صفرتاصد php

برای همه

(کاملا کاربردی با مثال های متعدد)

فصل ۱: آموزش مقدماتی php

صفر تا صد PHP – مقدمه

PHP یک زبان اسکریپتی سمت سرور (Server Side) می باشد که امروزه بیش از نصفی از وبسایتهای مطرح جهان، از آن برای حفظ و نگهداری داده ها و ایجاد سایتهای دینامیک و به روز استفاده می کنند.

PHP هم همانند تمامی زبانهای برنامه نویسی Server Side، ترجمه شده و در نهایت به صورت کدهای HTML در اختیار کاربر قرار می گیرد. از این رو کد شما، از دید دیگران پنهان می ماند.

اما فرق PHP با دیگر زبانها چیست؟ یکی از مهمترین رقیبهای PHP، توسط مایکروسافت تحت عنوان ASP که در نسخه جدید به ASP.NET تغییر نام داده است، به وجود آمده و هنوز بحث سر آن است که کدامیک از آنها قویتر و مناسب تر است.

در این مقاله نمی خواهم بگویم که کدام یک بهتر است. هر کس عقاید مربوط به خود را دارد. در زیر به مقایسه این دو زبان می پردازم:

- تقریبا در تمامی مراکز فروش هوست (Host)، سرور لینوکس ارزانتر از سرور ویندوز است و از آنجا که برای اجرای صفحات ASP.NET، نیازمند سرور ویندوز همراه با وب سرور IIS هستیم، می بایست پول بیشتری را پرداخت کنیم. این در حالی است که PHP بر روی تمامی Platformها از قبیل ویندوز و لینوکس نصب و اجرا می گردد.
- PHP کاملا Open-Source می باشد.
- به وضوح در اجرای صفحات ASP.NET و PHP دیده می شود که سرعت PHP سریعتر از ASP.NET می باشد.
- برای ایجاد کردن صفحات PHP، هیچ نرم افزار جامعی به بازار ارائه نشده است که بتواند نیازهای کاربران را بر طرف کند (از Dream Weaver می توان تا حدی استفاده کرد)، اما نرم افزار پر قدرت Visual Studio.NET، نرم افزاری جامع برای ایجاد صفحات دینامیک ASP.NET می باشد.
- در ASP.NET می توان از ترکیب زبانهای برنامه نویسی مختلف نظیر VB, C#, C++و غیره استفاده کرد ، اما PHP فقط از همان زبان خودش که PHP است پشتیبانی می کند.
- ASP.NET دارای ویژگی ای تحت عنوان Code Behind می باشد که به برنامه نویس اجازه می دهد کدهای HTML را جدا از کدهای ASP.NET نگهداری کند. این کار سبب آن می شود که در صورت نیاز به رفع مشکل، برنامه نویس مستقیما به فایل مربوط به کدهای ASP.NET برود و با کدهای HTML کاری نداشته باشد. در صورتی که PHP به صورت Default این طور نیست (به وسیله نرم افزارهایی می توان برای PHP هم، این کار را کرد).

همان طور که مشاهده کردید، هر کدام از آنها دارای معایب و مزایایی هستند و انتخاب آنها، کاملا به عهده خود شماست.

تاریخچه مختصری از PHP

فکر اولیه PHP در پاییز سال ۱۹۹۴ توسط Rasmus Lerdorf (rasmus @ php. Net) شکل گرفت.

در ابتدا نگارشی از PHP در صفحه شخصی وی به کار گرفته شد، تا اطلاعاتی از کسانی که رزومه وی را می بینند، نگاه داشته شود. اولین نگارش عمویم آن در اوایل سال ۹۵ ارایه شد و با نام Home Page Tolls Personal معرفی گردید. که البته شامل پارسری بسیار ساده بود که ماکروهای خاصی را می شناخت و نیز برخی کاربردهای مشترک در صفحات شخصی مانند شمارنده یا Guestbook و برخی ابزارهای دیگر را شامل می شد.

پارسر در نیمه سال ۹۵ بازنویسی شد و با نام PHP/FI نگارش ۲ ارائه گردید. FI نام بسته نرم افزاری دیگر از Rasmus بود که فرم های داده HTML را تفسیر می کرد. پس از آن بسیاری از PHP در کدهای خود استفاده کردند. در میانه سال ۹۶ میزان استفاده کنندگان به حدود ۱۵ هزار سایت رسید. این میزان در نیمه سال ۹۷ در حدود ۵۰ هزار سایت مختلف بود. در همین زمان PHP از حالت یک پروژه شخصی درآمد و توسط تیمی توسعه یافت. این گروه نگارش جدیدی از PHP از حالت یک پروژه شخصی درآمد و توسط تیمی توسعه یافت. این گروه نگارش جدیدی از PHP ارائه دادند و پارسر آن را بازنویسی نمودند. PHP4 به سرعت مورد استفاده قرار گرفت. هم اکنون نیز PHP5 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 web Server RedHat's Stronghold ارائه می گردد. هم اکنون برآورد می شود بیش ۵/۱۰۰/۰۰۰ سایت از PHP استفاده کرده اند که میزا ن اندکی از تمامی سایتهای که از سرورهای IIS مایکروسافت استفاده می کنند. (5.03 میلیون) بیشتر است.

چرا از PHP استفاده کنیم؟

گذشته از Open Source بودن آن، دلایل بسیاری برای انتخاب PHP برای ایجاد محتوی محاوره ای بر روی سایت های وب، وجود دارد.

ساختار و ترکیبی بسیار شبیه زبان C دارد.

نوع داده ها و ساختارهای (Structures) ، PHP به آسانی آموخته و به کار گرفته می شوند. PHP می داند منظور شما چیست و نوع داده های داده را، خود تغییر می دهد.

نیازی به دانستن دستور خاصی برای کامپایل برنامه ندارید، برنامه شما خود، در مرورگر اجرا می شود لازم نیست برای ابتدای کار و نوشتن برنامه های کاربردی درباره PHP زیاد اطلاع داشته باشید.

PHP سرویس از مجموعه فایل های کتابخانه ای C را ارایه می دهد که به آسانی درون زبان قرار گرفته و با انعطاف بسیار به ان قابلیت پاسخ دهی سریع برای تغییرات در وب را می دهد.

آنچه شما می توانید با PHP انجام دهید، با دیگر زبانها نیز قابل انجام است. اما PHP برای کار در زمینه وب طراحی شده است. بنابراین کارهایی مشکل و خسته کننده برای برنامه نویسان که نوشتن آن در Prel آنها را به زخمت می انداخت، به آسانی با PHP انجام می شود.

PHP وب سایت ها را قادر می سازد که با سرعت مبهوت کننده ای گسترش یابند، به این خاطر بسرعت برپا صفحات پویا و پشتیبانی پایگاه داده ها بکار گرفته شده است.

کدهای کوچک توکار در یک صفحه وب بسیار کارآمدند. به عنوان مثال، در یک صفحه وب ایستا ممکن است شما مقدار یک متغیر را بدست آورید و سپس آنرا برای تغییرات پویای محتوای صفحه به صفحه به کار برید. این مثال PHP، عبارتی را که نشانگر مرور گر وب کاربر است بر روی صفحه نمایش می دهد.

```
< ? PHP
$browser = getenv ("HTTP _ USER _ AGEENT");
?>
<P> You are using the <?php echo ($ browser); ? > web
</p>
```

چگونه از PHP استفاده کنیم؟

PHP نرم افزاری رایگان و Open sourcc است که می توان آنرا از سایت http: //www. Php. Net دریافت کرد. در این سایت بسیاری از نرم افزارهایی که برای کار با PHP طراحی شده اند معرفی گردیده می توانید از آنها استفاده نمایید. همچنین دستورالعمل استفاده PHP و معرفی دستورات و توابع آن در قالبهای مختلف از جمله Html، PDF و فایل Help ویندوز در دسترس شماست.

البته ابزارهایی مانند «PHP Traid» که یک بسته نرم افزاری رایگان است و در آن علاوه بر مفسر PHP، مفسر زبان Perl، موتور پایگاه داده PHP و پایگاه داده MySQL و نسخه تحت ویندوز وب سرور Apache PHP ، مفسر زبوان Perl، موتور پایگاه داده My SQL و نسخه تحت ویندوز وب سرور Apache قرار گرفته، ابزار مناسبی جهت کار با PHP و پایگاه داده MySQL می باشد و یا نرم افزار PHPED که محیطی دیداری جهت کار با PHP و Perl فراهم می آورد.

صفر تا صد PHP -نصب PHP

به چه چیزی نیاز دارم؟

برای شروع کار با PHP می توانید یکی از موارد زیر را استفاده نمایید:

۱. می توانید از یک وب هاست که PHP و MYSQL را پشتیبانی می کند، استفاده نمایید.
۲. یک web server مانند Apache، روی کامپیوترتان نصب کنید و سپس PHP و MYSQL را نصب نمایید.
۳. می توانید از پکیج هایی مانند Xampp یا Wamp استفاده نمایید.

۱- وب هاستی که PHP را پشتیبانی می کند

اگر سرور شما PHP را پشتیبانی می کند به چیز دیگری نیاز ندارید.

فقط باید یک فایل PHP ایجاد نمایید و آنرا در دایرکتوری وب قرار دهید. سرور بطور اتوماتیک آنرا ترجمه خواهد کرد.

بخاطر اینکه PHP رایگان است و اغلب سرورها آنرا پشتیبانی می کنند نیاز به کامپایل یا نصب ابزار های اضافی ندارید.

به هر حال اگر سرور شما PHP را پشتیبانی نمی کند، باید PHP را نصب نمایید.

۲- نصب جداگانه Web Server و PHP و MYSQL

اگر کامپیوترتان، PHP را پشتیبانی نمی کند باید کارهای زیر را انجام دهید:

- نصب وب سرور (Apache)
- نصب PHP
- نصب یک پایگاه داده، مانند MYSQL

در لینک روبرو، آموزش چگونگی نصب PHP به خوبی توضیح داده شده است: http://php.net/manual/en/install.php

دانلود Apache Server:

برای دانلود رایگان Appache از لینک روبرو استفاده کنید: http://httpd.apache.org/download.cgi

دانلود PHP:

برای دانلود رایگان PHP از لینک روبرو استفاده کنید: http://www.php.net/downloads.php

دانلود MySQL :

برای دانلود رایگان MySQL از لینک روبرو استفاده کنید: http://www.mysql.com/downloads

توجه: به جای نصب جداگانه PHP , MySQL, Apache می توانید از پکیج هایی مانند Xampp یا Wamp استفاده نمایید که کار شما را بسیار ساده تر می نمایند، البته در مطلب IIS & PHP چگونگی اجرای فایلهای PHP روی IIS توضیح داده شده است.

۳- استفاده از Xampp یا Wamp

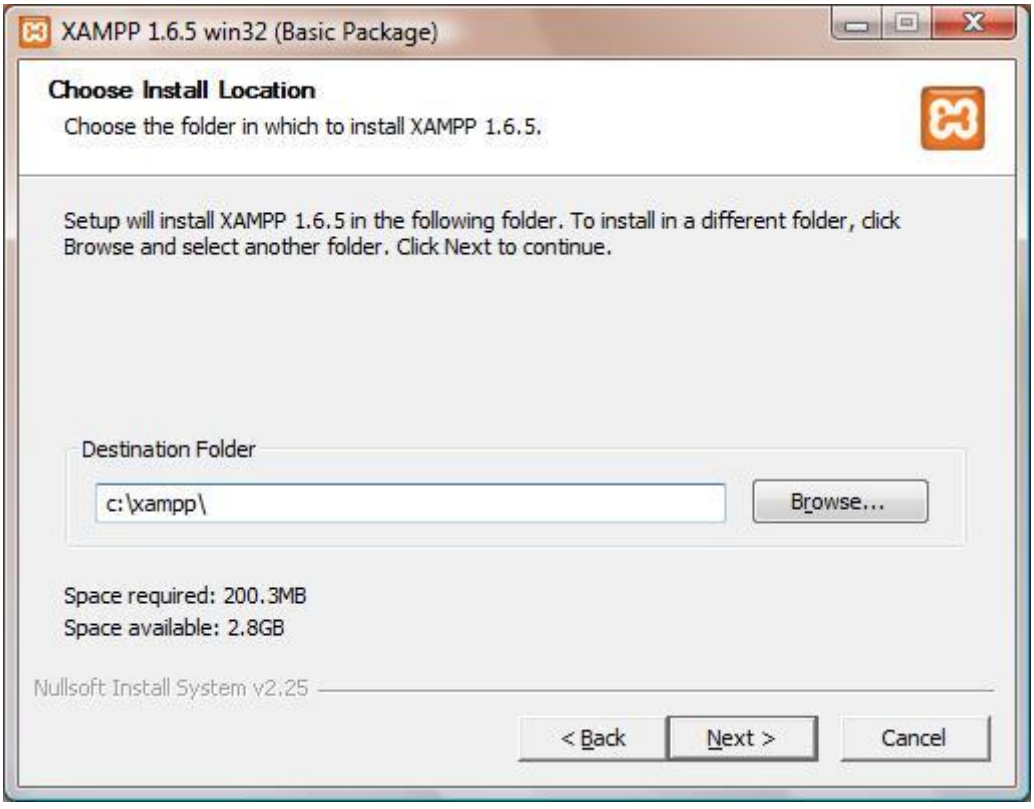
همانطور که قبلاً گفته شد برای شروع کار با php نیاز به نصب مفسر آن داریم که پکیج Xampp برای این منظور مناسب می باشد.

مراحل نصب XAMPP:

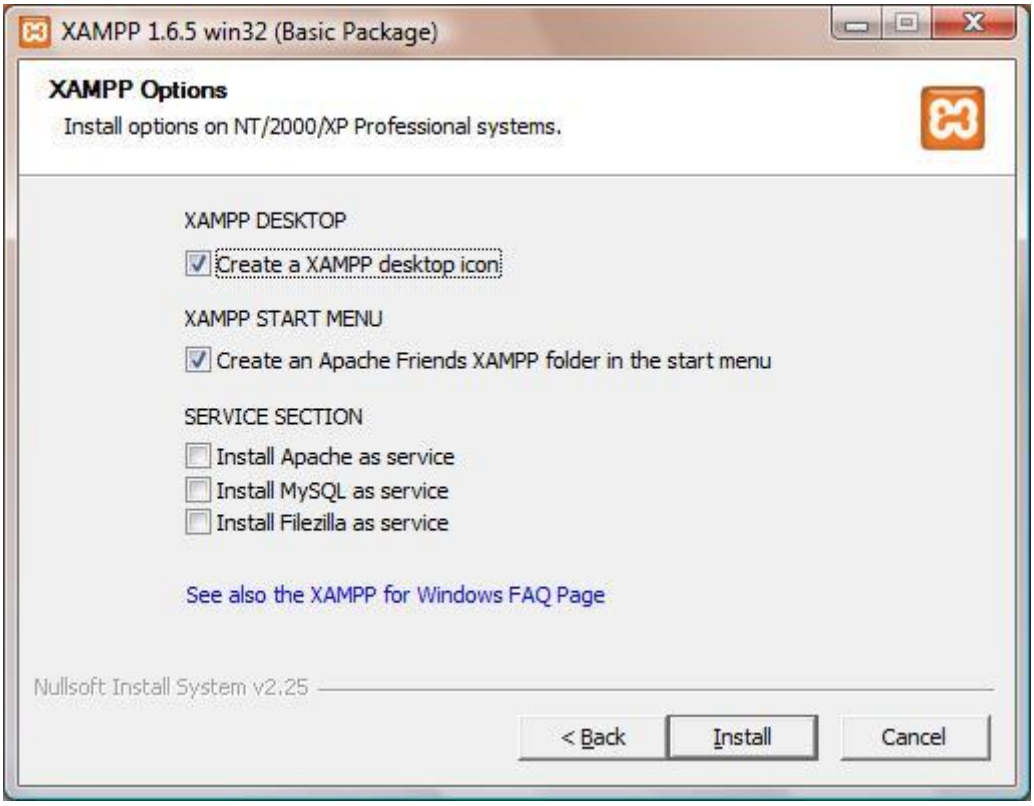
برای دانلود رایگان XAMPP از لینک روبرو استفاده کنید: http://www.apachefriends.org/en/xampp.html

۱- در اولین مرحله، خوش آمدگویی داریم، پس Next را مطابق شکل زیر کلیک کنید.

۲- مسیر نصب را مطابق شکل زیر البته به صورت دلخواه مشخص کنید.



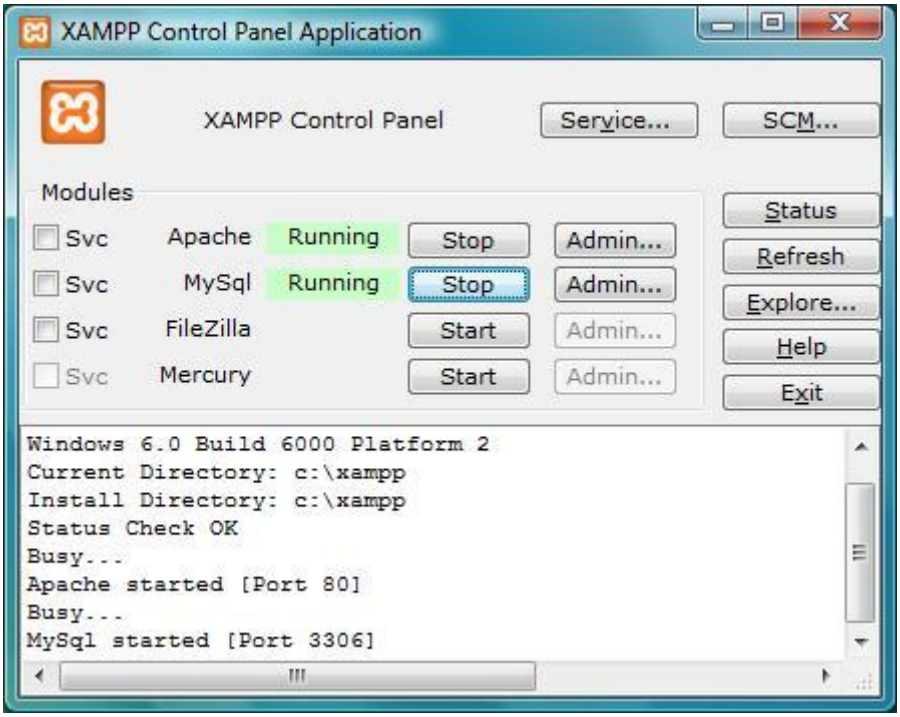
۳- در صورتی که میخواهید Apache و MySQL به عنوان سرویس های ویندوز نصب بشوند گزینه های Install Apache as Service و Install MySQL as Service را تیک بزنید، در غیر این صورت اگر تیک Install as Service را نزده باشید باید هر بار که ویندوز بالا می آید Apache و MySQL را فعال کنید، یعنی باید دکمه Start کنارشون را در کنترل پانل XAMPP کلیک کنید، در نهایت کلید Install را مطابق شکل زیر کلیک کنید.



نکته۱: در صورتیکه IIS روی سیستم شما نصب است و XAMPP را نصب کنید، در نهایت برای اجرای فایل های PHP به مشکل برخواهید خورد، بنابراین باید IIS را Stop کنید.

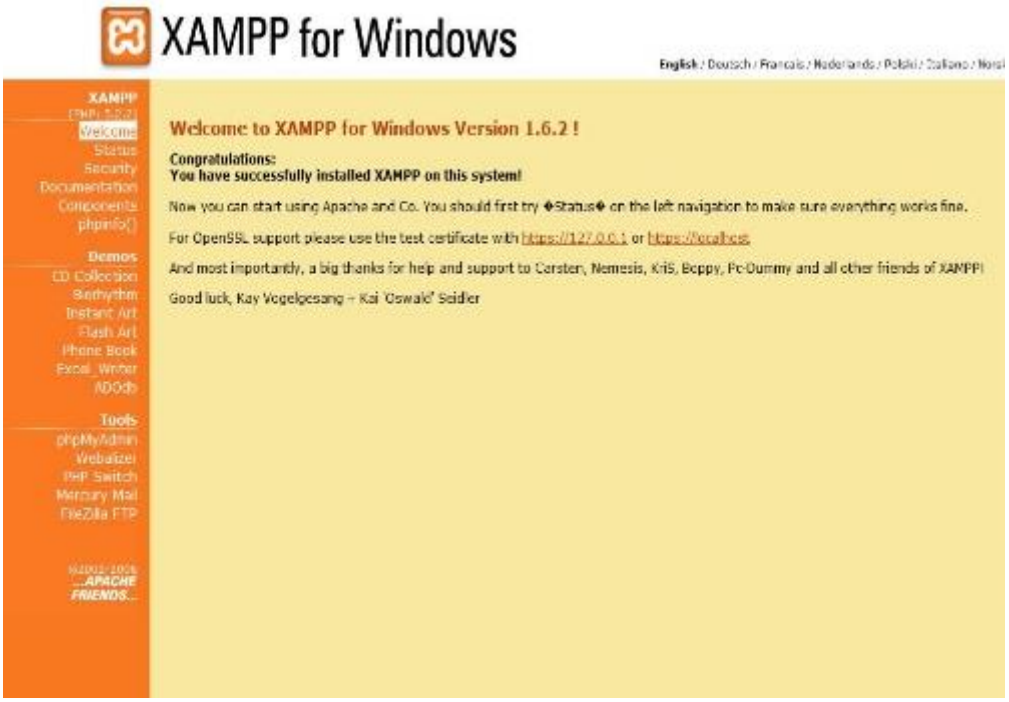
نکته۲: اگر SQL Server روی سیستم شما نصب است باید سرویس Reporting Services Configuration Manager آنرا Stop کنید تا XAMPP به درستی فایلهای PHP را اجرا کند.

در شکل زیر کنترل پنل Xampp را می بینید که می توانید Apache و MySQL را Start یا Stop کنید.



حالا مرورگرتان را باز کنید و در آدرس بار مرورگر localhost یا ۱۲۷.۰.۰.۱ را تایپ کنید و یا می توانید در شکل بالا دکمه Admin را کلیک کنید اگر همه چیز خوب پیش رفته باشه صفحه زیر را خواهید دید که می توانید زبان صفحات Xampp را انتخاب کنید پس روی English کلیک کنید!

در نهایت صفحه ای مانند شکل زیر نمایش داده می شود.



ایجاد و اجرای یک فایل PHP:

۱. یک فایل PHP ساده ایجاد کنید، به عنوان مثال در ادیتور NOTEPAD کد زیر را تایپ کنید و با نام hello و با پسوند php ذخیره کنید(hello.php):

```
<?php
echo "Hello";
?>
```

۲. فایل ایجاد شده در بالا را در مسیر C:\xampp\htdocs کپی کنید. (مسیر نصب XAMPP خواهد بود)

۳. در آدرس بار مرورگر، آدرس localhost/hello.php را وارد کنید.

۴. اگر همه چیز درست پیش رفته باشد، خروجی زیر را خواهید داشت:

```
Hello
```

صفر تا صد PHP -چند مثال

الف) Hello World

سریعترین راه برای آموختن PHP آنست که شروع به کار با آن کنید و ببینید که چه رخ می دهد. ما با مثال معروف " Hello World" که شما پیش از این در آموختن سایر زبانها نیز آنرا به کار برده اید.کار با PHP را آغاز می کنیم. این مثال ساده عبارت «HELLO WORLD» را در مرورگر به وسیله تابع ()echo می نویسد.

```
<? php
Echo ( "Hello World\ n");
?>
```

تابع ( ) echo یک یا چند رشته متنی را به مرورگر برای نمایش می فرستد. در واقع ()echo یک تابع نیست بلکه یک ساختار زبان است. اما مانند یک تابع عمل می کند. به این خاطر نیازی به پرانتزها نیست و می توانید از آنها صرف نظر کنید.

اکنون اجازه دهید به برخی از ترکیبهای دستوری معمول در اسکریپت های PHP بپردازیم. ابتدا و انتهای بخش PHP با علامتهای «کوچکتر از» و «بزرگتر از» نشانه گذاری می شود. که پس و یا پیش از آنها یک علامت سوال آمده است. پایان هر خط با یک «؛» مشخص می شود. « n \» نیز یک خط جدید ایجاد می کند و عبارات پس از آن در خط جدید نوشته می شود.

