال شما میونید یک فایل رو برحسب بایت یا برحسب لاین یا برحسب کاراکتر بخونید .

بزارید ابتدا یک مثال رو نگاه کنیم و بعد توضیحات مربوطه رو بخونیم :

PHP Code:

```
<?php
$filename = "test.txt";
$fp = fopen( $filename, "r" ) or die("Couldn't open $filename");
while ( ! feof( $fp ) )
 {
 $line = fgets( $fp, 1024 );
 print "$line<br>";
 }
 ?>
```

با استفاده از دستور ; feof() چک میکنیم ببینم به اخر فایل رسیدیم یا نه و اگه نه میایم خط به خط با دستور وایل خط های فایل رو داخل یک متغییر میریزیم و اونها رو چاپ میکنیم . دستور fgets($fp,1024) اینکارو میکنه و میگه که طول هر خط میتونه تا ۱۰۲۴ بایت باشه .

ما میتونیم مقدار خاصی از فایلمون رو بخونیم مثلا ۱۶ بایت یا ۱۶ کاراکتر از فایلمون رو بخونیم

PHP Code:

```
<?php
$filename = "test.txt";
$fp = fopen( $filename, "r" ) or die("Couldn't open $filename");
while ( ! feof( $fp ) )
{
$chunk = fread( $fp, 16 );
print "$chunk<br>";
}
?>
```

همونطور که دیدید با دستور fread($fp,16); ما ۱۶ کاراکتر از فایلمون رو میخونیم .

شما میتونید با تابع fseek() جای مشخصی از فایل رو بخونید به مثال زیر نگاه کنید

PHP Code:

```php
<?php
$filename = "test.txt";
$fp = fopen( $filename, "r" ) or die("Couldn't open $filename");
$fsize = filesize($filename);
$halfway = (int)( $fsize / 2 );
fseek( $fp, $halfway );
$chunk = fread( $fp, ($fsize − $halfway) );
print $chunk;
?>
```

در کد بالا ما نیمه دوم یک فایل رو چاپ میکنیم .همه چیز واضح و روشن هست ونیازی به توضیح نیست دستور ;()fgetcمثل دستور ;()fgetsمی‌باشد که اگه در کد بالا که خط به خط یک فایل رو اجرا میکرد بزارید کاراکتر به کاراکتر فایل رو نشون میده .

برای نوشتن یا اضافه کردن مقدار به یک فایل باید ابتدا فایل رو بصورت

```
fopen("file.txt",'w');
Or
fopen("file.txt",'a');
```

شما میتونید با تابع ;()fwrite داخل یک فایل مقداری رو قرار بدید ، دقت کنید که در اینصورت محتوای فایل قبلی پاک میشه و میتونید با تابع ;()fputs یک مقدار رو به فایل مورد نظر اضافه کنید .

کار کردن با این دستورها ساده هست با این حال یک مثال میارم :

PHP Code:

```php
<?php
$filename = "test.txt";
$fp = fopen( $filename, "w" ) or die("Couldn't open $filename");
fwrite( $fp, "Hello world\n" );
fclose( $fp );
print "Appending to $filename<br>";
$fp = fopen( $filename, "a" ) or die("Couldn't open $filename");
fputs( $fp, "And Hello To You\n" );
fclose( $fp );
?>
```

حال میرسیم به تابع تعیین دسترسی فایل شما میتونید با دستور ;flock() برای یک فایل دسترسی های متفاوتی رو اعمال کنید لیست شماره دسترسی ها به این صورت است

PHP Code:

```
1  ---- ◊Sharing اجازه خواندن میده ولی نوشتن خیر
2  ---- ◊Exclusive اجازه خواندن و نوشتن نمیدهد
3  ---- ◊Release دسترسی های بالا را ازاد میکند
```

کار با پوشه هاست :

شما میتونید با دستور rmdir , ()mkdir پوشه ای ایجاد یا پاک کنید .

توابع ساده ای هستن و نیازی به مثال نیست .

یک مثال برای بیشتر آشنا شدن با این نوع توابع ، میخوایم فایل های داخل یک پوشه رو نمایش بدیم

PHP Code:

```
<?php
$dirname = "testdir";
$dh = opendir( $dirname );
while ( gettype( $file = readdir( $dh )) != boolean )
{
if ( is_dir( "$dirname/$file" ) )
print "(D)";
print "$file<br>";
}
closedir( $dh );
?>
```

ما با دستور ()opendir پوشه مورد نظرمون رو در ابتدا باز میکنیم. سپس با دستور وایل مسیر فایلهامون رو میگیریم و چک میکنیم اگه متعلق به این دایرکتوری بودن عبارت (D) و سپس اسم فایل و مسیرش رو چاپ میکنیم در نهایت با دستور ()closedir میبندیم .

دستور readdir مقدار درست یا نادرست رو برمیگردونه و این بر حسب این هست که هر عددی بجز صفر مقدارش ترو میشه بزارید با مثالی توضیح بدم فرض کنیم ما چهار تا فایل داریم که بر حسب ایندکس ما میایم اینارو داخل وایل چک میکنیم و تا زمانی که ایندکس صفر نشده شرط وایل ما درست هست و ادامه میده و هنگامی که صفر شد از وایل خارج میشه .

تابع mail در PHP

در این مقاله ما خواهیم دید که چگونه زبان PHP را برای ارسال ایمیل تنظیم کنیم و همچنین نحوه فرستادن ایمیلهای HTML و ایمیلهای همراه با فایل ضمیمه (Attachment) را بررسی خواهیم کرد.

قبل از اینکه به کمک PHP بتوانیم ایمیل بفرستیم باید PHP را برای این کار تنظیم کنیم.دقیقا مانند اینکه بخواهیم برنامه ارسال و دریافت ایمیل (مانند outlook) را تنظیم کنیم.

برای این کار هم باید سراغ فایل php.ini رفته و آن را با editor دلخواه خودتان باز کنید.اگر میخواهید کدهای خودتان را بر روی سروری غیر از سیستم خودتان اجرا کنید از این مرحله صرف نظر کنید و فرض را بر این بگزارید که سرور شما برای انجام این کار تنظیم شده است و در نتیجه به مرحله بعد بروید.

در فایل php.ini در قسمتی که با [mail function] عنوان گزاری شده است گزینه ای دارید به نام SMTP که باید مقدار آن را SMTP ایمیلتان بگذارید مثلا mail.softhome.net

البته در فایل php.ini تنظیمات برای سرورهای ویندوز و لینوکس را جدا در نظر گرفته و شما باید بر اساس سیستمی که استفاده میکنید چیزی شبیه زیر را داشته باشید :

برای سیستمهای ویندوز :

[mail function]

Setup for Windows systems ;

SMTP = smtp.my.isp.net

sendmail_from = me@myserver.com

و برای سیستمهای لینوکس :

[mail function]

Setup for Linux systems ;

sendmail_path = /usr/sbin/sendmail -t

sendmail_from = me@myserver.com

وقتی تنظیمات را انجام دادید وب سرور خود را restart کنید و اکنون همه چیز برای ارسال ایمیل آماده است!

ارسال ایمیل ساده (Plain Email) :

حقیقتا از روشی که PHP برای ارسال ایمیل در نظر گرفته ساده تر نمی توان تصور کرد!

در حقیقت شما می توانید ارسال ایمیل را با تنها نوشتن یک خط انجام دهید! مانند زیر :

```
mail('recipient@some.net','Subject','Your message here.');
```

خط بالا یک ایمیل را به آدرس 'recipient@some.net' با موضوع 'Subject' و 'Your message here.' به عنوان متن نامه ارسال می کند.

همانطور که مشاهده کردید PHP ارسال ایمیل را بسیار ساده کرده است . ولی چندین راه حل پیشرفته وجود دارد که به ما این امکان را می دهد که ایمیلهای HTML و ایمیلهای همراه با فایل ضمیمه بفرستیم.