کد بالا را می توان به این شکل نیز نوشت.

```
<?PHP
echo " Hello World\ n" ;
```

ب) Hello Web

مثال معروف Hello Web را برای وب که پایه محاوره ای آن فرم هاست، باز نویسی می کنیم. این مثالی بسیار ساده از بکار گرفتن اسکریپت، توسط یک فرم در مرورگر است. این اسکریپت یک ورودی از کاربرد «نام وی» را دریافت می کند و در پاسخ این نام را ر پنجره محاوره مرورگر نمایش می دهد.

```
<form action = "hello web. Php3" method = "post">
<td> Name : </td>
<td> <input type = "text" name = " frmName "size= "24">
<input type = "submit"value= "submit">
</ form>
```

Hello – web. Html

چون PHP، مقادیر ارسال شده از فرم را می پذیرد و خود متغییرهایی به نام متغیرهای ورودی می سازد، فقط یک خط برای تولید پاسخ کافیست. و به خاطر اینکه PHP در خود این قابلیت را دارد که به عنوان یک پرونده وب عمل کند، اسکریپت به طور خودکار یک سرآمد (header) HTTP ایجاد می کند. تا با مرورگر مشخص می نماید که چه متنی را نمایش دهد.

Hello < ? php echo ($ frm Name) ; ? > !

Hello – web. Php 3

در ابتدا، PHP سرآمد فایل hTml را به مرورگر می فرستد. سپس تگ P و بعد از آن Hello را نمایش می دهد، هنگامی که به تگ آغازین PHP رسید آن را اجرا و نتیجه را به مرورگر ارسال کرده و بعد از تگ پایانی PHP، به حالت HTML می رود و سایر بخشها را بدون تغییر ارسال می کند.

یک متغیر می تواند از مقادیر ورودی یک فرم، یا پارامتر URL و یا متغیرهای محلی که دارای مقدار هستند، تشکیل شود.

صفر تا صد PHP - برنامه نویسی وب با PHP

در برنامه نویسی هیچ چیز لذت بخش تر از کد نویسی نیست. همین حالا برنامه Notepad خود را باز کرده و کدهای زیر را در آن بنویسید:

```
<html>
<head>
<title>PHP Info</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
</head>

<body>
<?php
phpinfo();
?>
</body>

</html>
```

این فایل را در محل %XAMPP%\htdocs\PHP\phpinfo.php ذخیره کرده و به وسیله مرورگر خود، آدرس http://localhost/PHP/phpinfo.php را اجرا نمایید. با انجام این کار، صفحه ای که مشخصات PHP نصب شده بر روی سیستم را به شما نشان می دهد، نمایان خواهد شد.

آشنایی با PHP

همان طور که در مثال بالا مشاهده کردید، کد زیر با بقیه کدهای HTML فرق داشت:

```
<php
phpinfo();
?>
```

این تکه کدی است که به وسیله PHP نوشته شده است. برای نوشتن کدهای PHP، می بایست آنها را در بین تگهای ?> و <?php یا ?> و <? قرار داد. همچنین یک راه میانبر نیز است که به صورت <?=SOMETHING?> به کار می رود. در این مقاله از همان روش ?> و <?php استفاده می شود.

عبارات (Statements)

به طور کلی در PHP، عبارات به دو دسته زیر تقسیم می شوند:

- تک خطی (Single Line)
- چندخطی (Multi Line)

در انتهای هر عبارت PHP، یک علامت سمی کالن (;) قرار می گیرد. به وسیله این علامت، PHP متوجه می شود که یک عبارت به اتمام رسیده و عبارت دیگری در حال شروع است. به مثالهای زیر توجه کنید:

```
<?php
echo("Hello World!");
echo("Mojtaba");
?>
```

```
<?php
echo("Hellow World"); echo("Mojtaba");
?>
```

هر دو کد بالا صحیح است. چرا که بعد از اتمام هر یک از عبارت PHP، یک علامت سمی کالن قرار داده شده است. برنامه نویسان معمولا از کد بالایی استفاده می کنند. چرا که در آن، هر یک از کدهای PHP در یک خط مجزا قرار گرفته و در نتیجه نگهداری کد آسان تر خواهد شد. حال به کد زیر توجه کنید:

```
<?php
echo("Hello World!")
```

```
echo("Mojtaba")
?>
```

اجرای این کد سبب خطا می شود. چرا که عبارت در آن از هم جدا نشده است و عبارتی همانند عبارت echo("Hello World") برای PHP ناآشناست. echo("Mojtaba")

تا به اینجای کار، عبارت تک خطی مورد بررسی قرار گرفتند و اما عبارات چند خطی. عبارات چندخطی که به کد بلاک (Code Block) معروف است، در PHP توسط آکولادها { } احاطه می شوند. این عبارات با همدیگر اجرا می شوند. به عنوان مثال در یک عبارت شرطی (عبارات شرطی در جلوتر مورد بررسی قرار می گیرند)، برای PHP تعیین می کنیم که اگر شرط درست بود، این سری عبارات و در غیر این صورت، سری دیگری عبارات را اجرا نماید:

```
<?php
if (8 > 5) {
    echo("True");
    echo("8 is greater than 5");
} else {
    echo("False");
    echo("The statement is false");
}
?>
```

فعلا زیاد به معنای این قطعه کد کار نداشته باشید. من فعلا می خواهم به شما ساختار عبارات چند خطی را نشان بدهم. در این قطعه کد اگر شرط درست باشد، دو عبارت بالایی و در غیر این صورت، دو عبارت پایینی اجرا می شوند. همان طور که می بینید، عبارت در بین آکولادها قرار گرفته اند. همچنین در انتهای هر عبارت، یک علامت سمی کالن قرار داده شده است. انجام این کار را هیچ موقع فراموش نکنید: قرار دادن یک سمی کالن در انتهای هر عبارت.

در اینجا بد نیست به نکته دیگری نیز اشاره کنم. شما می توانید کدهای PHP را همراه با کدهای HTML مخلوط کنید. ابتدا به مثال زیر توجه کنید تا درباره آن بیشتر توضیح دهم:

```
<?php
if (8 > 5) {
    echo("8");
?>
<br>is greater than<br>
```

```
<?php
    echo("5");
}
?>
```

همان طور که مشاهده می کنید، در قطعه کد بالا، ما از دو بلاک PHP استفاده کردیم و در بین آنها یک خط کد HTML قرار داده ایم. در این کد همان طور که مشاهده می کنید، از عبارات چند خطی استفاده شده است (به آکولادها دقت داشته باشید). آکولاد آغازین در قطعه کد اول و آکولاد پایانی در قطعه کد دوم قرار داده شده است. خروجی این عبارت به صورت زیر خواهد بود.

```
8
is greater than
5
```

توضیحات(Comments)

همانند توضیحات در HTML، در PHP هم می توان از توضیحات استفاده کرد. به نظر من توضیحات یکی از اصلی ترین بخشهای برنامه نویسی می باشد. به وسیله توضیحات، می توانید بعد از نگاه مجدد به کد نوشته شده، ساختار آن را بفهمید. شاید در نگاه اول با خود بگویید کدی را که خودم نوشتم، دیگر نیازی به توضیحات ندارد. اما فرض کنید شما برنامه ای نوشته اید و هم اکنون، ۱ سال است که دیگر به کدهای آن نگاهی نینداخته اید. در این صورت توضیحات، با ارزشترین چیز برای شما خواهد بود.

توضیحات در PHP به دو نوع توضیحات تک خطی و چند خطی تقسیم می شود. برای قرار دادن توضیحات تک خطی، از دو روش زیر می توانید استفاده کنید:

- استفاده از دابل اسلش (//)
- استفاده از کاراکتر شارپ (#)

همچنین برای قرار دادن توضیحات چندخطی، از توضیحات نوع جاوا یا C++ (که در CSS نیز از آنها استفاده می شود)، استفاده می کنیم. برای این کار توضیحات خود را در بین علامتهای */ و /* قرار می دهیم. همچنین به این نکته دقت داشته باشید که توضیحاتی که در بین کدهای PHP قرار می دهید، پس از اجرا در مرورگر، در Source صفحه به نمایش در نخواهند آمد. به کد زیر توجه کنید:

```
<?php
/* The below statements,
```

```
print "Hello World"
to the browser */
echo("Hello"); // Print "Hello" to the browser
echo("World"); # Print "World" to the browser
?>
```

همچنین می توانید توضیحات را در جاهای دیگر نظیر وسط کدهای PHP نیز، قرار دهید. هر دو قطعه کد زیر درست می باشند.

```
<?php
echo(/*Quantity*/ 2 /*Operation*/ * /*Price*/ 150);
?>
```

همان طور که مشاهده می کنید، تمامی عبارات /*Quantity*/ و /*Operation*/ و /*Price*/ به عنوان توضیح مد نظر گرفته شده اند. علامت * که در بین دو عبارت /*Price*/ و /*Operation*/ قرار گرفته، علامت ضرب می باشد که دو عبارت ۲ و ۱۵۰ را در یکدیگر ضرب می نماید. خروجی این مثال، ۳۰۰ می باشد.

```
<?php
echo(
    2 #Quantity
    * //Operation
    150 #Price
);
?>
```

PHP یک زبان غیرحساس به حروف بزرگ و کوچک (case-insensitive)

در PHP، هیچکدام از توابع، کلاس ها و کلمات کلیدی (مانند if و while و ...) به حروف بزرگ و کوچک، حساس نیستند.

در مثال زیر، هر سه دستور echo، صحیح و یکسان است:

مثال

```
<!DOCTYPE html>
<html>
<body>
<?php
```

```
ECHO "Hello World!<br>";
echo "Hello World!<br>";
EcHo "Hello World!<br>";
?>
</body>
</html>
```

خروجی کد بالا:

```
Hello World!
Hello World!
Hello World!
```

اما، در PHP تمام متغیرها به حروف بزرگ و کوچک حساس اند. (case-sensitive)

در مثال زیر، تنها دستور اول مقدار متغیر $color را نمایش می دهد: (دلیل این اتفاق، این است که با سه متغیر $color و $COLOR و $coLOR مانند سه متغیر متفاوت برخورد می شود)

مثال

```
<!DOCTYPE html>
<html>
<body>

<?php
$color="red";
echo "My car is " . $color . "<br>";
echo "My house is " . $COLOR . "<br>";
echo "My boat is " . $coLOR . "<br>";
?>

</body>
</html>
```

خروجی کد بالا:

```
My car is red
My house is
My boat is
```

صفر تا صد PHP -متغییرها

متغیر ها، ظرفی برای ذخیره اطلاعات هستند.

مثال

```
<?php
$x=5;
$y=6;
$z=$x+$y;
```

```
echo $z;
?>
```

خروجی کد بالا:

```
11
```

آیا جبر را از زمان مدرسه به یاد دارید؟

x=5, y=6, z=x+y

آیا به یاد می آورید که یک حرف (مانند x) می تواند برای نگهداری یک مقدار (مثل ۵) به کار رود و شما می توانید طبق اطلاعات بالا مقدار z را برابر ۱۱ ارزیابی کنید.

این حروف، متغیر نامیده می شوند و آنها را می توان برای نگهداری مقادیر (x=5) و یا عبارات (z=x+y) استفاده کرد.

x=5

y=6

z=x+y

نکته: به متغیرها مانند ظرفی برای نگهدازی داده ها نگاه کنید.

متغییر ها در PHP

- متغییرها برای ذخیره ی مقادیر مانند رشته های متنی، اعداد، یا آرایه ها استفاده می شوند.
- زمانی که یک متغییر را تعریف می کنید، می توانید آنرا در کدتان بارها و بارها استفاده کنید.
- در PHP متغییر ها برخلاف C نوع خاصی ندارند، و از متغییرها بدون تعریف قبلی می توان استفاده نمود.
- هر متغییر با علامت $ در ابتدای آن مشخص می شود.

روشی صحیح برای تعریف یک متغییر در PHP:

```
$var_name = value;
```

توجه: اغلب برنامه نویسان تازه کار در PHP فراموش می کنند که در ابتدای نام متغییر علامت $ را درج کنند، در این صورت برنامه کار نخواهد کرد.

مثال۱: در زیر یک متغییر با محتویات رشته و یک متغییر دیگر با محتویات عدد تعریف شده است:

```
<?php
$txt="Hello World!";
```

```
$x=16;
?>
```

قواعد نامگذاری متغییر ها در PHP

- نام یک متغییر باید با یک حرف یا یک "_" آغاز شود.
- نام یک متغییر می تواند شامل اعداد، حروف کوچک و بزرگ و "_" باشد. (A-Z,a-z,۰-۹)
- نام متغییر ها نباید شامل فضای خالی باشد.
- نام متغیرها به حروف بزرگ و کوچک حساس است. ($y و $Y دو متغیر متفاوت اند)

نکته: بیاد داشته باشید که متغیرها در PHP به حروف کوچک و بزرگ حساس هستند.

ایجاد یا اعلان متغیرها در PHP

در PHP دستوری برای ایجاد متغیرها وجود ندارد.

یک متغیر زمانی که مقداری را به آن اختصاص می دهید ایجاد می شود.

مثال

```
<?php
$txt="Hello world!";
$x=5;
$y=10.5;

echo $txt;
echo "<br>";
echo $x;
echo "<br>";
echo $y;
?>
```

خروجی کد بالا:

```
Hello world!
5
10.5
```

بعد از اجرای مثال بالا، متغیر txd مقدار "Hello world!"، متغیر x مقدار "۵" و متغیر y مقدار "۱۰.۵" را در خود نگهداری می کنند.

توجه: زمانی که می خواهید یک مقدار رشته ای را به یک متغیر اختصاص دهید از علامت ' استفاده نمایید.

PHP یک زبان بی ربط و بی قاعده (Loosely Type Language)

در PHP یک متغییر نیازی به تعریف پیش از استفاده ندارد.

در مثال بالا، می بینید که مجبور نیستید به PHP بگویید که متغییر از چه نوع داده ای است، PHP به طور خودکار متغییر ها را به نوع داده ای مناسب شان تبدیل می کند که بستگی دارد چگونه مقدار دهی شده باشند.

در یک زبان برنامه نویسی قوی (مانند C یا Java) شما باید نوع و اسم متغییر را پیش از استفاده تعیین کنید. در PHP متغییر ها زمانی که شما از آنها استفاده می کنید به طور خود کار تعریف می شوند.

محدوده متغیرها در PHP

در PHP، متغیرها در هرجایی از کد می توانند اعلان شوند.

منظور از محدوده، بخشی از اسکریپت است که متغیر می تواند در آن محدوده استفاده شود.

سه محدوده مختلف برای متغیرها وجود دارد:

- local
- global
- static

متغیر های محلی (local) در PHP

متغیرهایی که داخل یک تابع اعلان شده اند، تنها داخل همان تابع، قابل دسترسی اند. (متغیرهای با قلمرو محلی)

متغیرهای محلی به محض اینکه عملیات تابع تکمیل شد، حذف می شوند.

> **نکته:** شما می توانید در توابع مختلف، متغیرهای محلی با نام های یکسان داشته باشید. بخاطر اینکه متغیرهای محلی، تنها بوسیله تابعی که آنها را اعلان کرده است شناخته می شوند.

در بخش های بعدی، درباره توابع بیشتر خواهید آموخت.

متغیر های عمومی (global) در PHP

متغیرهایی که خارج از توابع اعلان شده اند، عمومی می شوند، و در تمام اسکریپت بجز توابع، به آنها دسترسی خواهد بود.

زمانی که یک صفحه وب را ببندید، متغیرهای عمومی حذف خواهند شد.

در مثال زیر، می توانید محدوده متغیرهای محلی و عمومی را مشاهده نمایید:

مثال

```
<?php
$x=5; // global scope

function myTest()
{
$y=10; // local scope
echo "<p>Test variables inside the function:<p>";
echo "Variable x is: $x";
echo "<br>";
echo "Variable y is: $y";
}

myTest();

echo "<p>Test variables outside the function:<p>";
echo "Variable x is: $x";
echo "<br>";
echo "Variable y is: $y";
?>
```

خروجی کد بالا:

```
Test variables inside the function:
Variable x is:
Variable y is: 10

variables outside the function:
Variable x is: 5
Variable y is:
```

در مثال بالا، دو متغیر $x و $y و یک تابع myTest() وجود دارد. متغیر $x، بخاطر اینکه خارج از تابع اعلان شده، عمومی است و متغیر $y، بخاطر اینکه داخل تابع اعلان شده، محلی است.

زمانی که داخل تابع myTest()، مقادیر متغیرها را چاپ می کنیم، مقدار متغیر $y بخاطر اینکه محلی است چاپ می شود، اما مقدار متغیر $x، چونکه خارج از تابع اعلان شده است نمی تواند چاپ شود.

سپس، زمانی که مقادیر متغیرها را خارج از تابع myTest() چاپ می کنیم، مقدار متغیر $x چاپ می شود، اما مقدار متغیر $y، چونکه داخل تابع mytest() اعلان شده است نمی تواند چاپ شود.

کلمه کلیدی global در PHP

با استفاده از کلمه کلیدی global، می توانید به متغیرهای عمومی در داخل یک تابع دسترسی داشته باشید.

برای انجام این کار، از کلمه کلیدی global، قبل از نام متغیر استفاده نمایید:

مثال

```
<?php
$x=5;
$y=10;

function myTest()
{
global $x,$y;
$y=$x+$y;
}

myTest();
echo $y; // outputs 15
?>
```

خروجی کد بالا:

```
15
```

در PHP، با استفاده از آرایه GLOBALS$ نیز می توانید به متغیرهای عمومی دسترسی داشته باشید. در واقع تمام متغیرهای عمومی در این آرایه ذخیره می شوند و با استفاده از نام متغیر بعنوان ایندکس آرایه می توانید به متغیرها، دسترسی داشته باشید.

بازنویسی مثال بالا، با استفاده از آرایه GLOBALS$:

مثال

```
<?php
$x=5;
$y=10;

function myTest()
{
$GLOBALS['y']=$GLOBALS['x']+$GLOBALS['y'];
}

myTest();
echo $y; // outputs 15
?>
```

خروجی کد بالا:

```
15
```

کلمه کلیدی static در PHP

بطور معمول زمانی که اجرای یک تابع به پایان می رسد، تمام متغیرهای آن نیز حذف خواهند شد. اما بعضی مواقع نیاز می شود که یک متغیر محلی برای استفاده های بعدی حذف نشود.

برای انجام این کار، از کلمه کلیدی static در ابتدای اعلان متغیر، استفاده نمایید:

مثال

```
<?php

function myTest()
{
static $x=0;
echo $x;
echo "<br />";
$x++;
}

myTest();
myTest();
myTest();

?>
```

خروجی کد بالا:

```
0
1
2
```

هر زمان که تابع صدا زده می شود، آخرین مقدار متغیر static، حفظ می شود.

توجه: در مثال بالا، متغیر $x محلی است و فقط در داخل تابع به آن دسیرسی وجود دارد.

صفر تا صد PHP -دستور echo/print

در PHP دو روش اصلی برای چاپ وجود دارد: echo و print

تقریباً در هر مثالی از این کتاب از echo (و print) استفاده شده است. بنابراین در این مطلب سعی شده است تا در این باره مقداری بیشتر توضیح داده شود.

دستور echo و print در PHP

تفاوت های echo و print:

- echo: یک یا بیشتر از یک آرگومان را می گیرد و مقداری را برنمی گرداند. برای اتصال دو رشته هم می توان از "." و هم از "," استفاده کرد.

- print: تنها یک آرگومان را می گیرد و همیشه مقدار ۱ را برمی گرداند. برای اتصال دو رشته فقط می توان از "." استفاده کرد.

شباهت های echo و print:

- هر دو تابع نیستند.
- جزئی از ساختار زبانی هستند و می توان از آنها با یا بدون پرانتز استفاده نمود.

نکته: echo سرعت بیشتری نسبت به print دارد.

دستور echo در PHP

نمایش رشته ها

در مثال زیر، نحوه نمایش رشته های مختلف با استفاده از دستور echo نشان داده شده است: (همچنین توجه داشته باشید که رشته چاپ شده می تواند شامل تگ های HTML باشد)

مثال

```
<?php
echo "<h2>PHP is fun!</h2>";
echo "Hello world!<br>";
echo "I'm about to learn PHP!<br>";
echo "This", " string", " was", " made", " with multiple parameters.";
?>
```

خروجی کد بالا:

!PHP is fun! Hello world

!I'm about to learn PHP

.This string was made with multiple parameters

نمایش متغیرها

در مثال زیر، نحوه ی چاپ رشته ها و متغیر ها با دستور echo نشان داده شده است:

مثال

```
<?php
$txt1="Learn PHP";
$txt2="Example.com";
$cars=array("Volvo","BMW","Toyota");

echo $txt1;
echo "<br>";
```

```
echo "Study PHP at $txt2";
echo "My car is a {$cars[0]}";
?>
```

خروجی کد بالا:

```
Learn PHP
Study PHP at Example.com
My car is a Volvo
```

توجه: اگر متغییری درون علامت " قرار بگیرد مقدار متغییر به جای آن قرار می گیرد. اما اگر درون علامت ' قرار بگیرد مقدار متغییر جایگزین نمی شود.

دستور print در PHP

نمایش رشته ها

در مثال زیر، نحوه نمایش رشته های مختلف با استفاده از دستور print نشان داده شده است: (همچنین توجه داشته باشید که رشته چاپ شده می تواند شامل تگ های HTML باشد)

مثال

```
<?php
print "<h2>PHP is fun!</h2>";
print "Hello world!<br>";
print "I'm about to learn PHP!";
?>
```

خروجی کد بالا:

!PHP is fun! Hello world

!I'm about to learn PHP

نمایش متغیرها

در مثال زیر، نحوه ی چاپ رشته ها و متغیر ها با دستور print نشان داده شده است:

مثال

```
<?php
$txt1="Learn PHP";
$txt2="Example.com";
$cars=array("Volvo","BMW","Toyota");

print $txt1;
print "<br>";
print "Study PHP at $txt2";
```

```
print "My car is a {$cars[0]}";
?>
```

خروجی کد بالا:

```
Learn PHP
Study PHP at Example.com
My car is a Volvo
```

صفر تا صد PHP -انواع داده ها

رشته ها، اعداد صحیح، اعداد با ممیز شناور، بولین، آرایه ها، اشیاء، تهی (NULL)

رشته ها در PHP

رشته، یک توالی از کاراکترهاست، مانند "Hello world!"

هر متنی داخل کوتیشن، می تواند رشته باشد. می توانید از کوتیشن (') یا دابل کوتیشن (") استفاده کنید:

مثال

```
<?php
$x = "Hello world!";
echo $x;
echo "<br>";
$x = 'Hello world!';
echo $x;
?>
```

خروجی کد بالا:

```
Hello world!
Hello world!
```

اعداد صحیح در PHP

اعداد بدون ممیز، صحیح هستند.

قوانین اعداد صحیح:

- یک عدد صحیح، حداقل باید یک رقم داشته باشد.
- یک عدد صحیح، نمی تواند شامل کاما یا فاصله باشد.
- یک عدد صحیح، نباید ممیز داشته باشد.
- یک عدد صحیح، می تواند منفی یا مثبت باشد.

- یک عدد صحیح، می تواند در سه فرمت مشخص شود: decimal (بر مبنای ۱۰)، hexadecimal (بر مبنای ۱۶ - با پیشوند ۰x) و octal (بر مبنای ۸ - با پیشوند ۰)

در مثال زیر، فرمت های مختلف اعداد صحیح نشان داده شده است. با استفاده از تابع var_dump()، می توانید مقدار و نوع متغیر را مشخص نمایید:

مثال

```
<?php
$x = 5985;
var_dump($x);
echo "<br>";
$x = -345; // negative number
var_dump($x);
echo "<br>";
$x = 0x8C; // hexadecimal number
var_dump($x);
echo "<br>";
$x = 047; // octal number
var_dump($x);
?>
```

خروجی کد بالا:

```
int(5985)
int(-345)
int(140)
int(39)
```

اعداد با ممیز شناور در PHP

اعداد با ممیز شناور، اعدادی هستند که شامل یک ممیز هستند یا اعدادی که در قالب نماد ریاضی نشان داده می شوند.

در مثال زیر، اعداد اعشاری در فرمت های مختلف، نشان داده شده است. با استفاده از تابع var_dump()، می توانید مقدار و نوع متغیر را مشخص نمایید:

مثال

```
<?php
$x = 10.365;
var_dump($x);
echo "<br>";
$x = 2.4e3;
var_dump($x);
echo "<br>";
$x = 8E-5;
```

```
var_dump($x);
?>
```

خروجی کد بالا:

```
float(10.365)
float(2400)
float(8.0E-5)
```

داده های Boolean در PHP

مقدار داده های Boolean می تواند TRUE یا FALSE باشد.

```
$X=true;
$y=false;
```

داده ی Boolean، معمولاً در تست مشروط استفاده می شود. در بخشهای بعدی درباره ی تست مشروط بیشتر خواهید آموخت.

آرایه ها در PHP

آرایه ها انواع خاصی از متغیرها به حساب می آیند که می توانند چندین داده را در قالب یک نام ذخیره کنند.

در مثال زیر، ابتدا یک آرایه ایجاد شده و سپس با استفاده از تابع var_dump()، مقدار و نوع هر سلول آرایه مشخص شده است:

مثال

```
<?php
$cars=array("Volvo","BMW","Toyota");
var_dump($cars);
?>
```

خروجی کد بالا:

```
array(3) { [0]=> string(5) "Volvo" [1]=> string(3) "BMW" [2]=> string(6) "Toyota" }
```

درباره آرایه ها در بخشهای بعدی توضیحات بیشتری را خواهید دید.

اشیاء (object) در PHP

یک شیء نوع داده ای است که هم داده ها و هم اطلاعات مربوط به نحوه پردازش آنها را ذخیره می کند.

یک شیء در PHP، باید بطور صریح اعلان شود.

ابتدا باید کلاس شیء اعلان شود، برای این کار، از کلمه کلیدی class استفاده نمایید. یک کلاس، ساختاری است که می تواند شامل چندین property و method باشد.

برای دسترسی به property و methodهای یک کلاس، باید از آن کلاس یک نمونه بسازید:

مثال

```
<?php
class My_class
{
  public $name="Amir";
  function SayHello()
  {
   print "Hello My Name is $this->name";
  }
}

$obj=new My_Class();
$obj->SayHello();
?>
```

خروجی کد بالا:

```
Hello My Name is Amir
```

در بخشهای بعدی، درباره ی اشیاء بیشتر توضیح داده خواهد شد.

مقدار NULL در PHP

با استفاده از مقدار NULL، می توان نشان داد که یک متغیر مقدار ندارد. NULL تنها مقدار ممکن از نوع داده NULL است.

مقدار NULL، خالی یا پر بودن یک متغیر را نشان می دهد. همچنین بهتر است بدانید که در پایگاه داده بین NULL و رشته خالی تفاوت وجود دارد.

می توان متغیرها را با تنظیم مقدار NULL خالی کرد:

مثال

```
<?php
$x="Hello world!";
$x=null;
var_dump($x);
?>
```

خروجی کد بالا:

```
NULL
```

صفر تا صد PHP -رشته ها

متغیرهای رشته ای در PHP

متغیر های رشته ای برای ذخیره مجموعه ای از کاراکترها استفاده می شود.

در این فصل قصد داریم متداولترین توابعی که برای دستکاری رشته ها(String) در PHP استفاده می شود را نشان دهیم.

بعد از اینکه یک متغیر رشته ای(String Variable) ایجاد کردید می توانید آنرا دستکاری کنید، یک رشته می تواند مستقیماً در یک تابع استفاده شود یا می توانید آنرا در یک متغیر ذخیره کنید.

در کد PHP زیر کلمه "سلام" را به یک متغیر رشته ای(String Variable) بنام $TXT انتساب داده و سپس آنرا چاپ می کنیم:

```
<? php
$TXT = "سلام";
echo $TXT;
?>
```

خروجی کد بالا:

```
سلام
```

حالا اجازه دهید تا بعضی از توابع و عمل کننده هایی(Operators) که برای دستکاری رشته ها بکار می رود را نشان دهیم:

عمل کننده الحاق(نقطه)

در PHP تنها یک عمل کننده برای رشته ها وجود دارد، و برای الحاق دو رشته بکار می رود، در کد زیر دو رشته "سلام" و "روز خوبی داشته باشید" به هم الحاق شده اند:

```
<? php
echo "سلام" . " " . "روز خوبی داشته باشید";
?>
```

خروجی کد بالا:

```
سلام روز خوبی داشته باشید
```

تابع ()strlen

این تابع برای بازگرداندن طول یک رشته استفاده می شود.

```
<?php
echo strlen("سلام!");
echo "<br />"
```

```
echo strlen("example.com");
?>
```

خروجی کد بالا:

```
9
12
```

توجه: در خط اول مثال بالا، چون در رشته "!سلام" از کاراکترهای فارسی استفاده شده است و به ازای هر کاراکتر فارسی ۲ بیت فضا در نظر گرفته می شود، خروجی تابع ()strlen به ازای رشته مذکور ۹ خواهد شد. یعنی ۸ بیت برای رشته "سلام" و ۱ بیت برای کاراکتر "!" در نظر گرفته خواهد شد.

تابع ()strpos

این تابع برای پیدا کردن مکان یک کاراکتر یا یک کلمه در یک رشته استفاده می شود.

```
<?php
echo strpos("example.com","m");
?>
```

خروجی کد بالا:

```
4
```

توجه: همانطور که مشاهده می شود کاراکتر "m" در مکان ۴ یافت شد. توجه داشته باشید که اولین مکان در رشته از ۰ شروع می شود و نه ۱.

صفر تا صد PHP -مقادیر ثابت

ثابت ها مانند متغیرها هستند با این تفاوت که یکبار آنها را تعریف می کنید و دیگر نمی توانید آنها را تغییر دهید.

ثابت ها (Constant) در PHP

یک ثابت، در واقع یک شناسه یا یک نام برای یک مقدار ساده است. در طول اسکریپت، مقدار ثابت نمی تواند تغییر نماید.

نام یک ثابت باید با یک حرف یا یک "_" آغاز شود. به علامت $ قبل از نام ثابت نیازی نیست.

توجه: برخلاف متغیرها، یک ثابت بطور اتوماتیک در طول تمام اسکریپت عمومی است.

تنظیم یک ثابت در PHP

با استفاده از تابع define()، می توانید یک ثابت را تنظیم نمایید. این تابع سه پارامتر می گیرد: پارامتر اول، نام ثابت را تعیین می کند، پارامتر دوم، مقدار ثابت و پارامتر اختیاری سوم، تعیین می کند که آیا نام ثابت به حروف بزرگ و کوچک حساس است یا خیر. (مقدار پیشفرض false است به این معنی که نسبت به حروف کوچک و بزرگ حساس است)

در مثال زیر، یک ثابت با نام GREETING و مقدار "Welcome to Example.com!" تعریف شده است:

مثال

```
<?php
define("GREETING", "Welcome to Example.com!");
echo GREETING;
?>
```

خروجی کد بالا:

```
Welcome to Example.com!
```

در مثال زیر، نام ثابت به حروف بزرگ و کوچک حساس نیست (greeting و GREETING یکسان است):

مثال

```
<?php
define("GREETING", "Welcome to Example.com!", true);
echo greeting;
echo GREETING;
?>
```

خروجی کد بالا:

```
Welcome to Example.com!
Welcome to Example.com!
```

صفر تا صد PHP -عملگرها

عملگرها در PHP

عملگرها برای انجام عملیات و محاسبات روی مقادیر استفاده می شوند، در زیر لیستی از عملگرهای مختلف که در PHP استفاده می شود، آورده شده است:

عملگرهای ریاضی

این عملگرها عملی را روی دو متغییر یا دو عدد انجام می دهند.

عملگر	توضیحات	مثال	نتیجه
+	جمع	x=2 x+2	۴
-	تفریق	x=2 5-x	۳

*	ضرب	x=4 x*5	۲۰
/	تقسیم	۱۵/۵ ۵/۲	۳ ۲.۵
%	باقیمانده تقسیم	۵%۲ ۱۰%۸ ۱۰%۲	۱ ۲ ۰
++	افزایش به میزان یک واحد	x=5 x++	x=6
--	کاهش به میزان یک واحد	x=5 x--	x=4

عملگرهای انتسابی

عملگر	مثال	هم ارز با
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
.=	x.=y	x=x.y
%=	x%=y	x=x%y

عملگرهای مقایسه ای

این عملگرها مقایسه ای بین دو متغییر انجام می دهند و نتیجه را به صورت true یا false برمی گردانند.

عملگر	توضیحات	مثال
==	برابری	5==8 returns false
!=	نابرابری	5!=8 returns true
>	بزرگتر از	5>8 returns false

<	کوچکتر از	5<8 returns true
>=	بزرگتر یا مساوی با	5>=8 returns false
<=	کوچکتر یا مساوی با	5<=8 returns true

عملگرهای منطقی

این عملگرها با مقدارهای true و false کار می کنند و آنها را با هم ادغام می کنند.

عملگر	توضیحات	مثال
&&	And	x=6 y=3 (x < 10 && y > 1) returns true
||	Or	x=6 y=3 (x==5 || y==5) returns false
!	not	x=6 y=3 !(x==y) returns true

عملگرهای رشته ای

در رشته ها تنها عملگری که استفاده می شود نقطه "." می باشد که دو رشته را به همدیگه متصل می کند.

عملگر	توضیحات	مثال
.	برای الحاق دو رشته استفاده می شود	'hello'.'world' return 'hello world'

عملگرهای بیتی

این عملگرها بر روی بیت های یک متغییر عملی را انجام میدهند و بیتها را به نسبت عملگر برمیگردانند. اگر متغییرها رشته هستند بر روی کدهای ASCII آنها عمل می کند.

عملگر	توضیحات	مثال	نتیجه
~	not	~$a;	بیت هایی را برمی گرداند که در $a نیستند.
&	and	$a & $b;	بیت هایی را برمی گرداند که هم در $a و هم در $b هستند.
|	or	$a | $b;	بیت هایی را که در $a یا در $b هستند برمی گرداند.
^	xor	$a ^$b;	بیت هایی را برمی گرداند که در $a یا در $b هستند اما در هر دو نیستند.
<<	شیفت به چپ	$a << $b;	بیت های $a را به اندازه $b تا به سمت چپ انتقال می دهد.

>>	شیفت به راست	$a >> $b;	بیت های $a را به اندازه $b تا به سمت راست انتقال می دهد.

صفر تا صد PHP -جملات شرطی

جملات شرطی

اغلب اوقات هنگام نوشتن یک اسکریپت، شما نیاز دارید که تصمیم های متفاوتی در مقابل نتایج مختلف بگیرید، برای تحقق این موضوع از جملات شرطی استفاده می کنیم.

انواع جملات شرطی در PHP:

۱. **if ...:** هنگامی که شرط درست باشد، دستور مقابل if اجرا می شود.
۲. **if ... else:** اگر شرط درست باشد دستور مقابل if وگرنه دستور مقابل else اجرا می شود.
۳. **if ... elseif ... else:** برای اجرای یک دستور از بین چند دستور کاربرد دارد.
۴. **PHP Switch:** برای انتخاب و اجرای یک دستور از بین چند دستور استفاده می شود.

دستور if

نحوه استفاده:

```
if (Condition)
{
Statement 1
Statement 2
...
}
```

دستور(Statement) مورد نظر تنها موقعی اجرا می شود که شرط(Condition) برقرار باشد، یعنی مقدار آن برابر true باشد.

مثال: در کد PHP زیر اگر امروز جمعه باشد، جمله "Have a nice weekend" چاپ می شود.

```
<html>
<body>

<?php
$d=date("D");
if ($d=="Fri") echo "Have a nice weekend!";
?>

</body>
</html>
```

دستور if...else

این دستور که در واقع می تواند کامل کننده if باشد، تعیین می کند که اگر شرط درست نبود چه دستوری اجرا شود.

نحوه استفاده:

```
if (condition)
  code to be executed if condition is true;
else
  code to be executed if condition is false;
```

مثال: در مثال زیر اگر امروز جمعه باشد جمله "Have a nice weekend" چاپ می شود وگرنه جمله "Have a nice day" چاپ خواهد شد.

```
<html>
<body>

<?php
$d=date("D");
if ($d=="Fri")
  echo "Have a nice weekend!";
else
  echo "Have a nice day!";
?>

</body>
</html>
```

دستور if...elseif....else

همان طور که در بالا گفته شد این نوع if برای اجرای یک دستور از بین چند دستور کاربرد دارد.

نحوه استفاده:

```
if (condition)
  code to be executed if condition is true;
elseif (condition)
  code to be executed if condition is true;
else
  code to be executed if condition is false;
```

مثال: در مثال زیر اگر امروز جمعه باشد جمله "Have a nice weekend" چاپ می شود و اگر یکشنبه بود جمله " Have a nice Sunday" چاپ می شود وگرنه جمله "Have a nice day" چاپ خواهد شد.

```
<html>
<body>

<?php
$d=date("D");
if ($d=="Fri")
  echo "Have a nice weekend!";
elseif ($d=="Sun")
  echo "Have a nice Sunday!";
else
  echo "Have a nice day!";
?>

</body>
</html>
```

توجه: نوع دیگر اجرای دستورات کنترلی(مثال):

```
if ($i == 1):
    echo "i is 1";
    $i+=2;
elseif ($i == 2):
    echo "i is 2";
    $i+=3;
else:
    echo "is not 1 or 2";
    $i-=1;
endif;
```

صفر تا صد PHP -دستور switch

دستور Switch

برای انتخاب و اجرای یک دستور از بین چند دستور استفاده می شود.

این دستور در برخی موارد شباهت بسیار زیادی به دستورات if ... elseif ... else دارد.

دستور switch یک حالت خاص هم دارد به نام default که اگر هیچکدام از مقادیر درست نبود، آن قسمت اجرا می شود.

نحوه استفاده:

```
switch (n)
{
case label1:
code to be executed if n=label1;
break;
case label2:
```

```
code to be executed if n=label2;
break;
default:
code to be executed if n is different from both label1 and label2;
}
```

کد بالا چگونه کار می کند:

- **case**

از کلمه case بجای if else استفاده می کنیم و حالت مختلف شرط را در جلوی آن می نویسیم و سپس علامت ":" می گذاریم، در حقیقت اگر متغییر ما یعنی n برابر با مقدار مقابل case بود، نتیجه این می شود که باید دستورات مربوط به آن case اجرا شود، در غیر اینصورت case بعدی مقایسه می شود.

- **break**

دستور break یکی از بخشهای switch است که به مفسر PHP می فهماند که در کجا، قسمت case تمام می شود، در حقیقت از تداخل قسمتهای مختلف جلوگیری می کند. دقت کنید که شما هم باید آنرا قبل از شروع case بعدی بنویسید.

- **default**

اگر هیچکدام از مقادیر مقابل caseها با متغییر n برابر نبود، در این صورت دستورات مربوط به default اجرا می شود.

مثال:

```
<html>
<body>

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

```
?>

</body>
</html>
```

خروجی کد بالا:

```
Number 3
```

صفر تا صد PHP -آرایه ها

آرایه ها در PHP

آرایه ها انواع خاصی از متغیرها به حساب می آیند که می توانند چندین داده را در قالب یک نام ذخیره کنند.

اگر لیستی از آیتمها (برای مثال یک لیست از نام ماشین ها) را داشته باشیم و بخواهیم آنها را در متغیر ها ذخیره کنیم،

چیزی شبیه زیر خواهیم داشت:

```
$cars1="Samand";
$cars2="Volvo";
$cars3="BMW";
```

حالا:

- اگر لیست شما بیشتر از ۳ آیتم باشد مثلاً ۳۰۰تا چه کار می کنید.
- اگر در این لیست به دنبال یک ماشین خاص باشید چه کار می کنید.

در اینجا بهترین راه حل استفاده از آرایه ها است.

یک آرایه می تواند مقادیر متغیرها را تحت یک نام برای شما نگه دارد. و شما از طریق نام آرایه می توانید به مقادیر دسترسی داشته باشید.

هر آیتم در آرایه ایندکس منحصر به فردی برای خود دارد که به راحتی از طریق ایندکس می توانید به مقادیر دسترسی پیدا کنید.

در PHP سه نوع آرایه وجود دارد:

- **آرایه عددی (Indexed array):** منظور از عددی ایندکس آرایه است، یعنی یک آرایه با ایندکس عددی
- **آرایه انجمنی (Associative array):** یک آرایه که بجای ایندکس عددی از یک نام یا مقدار برای ایندکس گذاری استفاده کرده است.
- **آرایه چند بعدی (Multidimensional array):** یک آرایه که مقادیر هر سلول آن آرایه ای دیگر است.

آرایه عددی

در یک آرایه عددی مقادیر هر سلول آرایه با یک ایندکس عددی مشخص می شود.

دو روش برای ایجاد چنین آرایه ای وجود دارد:

۱. در مثال زیر ایندکس به صورت اتوماتیک ساخته می شود(ایندکس از ۰ شروع می شود):

```
$cars=array("Samand","Volvo","BMW","Toyota");
```

۲. در مثال زیر به صورت دستی می توانید ایندکس را بسازید:

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

در مثال زیر بعد از مقداردهی آرایه شما می توانید به مقادیر هر سلول به وسیله نام و ایندکس آرایه دسترسی پیدا کنید:

```
<?php
$cars[0]="Saab";
$cars[1]="Volvo";
$cars[2]="BMW";
$cars[3]="Toyota";
echo $cars[0] . " and " . $cars[1] . " are Swedish cars.";
?>
```

خروجی کد بالا:

```
Saab and Volvo are Swedish cars.
```

توجه: با استفاده از تابع ()count، می توانید تعداد سلول های یک آرایه را تعیین نمایید.

آرایه های انجمنی

در یک آرایه انجمنی ایندکس هر سلول از آرایه، با یک نام یونیک مشخص می شود.

زمان ذخیره مقادیر سلولها باید یک نام مشخص و یونیک به آن سلول اختصاص دهید.

مثال ۱: در این مثال یک آرایه ای که سن افراد مختلف را نشان می دهد، تعریف شده است:

```
$ages = array("Ali"=>32, "Reza"=>30, "Amir"=>34);
```

مثال۲: این مثال مانند قبلی است اما می تواند نشان دهد که چگونه می توانیم به روش های مختلف یک آرایه را تعریف کنیم:

```
$ages['Ali'] = "32";
$ages['Reza'] = "30";
$ages['Amir'] = "34";
```

در زیر نشان داده شده که چگونه از طریق نام و ایندکس آرایه، توانسته ایم به محتویات هر سلول آرایه دسترسی پیدا کنیم:

```
<?php
$ages['Ali'] = "32";
$ages['Reza'] = "30";
$ages['Amir'] = "34";

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

خروجی کد بالا:

```
Ali is 32 years old.
```

آرایه های چند بعدی

هریک از عناصر آرایه در php می توانند از هر نوعی باشند پس می توانیم آرایه را نیز به عنوان عضو عناصر در نظر بگیریم. بنابراین به زبان ساده تر می توانیم یک آرایه داخل آرایه دیگر تعریف کنیم و داخل آن نیز یک آرایه دیگر و داخل آن نیز و به همین ترتیب. تعریف آرایه چندبعدی در زبان php به سادگی آرایه های معمولی است.

مثال: در این مثال یک آرایه چند بعدی تعریف شده که ایندکس آن به صورت اتوماتیک ایجاد می شود:

```
$families = array(array("Ali","Reza","Sara"),
                  array("Amir"),
                  array("Poya","Parniya")
                 );
```

اگر مایل باشید می توانید آرایه بالا را به صورت زیر تعریف کنید: (ایندکسِ مقداری)

```
$families = array("Ahmadi"=>array("a"=>"Ali","b"=>"Reza","c"=>"Sara"),
                  "Naderi"=>array("a"=>"Amir"),
                  "Mohamadi"=>array("a"=>"Poya","b"=>"Parniya")
                 );
```

مثال: اجازه بدهید تا با یک مثال نشان دهیم که چگونه می توان به یک سلول از آرایه های چند بعدی دسترسی پیدا کرد:

```
echo "Is " . $families['Ahmadi']['b'] . " a part of the Ahmadi family?";
```

خروجی کد بالا:

```
Is Reza a part of the Ahmadi family?
```

مرتب کردن آرایه ها در PHP

عناصر یک آرایه را می توان به صورت الفبایی یا عددی از کوچک به بزرگ (ascending) یا بزرگ به کوچک (descending) مرتب نمود.

توابع مرتب سازی آرایه ها:

- sort() - مرتب کردن مقادیر آرایه از کوچک به بزرگ
- rsort() - مرتب کردن مقادیر آرایه از بزرگ به کوچک
- asort() - مرتب کردن آرایه های انجمنی از نزولی به صعودی (بر اساس مقدار)
- ksort() - مرتب کردن آرایه های انجمنی از نزولی به صعودی (بر اساس کلید)
- arsort() - مرتب کردن آرایه های انجمنی از صعودی به نزولی (بر اساس مقدار)
- krsort() - مرتب کردن آرایه های انجمنی از صعودی به نزولی (بر اساس کلید)

در مثال زیر، آرایه انجمنی age$ بر اساس کلید، از کوچک به بزرگ مرتب شده است:

مثال

```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");
ksort($age);

foreach($age as $x=>$x_value)
    {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
    }
?>
```

خروجی کد بالا:

```
Key=Ben, Value=37
Key=Joe, Value=43
Key=Peter, Value=35
```

در مثال زیر، آرایه انجمنی age$ بر اساس مقادیر سلول ها، از بزرگ به کوچک مرتب شده است:

مثال

```
<?php
$age=array("Peter"=>"35","Ben"=>"37","Joe"=>"43");
arsort($age);
```

```
foreach($age as $x=>$x_value)
    {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
    }
?>
```

خروجی کد بالا:

```
Key=Joe, Value=43
Key=Ben, Value=37
Key=Peter, Value=35
```

برگرداندن تعداد عناصر یک آرایه

با استفاده از تابع ()count، می توان تعداد عناصر یک آرایه را برگرداند.

مثال

برگرداندن تعداد عناصر یک آرایه:

```
<?php
$cars=array("Volvo","BMW","Toyota");
echo count($cars);
?>
```

```
3
```

نحوه استفاده:

```
Count(array,mode);
```

پارامتر	توضیحات
array	الزامی است، نام آرایه را تعیین می کند.
mode	اختیاری است، در آرایه های چند بعدی، نحوه شمارش عناصر را تعیین می کند. ۰: مقدار پیشفرض، در آرایه های چند بعدی، تعداد کل عناصر را برنمی گرداند. ۱: در آرایه های چند بعدی، تعداد کل عناصر را برمی گرداند.

نکته: پارامتر mode از php 4.2 به بعد اضافه شده است.

یک مثال دیگر از تابع ()count

مثال

تعداد عناصر، در آرایه های چند بعدی:

```php
<?php
$cars=array
  (
  "Volvo"=>array
  (
  "XC60",
  "XC90"
  ),
  "BMW"=>array
  (
  "X3",
  "X5"
  ),
  "Toyota"=>array
  (
  "Highlander"
  )
  );

echo "Normal count: " . count($cars)."<br>";
echo "Recursive count: " . count($cars,1);
?>
```

```
Normal count: 3
Recursive count: 8
```

در بخشهای بعدی درباره ی حلقه ی foreach توضیح داده شده است.

صفر تا صد PHP -حلقه ها

حلقه ها در PHP

اغلب مواقعی که کد می نویسیم، می خواهیم که یک مجموعه از دستورات بارها و بارها اجرا بشوند، بجای اینکه آن چند خط که اغلب یکسان هستند را در دستورات تکرار کنیم از حلقه ها استفاده می کنیم.

انواع دستورات حلقه ای در PHP:

- **while:** تا زمانی که شرط حلقه درست باشد، دستورات اجرا می شود.
- **do...while:** دستورات داخل حلقه چه شرط حلقه درست باشد و چه نباشد برای یک بار اجرا می شود، و در مراحل بعدی اگر شرط حلقه درست بود، دستورات اجرا می شوند.
- **for:** دستورات داخل حلقه از یک عدد مشخص به تعدادی مشخص تکرار می شود.
- **foreach:** برای کار با آرایه ها کاربرد دارد.