قبل از هر چیز این نکته را متذکر شوم که اگر mail system ی که شما در php.ini تعریف کرده اید ایمیل ارسالی را برگشت (reject) دهد {برای مثال اگر در قسمت To آدرس یک ایمیل درست را ننوشته باشیم} این تابع یک پیغام خطا در مرورگر کاربر نمایش خواهد داد ، دقیقا مانند اتفاقی که در مورد سایر تابعهای PHP می افتد.

اما همانطور که می دانید ما می توانیم با نوشتن علامت @ قبل از تابع از نوشتن پیغام خطا در مرورگر کاربر جلوگیری کنیم.

اگر این نکته را با چیزی که تابع mail بر می گرداند (true یا false بسته به اینکه ایمیل ارسال شده باشد یا خیر) ترکیب کنیم کد زیر را خواهیم داشت :

```
if (@mail($to, $subject, $message)) {
echo('<p>Mail sent successfully.</p>');
} else {
echo('<p>Mail could not be sent.</p>');
}
```

به یاد داشته باشید که ارسال ایمیل نمی تواند تضمینی بر دریافت آن در مقصد باشد.

برای مثال اگر یک ایمیل به آدرس nonexistant.user@hotmail.com بفرستیم و فرض بر این باشد که این آدرس اصلاً وجود ندارد ، این آدرس برای تابع mail قابل قبول است و true را بر می گرداند ولی مطمئنا این ایمیل از بین می رود چون کسی صاحب آن نیست ، پس در این مورد کاری از دست PHP بر نمی آید.

وقتی که می خواهیم یک ایمیل را به چندین آدرس بفرستیم کافیست که در پارامتر اول تمام آدرس ها را پشت سر هم نوشته و آنها را با علامت کاما "،" از هم جدا کنیم. برای مثال :

```
mail('recipient1@some.net, recipient2@some.net',
'An email to two people', 'Message goes here.');
```

خب ، تا حالا اصول فرستادن یک ایمیل را بررسی کردیم ، اما بپردازیم به اصل مطلب و mail header ها و اینکه چه کارهایی میتوانیم با آنها انجام دهیم!

ایمیلهای HTML و header ها :

اکنون شما میتوانید از اسکریپتهای PHP خود ایمیل بفرستید ، چقدر جالب! من مطمئنم وقتی یاد بگیرید که چگونه ایمیلهای HTML بفرستید احساس قدرت بیشتری خواهید کرد!

پس ادامه میدهیم؛

برای اینکه ایملهای HTML را درک کنید ابتدا باید headerهای یک ایمیل را بشناسید.

هر ایمیل دریافتی از دو قسمت تشکیل شده است: headerها و متن نامه (message body) . در زیر نمونه یک ایمیل ساده که برنامه ایمیل شما دریافت کرده است را می بینیم :

```
Return-Path: <sender@elsewhere.com>
Delivered-To: you@some.net
Received: ...several lines like this...
From: Sender <sender@elsewhere.com>
To: You <you@some.net>
```

```
Subject: A Simple Message
Date: Mon, 11 Feb 2002 16:08:19 -0500
Organization: Sender's Company
X-Mailer: Microsoft Outlook, Build 10.0.2616

Hi there! <tap> <tap> Is this thing on?
```

تمام خطوط بالای خط سفید header ها هستند. در واقع یک ایمیل می تواند بیشتر از اینها هم header داشته باشد ولی برای اختصار در این مثال چند مورد اصلی را ذکر کرده ام.

همانطور که می بینید هر خط از header ها با نام آن header شروع می شود (From:, To:, Subject:, Date:, etc) و در ادامه آنها هم چند مقدار (value) قرار گرفته است. بیشتر header ها استاندارد شده هستند و یک مفهوم خاص برای mail program یا mail server ی که مسئول رساندن ایمیل به ما هستند، دارند. اما header های غیر استاندارد هم وجود دارند و مشخصه آنها این است که با X- شروع می شوند (مانند :X-Mailer که اغلب برای نشان دادن برنامه ای که برای ارسال ایمیل استفاده شده است به کار می رود)

نکته: اگر مقدار (value) یک header نیاز به بیش از یک خط داشته باشد ، خطوط اضافه باید با یک فاصله از سر خط شروع شوند. یک مثال در این زمینه را در قسمت بعد خواهیم دید.

وقتی که برنامه ایمیل شما به خط سفید (blank line) رسید می فهمد که header های نامه تمام شده و از این به بعد محتویات متن نامه است که باید نشان داده شود. در مثال ما ، متن نامه همان خط آخر است.

تابع mail در PHP به شما اجازه می دهد که headerهای مورد نظر خودتان را به نامه اضافه کنید و PHP آنها را به header هایی که خود به صورت اتوماتیک تولید می کند اضافه میکند. برای نمونه در مثال پایین یک header با عنوان :X-Mailer به نامه اضافه کرده ایم که PHP 4.x را به عنوان برنامه فرستنده ایمیل معرفی می کند.

```
mail('recipient@some.net', 'Subject', 'Your message here.',
'X-Mailer: PHP 4.x');
```

پارامتر چهارم که یک پارامتر اختیاری است اغلب برای نشان دادن From ایمیل استفاده می شود (علاوه بر Fromی که به صورت پیش فرض در php.ini تعریف کرده ایم). پس اجازه بدهید که یک header از نوع From به نامه اضافه کنیم تا این کار را برای ما انجام دهد:

```
mail('recipient@some.net', 'Subject', 'Your message here.',
"From: sender@some.net\nX-Mailer: PHP 4.x");
```

با توجه به اینکه headerها هر کدام در یک خط باید قرار داشته باشند پس ما باید هر دو خط را با \n از هم جدا کنیم (که این نیز خود نشان دهنده این است که ما باید پارامتر چهارم را درون " " قرار دهیم برای اینکه PHP به کاراکترهای خاص نظیر \n اگر درون ' ' قرار داشته باشند توجه نمی کند).

Header های دیگری هم هستند که نام فرستنده و گیرنده نامه را قبل از آدرس ایمیل آنها مینویسد؛ به این صورت : name <email>.

مثال :

```
mail('recipient@some.net', 'Subject', 'Your message here.',
   "To: The Receiver <recipient@some.net>\n" .
   "From: The Sender <sender@some.net>\n" .
   "X-Mailer: PHP 4.x");
```

توجه داشته باشید که برای اضافه کردن نام به قسمت To ، نمی توانیم نام را در پارامتر اول جا دهیم و تنها راه ممکن این است که یک header با عنوان :To به header ها اضافه کنیم.

Headerهای :CC و :Bcc هم وجود دارند که مورد استفاده آنها را حتما خودتان می دانید:

```
mail('recipient@some.net, someone@some.net, metoo@some.net',
   'Subject', 'Your message here.',
   "To: The Receiver <recipient@some.net>\n" .
   "From: The Sender <sender@some.net>\n" .
   "cc: Interested <someone@some.net>\n" .
```

```
"Bcc: Me Too <metoo@some.net>\n" .
"X-Mailer: PHP 4.x");
```

فقط توجه داشته باشید که آدرس ایمیل تمام گیرنده ها به ترتیب To و cc و Bcc در پارامتر اول نوشته شده است ، این نکته در جایی ذکر نشده است ولی من با تکیه بر تجربیات شخصی خودم به این نکته پی برده ام که اگر می خواهید ایمیل به تمام گیرنده ها برسد باید این کار را بکنید (مخصوصا در سرورهای ویندوز که زیادی حساس هستند!)

اخطار باگ:

دو باگ برای تابع mail در PHP وجود دارد که من اخیرا در PHP نسخه ۴,۱,۰ دیده ام ؛ اول اینکه هدر Cc: باید اینگونه تایپ شود: cc: یا CC: یعنی هر دو حروف بزرگ یا هر دو کوچک ... ترکیبی از حروف کوچک و بزرگ قاعدتا باید کار کند ولی اینطور نیست!

دوم اینکه در سرورهای ویندوز هدر Bcc: درست کار نمی کند. همانطور که می دانید هنگام ارسال نامه ، هدر Bcc: باید از بین headerها حذف شود ، ولی اینگونه نیست و گیرنده ایمیل می تواند هدر Bcc: را در بین هدر ها ببیند!

خب حتما سوال می کنید که این همه چه ربطی به فرستادن ایمیل های HTML داشت؟!

جواب : چند header خاص هستند که باعث می شوند برنامه دریافت کننده ایمیل آن را به عنوان ایمیل HTML بشناسد.

```
mail('recipient@some.net', 'Subject',
   '<html><body><p>Your <i>message</i> here.</p></body></html>',
   "To: The Receiver <recipient@some.net>\n" .
   "From: The Sender <sender@some.net>\n" .
   "MIME-Version: 1.0\n" .
   "Content-type: text/html; charset=iso-8859-1");
```

به متن نامه که با فرمت HTML نوشته شده و همچنین هدر های :Mime-Version و :Content-type توجه داشته باشید.

هدر :Mime-Version نشان دهنده این است که standard extended mail format مورد استفاده است.(Mime مخفف Multipurpose Internet Mail Extensions می باشد) که به ایمیل اجازه می دهد علاوه بر متن ساده دارای محتویات Content-type هم باشد. و هدر :Content-type مشخص می کند که متن از نوع HTML می باشد.

ترکیب Text و HTML در یک ایمیل (Mixed Messages):

یک ایمیل می تواند شامل ترکیبی از text ساده و html باشد که این باعث می شود ایمیل در بیشتر برنامه های ایمیل قابل دیدن باشد و دیگر شما قدرت html را به خاطر کاربرانی که از برنامه های قدیمی استفاده می کنند قربانی نمی کنید.

توجه داشته باشید که ایمیلهای ترکیبی (mixed messages) ضعف های خاص خود را نیز دارا هستند .مثلا به خاطر اینکه شما دو نسخه از ایمیل را در یک ایمیل ارسال می کنید پس قاعدتا حجم ایمیل ارسالی از حجمی که باید به طور معمول داشته باشد بیشتر است و این نکته را هم به خاطر داشته باشید که برنامه های ایمیل قدیمی که mixed messageرا تشخیص نمی دهند ممکن است هر دو نسخه از ایمیل را به صورت فایلهای ضمیمه نشان دهند (یکی text و دیگری html).

اجازه دهید نگاهی به یک ایمیل ترکیبی (mixed messag) ساده بیندازیم و سپس کد PHP برای ارسال آن را بنویسیم:

```
Date: Mon, 11 Feb 2002 16:08:19 -0500
To: The Receiver <recipient@some.net>
From: The Sender <sender@some.net>
Subject: A simple mixed message
MIME-Version: 1.0
Content-Type: multipart/alternative;
        boundary="==Multipart_Boundary_xc75j85x"
```

```
This is a multi-part message in MIME format.

--==Multipart_Boundary_xc75j85x
Content-Type: text/plain; charset="iso-8859-1"
Content-Transfer-Encoding: 7bit

This is the text portion of the mixed message.

--==Multipart_Boundary_xc75j85x
Content-Type: text/html; charset="iso-8859-1"
Content-Transfer-Encoding: 7bit

<html>
<body>
<p>This is the <b>HTML portion</b> of the mixed message.</p>
</body>
</html>

--==Multipart_Boundary_xc75j85x--
```

بعد از headerهای اصلی در بالای message ، هدر MIME-Version: 1.0 را داریم. Headerی که ما را قادر می سازد جنبه های پیشرفته تر ایمیل را بسازیم. هدر :Content-type جایی است که اصل ماجرا شروع می شود:

```
Content-Type: multipart/alternative;
        boundary="==Multipart_Boundary_xxc75885"
```

هدر :Content-type را برابر با multipart/alternative قرار داده ایم که یک نمونه خاص است و به ما این قدرت را می دهد که در ایمیل دو یا چند فرمت مختلف را داشته باشیم (تا برنامه ایمیل کاربر مناسب ترین آن را انتخاب کند و نمایش دهد).

به علاوه اینکه ما از Content-Type برای set کردن یک رشته boundary استفاده کرده ایم.

برای اینکه خطوط header ها کوتاه باشد این قسمت از هدر :Content-type را در خط دوم قرار داده ایم و همانطور که قبلا اشاره کردم باید با مقداری فاصله از سر خط آن را بنویسیم تا مشخص شود که ادامه header قبلی است.

در این مثال من از "==Multipart_Boundary_xc75j85x" استفاده کرده ام.

این رشته معنا و مفهوم خاصی ندارد (boundary در لغت به معنای مرز و سرحد می باشد و در تابع mail مرز هر قسمت از ایمیل را مشخص می کند مثلا اینکه از کجا تا کجا مربوط به قسمت text ایمیل است و از کجا تا کجا مربوط به قسمت html)

من از کاراکترهایی مانند علامت مساوی و underline و یک رشته تصادفی متشکل از اعداد و حروف برای درست کردن رشته boundary استفاده کرده ام و در آخر ما از این رشته برای تقسیم کردن message به چند قسمت استفاده می کنیم.

جمله "This is a multi-part message in MIME format." هم برای کاربران برنامه های ایمیل قدیمی آورده شده است تا اگر احیانا ایمیل در برنامه آنها درست نمایش داده نمی شود دلیل آن را بدانند!

پس از آن رشته boundary آمده است که شروع قسمت اول ایمیل را اعلام می کند :

```
--==Multipart_Boundary_xc75j85x
Content-Type: text/plain; charset="iso-8859-1"
Content-Transfer-Encoding: 7bit

This is the text portion of the mixed message.
```

حتما توجه کنید که وقتی می خواهیم رشته boundary را در کد بگذاریم باید با دو علامت dash پشت سر هم (--) شروع آن را اعلام کنیم.

پس از boundary اول ، قسمت text ایمیل را می نویسیم. هر قسمت از message با یک زوج header همراه است که Content-Type و Encoding آن را مشخص می کنند. در قسمت text مثال ما Content-Type برابر با text/plain (و با charset استاندارد iso-8859-1) و Encoding این قسمت bit۷ می باشد(plain ASCII text)

خط سفید نشان دهنده پایان header ها می باشد و در ادمه متن نامه آمده است.

و پس از همه اینها قسمت html شروع می شود :

```
--==Multipart_Boundary_xc75j85x
Content-Type: text/html; charset="iso-8859-1"
Content-Transfer-Encoding: 7bit

<html>
<body>
<p>This is the <b>HTML portion</b> of the mixed message.</p>
</body>
</html>
```

header ها در این قسمت نیز مانند قسمت text ایمیل هستند به جز در مورد Content-Type که text/html ذکر شده است. پس از متن (body) نامه که به زبان html نوشته شده است نوبت به بستن boundary می رسد :

```
--==Multipart_Boundary_xc75j85x--
```

(بستن boundary هم مانند شروع آن با دو علامت dash پشت سر هم مشخص می شود)

همانطور که می بینید ایمیلهای ترکیبی (ترکیب text و html) پیچیده به نظر می رسند ولی در حقیقت این طور نیست و با یک نگاه عمیق تر به سادگی آنها پی می بریم.