حلقه while

این دستور تا زمانی که شرط برقرار باشد دستورات را اجرا می کند و شرط قبل از اجرای دستورات چک می شود. بدین معنا که اگر در وسط اجرای دستورات متغیرها طوری تغییر کنند که شرط برقرار نباشد، دستورات همچنان تا پایان اجرا می شوند و نیز اگر شرط از ابتدا برقرار نباشد دستورات دیگر اجرا نخواهند شد.

نحوه استفاده:

```
while (condition)
  {
  code to be executed;
  }
```

مثال:

در مثال زیر یک حلقه تعریف کرده ایم که با i=0 شروع می شود و دستورات داخل حلقه تا زمانی که i<=۵ باشد ادامه می یابد، البته توجه داشته باشید که مقدار متغیر i در داخل حلقه هر بار یک واحد اضافه می شود و اگر این تغییر، شرط حلقه را false کند، دستورات داخل حلقه تا پایان حلقه ادامه خواهد یافت.

```
<html>
<body>

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

</body>
</html>
```

خروجی کد بالا:

```
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
```

دستور do...while

این دستور شباهت بسیار زیادی به دستور while دارد با این تفاوت که شرط در پایان اجرای دستورات چک می شود یعنی به این صورت که اگر از ابتدا هم شرط برقرار نباشد، دستورات یکبار اجرا می شوند و بعد شرط چک می شود.

نحوه استفاده:

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

مثال:

تفاوت این مثال با مثال قبلی در دو چیز است:

۱. چون شرط حلقه در پایان دستورات چک می شود، دستورات داخل حلقه حتماً برای یک بار اجرا می شوند.

۲. چون دستور $i++; قبل از دستور چاپ است، بنابراین اعداد از ۲ تا ۶ چاپ می شود.

```
<html>
<body>

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

</body>
</html>
```

خروچی کد بالا:

```
The number is 2
The number is 3
The number is 4
The number is 5
The number is 6
```

حلقه for و foreach در فصل بعدی توضیح داده خواهد شد.

صفر تا صد PHP -حلقه for

حلقه for در PHP

از حلقه for زمانی که تعداد دفعات اجرای دستورات را بدانید استفاده می شود یعنی دستورات داخل حلقه از یک عدد مشخص به تعدادی مشخص تکرار می شود.

نحوه استفاده:

```
for (init; condition; increment)
  {
  code to be executed;
  }
```

پارامترها:

- ***init*:** تنها یکبار در ابتدای حلقه اجرا می شود، و اغلب برای تعریف یک شمارنده استفاده می شود.
- ***condition*:** ارزیابی تکرار حلقه، اگر شرط برقرار باشد دستورات اجرا می شوند، در غیر اینصورت دستورات ادامه نمی یابند.
- ***increment*:** بعد از هربار اجرای دستورات حلقه، این دستور اجرا می شود و اغلب برای افزایش شمارنده استفاده می شود.

نکته:

۱. در for می توان هرکدام از پارامترهای بالا را خالی گذاشت، به این صورت که اگر شرط خالی باشد، مقدار آن برابر با true در نظر گرفته می شود و حلقه بینهایت بار اجرا میشود! (البته با استفاده از دستور break که بعداً به آن اشاره می شود، می توان آن را قطع نمود)

۲. در for همچنین می توان برای پارامترهای بالا چند دستور مختلف را اجرا نمود به اینصورت که با علامت "," آنها را باید از یکدیگر جدا نمود.

مثال:

در مثال زیر یک حلقه تعریف شده که با i=1 شروع می شود، و تا زمانی که i<=۵ باشد ادامه خواهد یافت و به مقدار i در هر بار اجرای حلقه یک واحد اضافه می شود.

```
<html>
<body>

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

</body>
</html>
```

خروجی کد بالا:

```
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
```

حلقه foreach درPHP

مخصوص کار با آرایه ها می باشد.

نحوه استفاده:

```
foreach ($array as $value)
  {
  code to be executed;
  }
```

این دستور تمام مقادیر آرایه را یکی یکی از ابتدایی تا انتهایی به درون value می ریزد و دستورات را اجرا می نماید.

نکته: مقدار value کپی داده می باشد و با تغییر آن داده اصلی در آرایه تغییری نمی کند، اما با قرار دادن علامت "&" در ابتدای value یک نشانگر از آن برگردانده می شود، بدین معنا که با تغییر آن داده اصلی درون آرایه نیز تغییر می کند.

مثال۱: مثال زیر یک حلقه را نشان می دهد که مقادیر گرفته شده از یک آرایه را چاپ می کند.

```
<html>
<body>

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

</body>
</html>
```

خروجی کد بالا:

```
one
two
three
```

مثال۲: با قرار دادن علامت "&" در ابتدای value اگر در حلقه این مقدار تغییر کند، داده اصلی نیز تغییر می کند.

```
<html>
<body>

<?php
$arr=array(1,2,3,4);
foreach($arr as &$value)
{
  echo $value;
  echo " ";
  $value *= 2;
}
echo "<br />";
foreach($arr as $value)
{
  echo $value;
  echo " ";
}
?>

</body>
</html>
```

خروجی کد بالا:

```
1 2 3 4
2 4 6 6
```

مثال۳: چاپ کلید و مقدار آن (key and value)

```
<html>
<body>

<?php
$a = array(
    "one" => 1,
    "two" => 2,
    "three" => 3,
    "seventeen" => 17
);

foreach ($a as $k => $v) {
    echo "a[$k] => $v"."<br />";
}
?>
```

```
</body>
</html>
```

خروجی کد بالا:

```
a[one] => 1
a[two] => 2
a[three] => 3
a[seventeen] => 17
```

مثال۴: چاپ کلید و مقدار آن (`key and value`) در یک آرایه دو بعدی

```
<html>
<body>

$families = array("Ahmadi"=>array("a"=>"Ali","b"=>"Reza","c"=>"Sara"),
                  "Naderi"=>array("a"=>"Amir"),
                  "Mohamadi"=>array("a"=>"Poya","b"=>"Parniya")
                 );
foreach ($families as $k => $v)
    foreach ($v as $b=>$v)
    echo "families[$k][$b] => $v ."<br />";

</body>
</html>
```

خروجی کد بالا:

```
families[Ahmadi][a] => Ali
families[Ahmadi][b] => Reza
families[Ahmadi][c] => Sara
families[Naderi][a] => Amir
families[Mohamadi][a] => Poya
families[Mohamadi][b] => Parniya
```

صفر تا صد PHP -توابع

تعریف توابع در PHP

قدرت واقعی PHP در توابع آن است، در PHP بیش از ۱۰۰۰ تابع از قبل نوشته شده وجود دارد.

در این فصل به شما نشان خواهیم داد که در PHP چگونه می توانید توابع خودتان را تعریف کنید.

یک تابع به وسیله صدا زدن آن اجرا خواهد شد، شما می توانید از هر جای صفحه آنرا صدا بزنید.

نحوه تعریف توابع:

```
function functionName()
{
```

```
code to be executed;
}
```

توجه: برای نامگذاری توابع موارد زیر را در نظر داشته باشید:

- اسامی توابع به نحوی انتخاب شوند که نام انتخاب شده معرفی کننده عملی باشد که تابع انجام می دهد.
- نام تابع می تواند با حروف و یا علامت "_" شروع شود(با عدد نمی تواند شروع شود).

مثال: یک تابع ساده که با صدا زدن آن نام من چاپ می شود:

```
<html>
<body>

<?php
function writeName()
{
echo "pahlavan sadegh";
}

echo "My name is ";
writeName();
?>

</body>
</html>
```

خروجی کد بالا:

```
My name is pahlavan sadegh
```

ارسال پارامتر به توابع

یک پارامتر چیزی شبیه یک متغیر است، پارامترها بعد از نام تابع داخل پرانتز تعریف می شوند.

مثال۱: در مثال زیر با ارسال نام شخص به تابع، بعد از صدا زدن تابع، نام های مختلف با فامیلی یکسان چاپ می شود.

```
<html>
<body>

<?php
function writeName($fname)
{
echo $fname . " Ahmadi.<br />";
}
```

```
echo "My name is ";
writeName("Ali");
echo "My sister's name is ";
writeName("Sara");
echo "My brother's name is ";
writeName("Amir");
?>

</body>
</html>
```

خروجی کد بالا:

```
My name is Ali Ahmadi.
My sister's name is Sara Ahmadi.
My brother's name is Amir Ahmadi.
```

مثال۲: تابع زیر دو پارامتر دارد.

```
<html>
<body>

<?php
function addNumbers($a , $b)
{
$sum = $a + $b;
echo $sum;
}

addNumbers(100,200);
?>

</body>
</html>
```

خروجی کد بالا:

```
300
```

نکته: به صورت پیش فرض پارامترهایی که به توابع ارسال می شوند، طوری هستند که در صورتی که در تابع تغییر کنند مقدار اصلی آنها تغییری نخواهد کرد و به همان صورت باقی خواهند ماند، اما اگر در تعریف تابع قبل از اسم متغیر از علامت "&" استفاده کنیم، این ویژگی تغییر می کند یعنی با تغییر یک متغیر در درون تابع، اصل متغیر هم تغییر خواهد نمود، به مثال زیر توجه کنید.

مثال:

```
<html>
<body>

<?php
function changeit(&$string)
{
echo "String is: " . $string . '<br />';
$string="Learning PHP";
echo "String Changed to: " . $string . '<br />';
}

$str="PLUS";
changeit($str);
echo $str;

?>

</body>
</html>
```

خروجی کد بالا:

```
String is: PLUS
String Changed to: Learning PHP
Learning PHP
```

بازگرداندن یک مقدار توسط توابع

شما می توانید از دستور return برای بازگرداندن یک مقدار استفاده کنید، یعنی کافی است نتیجه را جلوی دستور return قرار دهید.

مثال:

```
<html>
<body>

<?php
function add($x,$y)
{
$total=$x+$y;
return $total;
}

echo "1 + 16 = " . add(1,16);
?>
```

```
</body>
</html>
```

خروجی کد بالا:

```
1 + 16 = 17
```

توابع از پیش تعریف شده:

در php بیش از ۵۳۰۰ تابع از پیش تعریف شده وجود دارد که البته برای اجرای آنها ممکن است در ورژن های مختلف برخی توابع وجود نداشته باشند و یا برای اجرای بعضی توابع احتیاج باشد تا extension خاصی نصب باشد.

توابع از پیش تعریف شده همانند توابعی که کاربر تعریف می کند اجرا می شوند.

صفر تا صد PHP -فرم ها

فرم های HTML در PHP

مهمترین چیز موقع کار کردن با فرمهای HTML و PHP این است که هر عنصر در فرم HTML به طور اتوماتیک در فرم PHP قابل دسترس و ردیابی است.

در PHP با استفاده از متغیرهای $_POST و $_GET می توان به اطلاعات فرم های HTML دسترسی داشت.

مثال:

در مثال زیر یک فرم HTML با سه عنصر زیر تعریف شده است:

۱. فیلد input از نوع text که با نام "fname" در کد PHP قابل ردیابی است.
۲. فیلد input از نوع text که با نام "age" در کد PHP قابل ردیابی است.
۳. فیلد input از نوع submit که محتویات textها را به صفحه "welcome.php" می فرستد.

```
<html>
<body>

<form action="welcome.php" method="post">
Name: <input type="text" name="fname" />
Age: <input type="text" name="age" />
<input type="submit" />
</form>

</body>
</html>
```

توجه: در مثال بالا از دو خصوصیت در تگ form استفاده شده ولی به صورت کلی خصوصیات تگ form به صورت زیر است:

- **action:** اگر submit اتفاق افتاد، اطلاعات فرم HTML به صفحه ای که در این خصوصیت مشخص شده ارسال می شود.(به صورت پیش فرض صفحه جاری در نظر گرفته می شود)
- **method:** با استفاده از این خصوصیت مشخص می کنیم که ارسال اطلاعات از فرم HTML به صفحه PHP به چه شکل باشد، اگر آنرا با مقدار "get" پر کنیم، اطلاعات فرم در آدرس صفحه قرار می گیرد و ارسال می شود، و برعکس اگر آنرا با مقدار "post" پر کنیم، اطلاعات به صورت یک آرایه ارسال می شود.(به صورت پیشفرض Get در نظر گرفته می شود)
- **name:** نام فرم است که برای php نیازی به آن نداریم ولی اگه بخواهیم از جاوا اسکریپت استفاده کنیم نیاز می شود (مثل چك كردن صحت ایمیل قبل از ارسال)
- **target:** نوع باز شدن صفحه مقصد را مشخص می کند که شامل صفحه جاری ، صفحه جدید ، صفحه پدر ، يك فريم خاص و ... است.
- ...

فایل "welcome.php" می تواند چیزی شبیه زیر باشد:

```
<html>
<body>

Welcome <?php echo $_POST["fname"]; ?>!<br />
You are <?php echo $_POST["age"]; ?> years old.

</body>
</html>
```

اگر در فرم HTML تعریف شده تکس باکس ها را به صورت "fname=Amir" و "age=28" مقدار دهی کنیم خروجی برنامه بعد از کلیک روی دکمه "submit" به صورت زیر خواهد بود:

```
Welcome Amir!
You are 28 years old.
```

همانطور که قبلاً گفته شد متغیرها با علامت "$" شروع می شوند، در صفحه PHP بالا نیز از متغیری بنام "$_post" استفاده شده که در فصل بعد توضیح داده خواهد شد.

اعتبار سنجی فرم (Validation Form)

اعتبار سنجی فرم یا Validation Form یعنی چک کردن ورودی های کاربر، قبل از اینکه در دیتابیس ذخیره شود، مثلاً در بالا برای تکس باکس "age" باید حتماً کاراکترهای عددی وارد شود، در غیر این صورت برنامه باید پیغام خطا بدهد.

به دو صورت می توان اعتبار سنجی کرد:

۱. client scripts: اعتبار سنجی سمت سرویس گیرنده انجام می شود که در این صورت سرعت بیشتر خواهد بود.

۲. server validation: اعتبار سنجی سمت سرویس دهنده انجام می شود، در حقیقت این وظیفه را یک صفحه PHP که روی سرور قرار دارد انجام می دهد.

در فصل های آینده راجع به این موضوع به صورت کامل توضیح داده خواهد شد.

صفر تا صد PHP -متغییر $_GET

متغیرهای عمومی از پیش تعریف شده (Superglobal) در PHP

Superglobalها در واقع متغیرهای از پیش تعریف شده ای هستند که صرفنظر از محدوده یا قلمرو، همیشه می توانید در هر تابع، کلاس یا فایلی به آنها دسترسی داشته باشید.

متغیرهای Superglobal در PHP:

- $_GET
- $_POST
- $_REQUEST
- $GLOBALS
- $_SERVER
- $_FILES
- $_ENV
- $_COOKIE
- $_SESSION

در این بخش و بخشهای بعدی، درباره هر کدام از متغیرهای بالا، توضیح داده خواهد شد.

متغیر $_GET

$_GET یک متغیر یا آرایه از پیش تعریف شده است و مقادیر فرم HTMLی که خصوصیت method آن با مقدار "get" پر شده باشد، به صورت اتوماتیک در آن ذخیره می شوند.

توجه:

۱. توجه داشته باشید حروف متغیر $_GET، حروف بزرگ است.

۲. اطلاعات فرستاده شده از یک فرم HTMLی که خصوصیت method آن با مقدار "get" پر شده باشد توسط هر شخصی قابل مشاهده است (اطلاعات در نوار آدرس مرورگر نمایش داده می شود)، این موضوع در ارسال اطلاعات محدودیت ایجاد می کند.

مثال:

```
<form action="welcome.php" method="get">
   Name: <input type="text" name="fname" />
   Age:  <input type="text" name="age" />
         <input type="submit" />
</form>
```

زمانی که کاربر روی دکمه "Submit" کلیک می کند، URL زیر به سرور فرستاده می شود و البته این آدرس توسط هر شخصی قابل مشاهده است:

```
http://www.w3schools.com/welcome.php?fname=Amir&age=28
```

فایل "welcome.php" حالا می تواند با استفاده از متغیر $_GET اطلاعات فرم HTML را جمع آوری کند.

نکته: نام (name) هر فیلد در فرم HTML یکتا است. این نام در حقیقت ایندکس آرایه $_GET خواهد بود، و با آن می توانیم به مقادیر فیلدها دسترسی داشته باشیم.

```
Welcome <?php echo $_GET["fname"]; ?>.<br />
You are <?php echo $_GET["age"]; ?> years old!
```

چه موقع از متد "get' استفاده کنیم؟

۱. زمانی که می خواهیم کلمه عبور یا اطلاعات حساس کاربر را ارسال کنیم، نباید ار متد get استفاده کرد.
۲. متد get برای ارسال متغیرهای با مقادیر بزرگ، مانند متن ها مناسب نیست.(نباید مقادیرمان بیشتر از ۲۰۰۰ کاراکتر باشد)
۳. به خاطر اینکه متغیرها و مقادیرشان در URL نمایش داده می شود، بنابراین می توان از آنها در صفحات دیگر پروژه استفاده کرد، در حقیقت از این طریق می توان متغیر ها را به صفحات دیگر پاس داد.

صفر تا صد PHP -متغییر $_POST

متغیر $_POST

$_POST یک متغیر یا آرایه از پیش تعریف شده است و مقادیر فرم HTMLی که خصوصیت method آن با مقدار "post" پر شده باشد، به صورت اتوماتیک در آن ذخیره می شود.

توجه:

۱. توجه داشته باشید حروف متغیر $_POST، حروف بزرگ است.
۲. اطلاعات ارسال شده توسط دیگران قابل مشاهده نیست، و محدودیتی در ارسال وجود ندارد.
۳. به صورت پیش فرض ۸ مگابایت اطلاعات را می توانید از این طریق ارسال کنید(با تغییر مقدار post_max_size در فایل php.ini می توانید این مقدار را تغییر دهید)

مثال:

```
<form action="welcome.php" method="post">
   Name: <input type="text" name="fname" />
   Age:  <input type="text" name="age" />
         <input type="submit" />
</form>
```

زمانی که کاربر روی دکمه "Submit" کلیک می کند URL به صورت زیر خواهد بود:

```
http://www.w3schools.com/welcome.php
```

فایل "welcome.php" حالا می تواند با استفاده از متغیر $_POST اطلاعات فرم HTML را جمع آوری کند.

نکته: نام(name) هر فیلد در فرم HTML یونیک است این نام در حقیقت ایندکس آرایه $_POST خواهد بود، و با آن می توانیم به مقادیر فیلدها دسترسی داشته باشیم

```
Welcome <?php echo $_POST["fname"]; ?>!<br />
You are <?php echo $_POST["age"]; ?> years old.
```

متغیر $_REQUEST در PHP

$_REQUEST یک متغیر یا آرایه از پیش تعریف شده است و مقادیر فرم HTMLی که خصوصیت method آن با مقدار "post" یا "get"پر شده باشد، هنگام ارسال به صورت اتوماتیک در آن ذخیره می شود.

این متغیر در حقیقت می تواند شامل محتوای متغیر های زیر باشد:

- $_GET
- $_POST
- $_COOKIE

مثال:

```
Welcome <?php echo $_REQUEST["fname"]; ?>!<br />
You are <?php echo $_REQUEST["age"]; ?> years old.
```

متغیر $GLOBALS در PHP

متغیر $GLOBALS یک متغیر superglobal است که برای دسترسی به متغیرهای عمومی (global) در هر جای اسکریپت ها (داخل توابع و یا متدها) استفاده می شود.

در PHP تمام متغیر های عمومی در آرایه $GLOBALS ذخیره می‌شوند. ایندکس این آرایه در واقع نام متغیر خواهد بود.

در مثال زیرنحوه ی استفاده از آرایه $GLOBALS نشان داده شده است:

مثال

```
<?php
$x = 75;
$y = 25;

function addition() {
  $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];
}

addition();
echo $z;
?>
```

```
100
```

در مثال بالا، از آنجایی که z با استفاده از آرایه $GLOBALS تعریف شده است، در خارج از تابع ()addition نیز قابل دسترس خواهد بود.

متغیر $_SERVER در PHP

متغیر $_SERVER یک متغیر superglobal است که اطلاعاتی را درباره ی headerها، مسیر و مکان اسکریپت ها و ... در خود نگه می دارد.

در مثال زیر، نحوه استفاده از بعضی عناصر آرایه $_SERVER نشان داده شده است:

مثال

```
<?php
echo $_SERVER['PHP_SELF'];//نام فایل اسکریپتی که اخیراً اجرا شده است را برمی گرداند
echo "<br>";
echo $_SERVER['SERVER_NAME'];//نام سرور هاست را برمی گرداند
echo "<br>";
echo $_SERVER['HTTP_HOST'];//با توجه به درخواست جاری، هدر هاست را برمی گرداند
echo "<br>";
echo $_SERVER['HTTP_REFERER'];//آدرس کامل صفحه جاری را برمی گرداند
echo "<br>";
echo $_SERVER['HTTP_USER_AGENT'];//اطلاعات مرورگر و سیستم عامل و ... را برمی گرداند
echo "<br>";
echo $_SERVER['SCRIPT_NAME'];//مسیر اسکریپت جاری را برمی گرداند
?>
```

در آینده با متغیرهای superglobal زیر نیز آشنا خواهید شد:

- متغیر $_SESSION : آموزش PHP-متغیر session
- متغیر $_COOKIE: آموزش PHP-متغیر cookie
- متغیر $_FILES: آموزش PHP-فایل ها

فصل ۲: آموزش کار با فرم ها

صفر تا صد PHP -کار با فرم ها

حال که با مختصرات PHP آشنا شدید، بهتر است به چگونگی اینجاد فرمها و ارتباط با کاربر بپردازیم.

همان طور که می دانید، برای ساختن اجزاء صفحات وب، از تگهای HTML که معمولا به صورت <tag/> و <tag> می باشند، استفاده می کنیم (بعضی از اجزا، دارای تگ پایانی نیستند. همانند تگ <img>). برای ساختن فرمها در HTML، ما از تگ <form> استفاده می کنیم:

```
<form>
<!-- Form Components -->
</form>
```

تگ <form> دارای دو خاصیت بسیار مهم می باشد:

- خاصیت action
- خاصیت method

خاصیت action محل اسکریپت پردازش فرم را تعیین می کند و خاصیت method، نوع ارسال داده ها به اسکریپت مورد نظر.

خاصیت method، خود دارای دو مقدار می باشد:

- get
- post

اما فرق این دو متد در چیست؟ در متد get، داده های ما به صورت Query String به اسکریپت پردازش فرم ارسال می شود. اما در متد post این طور نیست. حالا شاید برای شما این سوال پیش بیاید که Query String چیست؟ به قطعه کد زیر توجه کنید تا در مورد آن توضیح دهم:

http://www.site.com/?name=mojtaba&familyname=dashti

این قطعه کد در واقع از دو خط کد زیر تشکیل شده است:

http://www.site.com/

?name=mojtaba&familyname=dashti

خط اول که آدرس سایت می باشد. اما خط دوم که به Query String معروف است، شامل داده های ما می باشد که در PHP، مقدار mojtaba در متغیر name و مقدار dashti در متغیر familyname ذخیره می شود.
معمولا برای ارسال داده های کم، از Query String استفاده می شود (متد get). اما برای ارسال داده های بلند و رمزهای عبور (Password)، متد post بهترین انتخاب می باشد.

ساخت یک Text-Box

Text-Boxها برای دریافت عبارات یک خطی، همانند نام، آدرس ایمیل، آدرس سایت، کلمه عبور و ... در وب به کار می روند.
برای ساخت Text-Boxها از تگ <input>، به صورت زیر استفاده می کنید:

<input type="text" [size="m"] [name="name"] [value="optional value"]>

خاصیت type="text" به مرورگر می گوید که با این تگ <input> یک کادر متنی بسازد (بعضی دیگر از اجزای فرم نیز با همین تگ، ولی با خاصیت type متفاوت ساخته می شوند). دیگر خاصیتها برای کادر متنی، اختیاری می باشد. برخی از این خاصیتهای مهم به صورت زیر است:

- خاصیت size: اندازه Text-Box را بر حسب تعداد کاراکتر نشان می دهد. دقت داشته باشید که این خاصیت، در واقع در اندازه نمایش Text-Box و نه مقدار کاراکتری که می تواند دریافت کند، اثر می گذارد.
- خاصیت name: این خاصیت برای ایجاد فرمهای تعاملی بسیار مهم می باشد. با اختصاص یک نام به یک Text-Box، به راحتی می تواند به وسیله اسکریپتها، داده وارد شده توسط کاربر را دریافت کرده و بر روی آن عملیات انجام دهید.
- خاصیت value: به طور Default، کادرهای متنی هنگام نمایش خالی می باشند. اگر می خواهید متنی را درون Text-Box قرار دهید، عبارت مورد نظر خود را در قسمت value قرار دهید.