تنها در قسمت تولید رشته boundary است که باید کمی مهارت به خرج دهیم ، من از این روش برای تولید رشته boundary استفاده می کنم:

```
$semi_rand = md5(time());
$mime_boundary = "==Multipart_Boundary_x{$semi_rand}x";
```

در حقیقت unix timestamp کنونی سیستم را با الگوریتم MD5 تبدیل به یک رشته شبه تصادفی کرده ایم. از این رشته هم برای استفاده در رشته boundary استفاده می کنیم.

با در نظر گرفتن تمام اینها شما باید بتوانید یک mixed message را با استفاده از PHP ارسال کنید.

به طریقه دو نیم کردن message با استفاده از رشته boundary خوب دقت کنید که در قسمت ضمیمه کردن فایل هم به آن احتیاج خواهیم داشت.

ضمیمه کردن فایل (File Attachments):

ارسال فایل ضمیمه همراه ایمیل هم تقریبا شبیه ارسال mixed message می باشد به جز در مورد Content-Type که برای کل message در نظر گرفته شده است (multipart/mixed به جای multipart/alternative).

یادآوری : منظور از "Content-Type که برای کل message در نظر گرفته شده است" همان Content-Type است که در پارامتر چهارم تابع mail قرار داده ایم و همانطور که دانستید یک ایمیل می تواند چندین Content-Type مختلف داشته باشد (قسمت های text و html و attachment هر کدام Content-Type خاص خود را دارند و تابع mail هم Content-Type کلی برای همه آنها در نظر می گیرد که در اینجا multipart/mixed است)

یک header جدید به نام Content-Disposition هم داریم که به برنامه ایمیل کاربر می گوید چطور با آن قسمت از ایمیل برخورد کند ، مثلا برای قسمت attachment این را داریم :

```
Content-Disposition: attachment
```

اجازه دهید که یک form را طراحی کنیم که پس از submit ایمیل را همراه با فایل ضمیمه (در صورت وجود) به مقصد ارسال کند.

من خط به خط این اسکریپت را توضیح خواهم داد ، در نتیجه در پایان شما علاوه بر داشتن یک تکه کد به درد بخور ، طریقه کارکرد آن را هم به خوبی فرا خواهید گرفت:

http://www.webmasterbase.com/examples/phpemail/phpemail.zip

ابتدا مقادیر submit شده را در متغیرهایی قرار میدهیم (فرض بر این است که register_globals=off می باشد)

```
// Read POST request params into global vars
$to      = $_POST['to'];
$from    = $_POST['from'];
$subject = $_POST['subject'];
$message = $_POST['message'];
```

فایلهایی را که upload می کنیم در یک آرایه خاص به نام $_FILES قرار می گیرند و در نتیجه ما به راحتی می توانیم مقادیر مورد نیاز را از آن بگیریم.

```
// Obtain file upload vars
$fileatt      = $_FILES['fileatt']['tmp_name'];
$fileatt_type = $_FILES['fileatt']['type'];
$fileatt_name = $_FILES['fileatt']['name'];
```

برای مختصر شدن مقاله فرض میکنیم که پارامتر های $to و $from دارای مقادیری valid هستند (آدرس ایمیل) و ما آنها را چک نمی کنیم (به طور عادی آنها باید با regular expressions چک شوند).

در مرحله بعد header ایمیل را کامل می کنیم ، برای شروع مقدار from را در آن قرار می دهیم:

```
$headers = "From: $from";
```

(این $header در پارامتر چهارم تابع mail قرار می گیرد)

سپس متغیر $fileatt را چک می کنیم که آیا path و نام فایل upload شده را در خود دارد یا خیر. برای این کار دستوری داریم به نام is_uploaded_file و از آن استفاده می کنیم:

```
if (is_uploaded_file($fileatt)) {
// Read the file to be attached ('rb' = read binary)
$file = fopen($fileatt,'rb');
$data = fread($file,filesize($fileatt));
fclose($file);
```

در این قسمت محتویات و مشخصات فایل را در متغیر $data قرار داده ایم.

الان باید header های ایمیل را جوری تنظیم کنیم تا بنواند نامه های multipart/mixed بفرستد :

```
// Generate a boundary string
$semi_rand = md5(time());
$mime_boundary = "==Multipart_Boundary_x{$semi_rand}x";

// Add the headers for a file attachment
$headers .= "\nMIME-Version: 1.0\n" .
        "Content-Type: multipart/mixed;\n" .
        " boundary=\"{$mime_boundary}\"";
```

و حالا می رسیم به متن نامه :

این دقیقا همان قسمت text در مبحث قبلی می باشد؛

```
// Add a multipart boundary above the plain message
$message = "This is a multi-part message in MIME format.\n\n" .
        "--{$mime_boundary}\n" .
        "Content-Type: text/plain; charset=\"iso-8859-1\"\n" .
        "Content-Transfer-Encoding: 7bit\n\n" .
        $message . "\n\n";
```

و اکنون با استفاده از روش Base64 فایل ضمیمه را به باینری تبدیل می کنیم (مناسب برای ارسال با ایمیل). تمامی برنامه های معروف ایمیل از روش Base64 encoding پشتیبانی می کنند ، پس ما هم از این روش استفاده می کنیم. خوشبختانه PHP هم یک تابع برای Base64 encoding در نظر گرفته است:

```
// Base64 encode the file data
$data = chunk_split(base64_encode($data));
```

ما الان تمام چیزهایی که برای ضمیمه کردن فایل لازم بود را داریم. این هم از کد:

```
// Add file attachment to the message
$message .= "--{$mime_boundary}\n" .
        "Content-Type: {$fileatt_type};\n" .
        " name=\"{$fileatt_name}\"\n" .
        "Content-Disposition: attachment;\n" .
        " filename=\"{$fileatt_name}\"\n" .
        "Content-Transfer-Encoding: base64\n\n" .
        $data . "\n\n" .
        "--{$mime_boundary}--\n";
}
```

این قسمت از کد تمام تغییرات و توضیحاتی را که لازم بود تا فایل را به عنوان attachment در نامه جا دهیم اعمال می کند.

اکنون باید با استفاده از تابع mail نامه را بفرستیم :

```
// Send the message
$ok = @mail($to, $subject, $message, $headers);
if ($ok) {
echo "<p>Mail sent! Yay PHP!</p>";
} else {
```

```
echo "<p>Mail could not be sent. Sorry!</p>";
}
```

و این بود تمام چیزی که برای ارسال ایمیل باید می دانستیم!

خلاصه :

در این مقاله به احتمال زیاد چیزهایی را درباره ایمیل آموختید که تا کنون شاید نمی دانستید.دانستیم که چگونه با تابع قدرتمند mail کارهای پیشرفته انجام دهیم.

آموختیم که :

- چگونه header های ایمیل را تنظیم کنیم ؛
- ایمیلهای html بفرستیم ؛
- هر دو فرمت html و plain text را در یک ایمیل قرار دهیم.
- و سر انجام ....
- با اضافه کردن چند تکنیک جدید به همه اینها توانستیم یک فایل را ضمیمه ایمیل کنیم.

اگر آدم پر حوصله ای هستید تلاش کنید تا این کد را برای ضمیمه کردن چند فایل در یک نامه گسترش دهید و یا متن های html را پشتیبانی کنید. و یا اگر یک برنامه نویس شی گرا هستید تلاش کنید تا یک کلاس بسازید که تمام تابع هایی که ما به کار بردیم را encapsulate کند.