ساخت Text-Area

Text-Areaها در واقع نوع پیشرفته تر Text-Box می باشند که می توانند بیش از یک خط داده را از کاربر دریافت کنند. برای ساخت Text-Area از تگ <textarea> استفاده می کنیم:

<textarea [name="name"] [rows="m"] [cols="m"]>

Optional Text

</textarea>

خاصیتهای rows و cols، به ترتیب نمایانگر تعداد سطرها و کاراکترهای Text-Area می باشند.

کادرهای پسورد (Password)

برای ساخت کادرهای پسورد، از تگ <input>، به صورت زیر استفاده می کنیم. کادرهای Password در واقع نوعی Text-Box هستند، با این تفاوت که در این نوع کادرها، داده های وارد شده به صورت ستاره (*) در ویندوزهای ۹۸//ME۲۰۰۰ و دایره (●) در ویندوز XP نمایش داده می شوند.

<input type="password" [name="name"] [size="m"] [value="optional value"]>

کادرهای انتخاب (Check-Box)

حتما تا به حال با Check-Boxها در ویندوز برخورد کرده اید: مربع های کوچکی که کاربر می تواند آنها را انتخاب کند. برای ایجاد کردن Check-Boxها از تگ <input> به صورت زیر استفاده می کنیم:

<input type="checkbox" [name="name"] [value="value"] [checked="checked"]>

مقدار خاصیت value در صورتی که Check-Box انتخاب شده باشد، به سرور فرستاده می شود. اگر خاصیت value را برای Check-Box در نظر نگیرید، به طور اتوماتیک مقدار on به سرور فرستاده می شود. خاصیت checked="checked" سبب انتخاب شدن Check-Box هنگام لود شدن صفحه می شود.

دکمه های رادیویی (Radio Buttons)

Button Radioها همانند Check-Boxها می باشند. با این تفاوت که در Check-Boxها، کاربر می تواند از بین چندین انتخاب، هر کدام را که بخواهد انتخاب کند (۱، ۲، ۳ و یا بیشتر). اما در Radio Buttonها، کاربر از بین چندین انتخاب، تنها می تواند یکی از آنها را انتخاب کند.

حال که با بعضی از اجزای فرم آشنا شدید، بهتر است یک فرم تعاملی بسازیم که یکسری اطلاعات از کاربر گرفته و آنها را در صفحه ای مجزا نمایش دهد. کد HTML مورد نظر ما، به صورت زیر می باشد. آن را در فایلی تحت عنوان index.php در محل دلخواه خود، ذخیره کنید (در محلی که فایلهای php قادر به اجرا شدن هستند).

<html>

```
<head>
   <title>Getting Information</title>
   <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
</head>

<body>
<div style="text-align: center">
<div style="width: 420px; text-align: justify ">
<form method="post" action="result.php">
<table border="0" width="100%">
   <tr>
      <td width="200px" style="text-align: right">Username</td>
      <td width="20px"> </td>
      <td width="200px" style="text-align: left">
         <input type="text" name="username" size="20">
      </td>
   </tr>
   <tr>
      <td width="200px" style="text-align: right">Password</td>
      <td width="20px"> </td>
      <td width="200px" style="text-align: left">
         <input type="password" name="password" size="20">
      </td>
   </tr>
   <tr valign="top">
      <td width="200px" style="text-align: right">Feedback</td>
      <td width="20px"> </td>
      <td width="200px" style="text-align: left">
         <textarea name="feedback"
         rows="5" cols="25"></textarea>
```

```
    </td>
</tr>
<tr valign="top">
    <td width="200px" style="text-align: right">Hobbies</td>
    <td width="20px"> </td>
    <td width="200px" style="text-align: left">
        <input type="checkbox" id="internet" name="internet"
        value="Internet" checked="checked">
        <label for="internet">Internet</label><br>
        <input type="checkbox" id="computer"
        name="computer" value="Computer">
        <label for="computer">Computer</label><br>
        <input type="checkbox" id="hack"
        name="hack" value="Hack">
        <label for="hack">Hack, Crack</label><br>
    </td>
</tr>
<tr valign="top">
    <td width="200px" style="text-align: right">Show Password</td>
    <td width="20px"> </td>
    <td width="200px" style="text-align: left">
        <input type="radio" id="no" name="passwordio"
        value="no" checked="checked">
        <label for="no">No</label><br>
        <input type="radio" id="yes"
        name="passwordio" value="yes">
        <label for="yes">Yes</label>
    </td>
</tr>
<tr>
```

```
        <td colspan="3" style="text-align: center">
            <input type="submit" value=" Submit Data ">
        </td>
    </tr>
</table>
</form>
</div>
</div>
</body>

</html>
```

حال که فرم HTML خود را ساختیم، ساخت اسکریپت مورد نظر خود را، که در اینجا result.php می باشد، شروع می کنیم.

```
<html>

<head>
<title>Result</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
</head>

<body>
<p>Hello, you can see your information below ;)</p>
Username: <b><?=$username?></b><br>
<?php if ($passwordio == "yes") { ?>
Password: <b><?=$password?></b><br>
<?php } else { ?>
Password: <b><font color="red">Password is hidden</font></b><br>
<?php } ?>
Feedback:<br>
<b><?=$feedback?></b><br>
```

```
Hobbies:<br>
<?php echo("$internet<br>$computer<br>$hack");?><br>

</body>

</html>
```

دقت داشته باشید که این یک اسکریپت بسیار ساده می باشد و برای کامل کردن آن، نیازمند دانش بیشتری هستیم. پس فعلا به همین اسکریپت ساده اکتفا می کنیم.

با استفاده از متغیرهای $_GET و $_POST، می توانید سمت سرور مقادیر فرم HTML را دریافت نمایید.

یک فرم ساده HTML

در مثال زیر، یک فرم ساده HTML با دو فیلد ورودی و یک دکمه submit نشان داده شده است:

مثال

```
<html>
<body>

<form action="welcome.php" method="post">
نام: <input type="text" name="name"><br>
ایمیل: <input type="text" name="email"><br>
<input type="submit" value="ارسال اطلاعات">
</form>

</body>
</html>
```

خروجی کد بالا:

ایمیل:

ارسال اطلاعات

دازش به

یک فایل php بنام "welcome.php" ارسال می شود. اطلاعات فرم با متد POST ارسال می شود.

فایل welcome.php

برای نمایش اطلاعات ارسال شده به سرور، بصورت ساده می توانید همه متغیرها را چاپ نمایید:

```
<html>
<body>

نام:<php echo $_POST["name"]; ?><br?>
ایمیل:<? php echo $_POST["email"];?>
</body>
</html>
```

خروجی کد بالا می تواند شبیه زیر باشد:

```
نام: example
ایمیل : admin@example.com
```

توجه: مثال بالا را می توان از طریق متد get نیز انجام داد. که البته نحوه انجام کار در بخشهای قبلی ذکر شده است.

کد بالا، یک مثال کاملاً ساده است. اما توجه داشته باشید که یک چیز بسیار مهم در آن از قلم افتاده است و آن اعتبار سنجی یا Validate اطلاعات ارسالی است. از این طریق اسکریپتتان امن تر خواهد بود.

نکته: در مثال بالا، اعتبار سنجی انجام نشده است، توجه داشته باشید که در این مثال فقط نحوه ی ارسال و دریافت اطلاعات نشان داده شده است.
اما در بخش بعدی، نحوه ی پردازش اطلاعات فرم ارسال شده بصورتی که امنیت حفظ شود، نشان داده خواهد شد.
اعتبار سنجی مناسب اطلاعات فرم، شما را از حمله هکرها حفظ خواهد کرد.

صفر تا صد PHP -اعتبارسنجی فرم

در این بخش و همچنین بخش بعدی، نحوه ی اعتبارسنجی داده های فرم با استفاده از PHP آموزش داده خواهد شد.

اعتبارسنجی فرم در PHP

زمان پردازش فرم های PHP، به امنیت بیاندیشید!
در این بخش نحوه ی پردازش فرم های PHP بصورتی که حداقل موارد امنیتی که در ذهن دارید حفظ شود، نشان داده خواهد شد. اعتبار سنجی مناسب اطلاعات فرم، بمنظور جلوگیری از حمله هکرها دارای اهمیت است.

فرم HTMLی که در این بخش روی آن کار خواهیم کرد شامل فیلدهای مختلفی است: فیلدهای متنی الزامی و اختیاری، radio button و یک دکمه submit:

در جدول زیر قوانین اعتبارسنجی ذکر شده است:

فیلد	قوانین اعتبارسنجی
نام	الزامی است. فقط باید شامل حروف الفبا و خط فاصله باشد.
ایمیل	الزامی است. باید شامل فرمت صحیح ایمیل باشد (همراه با علامت @ و .)
وب سایت	اختیاری است. اگر پر شد، باید شامل فرمت صحیح URL باشد.
توضیحات	اختیاری است. می تواند شامل چند خط متن باشد (textarea)

جنسیت	الزامی است. یکی از دو مورد باید انتخاب شود.

اجازه دهید تا در ابتدا نگاهی به کد HTML فرم بیاندازیم.

فیلدهای متنی

برای فیلدهای متنی "نام"، "ایمیل" و "وب سایت" از عنصر متنی <input> و برای فیلد "توضیحات" از عنصر <textarea>

استفاده می کنیم:

```
نام: <input type="text" name="name">
ایمیل: <input type="text" name="email">
وب سایت: <input type="text" name="website">
توضیحات: <textarea name="comment" rows="5" cols="40"></textarea>
```

فیلد "جنسیت"

برای فیلد "جنسیت" از Radio Button استفاده می کنیم: (عنصر <input> از نوع radio)

```
جنسیت:
<input type="radio" name="gender" value="female">Female
<input type="radio" name="gender" value="male">Male
```

عنصر <form>

کد HTML فرم شبیه زیر است:

```
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
```

زمانی که روی دکمه submit کلیک می شود، اطلاعات با متد "post" به سرور ارسال می شود.

متغییر $_SERVER

متغییر $_SERVER یکی از متغییرهای عمومی از پیش تعریف شده است (آرایه). "PHP_SELF" نام فایلِ اسکریپت جاری را برمی گرداند.

بنابراین زمانی که کاربر روی دکمه submit کلیک می کند، بجای اینکه داده ها به یک فایل متفاوت دیگر ارسال شود به خودش ارسال خواهد شد. در این روش، کاربر خطاها را در همان صفحه ای که فرم وجود دارد مشاهده خواهد کرد.

تابع ()htmlspecialchars

با استفاده از تابع ()htmlspecialchars، می توان کاراکترهای خاص را به HTML entity تبدیل نمود. منظور این است که کاراکترهایی مثل علامت کوچکتر(<) و بزرگتر(>) در پارامتر ورودی را به $lt; و $gt; تبدیل می کند. با این کار از حمله ی هکرهایی که می خواهند از طریق تزریق HTML یا JavaScript اخلال ایجاد کنند، جلوگیری می شود.

یک تذکر مهم درباره امنیت فرم PHP

PHP_SELF می تواند توسط هکرها مورد استفاده قرار گیرد.

یک هکر می تواند در آدرس بار مرورگرش بعد از آدرس فایل، یک اسلش (/) قرار دهد و سپس دستورات XSS را برای اجرا تایپ کند.

XSS یا Cross-site scripting
XSS یک نوع قابلیت آسیب پذیری امنیت کامپوتر است. معمولاً در برنامه های کاربردی web بکار می رود. XSS، هکرها را قادر می سازد تا صفحات وب را از طریق تزریق اسکریپت سمت client هک کنند.
دلیل به وجود آمدن این آسیب پذیری عدم اعتبارسنجی ورودی‌های کاربر می باشد، و مهاجم می تواند با تزریق اسکریپت‌های مخرب در سایت از این آسیب پذیری سو استفاده کند.

فرض کنید، فرم زیر را در یک فایل بنام "test_form.php" داریم:

```
<form method="post" action="<?php echo $_SERVER["PHP_SELF"];?>">
```

حالا اگر یک کاربر در آدرس بار مرورگرش "http://www.example.com/test_form.php" را وارد کند، کد بالا بصورت زیر ترجمه خواهد شد:

```
<form method="post" action="test_form.php">
```

خوب تا اینجا همه چیز خوب است.

اما درنظر بگیرید که کاربری URL زیر را در آدرس بار وارد کند:

```
http://www.example.com/test_form.php/%22%3E%3Cscript%3Ealert('hacked')%3C/script%3E
```

در این صورت کد بالا بصورت زیر ترجمه خواهد شد:

```
<form method="post" action="test_form.php"/><script>alert('hacked')</script>
```

دومین URL، باعث اضافه شدن تگ <script> و یک دستور alert در بین کدهای ما شده است. و زمانی که صفحه لود می شود، کد JavaScript اجرا می شود (کاربر یک جعبه پیغام خواهد دید). این فقط یک مثال ساده و بی ضرر است که نحوه هک کردن متغییر PHP_SELF را نشان می دهد.

توجه داشته باشید که هر کد JavaScript دیگری را می توان در تگ <script> قرار داد...! یک هکر می تواند کاربر را به یک فایل دیگر روی سروری دیگر redirect کند، و از طریق آن فایل، اطلاعات کاربر را ذخیره کند.

نحوه مقابله با هک از طریق"PHP_SELF"

با استفاده از تابع htmlspecialchars()، می توان با هک از طریق "PHP_SELF" مقابله نمود.

کد آن شبیه زیر است:

```
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
```

تابع htmlspecialchars()، کاراکترهای خاص را به HTML entity تبدیل می کند. حالا اگر کاربر بخواهد از متغییر "PHP_SELF" سوء استفاده کند، با نتیجه زیر روبرو خواهد شد:

```
<form method="post" action="test_form.php/&quot;&gt;&lt;script&gt;alert('hacked')&lt;/script&gt;">
```

و از این طریق هیچ آسیبی وارد نخواهد شد...!

اعتبارسنجی داده های فرم با PHP

اولین کاری که باید انجام دهید این است که تمام متغییرها را به تابع htmlspecialchars() پاس دهیم.

حالا اگر کاربر تلاش کند که متنی مانند زیر را ارسال کند:

<script>location.href('http://www.hacked.com')</script>

در اینصورت اسکریپت بالا اجرا نخواهد شد، چونکه کاراکترهای خاص در متن بالا به HTML entity معادلاشان تبدیل شده اند:

<script>location.href('http://www.hacked.com')</script>

حالا این کد برای نمایش در یک صفحه یا داخل یک ایمیل، امن شده است.

همچنین ما دو کار دیگر را هنگام ارسال داده ها به سرور انجام می دهیم:

۱. با استفاده از تابع trim() کاراکترهای غیرضروری (مثل: فاصله های اضافی، tab و خطوط خالی) را حذف می کنیم.
۲. با استفاده از تابع stripslashes()، بک اسلش ها (\) را حذف می کنیم

گام بعدی، ایجاد یک تابع، برای انجام تمام کارهای بالاست (بجای اینکه کدهای مربوط به این قسمت را بارها و بارها بنویسیم، مناسب تر است که از یک تابع استفاده کنیم)

ما این تابع را test_input() می نامیم.

حالا ما می توانیم به ازای هر متغییر $_POST مقدار آنرا با تابع test_input() چک کنیم و کد آن شبیه زیر است:

مثال

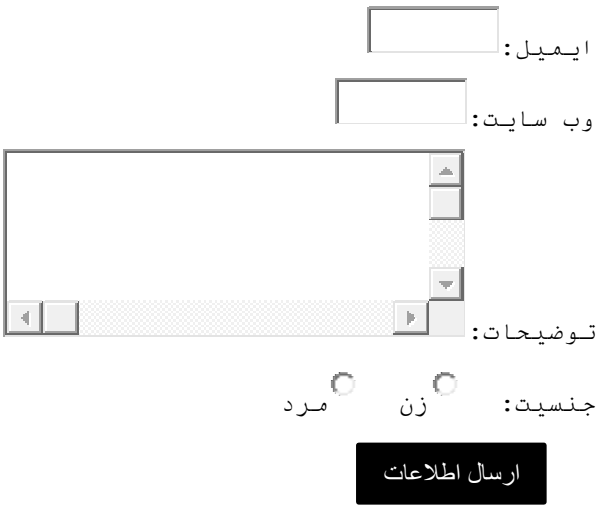
```
<!DOCTYPE HTML>
<html>
<head>
<style>
  span{min-width: 200px;float: right;}
</style>
</head>
<body style="direction:rtl;">

<?php
$name = $email = $gender = $comment = $website = "";
if ($_SERVER["REQUEST_METHOD"] == "POST")
{
  $name = test_input($_POST["name"]);
```

```
  $email = test_input($_POST["email"]);
  $website = test_input($_POST["website"]);
  $comment = test_input($_POST["comment"]);
  $gender = test_input($_POST["gender"]);
}

function test_input($data)
{
  $data = trim($data);
  $data = stripslashes($data);
  $data = htmlspecialchars($data);
  return $data;
}
?>

<h2>مثال اعتبارسنجی فرم ها در PHP</h2>
<form method="post" action="<?php echo
htmlspecialchars($_SERVER["PHP_SELF"]);?>">
<div><span>نام:</span><input type="text" name="name"></div>
<div><span>ایمیل:</span><input type="text" name="email"></div>
<div><span>وب سایت:</span><input type="text" name="website"></div>
<div><span>توضیحات:</span><textarea name="comment" rows="5"
cols="40"></textarea>
</div>
<div><span>جنسیت:</span>
<input type="radio" name="gender" value="female">زن
<input type="radio" name="gender" value="male">مرد
</div>
</div><input type="submit" name="submit" value="ارسال اطلاعات"></div>
</form>

<?php
if (isset($name) || isset($email) || isset($gender) || isset($comment)
|| isset($website))
 {
  echo "<br /><h2>خروجی کدتان</h2>";
  echo "نام :$name";
  echo "<br />";
```

```
  echo "ایمیل: $email";
  echo "<br />";
  echo "وب سایت: $website";
  echo "<br />";
  echo "توضیحات: $comment";
  echo "<br />";
  echo "جنسیت: $gender";
 }
?>

</body>
</html>
```

خروجی کد بالا:

مثال اعتبارسنجی فرم ها در PHP



توجه داشته باشید که در ابتدای اسکریپت، با استفاده از متغییر "REQUEST_METHOD" نحوه ارسال داده های فرم را چک می کنیم. اگر نحوه ی ارسال داده های فرم، از طریق متد "POST" است، اطلاعات فرم پردازش خواهد شد وگرنه با یک صفحه خالی روبرو خواهیم شد.

توجه: برای بار اول که کاربر درخواست مشاهده فایل مثال بالا را به سرور ارسال می کند، بدلیل اینکه متد پیش فرض برای مشاهده صفحات از نوع get است، بنابراین شرط `SERVER["REQUEST_METHOD"] == "POST_$"` درست نخواهد بود و دستورات داخل شرط اجرا نخواهد شد. اما بعد از اینکه کاربر اطلاعات فرم را پر کرده و روی دکمه ارسال (submit) کلیک کرد، چون ویژگی method فرم را با مقدار "post" تنظیم کرده ایم، شرط ذکر شده درست خواهد بود و دستورات داخل آن اجرا خواهد شد.

اما در مثال بالا، تمام فیلدهای ورودی اختیاری است. حتی اگر کاربر هیچ کدام از فیلدها را پر نکند، اسکریپت بالا باز هم کار خواهد کرد.

در بخش بعدی برای فیلدهای الزامی، یک پیغام مناسب نمایش خواهیم داد.

صفر تا صد PHP -فیلدهای الزامی

در این آموزش، نحوه ی ساخت فیلدهای الزامی (required) و نمایش پیغام خطای مناسب نشان داده خواهد شد.

فیلدهای الزامی در PHP

در آموزش قبلی، تمام فیلدهای ورودی اختیاری بودند. اما همانطور که در جدول اعتبار سنجی زیر مشخص شده است، فیلدهای "نام"، "ایمیل" و "جنسیت" الزامی اند. این فیلدها نمی توانند خالی باشند و باید حتماً با یک مقدار تنظیم شوند.

قوانین اعتبارسنجی:

فیلد	قوانین اعتبارسنجی
نام	**الزامی است**. فقط باید شامل حروف الفبا و خط فاصله باشد.
ایمیل	**الزامی است**. باید شامل فرمت صحیح ایمیل باشد (همراه با علامت @ و .)
وب سایت	اختیاری است. اگر پر شد، باید شامل فرمت صحیح URL باشد.
توضیحات	اختیاری است. می تواند شامل چند خط متن باشد (textarea)
جنسیت	**الزامی است**. یکی از دو مورد باید انتخاب شود.

در کد زیر، تعدادی متغیر جدید با نام های $nameErr و $emailErr و $genderErr و $websiteErr اضافه شده است. این متغیرهای خطا، متن خطای مورد نظر را برای فیلدهای الزامی در خود نگه می دارند. ما همچنین برای هر متغیر $_POST یک دستور if ... else اضافه کرده ایم. این دستور شرطی، چک می کند که اگر متغیر $_POST خالی است (با استفاده از تابع empty()) یک متن خطای مناسب در متغیرهای خطا ذخیره می کند، و اگر خالی نبود، مقدار برگشتی تابع test_input() در متغیر متناظرش ذخیره می شود:

```
<?php
// define variables and set to empty values
$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = $comment = $website = "";

if ($_SERVER["REQUEST_METHOD"] == "POST")
{
```

```
 if (empty($_POST["name"]))
  {$nameErr = "Name is required";}
 else
  {$name = test_input($_POST["name"]);}

 if (empty($_POST["email"]))
  {$emailErr = "Email is required";}
 else
  {$email = test_input($_POST["email"]);}

 if (empty($_POST["website"]))
  {$website = "";}
 else
  {$website = test_input($_POST["website"]);}

 if (empty($_POST["comment"]))
  {$comment = "";}
 else
  {$comment = test_input($_POST["comment"]);}

 if (empty($_POST["gender"]))
  {$genderErr = "Gender is required";}
 else
  {$gender = test_input($_POST["gender"]);}
}
?>
```

نمایش متن خطا در PHP

در فرم HTML، باید یک اسکریپت PHP اضافه نمایید. که متن خطای مورد نظر را نمایش می دهد:

مثال

```
<!DOCTYPE HTML>
<html>
<head>
<style>
  .error {color: #FF0000;}
  span{min-width: 200px;float: right;}
</style>
</head>
<body>

<?php
// define variables and set to empty values
$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = $comment = $website = "";

if ($_SERVER["REQUEST_METHOD"] == "POST")
```

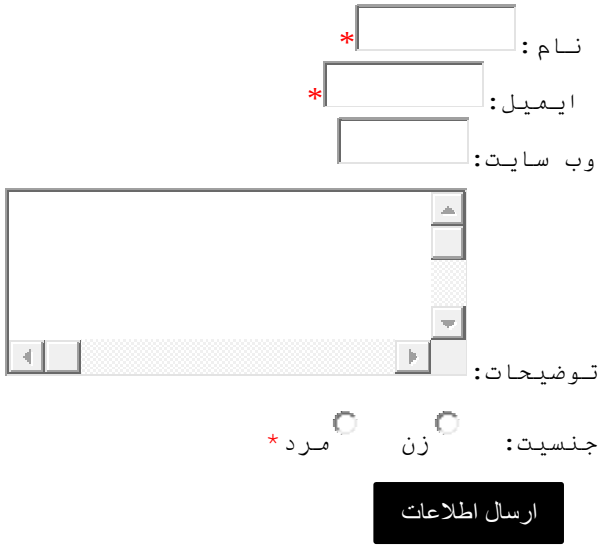
```
{

   if (empty($_POST["name"]))
     {$nameErr = "وارد کردن نام الزامی است";}
   else
     {$name = test_input($_POST["name"]);}

   if (empty($_POST["email"]))
     {$emailErr = "وارد کردن ایمیل الزامی است";}
   else
     {$email = test_input($_POST["email"]);}

   if (empty($_POST["website"]))
     {$website = "";}
   else
     {$website = test_input($_POST["website"]);}

   if (empty($_POST["comment"]))
     {$comment = "";}
   else
     {$comment = test_input($_POST["comment"]);}

   if (empty($_POST["gender"]) || $_GET["gender"]=="undefined")
     {$genderErr = "انتخاب جنسیت الزامی سات";}
   else
     {$gender = test_input($_POST["gender"]);}

}

function test_input($data)
{
     $data = trim($data);
     $data = stripslashes($data);
     $data = htmlspecialchars($data);
     return $data;
}
?>
<h2>مثال اعتبارسنجی فرم ها در PHP</h2>
```

```
<form method="post" action="<?php echo
htmlspecialchars($_SERVER["PHP_SELF"]);?>">
<div>
  <span>نـام:</span><input type="text" name="name">
  <span class="error">*</span>
</div>
<div>
  <span>ایـمیل:</span><input type="text" name="email">
  <span class="error">*</span>
</div>
<div>
  <span>وب سایـت:</span><input type="text" name="website">
  <span class="error"></span>
</div>
<div>
  <span>تـوضیحات:</span><textarea name="comment" rows="5"
cols="40"></textarea>
  <span class="error">*</span>
</div>
<div>
<span>جنسیت:</span>
<input type="radio" name="gender" value="female">زن
<input type="radio" name="gender" value="male">مـرد
</div>
</div><input type="submit" name="submit" value="ارسال اطلاعات"></div>
</form>

<?php
if ($nameErr!="" || $emailErr!="" || $genderErr!="" || $websiteErr!="")
echo "<span style='color:red' >
         $nameErr <br /> $emailErr <br /> $genderErr <br /> $websiteErr
      </span>";
else
{
echo "<br /><h2>خروجی کـدتـان</h2>";
echo "نـام :$name";
echo "<br />";
echo "ایـمیل: $email";
```

```
echo "<br />";
echo "وب سایت: $website";
echo "<br />";
echo "توضیحات: $comment";
echo "<br />";
echo "جنسیت: $ender";
}
?>
</body>
</html>
```

خروجی کد بالا:

مثال اعتبارسنجی فرم ها در PHP



در آموزش بعدی نحوه اعتبارسنجی فیلدهای "نام"، "ایمیل" و "وب سایت" توضیح داده خواهد شد. (آیا فیلد "نام"، فقط شامل حروف و فاصله خالی است - آیا فیلد "ایمیل"، شامل یک فرمت صحیح است - اگر فیلد "وب سایت" پر شد، آیا شامل یک فرمت صحیح است)

صفر تا صد PHP -اعتبارسنجی ایمیل

در این آموزش، نحوه ی اعتبارسنجی فیلدهای "نام"، "ایمیل" و "وب سایت" نشان داده خواهد شد.