اگر واقعا علاقه مند شده اید RFC for MIME extensions را چک کنید تا از تمام قابلتهایی که ایمیل دارد مطلع شوید. (RFC 2045)

ذخیره سازی مشخصات بازدید کنندگان

ذخیره سازی مشخصات بازدید کنندگان بر روی فایل ساده:

این مقاله به عزیزانی که میخواهند اطلاعاتی در مورد سیستم بازدیدکننده های سایت خود به دست آورند کمک بزرگی خواهد کرد. وقتی شما هزینه ای را برای گرفتن فضا پرداخت کرده اید و از یک فضا بر روی یک سرور استفاده میکنید نمیتوانید از فایلهای Log سرور استفاده نمایید، اگر سرور در اختیار خودتان باشد این کار امکان پذیر است و حتی اطلاعات مفیدی نیز ذخیره میشود، ولی وقتی سرور در اختیارتان نباشد میبایست از طریق برنامه نویسی اقدام به اینکار نمایید ، در این مقاله به بررسی برنامه نویسی این سیستم توسط PHP خواهیم پرداخت. با استفاده از این روش میتوانید آماری از ترافیک سایتتان را بدست آورید و همچنین متوجه شوید که کاربر از کجا به سایتتان آمده و چه صفحاتی را مشاهده کرده اند. با خواندن این مقاله میتوانید اعمال فوق را بدون استفاده از بانک اطلاعاتی MySQL انجام دهید، ما فقط به یک فایل متنی معمولی برای ذخیره ترافیک استفاده میکنیم و دو متد را مورد بررسی قرار میدهیم. اولین متد در رابطه با Session منحصر به فرد هر کاربر و دومین متد در رابطه با اطلاعات وارد شدن کاربران به همه صفحات میباشد. در این مقاله علاوه بر مرور این سیستم ، اطلاعات مناسبی در مورد متغیرهای از پیش تعریف شده در PHP در اختیار شما قرار خواهد گرفت.

متد ۱ : (Session Logging)

با استفاده از این متد میتوانید اطلاعات منحصر به فرد در بازدید کننده سایت را بدست آورید :

```
<?php
session_start();
if(!session_is_registered('counted')){
   $agent = $_SERVER['HTTP_USER_AGENT'];
   $uri = $_SERVER['REQUEST_URI'];
   $user = $_SERVER['PHP_AUTH_USER'];
   $ip = $_SERVER['REMOTE_ADDR'];
   $ref = $_SERVER['HTTP_REFERER'];
   $dtime = date('r');

if($ref==     \"\"){
```

```
    $ref = \"None\";
  }
  if($user == \"\"){
    $user = \"None\";
  }

  $entry_line = \"$dtime - IP: $ip | Agent: $agent  | URL: $uri | Referrer: $ref | Username: $user
n\";
  $fp = fopen(\"logs.txt\", \"a\");
  fputs($fp, $entry_line);
  fclose($fp);
  session_register('counted');
}
?>
```

اجازه دهید که در مورد سورس کد بالا کمی صحبت کنیم :

```
<?session_start(); ?>
```

با استفاده از این دستور به PHP اعلام میکنید که Session فردی را آماده کند که ما اطلاعات مرورگر و سیستم آن را بدست آوریم.

```
<?if(!session_is_registered('counted')){ ?>
```

در این کد ما Session مورد نظر را چک میکنیم که اگر قبلا Session با نام Counted وجود نداشته باشد دستورات داخل شرط اجرا شود و اطلاعات سیستم کاربر، ذخیره شود.

```
<?
$agent = $_SERVER['HTTP_USER_AGENT'];
$uri = $_SERVER['REQUEST_URI'];
$user = $_SERVER['PHP_AUTH_USER'];
$ip = $_SERVER['REMOTE_ADDR'];
$ref = $_SERVER['HTTP_REFERER'];
$dtime = date('r');
?>
```

کدهای بالا همان متغیرهای از پیش تعریف شده هستند که توسط خود PHP برای ما فراهم شده اند و میتوانیم از آنها استفاده کنیم. در ادامه در مورد هر کدام از این متغیرهای از پیش تعریف شده ، توضیح کمی خواهم داد.

```
<? $agent = $_SERVER['HTTP_USER_AGENT']; ?>
```

نوع مرورگر را نشان میدهد.

```
<? $uri = $_SERVER['REQUEST_URI']; ?>
```

URL صفحه ای کاربر در آن است را نشان میدهد.

```
<? $ip = $_SERVER['REMOTE_ADDR']; ?>
```

آدرس IP کامپیوتر مورد نظر را میدهد.

```
<? $ref = $_SERVER['HTTP_REFERER']; ?>
```

مشخص میکند که کاربر از چه سایتی به سایت شما آمده.

```
<? $dtime = date('r'); ?>
```

این دستور فقط یک ساختار تاریخی است که زمان دسترسی کاربر به سایت را در متغیر $dtime ذخیره میکند.

```
<?
if(!$ref){
   $ref = \"None\";
}
if(!$user){
   $user = \"None\";
} ?>
```

بعد از قرار دادن متغیرهای از پیش تعریف شده در متغیرهای برنامه در کد بالا، میتوانیم متغیرهایی که مقادیری در آن قرار نگرفته شناسایی کرده و مقدار None را در آنها قرار میدهیم.

```
<?php
$entry_line = \"$dtime - IP: $ip | Agent: $agent  | URL: $uri | Referrer: $ref | Username: $user n\";
$fp = fopen(\"logs.txt\", \"a\");
fputs($fp, $entry_line);
fclose($fp);
session_register('counted');
?>
```

در دستورات بالا مقادیر بدست آورده از سیستم کاربران را در یک فایل متنی ذخیره میکنیم. در متغیر $entry_line مشخص میکنیم که چه اطلاعاتی در فایل ذخیره شود و این اطلاعات را در ابتدا در این متغیر ذخیره میکنیم، این اطلاعات میتواند تغییر داده شود و به دلخواه خودتان اطلاعات مورد نظر ذخیره شود. در خط بعدی با استفاده از دستور fopen یک فایل متنی را انتخاب کرده و آن را در حالت درج اطلاعات قرار میدهیم و مقدار را در یک اشاره گر فایل که $fp میباشد قرار میدهیم. در خط بعد با استفاده از تابع fputs در پارامتر اول اشاره گر فایل و در پارامتر دوم متغیری را که حاوی مقداری است که مایل به ذخیره سازی در فایل میباشیم را قرار میدهیم. در خط بعدی و با تابع fclose فایل مورد نظر را میبندیم و در خط بعدی یعنی ;('counted'); session_register باعث میشود که مشخص گردد که اطلاعات این کاربر ذخیره شده است که مجددا اقدام به ذخیره سازی اطلاعات این کاربر نکند.

تا اینجا تمام موارد مربوط به متد ۱ و ذخیره اطلاعات منحصر به فرد هر کاربر میباشد. حال به بررسی متد ۲ میپردازیم.

متد ۲ : (Logging All Page)

در این روش هر صفحه ای که مورد بازدید قرار میگیرد اطلاعات را ثبت میکند، این متد بسیار شبیه متد ۱ میباشد با این تفاوت که Session را از سورس برنامه حذف میکنیم. البته روش متد ۲ همیشه مناسب نیست چونکه بسرعت باعث افزایش حجم فایل Log میگردد.

```
<?php
   $agent = $_SERVER['HTTP_USER_AGENT'];
   $uri = $_SERVER['REQUEST_URI'];
```

```
   $user = $_SERVER['PHP_AUTH_USER'];
   $ip = $_SERVER['REMOTE_ADDR'];
   $ref = $_SERVER['HTTP_REFERER'];
   $dtime = date('r');

   if($ref == \"\"){
      $ref = \"None\";
   }
   if($user == \"\"){
      $user = \"None\";
   }

   $entry_line = \"$dtime - IP: $ip | Agent: $agent  | URL: $uri | Referrer: $ref | Username: $user
n\";
   $fp = fopen(\"logs.txt\", \"a\");
   fputs($fp, $entry_line);
   fclose($fp);
?>
```

همانطور که متوجه شده اید کدهای زیر از سورس کد اصلی حذف شده اند.

```
<?php
session_start();
if(!session_is_registered('counted')){

session_register('counted');
}
?>
```

در پایان بد نیست کمی در مورد تنظیمات فایل Log صحبت کنیم. در ابتدا شما میبایست یک فایل خالی به وب سرور خود ارسال کنید و سعی کنید در اشاره گر فایل $fp آدرس کامل فایل را مثل /www/htdocs/logs.txt وارد نمایید. بعد از انجام این عملیات در نهایت با اجرای دستور CHMOD 755 روی فایل Permission لازم را ایجاد نمایید. [12]

استفاده از تابع تبدیل تاریخ شمسی به میلادی و برعکس

استفاده از تابع تبدیل تاریخ شمسی به میلادی و بالعکس در PHP:

نویسنده: (سید حمید رضا هاشمی گلپایگانی)

برای دریافت توابع کافیست که کد توابع را از این آدرس دریافت کنید :

http://www.iranphp.net/modules/sections/index.php?op=viewarticle&artid=19

بوسیله یک Copy و Paste متن این توابع را درون یک فایل جدید بنام به عنوان مثال jalali.php ذخیره کنید .