اعتبارسنجی فیلد "نام" در PHP

کد زیر، یک روش ساده برای چک کردن اینکه آیا فیلد "نام" معتبر است یا نه را نشان می دهد. (فیلد نام فقط شامل حروف و خط فاصله است)

اگر فیلد "نام" معتبر نباشد، در متغیر $nameErr یک متن مناسب، تنظیم می شود:

```
$name = test_input($_POST["name"]);
if (!preg_match("/^[a-zA-Z ]*$/",$name))
 {
 $nameErr = "فیلد نام فقط شامل حروف و خط فاصله است" ;
 }
```

تابع preg_match()

با استفاده از تابع preg_match()، می توانید یک الگوی خاص را در یک رشته جستجو کنید. اگر الگوی مورد نظر در رشته وجود داشت مقدار true و اگر وجود نداشت مقدار false را برمی گرداند.

اعتبارسنجی فیلد "ایمیل" در PHP

روش اول: استفاده از تابع preg_match()

کد زیر، یک روش ساده برای چک کردن اینکه آیا فیلد "ایمیل" معتبر است یا نه را نشان می دهد. (باید شامل فرمت صحیح ایمیل باشد، همراه با علامت @ و .)

اگر فیلد "ایمیل" معتبر نباشد، در متغیر $emailErr یک متن مناسب، تنظیم می شود:

```
$email = test_input($_POST["email"]);
if (!preg_match("/([\w\-]+\@[\w\-]+\.[\w\-]+)/",$email))
 {
 $emailErr = "فرمت فیلد ایمیل صحیح نیست ";
 }
```

روش دوم: استفاده از تابع filter_var()

یکی دیگر از روش های چک کردن اعتبار ایمیل، استفاده از تابع filter_var() در PHP است. برای کسب اطلاعات بیشتر درباره تابع filter_var() به لینک آموزش PHP-فیلتر ورودی ها مراجعه نمایید.

```
$email = test_input($_POST["email"]);
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
 $emailErr ="فرمت فیلد ایمیل صحیح نیست ";
}
```

در این آموزش، از روش اول یعنی تابع preg_match() استفاده شده است.

اعتبارسنجی فیلد "وب سایت" در PHP

کد زیر، یک روش ساده برای چک کردن اینکه آیا فیلد "وب سایت" معتبر است یا نه را نشان می دهد. (باید شامل فرمت صحیح URL باشد، همچنین عبارت منظم زیر اجازه می دهد که در URL علامت dash "-" نیز استفاده شود)

اگر فیلد "وب سایت" معتبر نباشد، در متغیر $websiteErr یک متن مناسب، تنظیم می شود:

```
$website = test_input($_POST["website"]);
if (!preg_match("/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-9+&@#\/%?=~_|!:,.;]*[-a-z0-9+&@#\/%=~_|]/i",$website))
 {
 $websiteErr = "فرمت فیلد وب سایت صحیح نیست ";
 }
```

اعتبارسنجی فیلدهای "نام"، "ایمیل" و "وب سایت" در PHP

تا اینجا، اسکریپت مان شبیه زیر شده است:

مثال

```
<!DOCTYPE HTML>
<html>
<head>
<style>
  .error {color: #FF0000;}
  span{min-width: 200px;float: right;}
</style>
</head>
<body>

<?php
// define variables and set to empty values
$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = $comment = $website = "";

if ($_SERVER["REQUEST_METHOD"] == "POST")
{
   if (empty($_POST["name"]))
     {$nameErr = "Name is required";}
   else
     {
     $name = test_input($_POST["name"]);
     // check if name only contains letters and whitespace
     if (!preg_match("/^[a-zA-Z ]*$/",$name))
       {
       $nameErr = "Only letters and white space allowed";
       }
```

```
    }

  if (empty($_POST["email"]))
    {$emailErr = "Email is required";}
  else
    {
    $email = test_input($_POST["email"]);
    // check if e-mail address syntax is valid
    if (!preg_match("/([\w\-]+\@[\w\-]+\.[\w\-]+)/",$email))
      {
      $emailErr = "Invalid email format";
      }
    }

  if (empty($_POST["website"]))
    {$website = "";}
  else
    {
    $website = test_input($_POST["website"]);
    // check if URL address syntax is valid (this regular expression
also allows dashes in the URL)
    if (!preg_match("/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-
9+&@#\/%?=~_|!:,.;]*[-a-z0-9+&@#\/%=~_|]/i",$website))
      {
      $websiteErr = "Invalid URL";
      }
    }

  if (empty($_POST["comment"]))
    {$comment = "";}
  else
    {$comment = test_input($_POST["comment"]);}

  if (empty($_POST["gender"]) || $_GET["gender"]=="undefined")
    {$genderErr = "Gender is required";}
  else
    {$gender = test_input($_POST["gender"]);}
}
```

```
function test_input($data)
{
     $data = trim($data);
     $data = stripslashes($data);
     $data = htmlspecialchars($data);
     return $data;
}
?>
<h2>مثال اعتبارسنجی فرم ها در PHP</h2>
<form method="post" action="<?php echo
htmlspecialchars($_SERVER["PHP_SELF"]);?>">
<div>
  <span>نام:</span><input type="text" name="name">
  <span class="error">*</span>
</div>
<div>
  <span>ایمیل:</span><input type="text" name="email">
  <span class="error">*</span>
</div>
<div>
  <span>وب سایت:</span><input type="text" name="website">
  <span class="error"></span>
</div>
<div>
  <span>توضیحات:</span><textarea name="comment" rows="5"
cols="40"></textarea>
  <span class="error">*</span>
</div>
<div>
<span>جنسیت:</span>
<input type="radio" name="gender" value="female">زن
<input type="radio" name="gender" value="male">مرد
</div>
</div><input type="submit" name="submit" value="ارسال اطلاعات"></div>
</form>

<?php
```

```
if ($nameErr!="" || $emailErr!="" || $genderErr!="" || $websiteErr!="")
echo "<span style='color:red' >
        $nameErr <br /> $emailErr <br /> $genderErr <br /> $websiteErr
     </span>";
else
{
echo "<br /><h2>خروجی کدتان</h2>";
echo "نام :$name";
echo "<br />";
echo "ایمیل: $email";
echo "<br />";
echo "وب سایت: $website";
echo "<br />";
echo "توضیحات: $comment";
echo "<br />";
echo "جنسیت: $gender";
?>
}
</body>
</html>
```

خروجی کد بالا:

مثال اعتبارسنجی فرم ها در PHP

نام : *

ایمیل: *

وب سایت:

توضیحات:

جنسیت: زن مرد*

ارسال اطلاعات

در آموزش بعدی نحوه ی جلوگیری از خالی شدن فیلدهای فرم، زمانی که کاربر روی دکمه submit کلیک می کند را نشان خواهیم داد.

آموزش PHP-حفظ مقادیر فرم

در این آموزش نحوه ی جلوگیری از خالی شدن فیلدهای فرم، زمانی که کاربر روی دکمه submit کلیک می کند را نشان خواهیم داد.

حفظ مقادیر فرم در PHP

زمانی که کاربر، مقادیر فیلدهای فرم را تنظیم و روی دکمه submit کلیک می کند، تمام فیلدها، مقادیرشان را از دست می دهند. حالا تصور نمایید که کاربر، در ورود اطلاعات یک فیلد (مثلاً فیلد "ایمیل") خطایی داشته باشد، بنابراین باید از ابتدا شروع به وارد کردن اطلاعات کند...!

بمنظور حفظ مقادیر فیلدهای فرم، برای فیلدهای متنی، باید یک کد PHP کوچک در خصوصیت value تگ <input> قرار دهیم. همچنین برای عنصر <textarea> باید کد PHP را بین تگ های باز و بسته <textarea> و </textarea> قرار دهیم. این قطعه کدها، مقادیر متغیرهای $name, $email, $website, $comment را نمایش می دهند.

سپس، در مورد فیلد "جنسیت" باید مشخص کنیم که کدام گزینه انتخاب شده است. بنابراین باید خصوصیت ckecked آنرا دستکاری کنیم (نه خصوصیت value):

```
نام : <input type="text" name="name" value="<?php echo $name;?>">

ایمیل : <input type="text" name="email" value="<?php echo $email;?>">

وب سایت : <input type="text" name="website" value="<?php echo $website;?>">

توضیحات : <textarea name="comment" rows="5" cols="40"><?php echo $comment;?></textarea>

جنسیت:
<input type="radio" name="gender"
<?php if (isset($gender) && $gender=="female") echo "checked";?>
 value="female">زن
<input type="radio" name="gender"
<?php if (isset($gender) && $gender=="male") echo "checked";?>
 value="male">مرد
```

مثال تکمیل شده اعتبارسنجی فرم در PHP

در زیر، کد تکمیل شده اعتبارسنجی فرم در PHP آمده است:

مثال

```
 <!DOCTYPE HTML>
<html>
<head>
<style>
  .error {color: #FF0000;}
```

```
  span{min-width: 200px;float: right;}
</style>
</head>
<body>

<?php
// define variables and set to empty values
$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = $comment = $website = "";

if ($_SERVER["REQUEST_METHOD"] == "POST")
{
   if (empty($_POST["name"]))
     {$nameErr = "Name is required";}
   else
     {
     $name = test_input($_POST["name"]);
     // check if name only contains letters and whitespace
     if (!preg_match("/^[a-zA-Z ]*$/",$name))
       {
       $nameErr = "Only letters and white space allowed";
       }
     }

   if (empty($_POST["email"]))
     {$emailErr = "Email is required";}
   else
     {
     $email = test_input($_POST["email"]);
     // check if e-mail address syntax is valid
     if (!preg_match("/([\w\-]+\@[\w\-]+\.[\w\-]+)/",$email))
       {
       $emailErr = "Invalid email format";
       }
     }

   if (empty($_POST["website"]))
     {$website = "";}
```

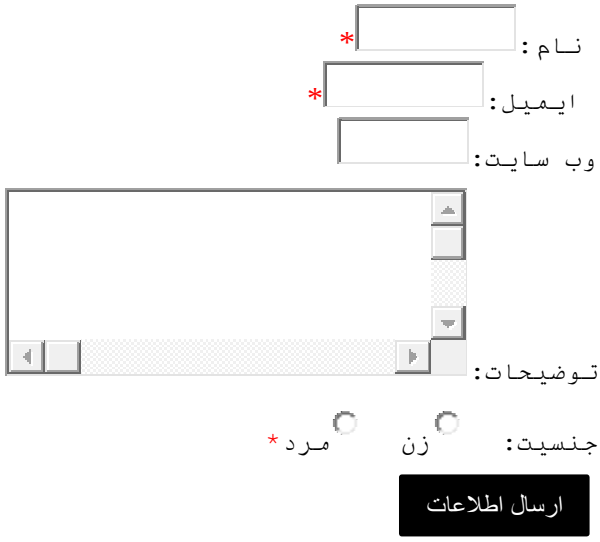
```
   else
     {
     $website = test_input($_POST["website"]);
     // check if URL address syntax is valid (this regular expression
also allows dashes in the URL)
     if (!preg_match("/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-
9+&@#\/%?=~_|!:,.;]*[-a-z0-9+&@#\/%=~_|]/i",$website))
       {
       $websiteErr = "Invalid URL";
       }
     }

   if (empty($_POST["comment"]))
     {$comment = "";}
   else
     {$comment = test_input($_POST["comment"]);}

   if (empty($_POST["gender"]) || $_GET["gender"]=="undefined")
     {$genderErr = "Gender is required";}
   else
     {$gender = test_input($_POST["gender"]);}
}

function test_input($data)
{
     $data = trim($data);
     $data = stripslashes($data);
     $data = htmlspecialchars($data);
     return $data;
}
?>
<h2>مثال اعتبارسنجی فرم ها در PHP</h2>
<form method="post" action="<?php echo
htmlspecialchars($_SERVER["PHP_SELF"]);?>">
<div>
  <span>نام:</span><input type="text" name="name" value="<?php echo
$name;?>">
  <span class="error">*</span>
```

```
</div>
<div>
  <span>ایمیل:</span><input type="text" name="email" value="<?php echo
$email;?>">
  <span class="error">*</span>
</div>
<div>
  <span>وب سایت:</span>
  <input type="text" name="website" value="<?php echo $website;?>">
  <span class="error"></span>
</div>
<div>
  <span>توضیحات:</span><textarea name="comment" rows="5" cols="40">
                          <?php echo $comment;?>
                      </textarea>
  <span class="error">*</span>
</div>
<div>
<span>جنسیت:</span>
<input type="radio" name="gender" value="female" <?php if
(isset($gender) && $gender=="female") echo "checked";?> >زن
<input type="radio" name="gender" value="male" <?php if (isset($gender)
&& $gender=="male") echo "checked";?> >مرد
</div>
</div><input type="submit" name="submit" value="ارسال اطلاعات"></div>
</form>

<?php
if ($nameErr!="" || $emailErr!="" || $genderErr!="" || $websiteErr!="")
echo "<span style='color:red' >
        $nameErr <br /> $emailErr <br /> $genderErr <br /> $websiteErr
     </span>";
else
{
echo "<br /><h2>خروجی کدتان</h2>";
echo "نام :$name";
echo "<br />";
echo "ایمیل: $email";
```

```
echo "<br />";
echo "وب سایت: $website";
echo "<br />";
echo "توضیحات: $comment";
echo "<br />";
echo "جنسیت: $gender";
?>
}
</body>
</html>
```

خروجی کد بالا:

مثال اعتبارسنجی فرم ها در PHP



فصل ۳:آموزش پیشرفته PHP

صفر تا صد PHP -تاریخ

تابع Date در PHP

این تابع، رشته ای از داده ها را به عنوان پارمتر می گیرد و نتایج آنها را به صورت زمان یا تاریخ نمایش می دهد.

در حقیقت با استفاده از این تابع می توان تاریخ یا زمان را در شکل های مختلف نشان داد.

نحوه استفاده:

```
date(format,timestamp)
```

پارامتر	توضیحات
format	الزامی است، فرمت تاریخ یا زمان را مشخص می کند.
timestamp	اختیاری است، یک برچسب زمان که باید به فرمتی که در پارامتر اول مشخص کرده ایم تبدیل شود(به صورت پیشفرض تاریخ جاری در نظر گرفته می شود).

بعضی از کاراکترهایی که می توان به عنوان پارامتر الزامی، در تابع date استفاده کرد:

- a: صبح یا عصر، نماینده am و pm است.
- A: صبح یا عصر، نماینده AM و PM است.
- B: زمان اینترنت سوئچ (swatch) که یک زمان جهانی است.
- C: تاریخ ISO 8601، تاریخ به صورت YYYY_MM_DD . این کد فرمت در نسخه php 5 ارائه شده است.
- d: روز به صورت عدد دو رقمی از ۰۱ تا ۳۱
- D: روز هفته به صورت ۳ کاراکتر مانند :Mon و Sun
- F: ماه در فرمت متنی . محدوده آن از January تا December است.
- g: ساعت در فرمت ۱۲ ساعت از ۱ تا ۱۲
- G: ساعت در فرمت ۲۴ ساعت از ۰ تا ۲۳
- h: ساعت در فرمت ۱۲ ساعت از ۰۱ تا ۱۲
- H: ساعت در فرمت ۲۴ ساعت از ۰۰ تا ۲۳
- i: دقیقه به صورت عدد دو رقمی از ۰۰ تا ۵۹
- I: این مقدار بولی ساعت را یک ساعت به جلو و عقب می برد (در آخر تابستان و اول فروردین)
- j: روز را در یک ماه نشان می دهد. از ۱ تا ۳۱
- l: روز را در هفته از Sonday تا Monday مشخص می کند.
- L: این مقدار بولی سال کبیسه را مشخص می کند و اگر سال کبیسه باشد مقدار یک را بر می گرداند.
- m: ماه را در فرمت دو رقمی از ۰۱ تا ۱۲ مشخص می کند.
- M: ماه در فرمت ۳ کاراکتر مانند: Jan و Dec
- n: ماه در فرمت عددی بدون صفر از ۱ تا ۱۲
- o: تفاوت زمان منطقه جغرافیایی موجود و GMT را در واحد ساعت بیان می کند.
- r: زمان و تاریخ در فرمت RFC822
- s: ثانیه در محدوده ۰۰ تا ۵۹
- S: پسوند های ترتیبی اعداد در فرمت دو کاراکتر .مانند: st۱ یا th۲۳
- t: تعداد روز های ماه را از ۲۸ تا ۳۱ مشخص می کند.

- T: تنظیمات منطقه زمانی سرور در فرمت ۳ کاراکتر مانند EST
- U: تعداد کل ثانیه ها از اول ژانویه ۱۹۷۰ تاکنون را بیان می کند.
- w: روز هفته در یک رقم. ۰ برای یکشنبه و ۶ برای شنبه
- W: شماره هفته از اول سال. مانند ۲۱ برای نشان دادن هفته ۲۱ام سال
- y: سال در فرمت دورقمی، ۰۵ برای ۲۰۰۵
- Y: سال در فرمت چهار رقمی، مانند ۲۰۰۶
- z: روز را مشخص می کند. محدوده آن ۰ تا ۳۶۵ است.
- Z: افست منطقه زمانی را بر حسب ثانیه بیان می کند. محدوده آن بین ۴۳۲۰۰- و ۴۳۲۰۰ است.

نکته: کاراکتر هایی که در لیست بالا نباشند، مستقیماً چاپ می شوند. مانند: / و , و ... .

مثال:

```
<?php
echo date("Y/m/d") . "<br />";
echo date("Y.m.d") . "<br />";
echo date("Y-m-d");
?>
```

خروجی کد بالا:

```
2009/05/11
2009.05.11
2009-05-11
```

پارامتر timestamp در تابع date

تابع ()mktime یک برچسب زمان یونیکسی برمی گرداند که می توان از آن، به عنوان پارامتر اختیاری تابع ()date استفاده کرد.

برچسب زمان یونیکسی: منظور تعداد ثانیه هایی است که از زمان (January 1 1970 00:00:00 GMT) تا زمانی که در تابع mktime مشخص شده است.

نحوه استفاده:

```
mktime(hour,minute,second,month,day,year,is_dst)
```

مثال: در مثال زیر با استفاده از تابع mktime و date تاریخ فردا را چاپ می کنیم:

```
<?php
$tomorrow = mktime(0,0,0,date("m"),date("d")+1,date("Y"));
```

```
echo "Tomorrow is ".date("Y/m/d", $tomorrow);
?>
```

خروجی کد بالا: (البته هنگام اجرای مثال)

```
Tomorrow is 2013/05/12
```

صفر تا صد PHP -دستور include

درج کردن یک فایل PHP در یک فایل PHP دیگر

شما می توانید با استفاده از دو تابع include یا require محتویات یک فایل PHP را داخل یک فایل PHP دیگر درج کنید، البته قبل از اینکه فایل دوم اجرا شود.

این دو تابع در همه موارد یکسان عمل می کنند، بجز در چگونگی برخورد با خطا:

- include: پیغام خطا را تولید می کند، اما ادامه اسکریپت اجرا خواهد شد.
- require: پیغام خطا را تولید می کند و اجرای اسکریپت متوقف خواهد شد.

موارد استفاده دو تابع include و require می تواند شامل موارد زیر باشد:

- توابع
- Header
- Footer
- المانهایی که در چندین صفحه استفاده می شود مانند منوها، کامپوننت تاریخ و ...

شما می توانید یک فایل استاندارد Header یا Footer و یا منو ایجاد کنید و در همه صفحات پروژه تان از آنها استفاده کنید، و موقعی که نیاز به تغییر داشت، تنها با تغییر یک فایل، کل پروژه تان را تازه سازی کنید.

دستور include_once و require_once:

دو تابع بالا دقیقاً مثل include و require عمل می کنند، با این تفاوت که اگر فایل مورد نظر قبلاً خوانده شده است، دیگر خوانده نمی شود.

مثال ۱:

فرض کنید یک فایل Header با نام "header.php" دارید، برای اضافه کردن آن به صفحه ای از پروژه از کد زیر استفاده کنید:

```
<html>
<body>

<?php include("header.php"); ?>
```

```
<h1>Welcome to my home page!</h1>
<p>Some text.</p>

</body>
</html>
```

دستورات include, include_once, require و require_once را می توان بدون پرانتز نیز بکار برد.

مثال ۲:

فرض کنید یک فایل منو طبق زیر با نام "menu.php" داریم، که باید به تمام صفحات پروژه اضافه شود:

```
<a href="/default.php">Home</a>
<a href="/tutorials.php">Tutorials</a>
<a href="/references.php">References</a>
<a href="/examples.php">Examples</a>
<a href="/about.php">About Us</a>
<a href="/contact.php">Contact Us</a>
```

در زیر فایل "menu.php" را به صفحه دلخواهمان اضافه کرده ایم:

```
<html>
<body>

<div class="leftmenu">
<?php include("menu.php"); ?>
</div>

<h1>Welcome to my home page.</h1>
<p>Some text.</p>

</body>
</html>
```

اگر بعد از اجرای فایل بالا در IE ، روی صفحه راست کلیک کرده و View Source را کلیک کنید، کد html شبیه زیر خواهیم داشت:

```
<html>
<body>

<div class"leftmenu">
<a href="/default.php">Home</a>
<a href="/tutorials.php">Tutorials</a>
<a href="/references.php">References</a>
<a href="/examples.php">Examples</a>
<a href="/about.php">About Us</a>
```

```
<a href="/contact.php">Contact Us</a>
</div>

<h1>Welcome to my home page!</h1>
<p>Some text.</p>

</body>
</html>
```

مثال ۳: خطا در تابع include:

در مثال زیر فایل wrongFile.php وجود ندارد و همان طور که مشاهده می کنید بعد از چاپ پیغام خطا ادامه اسکریپت اجرا می شود.

```
<html>
<body>

<?php
include("wrongFile.php");
echo "Hello World!";
?>

</body>
</html>
```

پیغام خطا:

```
Warning: include(wrongFile.php) [function.include]:
failed to open stream:
No such file or directory in C:\home\website\test.php on line 5

Warning: include() [function.include]:
Failed opening 'wrongFile.php' for inclusion
(include_path='.;C:\php5\pear')
in C:\home\website\test.php on line 5

Hello World!
```

مثال ۴:پیغام خطا در تابع require:

```
<html>
<body>

<?php
```

```
require("wrongFile.php");
echo "Hello World!";
?>

</body>
</html>
```

پیغام خطا:

```
Warning: require(wrongFile.php) [function.require]:
failed to open stream:
No such file or directory in C:\home\website\test.php on line 5

Fatal error: require() [function.require]:
Failed opening required 'wrongFile.php'
(include_path='.;C:\php5\pear')
in C:\home\website\test.php on line 5
```

بعد از ایجاد خطا، ادامه اجرای اسکریپت متوقف می شود، و در حقیقت خطی که قرار است Hello World را چاپ کند، اجرا نمی شود.

پیشنهاد می شود از تابع require بجای include استفاده شود، چون اسکریپت ها بعد از برخورد با خطا نباید ادامه پیدا کنند.

صفر تا صد PHP -فایل ها

باز کردن فایل در PHP

تابع ()fopen برای باز کردن فایل ها در php مورد استفاده قرار می گیرد.

پارامتر اول این تابع شامل نام فایلی است که می خواهیم باز شود و پارامتر دوم مشخص می کند که فایل در چه حالتی باز شود:

```
<html>
<body>

<?php
$file=fopen("welcome.txt","r");
?>

</body>
</html>
```

فایل ممکن است در یکی از حالات زیر باز شود:

حالت	توضیحات

R	فقط خواندنی. از ابتدای فایل آغاز می شود
r+	خواندنی/نوشتنی. از ابتدای فایل آغاز می شود
W	فقط نوشتنی. فایل را باز و محتویات آن را پاک می کند، یا اگر فایلی وجود نداشت آن را ایجاد می کند
w+	خواندنی/نوشتنی. فایل را باز و محتویات آن را پاک می کند، یا اگر فایلی وجود نداشت آن را ایجاد می کند
A	افزودن(الحاق). فایل را باز و در انتهای آن می نویسد، یا اگر فایلی وجود نداشت آن را ایجاد می کند
a+	خواندنی/افزودنی. محتویات فایل را با نوشتن در انتهای آن حفظ می کند
X	فقط نوشتنی. یک فایل جدید ایجاد می کند.اگر فایل درحال حاضر وجود داشته باشد false و یک خطا برمیگرداند
x+	خواندنی/نوشتنی. یک فایل جدید ایجاد می کند. اگر فایل درحال حاضر وجود داشته باشد false و یک خطا برمیگرداند

نکته: اگر تابع ()fopen قادر به باز کردن فایل مشخص شده نباشد, مقدار "۰" یا (false) را برمی گرداند.

مثال:

در مثال زیر اگر تابع ()fopenقادر به باز کردن فایل مشخص شده نباشد یک پیغام تولید می کند:

```
<html>
<body>

<?php
$file=fopen("welcome.txt","r") or exit("Unable to open file!");
?>

</body>
</html>
```

بستن فایل در PHP

تابع ()fclose برای بستن یک فایل باز استفاده می شود:

```
<?php
$file = fopen("test.txt","r");

//some code to be executed

fclose($file);
?>
```

چک کردن پایان فایل

تابع ()feof بررسی می کند که آیا به پایان فایل (eof) رسیده ایم یا نه.

تابع ()feof برای ایجاد حلقه در داده های با طول نامشخص مناسب است.

توجه: شما نمی توانید از فایل هایی که در حالت w و a و x باز شده اند، بخوانید!

```
if (feof($file)) echo "End of file";
```

خواندن خط به خط یک فایل

تابع ()fgets برای خواندن یک خط از یک فایل، مورد استفاده قرار می گیرد.

نکته: بعد از هر بار فراخوانی این تابع، اشاره گر فایل، به خط بعدی جابه جا می شود.

مثال:

مثال زیر یک فایل را تا پایان فایل، خط به خط می خواند:

```
<?php
$file = fopen("welcome.txt", "r") or exit("Unable to open file!");
//Output a line of the file until the end is reached
while(!feof($file))
  {
  echo fgets($file). "<br />";
  }
fclose($file);
?>
```

خواندن کاراکتر به کاراکتر یک فایل

تابع ()fgetc برای خواندن یک کاراکتر تنها از یک فایل، مورد استفاده قرار می گیرد.

نکته: بعد از فراخوانی این تابع، اشاره گر فایل به کاراکتر بعدی جابه جا می شود.

مثال:

مثال زیر یک فایل را تا پایان فایل، کاراکتر به کاراکتر می خواند:

```
<?php
$file=fopen("welcome.txt","r") or exit("Unable to open file!");
while (!feof($file))
  {
  echo fgetc($file);
  }
fclose($file);
?>
```

صفر تا صد PHP -آپلود فایل

ارسال یا آپلود فایل در PHP

برای ارسال فایل از Client به Server مراحل زیر را طی کنید:

۱- ایجاد یک فرم HTML ارسال فایل

به فرم HTML زیر برای ارسال فایل توجه کنید:

```
<html>
<body>

<form action="upload_file.php" method="post" enctype="multipart/form-data">
  <label for="file">Filename:</label>
  <input type="file" name="file" id="file" />
  <br />
  <input type="submit" name="submit" value="Submit" />
</form>

</body>
</html>
```

- **خاصیت enctype در تگ <form>:** هنگامی که یک فرم، داده های دودویی، مثل محتوای یک فایل را برای ارسال شدن نیاز دارد، این خاصیت با مقدار "multipart/form-data" پر می شود، در حقیقت مشخص می کند که چه نوع محتوایی هنگام ارائه ی فرم استفاده می شود.
- **خاصیت action در تگ <form>:** اگر submit اتفاق افتاد، اطلاعات فرم HTML به صفحه ای که در این خصوصیت مشخص شده ارسال می شود.(به صورت پیش فرض صفحه جاری در نظر گرفته می شود)
- **خاصیت type در تگ <input>:** با مقدار "file" پر شده است و مشخص می کند که ورودی کاربر، باید به عنوان یک فایل پردازش شود.(یک تکس باکس به همراه دکمه Browse، جهت جستجوی فایل از کامپیوتر کاربر)

نکته: دادن سطح دسترسی ارسال فایل به کاربران ریسک بزرگی برای امنیت است، بنابراین تنها به کاربران مورد اعتماد اجازه ی ارسال فایل بدهید.

۲- ایجاد یک اسکریپت ارسالی

فایل "upload_file.php" که در فرم HTML بالا در تگ <form> مشخص کردیم، می تواند چیزی شبیه زیر باشد:

```
<?php
if ($_FILES["file"]["error"] > 0)
  {
  echo "Error: " . $_FILES["file"]["error"] . "<br />";
  }
else
  {
  echo "Upload: " . $_FILES["file"]["name"] . "<br />";
  echo "Type: " . $_FILES["file"]["type"] . "<br />";
  echo "Size: " . ($_FILES["file"]["size"] / 1024) . " Kb<br />";
  echo "Stored in: " . $_FILES["file"]["tmp_name"];
```

```
  }
?>
```

با استفاده از متغیر سراسری $_FILES می توانید فایل های مشخص شده توسط کاربر را از client به Server ارسال کنید، در حقیقت $_FILES یک آرایه دوبعدی از پیش تعریف شده است که با استفاده از ایندکس مقداری آن می توانید به مقادیر زیر دسترسی داشته باشید:

- $_FILES["file"]["name"] : نام فایل ارسالی
- $_FILES["file"]["type"] : نوع فایل ارسالی
- $_FILES["file"]["size"] : اندازه فایل ارسالی با واحد بایت
- $_FILES["file"]["tmp_name"] : نام کپی موقت از فایل که روی سرور ذخیره شده
- $_FILES["file"]["error"] : کد خطا که از ارسال فایل به وجود آمده

توجه: به دلایل امنیتی, باید محدودیتی را روی آنچه که کاربر ارسال می کند قرار دهیم.

۳- محدودیت های ارسال

دراین اسکریپت چند شرط را برای ارسال فایل اضافه می کنیم. کاربر فقط قادر است فایل های "jpg." یا "gif." را ارسال کند و اندازه ی فایل باید کمتر از "KB۲۰" باشد:

```
<?php
if ((($_FILES["file"]["type"] == "image/gif")
|| ($_FILES["file"]["type"] == "image/jpeg")
|| ($_FILES["file"]["type"] == "image/pjpeg"))
&& ($_FILES["file"]["size"] < 20000))
  {
  if ($_FILES["file"]["error"] > 0)
    {
    echo "Error: " . $_FILES["file"]["error"] . "<br />";
    }
  else
    {
    echo "Upload: " . $_FILES["file"]["name"] . "<br />";
    echo "Type: " . $_FILES["file"]["type"] . "<br />";
    echo "Size: " . ($_FILES["file"]["size"] / 1024) . " Kb<br />";
    echo "Stored in: " . $_FILES["file"]["tmp_name"];
    }
  }
else
  {
  echo "Invalid file";
```

```
  }
?>
```

نکته: در کد بالا برای اینکه IE فایل های با فرمت jpg را بشناسد، باید نوع pjpg باشد و برای FireFox باید jpeg باشد.

۴- ذخیره فایل های ارسالی

مثال های بالا یک کپی موقت از فایل های ارسال شده در پوشه ی موقت php روی سرور ایجاد می کنند و وقتی که اسکریپت به پایان رسید، از بین می روند. بنابراین باید آن ها را به یک مکان دیگر منتقل کنیم.

```
<?php
if ((($_FILES["file"]["type"] == "image/gif")
|| ($_FILES["file"]["type"] == "image/jpeg")
|| ($_FILES["file"]["type"] == "image/pjpeg"))
&& ($_FILES["file"]["size"] < 20000))
  {
  if ($_FILES["file"]["error"] > 0)
    {
    echo "Return Code: " . $_FILES["file"]["error"] . "<br />";
    }
  else
    {
    echo "Upload: " . $_FILES["file"]["name"] . "<br />";
    echo "Type: " . $_FILES["file"]["type"] . "<br />";
    echo "Size: " . ($_FILES["file"]["size"] / 1024) . " Kb<br />";
    echo "Temp file: " . $_FILES["file"]["tmp_name"] . "<br />";

    if (file_exists("upload/" . $_FILES["file"]["name"]))
      {
      echo $_FILES["file"]["name"] . " already exists. ";
      }
    else
      {
      move_uploaded_file($_FILES["file"]["tmp_name"],
      "upload/" . $_FILES["file"]["name"]);
      echo "Stored in: " . "upload/" . $_FILES["file"]["name"];
      }
    }
  }
else
  {
  echo "Invalid file";
  }
?>
```

در اسکریپت بالا در قسمت if چک می شود که آیا فایل ارسال شده در پوشه "upload" قبلاً وجود داشته یا نه، اگر موجود نباشد، آن را به پوشه مشخص شده منتقل یا کپی می کند.

نکته: این مثال فایل را در یگ پوشه ی جدید به نام "upload" ذخیره می کند، احتمالاً شما نیاز خواهید داشت که نام فایل و مسیر آن را در پایگاه داده خود ذخیره کنید بنابراین در ادامه مثال بالا به راحتی می توانید با استفاده از دستور insert اینکار را انجام دهید.

آموزش PHP-متغییر cookie

Cookie ها در php

Cookie چیست؟ cookie یک فایل کوچک است که سرور آن را درون کامپیوتر کاربر قرار می دهد. هر بار که همان کامپیوتر یک صفحه را از طریق مرورگرش درخواست می کند، مقدار cookie همراه با درخواست به سرور ارسال می شود. با php، شما می توانید هم مقادیر cookie را بسازید و هم بازیابی کنید.

توجه: یک cookie اغلب برای شناسایی یک کاربر استفاده می شود.

چگونه یک cookie ایجاد کنیم؟

تابع ()setcookie برای ایجاد cookie استفاده می شود.

نکته: تابع ()setcookie باید قبل از تگ <html> قرار گیرد.

نحوه استفاده:

```
setcookie(name, value, expire, path, domain);
```

مثال:

در مثال زیر, یک cookie با نام "user" ایجاد می کنیم و مقدار "Ali Ahmadi" را به آن اختصاص می دهیم. ما همچنین مشخص می کنیم که cookie پس از یک ساعت از بین برود:

```
<?php
setcookie("user", "Ali Ahmadi", time()+3600);
?>

<html>
.....
```

نکته: مقدار cookie به طور خودکار هنگام ارسال کد(URLencoding) می شود و موقع دریافت رمزگشایی می شود (برای پیشگیری از رمزگذاری, از ()setrawcookie استفاده کنید.

مثال:

شما می توانید زمان انقضای cookie را به روشی دیگر مشخص کنید، احتمالاً این راه ساده تر از مشخص کردن ثانیه هاست:

```
<?php
$expire=time()+60*60*24*30;
setcookie("user", "Ali Ahmadi", $expire);
?>

<html>
.....
```

در مثال بالا زمان انقضا با مقدار "۱ ماه" پر شده است(*۶۰sec * 60 min * 24 hours * 30 days*)

چگونه مقدار یک cookie را بازیابی کنیم؟

متغیر $_COOKIE در php برای بازیابی مقدار یک cookie مورد استفاده قرار می گیرد.

مثال:

در مثال زیر, مقدار cookie با نام "user" را بازیابی می کنیم و بر روی صفحه، نمایش می دهیم:

```
<?php
// Print a cookie
echo $_COOKIE["user"];

// A way to view all cookies
print_r($_COOKIE);
?>
```

در مثال زیر با استفاده از تابع ()isset چک می کنیم که cookie با نام "user" قبلاً set شده است یا نه؟

```
<html>
<body>

<?php
if (isset($_COOKIE["user"]))
  echo "Welcome " . $_COOKIE["user"] . "!<br />";
else
  echo "Welcome guest!<br />";
?>

</body>
</html>
```

چگونه یک cookie را حذف کنیم؟

هنگامی که می خواهید یک cookie را حذف کنید باید اطمینان دهید که تاریخ انقضاء آن به پایان رسیده است.

مثال حذف:

```
<?php
// set the expiration date to one hour ago
setcookie("user", "", time()-3600);
?>
```

اگر یک مرورگر cookieها را پشتیبانی نکند چه باید کرد؟

اگر برنامه شما با مرورگرهایی سروکار دارد که cookieها را پشتیبانی نمی کنند، باید روشهای دیگری رابرای انتقال اطلاعات از یک صفحه به صفحه ی دیگر انتخاب کنید. یک روش انتقال اطلاعات از طریق فرم ها و متغیرهای از پیش تعریف شده است(فرم ها و متغیر های از پیش تعریف شده، قبل از این توضیح داده شده اند).

در فرم زیر وقتی که کاربر دکمه ی "submit" را کلیک می کند، ورودی های کاربر را به "welcome.php" می فرستد:

```
<html>
<body>

<form action="welcome.php" method="post">
   Name: <input type="text" name="name" />
   Age: <input type="text" name="age" />
   <input type="submit" />
</form>

</body>
</html>
```

بازیابی اطلاعات در صفحه "welcome.php" شبیه زیر خواهد بود:

```
<html>
<body>

Welcome <?php echo $_POST["name"]; ?>.<br />
You are <?php echo $_POST["age"]; ?> years old.

</body>
</html>
```

صفر تا صد PHP -متغییر session

متغیرهای جلسه (session) در php

وقتی یک Application تحت ویندوز را باز می کنید و چند تغییر روی آن می دهید و سپس آنرا می بندید، کامپیوتر می داند شما چه کسی هستید، چه موقع درخواست را آغاز و چه موقع آن را به اتمام رسانده اید. اما در اینترنت یک مشکل وجود دارد، سرور وب نمی داند شما که هستید و چه می کنید چون آدرس HTTP، حالت State را پشتیبانی نمی کند.

یک جلسه یا session این مشکل را برای شما حل می کند، در حقیقت اطلاعات کاربر(مثل: نام کاربری و غیره) بر روی سرور برای استفاده های بعدی ذخیره می شود. اما اطلاعات جلسه موقتی اند و پس از اینکه کاربر، وب سایت را رها کند، اطلاعات حذف خواهند شد، اگر نیاز به ذخیره سازی دائمی دارید باید داده ها را در یک پایگاه داده ذخیره کنید.

جلسه ها با ایجاد یک شماره ID منحصر به فرد برای هر بیننده و ذخیره متغیرها براساس این شماره ها کار می کنند.

نکته: در یک پروژه تحت وب، بعد از ایجاد یک متغیر جلسه، مقدار آن برای همه ی صفحات پروژه قابل دستیابی است.

آغاز یک جلسه (session) در php

قبل از اینکه بتوانید اطلاعات کاربر، را در جلسه تان ذخیره کنید، ابتدا باید یک جلسه آغاز شود، تابع session_start() برای این منظور استفاده می شود.

نکته: تابع session_start() باید قبل از تگ <html> بیاید:

```
<?php session_start(); ?>

<html>
<body>

</body>
</html>
```

تعریف یک متغیر جلسه (session) در PHP

روش صحیح برای ذخیره و بازیابی متغیرهای جلسه، استفاده از $_SESSION است:

```
<?php
session_start();
// store session data
$_SESSION['views']=1;
?>

<html>
<body>

<?php
```

```
//retrieve session data
echo "Pageviews=". $_SESSION['views'];
?>

</body>
</html>
```

خروجی کد بالا:

```
Pageviews=1
```

در مثال زیر یک "شمارنده بازدید صفحه" ایجاد کرده ایم، تابع isset() بررسی می کند که آیا متغیر "views" در حال حاضر تنظیم شده است یا نه:

```
<?php
session_start();

if(isset($_SESSION['views']))
   $_SESSION['views']=$_SESSION['views']+1;
else
   $_SESSION['views']=1;
echo "Views=". $_SESSION['views'];
?>
```

از بین بردن جلسه (session) در PHP

برای از بین بردن جلسه ها در PHP می توان از توابع زیر استفاده کرد:

- **unset()** یک جلسه مشخص را به عنوان پارامتر دریافت می کند و آنرا از بین می برد.
- **session_destroy()** کلیه جلسه های ایجاد شده را از بین می برد.

```
<?php
unset($_SESSION['views']);
?>

<?php
session_destroy();
?>
```

جی‌میل چند وقتی آمده است و چشمان تمامی کاربران را به خود دوخته است. یک گیگابایت فضای رایگان، سرعت بالا. دیگر چه می‌خواهید؟ اما به همین سادگی‌ها هم نیست. کمی باید بیشتر بر روی جی‌میل کار کنیم.

امیر عظمتي در مقاله‌ای، چگونگی کارکردن با جی‌میل را به طور کامل نوشته است، اما او به عنوان کسی که از جی‌میل راضی است و از آن استفاده می‌کند، این مقاله را نوشته است. حال من می‌خواهم به عنوان کسی که دید چندان خوبی به جی‌میل ندارم، درباره‌اش حرف بزنم.

نامه‌ای از یکی از دوستانم دریافت می‌کنم که به زبان انگلیسی نوشته شده است. در حین خواندن نامه، تبلیغاتی توجه مرا جلب می‌کند که می‌بینم یک‌جورایی به نامه‌ام در ارتباط است. یعنی چه؟ بله. گوگل هم خودش گفته است. گوگل تمامی نامه‌ها را چک می‌کند. نامه‌هایی که تنها باید دو نفر از آن‌ها خبر داشته باشند: گیرنده و فرستنده. کاری ندارم که گوگل توسط روبات نامه‌ها را چک می‌کند یا انسان! مهم این است که نامه‌های ما را زیر نظر می‌گیرد و این یعنی تجاوز به حریم شخصی. من در هیچ صورتی، حاضر نیستم به حریم شخصیم در این دنیای مجازی تجاوز شود.

اما آخر گوگل یک گیگابایت ایمیل رایگان می‌دهد. خیلی خب، صبر داشته باشید. قبل از عجله ببینید واقعا یک گیگابایت به دردتان می‌خورد یا فقط دوست دارید بگویید یک گیگابایت ایمیل دارم، در حالی که هنوز یک مگابایت آن را هم استفاده نکرده‌اید. یاهو هم صدمگابایت ایمیل می‌دهد.

قبول دارم سیستم لیبل‌گذاری جی‌میل فوق‌العاده است. به طوری که در عرض کمتر از پنج دقیقه، می‌توانید بیش از صد نامه را آرشیو بندی کنید.

جی‌میل اچ‌تی‌امال را پشتیبانی نمی‌کند. از مجلات خارجی برایم نیوزلتر می‌آید که حاوی مقالات و نکته‌های فراوان است. اما جی‌میل نمی‌تواند آن‌ها را مثل بچه آدمیزاد نشان دهد و همه را به صورت متن‌های ساده نشان می‌دهد. حتی متن بولد شده را هم نمی‌تواند نشان دهد. این دیگر چه صیغه‌ای است!

و اما فجیع‌ترین مصیبت برای کاربران ایرانی جی‌میل، مشکلاتش با فارسی است. بارها و بارها ایمیل‌هایی دریافت کرده‌ام که فارسی نوشته شده بودند، اما به هیچ عنوان نمی‌توانستم بخوانم. کاربران دیگر هم همین مشکل را دارند.

البته جی‌میل فعلا نسخه‌ی بتای خود را ارائه داده است و همه ما امیدواریم که در نسخه نهایی، تمامی این مشکلات برطرف شده باشد. اما من اگر تمامی این مشکلات هم برطرف شود، حاضر نیستم به خاطر کمی سرعت بالای جی‌میل و سیستم لیبل‌گذاری‌اش، حریم شخصی‌ام را در خطر بیندازم. شما را نمی‌دانم.

+ چگونگی حل مشکلات فارسی در جی‌میل

اگر شما هم جزء آن دسته از افراد هستید که با نامه‌های فارسی در جی‌میل مشکل دارید، و هنوز هم اصرار به استفاده از آن دارید، نگران نباشید. من دیروز راه حلی پیدا کردم که ممکن است جواب دهد (من از همین راه برای خواندن نامه‌ای که نتوانستم بخوانمش استفاده کردم):

فرمت را به همان صورتی که هست باقی بگذارید (همیشه به صورت UTF-۸ است). سپس نامه را به یکی دیگر از میل‌باکس‌هایتان فوروارد (forward) کنید و بروید نامه‌اتان را با خیال راحت بخوانید. البته ممکن است این راه برای تمامی نامه‌ها جواب ندهد. اما برای من که جواب داد.

ارسال ایمیل در PHP

در PHP با استفاده از تابع ()mail می توان از داخل یک اسکریپت ایمیل ارسال کرد.

نحوه استفاده:

```
mail(to,subject,message,headers,parameters)
```

پارامتر	توضیحات
To	الزامی است، دریافت کننده یا دریافت کنندگان ایمیل را مشخص می کند.
Subject	الزامی است، موضوع ایمیل را مشخص می کند. توجه داشته باشید که این پارامتر Enter را قبول نمی کند(\n)
Message	الزامی است، متن ایمیل را مشخص می کند.
Headers	اختیاری است، headerهای دلخواه مانند From,CC,BCC و غیره را مشخص می کند، هر کدام از این Headerها باید با کاراکتر "\n" یا "\r" از یکدیگر جدا شوند
Parameters	اختیاری است، یک پارامتر اضافی برای برنامه ارسال ایمیل مشخص می کند.