برای اینکه بتوانید از این توابع در برنامه خود استفاده کنید کافیست در ابتدای برنامه خود دستور زیر را وارد کنید :

```
<?php require_once "jalali.php\";?>
```

با این کار دو تابع با نامهای gregorian_to_jalali و jalali_to_gregorian در اختیار شما قرار می گیرند که به ترتیب برای تبدیل تاریخ میلادی به شمسی و شمسی به میلادی مورد استفاده قرار می گیرند .

برای اینکه بتوانم نحوه استفاده از این دو تابع را شرح دهم به ذکر مثالهایی در این مورد می پردازم .

۱- تبدیل تاریخ میلادی به شمسی :

برای این کار باید از تابع gregorian_to_jalali استفاده کنید . این تابع ۳ متغیر را از ورودی دریافت می کند که به ترتیب سال، ماه و روز می باشد که همگی از نوع عددی می باشند .

برای مثال اگر بخواهیم تاریخ **۲۰۰۲-۱۱-۲۵** را که درون یک متغیر داریم به شمسی تبدیل کنیم و سپس آنرا درون یک متغیر قرار دهیم ابتدا باید عناصر تاریخ میلادی رو جدا کنیم . برای این کار می توانیم در دستور زیر استفاده کنیم :

```
<?php $gdate='2002-11-25';
list( $gyear, $gmonth, $gday ) = preg_split ( '/-/', $gdate );
?>
```

در اینجا از دستور preg_spilit برای جدا کردن عناصر تاریخ $gdate توسط جداکننده – استفاده کرده ایم . همانطور که می بینید syntax این دستور به این صورت است که ۲ متغیر به عنوان ورودی دریافت می کند . اولی به عنوان delimiter می باشد که چون می توان به صورت regex نیز آنرا وارد کرد باید بین دو slash (/) قرار گیرد و متغیر دوم هم که همان متغییری است که تاریخ میلادی ما در آن قرار دارد .

چون خروجی این دستور یک آرایه است از دستور list استفاده کرده ایم تا عناصر آرایه را هر کدام در یک متغیر قرار دهیم . هم اکنون روز، ماه و سال میلادی را هر کدام در یک متغیر داریم . حال می توانیم از تابع gregorian_to_jalali استفاده کنیم :

```
<?php
list( $jyear, $jmonth, $jday ) = gregorian_to_jalali($gyear, $gmonth, $gday);
?>
```

در اینجا مشخص است که ۳ متغیر سال، ماه و روز میلادی را به عنوان متغیر های ورودی تابع grogorian_to_jalali وارد کرده ایم و چون خروجی این تابع از نوع آرایه است توسط دستور list هر کدام از عناصر این آرایه را در یک متغیر قرار می دهیم . این متغیر های حاوی اطلاعات سال، ماه و روز همان تاریخ به شمسی می باشند . برای اینکه آنها را به صورت قابل نمایش در یک متغیر قرار دهیم می توانیم این خط را به برنامه اضافه کنیم :

```
<?php
$jdate = $jyear."/".$jmonth."/".$jday;
?>
```

در صوتیکه مقدار $jdate را نمایش دهید همان تاریخ از نوع شمسی خواهد بود که چیزی شبیه به مقدار زیر است :

حال در صورتیکه بخواهید تاریخ همین لحظه را به صورت شمسی بدست آورید می توانید از برنامه زیر استفاده کنید :

```
<?php
list($gyear, $gmonth, $gday ) = preg_split ('/-/', date("Y-m-d\"));
list( $jyear, $jmonth, $jday ) = gregorian_to_jalali($gyear, $gmonth, $gday);

$jdate = $jyear.\"/\".$jmonth.\"/\".$jday;
?>
```

همانطور که ملاحظه کردید از دستور date با آرگومان داده شده تاریخ فعلی سیستم استخراج می شود و در همان روال توضیح داده شده تبدیل به شمسی شده و در متغیر $jdate قرار می گیرد .

برای اینکه کارتان کمی ساده تر شود می توانید تابعی به شکل زیر تعریف کنید :

```
<?php
function get_jalali_date( $gdate='now' )
{

if ( $gdate == 'now' )
{
list($gyear, $gmonth, $gday ) = preg_split ('/-/', date("Y-m-d\"));
}
else
{
list( $gyear, $gmonth, $gday ) = preg_split ( '/-/', $gdate );
}
list( $jyear, $jmonth, $jday ) = gregorian_to_jalali($gyear, $gmonth, $gday);
return $jyear.\"/\".$jmonth.\"/\".$jday;

}
?>
```

تابعی با نام get_jalali_date تعریف کردیم که یک متغیر به عنوان ورودی دریافت می کند که این همان تاریخ به میلادی است . در صوتیکه هنگام صدا کردن این تابع تاریخ میلادی را به عنوان ورودی برای این تابع وارد کرده باشید، تابع همان تاریخ را به شمسی تبدیل کرده و در خروجی بر می گرداند، ولی اگر هیچ متغیری به عنوان ورودی به این تابع ندهید، تاریخ همان لحظه را به شمسی برای شما در خروجی بر می گرداند .

مثلا برای تبدیل همان تاریخ ۲۰۰۲-۱۱-۲۵ به شمسی و قرار دادن آن در یک متغیر می توانیم با استفاده از تابع بالا اینگونه عمل کنیم :

```
<?php $jdate = get_jalali_date("2002-11-25\");?>
```

و یا برای بدست آوردن تاریخ همین لحظه و قرار دادن آن در یک متغیر اینگونه عمل می کنیم :

```
<?php $jdate = get_jalali_date();?>
```

همانطور که می بینید کار بسیار ساده تر شد.

۲- تبدیل تاریخ شمسی به میلادی :

با توضیحاتی مشابه می توانید از تابع jalali_to_gregorian استفاده کنید . مانند تابع قبل این تابع هم سه متغیر به عنوان ورودی دریافت می کند که عبارتند از سال، ماه و روز که همگی به شمسی هستند . سپس در جواب آرایه ای شامل سه قسمت که سال، ماه و روز میلادی همان تاریخ هستند را بر می گرداند .

برای نمونه اگر بخواهیم تاریخ ۱۳۸۱/۹/۴ را به میلادی تبدیل کنیم کاری مشابه برنامه زیر انجام می دهیم :

```
<?php
$jdate="1381/9/4\";
list( $jyear, $jmonth, $jday ) = preg_split ( '/\//', $jdate );
list( $gyear, $gmonth, $gday ) = jalali_to_gregorian($jyear, $jmonth, $jday);
$gdate = $gyear.\"-\".$gmonth.\"-\".$gday;
?>
```

تنها تفاومت موجود با تابع بالا استفاده از / برای جداسازی اجزاء تاریخ شمسی می باشد . همانطور که گفتیم تابع preg_split آرگومان اول خود که همان delimiter (جدا کننده) می باشد را به صورت regex می گیرد که باید بین دو علامت / باشد ، در اینجا باید قبل / که نشان دهنده کاراکتر delimiter ما می باشد یک \ (slash back) قرار می دهیم که جداکننده مشخص باشد .

توضیح خاص دیگری هم ندارد و همانند تابع تبدیل میلادی به شمسی اینجا هم می توانیم تابعی با نام get_gregorian_date بسازیم که کار را راحتتر کند . ایجاد این تابع را به خواننده واگذار می کنم.

چگونگی توسعهء PHP

چکیده

در این فصل به بررسی نحوهء توسعهء PHP می پردازیم. یکی از بزرگترین محاسن برنامه های open source از جمله PHP، باز بودن متن برنامهء آنهاست که به توسعه دهندگان این امکان را می دهد تا برنامهء مورد نظر را طبق درخواست ها و نیاز های خودشان تغییر داده و از آن استفاده کنند. پس از خواندن این مقاله شما خواهید توانست توسعه های ساده ای برای PHP نوشته و از آنها در پروژه های خود استفاده کنید.

بخشهای تشکیل شده این فصل این فصل

۱. مقدمه فصل

در این فصل به بررسی نحوهء توسعهء PHP می پردازیم. یکی از بزرگترین محاسن برنامه های open source از جمله PHP، باز بودن متن برنامهء آنهاست که به توسعه دهندگان این امکان را می دهد تا برنامهء مورد نظر را طبق درخواست ها و نیاز های خودشان تغییر داده و از آن استفاده کنند. به عنوان مثال از تاریخ شمسی در PHP پشتیبانی نمی شود و شما می توانید با توسعهء این زبان، برای پشتیبانی این تقویم، آن را برای خود مناسب و قابل استفاده کنید.

پس از خواندن این فصل شما خواهید توانست توسعه های ساده ای برای PHP نوشته و از آنها در پروژه های خود استفاده کنید. در این فصل با هم توسعه ای ساده برای PHP خواهیم نوشت.

برای درک مطالب این فصل احتیاج به داشتن اطلاعاتی در مورد زبان برنامه نویسی C و PHP و روش کامپایل کردن در لینوکس دارید. زبان برنامه نویسی این فصل C بوده و محیط کاری لینوکس می باشد. پیش فرض آن است که شما توانایی نصب برنامه های مختلف تحت لینوکس را دارید. تمامی کار های این فصل تحت خط فرمان یا command line انجام می شود و شما باید تسلط و درک کافی در این مورد را نیز داشته باشید.

۲.برنامه های مورد نیاز

برای توسعهء PHP احتیاج به نرم افزار هایی داریم که مسئولیت عملیات کامپایل کردن در لینوکس را دارند. لیستی از برنامه های مورد نیاز را در زیر می بینید، این نرم افزار ها را می توانید از آدرس www.gnu.org گرفته و روی سیستم لینوکس تان نصب کنید. در صورتی که از توسعه های معروف لینوکس استفاده می کنید اکثر این برنامه ها را نصب شده دارید.

- bison
- flex
- m4
- autoconf
- automake

- libtool
- gcc یا هر نوع کامپایلر دیگر
- make
- cvs از سایت www.cvshome.org

پس از نصب برنامه های فوق، احتیاج دارید نسخهء قابل توسعهء PHP یعنی نسخهء CVS یا Concurrent Version System آن را از آدرس http://cvs.php.net بگیرید و آمادهء توسعه شوید. برای این کار باید تحت خط فرمان لینوکس تان دستوری مشابه دستور زیر تایپ کرده و به جای X، Y و Z شمارهء نگارش PHP ای که قصد گرفتن آن را دارید را وارد کنید. با اجرای این دستور متن برنامه های PHP از اینترنت گرفته می شود و با شاخه ای به نام php-src روی دیسک سخت شما ذخیره می شود.

```
cvs -d :pserver:cvsread@cvs.php.net:/repository checkout -r php_X_Y_Z php-src
```

۳. توسعهء PHP

۱.۳. هدف برنامه و آماده کردن محیط

می خواهیم تابعی به PHP اضافه کنیم به شکل salam() که به عنوان ورودی یک رشته را دریافت کرده و رشته ای به صورت salam STRING را بر گرداند.

به عنوان اولین قدم، به شاخهء PHP رفته و سپس به شاخهء ext آن می رویم و فایلی را در مورد نوع توابعی که قصد توسعه داریم می سازیم. در این فایل که ما نام آن را salam.def می گذاریم به ترتیب نوع خروجی توابع، نام تابع، نام و نوع پارامترهای ورودی، یک " " (فاصله) به عنوان جدا کننده و رشته ای که توضیح مختصری از آن تابع باشد را می نویسیم. در این فایل، هر تابع را باید در یک خط بنویسیم. در زیر به نمونه فایل مورد نیاز برای توسعهء تابع مان توجه می کنیم:

string salam(string arg) return "salam ARG"

توجه کنید که تمامی مقادیر بعد از پرانتز بسته تا آخر خط ، توضیحات تابع می باشند.

PHP برنامه ای به نام ext_skel برای آماده سازی مقدمات توسعه دارد، که در شاخهء ext قرار دارد و شما می توانید به عنوان دومین قدم توسعه، از آن استفاده کنید. برای ادامهء توسعه احتیاج داریم از خط زیر استفاده کرده تا مجموعهء فایل های مورد نیاز توسط این برنامه ساخته شود.

./ext_skel –extname=salam –proto=salam.def

با اجرای این دستور شاخه ای به نام مقدار جلوی extname– ساخته شده و فایل های مورد نیاز نیز در آن قرار می گیرد. این فایل ها عبارتند از:

- config.m4
- CREDITS
- EXPERIMENTAL
- salam.c
- salam.php
- Makefile.in
- php_salam.h
- شاخهء tests

۲.۳. توسعهء فایل ها

یکی از مهمترین و اصلی ترین فایل هایی که لازم است تغییر دهیم و تمامی کدهای اصلی برنامهء ما در آن قرار دارد، فایل salam.c می باشد. با باز کردن این فایل، اولین قسمت مهمی که مشاهده می کنید، خطوط زیر می باشد:

```
/* {{{ salam_functions[]
 *
 * every user-visible function must have an entry in salam_functions[]
*/
```

```
function_entry salam_functions[] = {
    PHP_FE(confirm_salam_compiled,  NULL)  /* for testing; remove later */
    PHP_FE(salam,   NULL)
    {NULL, NULL, NULL}  /* must be the last line in salam_functions[] */
};
                                                    /* }}} */
```

این مقادیر توسط فایل salam.def که تعریف کرده بودیم ساخته شده و در صورتی که فایل salam.def را بدون نقص نوشته باشیم، احتیاجی به ایجاد تغییر در این قسمت نداریم. توجه کنید که اولین تابع معرفی شده یعنی confirm_salam_compiled یک تابع برای تست کامپایل شدن یا نشدن سری توابع مان بوده که پس از موفقیت در کامپایل، می توانیم خط مربوطه را حذف کنیم.

کدهای مهم بعدی، مشابه بخش زیر خواهد بود که شامل توابع پیش فرض زمان فراخوانی و اتمام کار توابع مورد نظر می باشد. این بخش، اطلاعاتی در مورد نوع، نگارش و نام توسعهء ما را نیز داراست.

```
zend_module_entry salam_module_entry = {
    STANDARD_MODULE_HEADER,
    "salam",
    salam_functions,
    PHP_MINIT(salam),
    PHP_MSHUTDOWN(salam),
    PHP_RINIT(salam), /* replace with NULL if no request init code */
    PHP_RSHUTDOWN(salam), /* replace with NULL if no request shutdown code */
    PHP_MINFO(salam),
    "0.1", /* replace with version number for your extension */
    STANDARD_MODULE_PROPERTIES
};
```

برای تابعی که ما می خواهیم توسعه دهیم باید مقادیر PHP_RINIT(salam) و PHP_RSHUTDOWN(salam) را با NULL جایگزین کنیم، پس کد شما چیزی شبیه مقادیر زیر خواهد شد:

```
zend_module_entry salam_module_entry = {
        STANDARD_MODULE_HEADER,
        "salam",
        salam_functions,
        PHP_MINIT(salam),
        PHP_MSHUTDOWN(salam),
        NULL,
        NULL,
        PHP_MINFO(salam),
        "0.1", /* replace with version number for your extension */
        STANDARD_MODULE_PROPERTIES
};
```

سپس توابع(PHP_MINIT(salam و PHP_MSHUTDOWN(salam را پیدا کرده و تعیین می کنیم که مقدار SUCCESS را برگردانند. پس کد این بخش برنامهء ما شبیه کد زیر می شود:

```
PHP_MINIT_FUNCTION(salam) {
   return SUCCESS;
}
PHP_MSHUTDOWN_FUNCTION(salam) {
   return SUCCESS;
}
```

سپس به اصلی ترین تابع می رسیم که باید تمام وظایف مورد نظر برای تابع salam را در آن توسط زبان C نوشته تا خروجی مناسب را برای ما تولید کند. کدهای شما باید بعد از خط return بیاید و خط های بالای آن بدون تغییر بماند (این بدون تغییر ماندن به دلیل آگاهی نداشتن ما برای تغییر آنهاست. صحبت کامل در مورد این خطوط از حوصلهء این فصل خارج بوده و فقط کافیست بدانید که این کدها شامل مراحل پردازش مقادیر ورودی تابع است).

```
/* {{{ proto string salam(string arg)
   return "salam ARG" */
PHP_FUNCTION(salam)
{
```

```
    char *arg = NULL;
    int argc = ZEND_NUM_ARGS( );
    int arg_len;

    if (zend_parse_parameters(argc TSRMLS_CC, "s", &arg, &arg_len)
      == FAILURE)
      return;

    // YOUR CODES SHOULD COME HERE
  }
/* }}} */
```

تابع PHP_FUNCTION برای PHP نام یک تابع را مشخص می کند و مقادیر داخل این تابع، اعمالی را که تابع PHP در هنگام فراخوانی آن در برنامه های PHP باید انجام دهد را نشان می دهد. همانطور که گفته شد، بعد از چند خط اول، کدهای ما قرار خواهد گرفت و برنامهء ما به صورت زیر در خواهد آمد:

```
  PHP_FUNCTION(salam) {
    char *arg = NULL, *sout;
    int argc = ZEND_NUM_ARGS( );
    int arg_len, len;

    if (zend_parse_parameters(argc TSRMLS_CC, "s/", &arg, &arg_len)
      == FAILURE)
      return;

    strcpy(sout, "salam ");
    strcat(sout, arg);
    len = strlen(sout);
    RETURN_STRINGL(sout, len, 1);
  }
```

ابتدا متغیر هایی به نام sout از نوع کاراکتر و len از نوع عددی تعریف کرده ایم. در خط بعد از return مقدار " salam" را در متغیر sout کپی کرده و سپس مقادیر sout و arg که رشتهء ورودی ما می باشد را به هم متصل می

کنیم و به sout انتساب می دهیم. سپس طول sout را به دست آورده و از طریق تابع RETURN_STRINGL (این تابع جزو توابع ZEND API بوده و برای برگرداندن مقادیر استفاده می شوند، این توابع محدود و تعیین شده می باشند) مقدار رشته و طول آنرا بر می گردانیم. سومین متغیر در تابع RETURN_STRINGL مشخص کنندهء صحت خروجی یا وجود اشکال در خروجی می باشد.

هم اکنون توسعه تابع ما به پایان رسیده، اما برای امکان کامپایل تابع توسعه داده شده لازم است مقادیر لازم برای درست کامپایل شدن تابع را تصحیح و کامل کنیم. پس فایل salam.c را ذخیره کرده و می بندیم و فایل config.m4 را برای اِعمال تغییرات باز می کنیم. در این فایل، از قبل، خطوط لازم وجود دارند ولی به صورت توضیح یا comment در آمده اند که شما باید این خطوط را از این حالت خارج کنید. پس از این کار خطوطی دقیقاً مشابه خطوط زیر خواهید داشت (توجه کنید که هیچ علامتی در ابتدای این خطوط نباشند):

```
PHP_ARG_ENABLE(salam, whether to enable salam support,
[  --enable-salam         Enable salam support])
```

توسعهء تابع ما به پایان رسید، حال نوبت کامپایل کردن PHP است تا بتوانید از تابعی که نوشتیم در آن استفاده کنیم. در بخش بعد به این مسئله می پردازیم.

۴. کامپایل PHP

قبل از اجرای دستورات کامپایل، برای اینکه PHP، توسعهء جدید ما را بشناسد، لازم است دستور buildconf را که در شاخهء اصلی PHP قرار دارد، اجرا کنیم. این دستور را باید به شکل زیر وارد کنیم:

```
./buildconf
```

سپس باید برنامه را با توسعه ای که نوشتیم کامپایل کنیم. این کار در PHP شبیه خیلی از برنامه های دیگر تحت لینوکس انجام می شود. به این صورت که شما باید مقداری مشابه مقدار --enable-salam را در هنگام configure کردن PHP اِعمال کنید. سپس سری دستورات make را برای نصب PHP اجرا می کنیم. دستوراتی که

اجرا می کنیم به ترتیب، شبیه دستورات زیر خواهد بود، توجه کنید که تمامی این دستورات باید در شاخهء اصلی PHP اجرا شوند:

```
./configure --enable-salam
make
make install
```

اکنون PHP شما با تابع شما توسعه پیدا کرده است. حال نوبت آزمایش تابع است تا مشاهده کنیم که درست کار می کند. برای این منظور فایلی با توسعهء php و نام test ساخته و مقادیر زیر را در آن می نویسیم:

```
<?php
    echo salam('donya');
?>
```

سپس، در خط فرمان دستور زیر را اجرا کنید. می بینید که مقدار "salam donya" در خروجی چاپ خواهد شد.

```
/usr/bin/php –q test.php
```

۵. جمع بندی این فصل

در این فصل یاد گرفتیم که چطور از برتری های برنامه های open source استفاده کرده و از آنها برای رسیدن به اهداف مان استفاده کنیم و چطور آنها را توسعه دهیم.

تابعی برای PHP ساختیم، که قبلا در آن وجود نداشت و از آن در برنامه ای که با PHP نوشتیم استفاده نمودیم.

به یاد داشته باشید که این فصل، فقط توضیحات مختصری در مورد توسعهء PHP ارایه کرده و قصد آشنا کردن شما با این مقوله را داشته است. [11]

منابع و مآخذ:

۱) www.prdev.com

۲) www.majidonline.com

۳) www.pardise.net

۴) کتاب آموزش PHP در ۲۴ ساعت نوشته Matt Zandstra

۵) www.PersianTools.com

۶) www.barnamenevis.org

۷) sitepoint.com

۸) www.iranphp.net

۹) http://www.omidpc.com

۱۰) http://www.ccwmagazine.com/CCW مقاله شماره ۱۴

۱۱) "Programming PHP", O'reilly, Kevin, Rasmus and Tatroe, Lerdorf , March 2002

۱۲) http://www.fekrinejat.com فرشاد فکری نجات

۱۳) دیگر کتب و منابع اینترنتی دیگر

۱۴) dev.ir

۱۵) http://www.phpmystery.com