توجه: برای تغییر تنظیمات پیکربندی ارسال ایمیل می توانید از فایل php.ini استفاده کنید، احتمالاً اگر می خواهید مطالب بیشتری راجع به توابع ارسال ایمیل بدانید از لینک روبرو استفاده کنید:

مثال: در مثال زیر، یک متن ساده را ارسال کرده ایم:

```
<?php
$to = "someone@example.com";
$subject = "Test mail";
$message = "Hello! This is a simple email message.";
$from = "someonelse@example.com";
$headers = "From:" . $from;
mail($to,$subject,$message,$headers);
echo "Mail Sent.";
?>
```

فرم ارسال ایمیل

همان طور که می دانید با استفاده از دستور echo می توان هر متغیر یا رشته ای را چاپ کرد، این رشته می تواند تگ های HTML باشد، ایجاد فرم HTML از این طریق feedback-form نامیده می شود، به مثال زیر برای ارسال ایمیل توجه کنید:

```
<html>
<body>

<?php
//اگر تکس باکس ایمیل پرشده باشد،ایمیل ارسال می شود
if (isset($_REQUEST['email']))
  {
  //ارسال ایمیل
  $email = $_REQUEST['email'] ;
  $subject = $_REQUEST['subject'] ;
  $message = $_REQUEST['message'] ;
  mail("someone@example.com", "$subject",
  $message, "From:" . $email);
  echo "Thank you for using our mail form";
  }
else
//اگر تکس باکس ایمیل پرنشده باشد، فرم ارسال، نمایش داده می شود
  {
  echo "<form method='post' action='mailform.php'>
  Email: <input name='email' type='text' /><br />
  Subject: <input name='subject' type='text' /><br />
  Message:<br />
  <textarea name='message' rows='15' cols='40'>
  </textarea><br />
  <input type='submit' />
  </form>";
  }
?>

</body>
</html>
```

مثال بالا چه طور کار می کند:

- ابتدا بررسی می کند که آیا فیلد ورودی email پر شده است یا نه
- اگر پرنشده است(مثل وقتی که صفحه تازه دیده می شود) فرم HTML را نمایش می دهد
- اگر پر شده است(بعد از اینکه فرم پر شده) email را ارسال می کند

نکته: این ساده ترین راه برای ارسال email است، اما ایمن نیست. در فصل بعد درمورد آسیب پذیری scriptهای ارسال ایمیل و اینکه چه طور ورودی کاربر را برای ایمن تر کردن آن معتبر کنیم، توضیح داده شده است.

صفر تا صد PHP -امنیت در ایمیل

امنیت در کد ارسال ایمیل

یک ضعف در اسکریپت e-mail فصل قبل وجود دارد. ابتدا، به کد آن نگاهی می اندازیم:

```
<html>
<body>

<?php
//اگر تکست باکس ایمیل پرشده باشد،ایمیل ارسال می شود
if (isset($_REQUEST['email']))
  {
  //ارسال ایمیل
  $email = $_REQUEST['email'] ;
  $subject = $_REQUEST['subject'] ;
  $message = $_REQUEST['message'] ;
  mail("someone@example.com", "Subject: $subject",
  $message, "From: $email" );
  echo "Thank you for using our mail form";
  }
else
//اگر تکست باکس ایمیل پرنشده باشد، فرم ارسال، نمایش داده می شود
  {
  echo "<form method='post' action='mailform.php'>
  Email: <input name='email' type='text' /><br />
  Subject: <input name='subject' type='text' /><br />
  Message:<br />
  <textarea name='message' rows='15' cols='40'>
  </textarea><br />
  <input type='submit' />
  </form>";
  }
?>

</body>
</html>
```

مشکل کد بالا این است که کاربران غیرمجاز می توانند از طریق فرم ورودی، درون header نامه، داده درج کنند.

توضیح بیشتر: چه اتفاقی خواهد افتاد اگر کاربر متن زیر را در فیلد ورودی email در فرم اضافه کند؟

```
someone@example.com%0ACc:person2@example.com
%0ABcc:person3@example.com,person3@example.com,
anotherperson4@example.com,person5@example.com
%0ABTo:person6@example.com
```

تابع ()mail طبق معمول متن بالا را درون header نامه قرار می دهد و اکنون header فیلدهای اضافی cc و Bcc و to را دارد.

وقتی که کاربر بر روی دکمه submit کلیک می کند ,email به تمام آدرس های بالا ارسال خواهد شد!

متوقف کردن تزریقات E-mail در php

بهترین راه برای متوقف کردن تزریقات email، معتبر کردن ورودی است.

کد زیر مانند کد فصل قبل است, اما حالا ما یک معتبرساز ورودی که فیلد email را در فرم چک می کند اضافه کرده ایم:

```
<html>
<body>
<?php
function spamcheck($field)
  {
  $field=filter_var($field, FILTER_SANITIZE_EMAIL);
  if(filter_var($field, FILTER_VALIDATE_EMAIL))
    {
    return TRUE;
    }
  else
    {
    return FALSE;
    }
  }
//اگر تکست باکس ایمیل پرشده باشد،ایمیل ارسال می شود
if (isset($_REQUEST['email']))
  {
  //چک می کنیم که آدرس ایمیل معتبر است یا نه
  $mailcheck = spamcheck($_REQUEST['email']);
  if ($mailcheck==FALSE)
    {
    echo "Invalid input";
    }
  else
    //ارسال ایمیل
    {
    $email = $_REQUEST['email'] ;
    $subject = $_REQUEST['subject'] ;
    $message = $_REQUEST['message'] ;
    mail("someone@example.com", "Subject: $subject",
    $message, "From: $email" );
    echo "Thank you for using our mail form";
    }
  }
else
  //اگر تکست باکس ایمیل پرنشده باشد، فرم ارسال، نمایش داده می شود
  {
  echo "<form method='post' action='mailform.php'>
  Email: <input name='email' type='text' /><br />
  Subject: <input name='subject' type='text' /><br />
```

```
  Message:<br />
  <textarea name='message' rows='15' cols='40'>
  </textarea><br />
  <input type='submit' />
  </form>";
  }
?>

</body>
</html>
```

در کد بالا ما از فیلترهای php زیر، برای معتبرسازی ورودی استفاده کرده ایم:

- **فیلتر FILTER_SANITIZE_EMAIL:** تمام کاراکترهای غیرمجاز ایمیل را از رشته حذف می کند.
- **فیلتر FILTER_VALIDATE_EMAIL:** آدرس ایمیل را معتبر می کند.

برای اطلاعات بیشتر درمورد فیلتر های PHP می توانید به بخش PHP Filter مراجعه کنید.

صفر تا صد PHP -مدیریت خطا

رسیدگی یا مدیریت خطاها در php

رسیدگی به خطاها در php بسیار ساده است. به صورت پیشفرض اگر خطایی رخ دهد، یک پیغام خطا که توضیح دهنده ی خطا است، همراه با نام فایل و شماره خط به مرورگر فرستاده می شود.

هنگام ایجاد برنامه های تحت وب، بررسی خطاها یک بخش مهم است. اگر کد شما امکان بررسی خطا را نداشته باشد، در این صورت بسیار غیرحرفه ای عمل کرده اید و ممکن است در معرض خطر امنیتی قرار بگیرید.

در زیر بعضی روش های معمول بررسی خطا در php توضیح داده خواهد شد:

- استفاده از دستور ساده die()
- خطاهای معمولی و راه اندازی رسیدگی کننده خطا
- واقعه نگاری خطاها

استفاده از تابع die()

این تابع متنی را که به عنوان پارامتر برای آن مشخص کرده ایم، چاپ می کند و از کد جاری خارج می شود.

مثال: در این مثال با استفاده از تابع fopen() یک فایل متنی را به صورت فقط خواندنی باز می کنیم:

```
<?php
$file=fopen("welcome.txt","r");
?>
```

اگر فایل وجود نداشته باشد، با خطای زیر مواجه می شویم:

```
Warning: fopen(welcome.txt) [function.fopen]: failed to open stream:
No such file or directory in C:\webfolder\test.php on line 2
```

برای اینکه کاربر با پیغام خطای بالا مواجه نشود، قبل از هر چیز، وجود فایل "welcome.txt" را بررسی می کنیم:

```
<?php
if(!file_exists("welcome.txt"))
die("File not found");
else
$file=fopen("welcome.txt","r");
?>
```

کد بالا کارامدتر از کد قبلی است، حالا اگر فایل وجود نداشته باشد بعد از اجرای اسکریپت، پیغام خطای زیر نمایش داده می شود:

```
File not found
```

توجه: در اسکریپت بالا، بعد از اجرای دستور ()die، ادامه اجرای اسکریپت متوقف می شود.

متوقف کردن اسکریپت، به این سادگی، همیشه راه درست نیست. بیایید نگاهی به توابع تناوبی رفع خطا در php بیندازیم.

ایجاد یک تابع رسیدگی کننده خطا (error handler)

ایجاد این تابع کاملاً ساده است، و هرگاه که در کد PHP خطایی رخ داد این تابع فراخوانی می شود.

این تابع باید بتواند حداقل با دو پارامتر کار کند(سطح خطا و پیغام خطا) اما تا پنج پارامتر را بپذیرد(اختیاری: فایل، شماره خط و متن پیغام):

نحوه استفاده:

```
error_function(error_level,error_message,error_file,error_line,error_context)
```

پارامتر	توضیحات
error_level	الزامی است، سطح گزارش خطا را مشخص می کند. جدول زیر را برای انواع سطوح گزارش خطا ببینید.
error_message	الزامی است، پیغام خطا را برای خطای تعریف شده توسط کاربر مشخص می کند.
error_file	اختیاری است. نام فایلی را که خطا در ان رخ داده است مشخص می کند.
error_line	اختیاری است. شماره خطی را که خطا در آن رخ داده است مشخص می کند.
error_context	اختیاری است. یک آرایه که شامل تمام متغییرهای در حال استفاده هنگام رخداد خطا با مقادیرشان است را مشخص می کند.

سطوح گزارش خطا:

مقدار	نام ثابت	توضیحات
۲	E_WARNING	خطاهای زمان اجرا غیر جدی. اجرای script متوقف نمی شود
۸	E_NOTICE	اخطارهای زمان اجرا.script چیزی پیدا کرده که ممکن است خطا باشد؟ اما می تواند در اجرای معمولی script نیز رخ دهد
۲۵۶	E_USER_ERROR	خطای جدی کاربرساز. مانند یک E-ERROR که توسط کابر با استفاده از تابع trigger_error( ) در php به وجود می آید.
۵۱۲	E_USER_WARNING	هشدار غیرجدی کاربرساز. مانند یک E_WARNING است که توسط کابر با استفاده از تابع trigger_error( ) در php به وجود می آید.
۱۰۲۴	E_USER_NOTICE	اخطار کاربرساز. مانند یک E_NOTICE است که توسط کابر با استفاده از تابع trigger_error( ) در php به وجود می آید.
۴۰۹۶	E_RECOVERABLE_ERROR	خطای جدی گرفتنی. مانند یک E_ERROR است اما بوسیله ی رفع خطای تعریف شده توسط کاربر می توان ان را گرفت(set_error_handler( ) را نیز ببینید)
۸۱۹۱	E_ALL	همه ی خطاها و هشدارها؟به جز سطح E_STRICT (E_STRICT در php 6.0 جزئی از E_ALL خواهد شد.)

حالا اجازه دهید تا یک تابع برای رسیدگی کردن به خطاها ایجاد کنیم، که شامل ورودی های زیر است:

- **$errno:** سطح خطا
- **$errstr:** متن خطا

```
function customError($errno, $errstr)
{
echo "<b>Error:</b> [$errno] $errstr<br />";
echo "Ending Script";
die();
}
```

کد بالا موقعی که راه اندازی شد، "سطح خطا" و "متن خطا" را می گیرد و سپس با چاپ کردن دو مورد ذکر شده به اجرای اسکریپت پایان می دهد.

در حال حاضر باید تصمیم بگیریم که چه موقع تابع بالا، راه اندازی شود.

تنظیم رسیدگی کننده خطا

ما قصد داریم تابع "customError" را به عنوان پیشفرض رسیدگی کننده خطا در PHP معرفی کنیم، این کار به سادگی با استفاده از تابع زیر در PHP امکان پذیر است:

```
set_error_handler("customError");
```

نکته: یک پارامتر دوم می تواند برای مشخص کردن سطح خطا به تابع set_error_handler() اضافه شود، ولی چون می خواهیم که تابع معمولی مان همه ی خطاها را رفع کند، set_error_handler() تنها با یک پارامتر فراخوانی می شود.

مثال: در زیر سعی کرده ایم با چاپ متغییر "$test" که وجود ندارد تابع "رسیدگی کننده خطا" یا همان "customError" را تست کنیم:

```
<?php
function customError($errno, $errstr)
{
echo "<b>Error:</b> [$errno] $errstr";
}
set_error_handler("customError");
echo($test);
?>
```

خروجی کد بالا:

```
Error: [8] Undefined variable: test
```

راه اندازی یک خطا

در اسکریپتی که کاربران می توانند داده وارد کنند، می توان موقع رخداد یک ورودی غیرمجاز خطاها را راه اندازی کرد. در php، این کار با تابع trigger_error() انجام می شود.

مثال: در این مثال اگر متغییر "test" بزرگتر از "۱" باشد، یک خطا رخ می دهد:

```
<?php
$test=2;
if ($test>1)
{
trigger_error("Value must be 1 or below");
}
?>
```

خروجی کد بالا:

```
Notice: Value must be 1 or below
in C:\webfolder\test.php on line 5
```

یک خطا می تواند هر جایی در اسکریپت که بخواهید راه اندازی شود و با افزودن پارامتر دوم شما می توانید مشخص کنید که چه سطح خطایی راه اندازی شود.

مثال: در این مثال یک خطا با سطح E_USER_WARNING رخ می دهد، اگر متغییر "test" بزرگتر از "۱" باشد، تابع رسیدگی کننده خطا، راه اندازی شده و اسکریبت را خاتمه می دهیم:

```
<?php
function customError($errno, $errstr)
{
echo "<b>Error:</b> [$errno] $errstr<br />";
echo "Ending Script";
die();
}

set_error_handler("customError",E_USER_WARNING);

$test=2;
if ($test>1)
{
trigger_error("Value must be 1 or below",E_USER_WARNING);
}
?>
```

خروجی کد بالا:

```
Error: [512] Value must be 1 or below
Ending Script
```

اکنون که طریقه ی رسیدگی به خطاها و چگونگی راه اندازی آن ها را یاد گرفتیم، اجاز دهید نگاهی به واقعه نگاری (Log) خطاها بیندازیم.

واقعه نگاری خطاها (Log)

به طور پیش فرض، php یک گزارش خطا به سیستم واقعه نگاری سرورها یا یک فایل، بسته به اینکه پیکر بندی گزارش خطا چگونه درفایل php.ini تنظیم شده است ارسال می کند. با استفاده از تابع ()error_log شما می توانید گزارش های خطا را به به یک فایل مشخص شده یا یک مقصد دور ارسال کنید.

ارسال پیغام های خطاها به خودتان از طریق email می تواند روش خوبی برای آگاه شدن از خطاهای خاص باشد.

مثال: در این مثال اگر یک خطای خاص رخ دهد ما یک e-mail را همراه با یک پیغام خطا به آدرس مشخص شده می فرستیم و اسکریبت خاتمه خواهد یافت:

```
<?php
function customError($errno, $errstr)
{
echo "<b>Error:</b> [$errno] $errstr<br />";
```

```
echo "Webmaster has been notified";
error_log("Error:[$errno]$errstr",1,"someone@example.com","From:webmaster@example.com");
}

set_error_handler("customError",E_USER_WARNING);

$test=2;
if ($test>1)
{
trigger_error("Value must be 1 or below",E_USER_WARNING);
}
?>
```

خروجی کد بالا:

```
Error: [512] Value must be 1 or below
Webmaster has been notified
```

و در ادامه ایمیل دریافت شده چیزی شبیه زیر خواهد بود:

```
Error: [512] Value must be 1 or below
```

توجه: این نباید برای همه ی خطاها استفاده شود، خطاهای معین باید با سیستم واقعه نگاری پیش فرض php در سرور ثبت شوند.

آموزش PHP-بررسی استثناها

بررسی استثناها در php

همراه با php 5 یک روش جدید شیء گرا برای کار با خطاها ارائه شده است.

استثنا یا Exception چیست؟

اگر در حین اجرای عادی اسکریپت، یک خطای خاص رخ دهد، بررسی کننده استثنا، راه اندازی شده و ادامه اجرای اسکریپت را تغییر می دهد، این خطا را می توان با دستورات شرطی، مشخص کرد، این شرط در حقیقت همان استثنای ماست.

برای استفاده از استثناها می توان مستقیماً از کلاس Exception استفاده کرد و آنرا به صورت زیر راه اندازی یا پرتاب (throw) کرد:

```
<?php
if($error)
  {
     throw new Exception("خطایی رخ داده است\n");
  }
?>
```

به طور معمول وقتی یک استثنا راه اندازی می شود چه چیزهایی اتفاق می افتد؟

۱. وضعیت فعلی کد ذخیره می شود.
۲. ادامه اجرای کد به یک تابع از پیش تعریف شده (بررسی کننده استثنا) منتقل می شود.
۳. بسته به موقعیت، بررسی کننده استثنا ممکن است اجرای اسکریپت را از وضعیت ذخیره شده از سر بگیرد، یا اجرای اسکریپت را متوقف کند و یا اجرای اسکریپت را از یک مکان متفاوت در کد ادامه دهد.

روش های مختلف بررسی خطا:

۱. استفاده اصلی از استثناها
۲. تعریف کلاس استثایِ خودمان
۳. استثناهای چندگانه
۴. راه اندازی مجدد استثنا
۵. تعیین یک مهارکننده استثنای سطح بالا

نکته: یک استثنا همیشه با یک شرط همراه است و نباید برای پرش به جای دیگر استفاده شود.

۱- استفاده اصلی از استثناها

زمانی که یک استثنا راه اندازی می شود، اجرای کدهای بعد از آن متوقف شده و php سعی به پیدا کردن بلاک تطابقی استثنای مذکور که catch نامیده می شود، خواهد کرد. به زبان ساده تر بعد از بروز خطا و راه اندازی استثنا، ادامه اجرای اسکریپت در یک بلاک که catch نامیده می شود، گرفتار می شود.

نکته: اگر php برای یک استثنا بلاک تطابقی آنرا نیابد، یک fatal error یا خطای مهلک، همراه با پیغام "Uncaught Exception" صادر خواهد شد.

مثال: در این مثال تابع "checkNum" بررسی می کند، اگر پارامتر ورودی آن بزرگتر از "۱" بود استثنا راه اندازی شود، البته توجه داشته باشید که استثنای تعریف شده بدون Catch است:

```
<?php
function checkNum($number)
  {
  if($number>1)
    {
    throw new Exception("Value must be 1 or below");
    }
  return true;
  }

checkNum(2);
?>
```

خروجی کد بالا:

```
Fatal error: Uncaught exception 'Exception'
with message 'Value must be 1 or below' in C:\webfolder\test.php:6
Stack trace: #0 C:\webfolder\test.php(12):
checkNum(28) #1 {main} thrown in C:\webfolder\test.php on line 6
```

راه اندازی استثنا همراه با بلاک های try و catch

در مثال بالا برای مدیریت بهتر خطا، ما به یک کد مناسب برای بررسی کردن استثنا نیاز داریم که می تواند شامل بخش های زیر باشد:

۱. **بلاک try:** تابعی که یک استثنا را استفاده می کند باید در بلاک "try" باشد. (اگر استثنا راه اندازی نشود، کد داخل این بلاک به طور معمول، ادامه خواهد یافت اما اگر شرط استثنا برقرار شود استثنا اجرا خواهد شد)
۲. **راه انداز یا پرتاب استثنا(throw):** یعنی چه طور یک استثنا را راه اندازی شود. هر "throw" باید حداقل یک "catch" داشته باشد.
۳. **بلاک catch:** اگر استثنای پرتاب شود، بلاک "catch" آنرا می گیرد و یک شیء شامل اطلاعات استثنا ایجاد می کند.

مثال:

```
<?php
function checkNum($number)
  {
  if($number>1)
    {
    throw new Exception("Value must be 1 or below");
    }
  return true;
  }

try
  {
  checkNum(2);
  //اگر استثنا راه اندازی بشود، خط بعدی اجرا نخواهد شد
  echo 'If you see this, the number is 1 or below';
  }

catch(Exception $e)
  {
  echo 'Message: ' .$e->getMessage();
  }
?>
```

خروجی کد بالا:

```
Message: Value must be 1 or below
```

توضیح مثال بالا:

۱. تابع checkNum() بررسی می کند که آیا پارامتر ورودی آن بزرگتر از ۱ است. اگر بزرگتر باشد یک استثنا پرتاب می شود.

۲. تابع checkNum() در بلاک "try" فراخوانی می شود

۳. بلاک "catch" استثناهای پرتاب شده را می گیرد و یک شیء ($e) شامل اطلاعات استثنا ایجاد می کند.

۴. پیغام خطای استثنا، با فراخوانی تابع getMessage() از شیء ($e) تولید می شود.

۲- ایجاد کلاس استثنایِ خودمان

به سادگی می توانیم یک کلاس سفارشی، همراه با توابعی که بتوانند هنگام رخداد یک استثنا در php فراخوانی شوند، ایجاد کرد، لطفاً به مثال زیر توجه فرمایید:

نکته: کلاس جدیدمان باید از کلاس Exception ارثبری داشته باشد.

مثال:

```
<?php
class customException extends Exception
  {
  public function errorMessage()
    {
    $errorMsg = 'Error on line '.$this->getLine().' in '.$this->getFile()
    .': <b>'.$this->getMessage().'</b> is not a valid E-Mail address';
    return $errorMsg;
    }
  }

$email = "someone@example...com";

try
  {
  if(filter_var($email, FILTER_VALIDATE_EMAIL) === FALSE)
    {
    throw new customException($email);
    }
  }

catch (customException $e)
  {
  echo $e->errorMessage();
  }
?>
```

توضیح مثال:

۱. کلاس customException از کلاس Exception ارثبری دارد، بنابراین، این کلاس همه ی متدها و خصوصیات را از کلاس قبلی به ارث می برد (getMessage,getFile,getLine)

۲. تابع errorMessage() یک پیغام خطا برمی گرداند (درصورتی که آدرس ایمیل نامعتبر باشد).

۳. متغییر email$ با یک مقدار نامعتبر برای ایمیل تنظیم شده است.

۴. بلاک try اجرا می شود و چون که آدرس ایمیل نامعتبر است، یک استثنا راه اندازی می شود

۵. بلاک catch استثنا را می گیرد و یک پیغام خطا نمایش می دهد

۳- استثناهای چندگانه

برای یک اسکریپت این امکان وجود دارد که به ازای اتفاق افتادن شرط های متفاوت، استثناهای متفاوت با بلاک های catch متفاوت را تعریف کنیم.

مثال:

```
<?php
class customException extends Exception
{
public function errorMessage()
   {
     $errorMsg = 'Error on line '.$this->getLine().' in '.$this->getFile()
     .': <b>'.$this->getMessage().'</b> is not a valid E-Mail address';
     return $errorMsg;
   }
}

$email = "someone@example.com";

try
  {
  // style="font-family: Courier New;">اگر ایمیل معتبر نباشد، استثنا راه اندازی می شود
  if(filter_var($email, FILTER_VALIDATE_EMAIL) === FALSE)
    {
     throw new customException($email);
    }
  // style="font-family: Courier New;"> اگر در آدرس ایمیل کلمه مشخص شده وجود داشت، استثنا راه
اندازی می شود
  if(strpos($email, "example") !== FALSE)
    {
     throw new Exception("$email is an example e-mail");
    }
  }
```

```
catch (customException $e)
  {
  echo $e->errorMessage();
  }

catch(Exception $e)
  {
  echo $e->getMessage();
  }
?>
```

خروجی کد بالا:

```
someone@example.com is an example e-mail
```

توضیح مثال:

۱. متغییر email$ با یک آدرس ایمیل معتبر تنظیم می شود، اما شامل رشته ی "example" است.

۲. چون متغییر تعریف شده حاوی رشته ی "example" است؟ استثنا دوم در بلاک try راه اندازی می شود.

۳. بلاک catch دوم استثنا را می گیرد و خطای مناسب را برمی گرداند.

نکته: اگر استثنای بوجود آمده، بلاک catch خود را نیابد، شروع به جستجوی یک بلاک catch دیگر در "سطوح بالاتر" خواهد کرد.

دوباره راه اندازی استثناها

بعضی مواقع ممکن است مایل باشید که یک استثنا به صورتی متفاوت از آنچه که استاندارد است بررسی شود. این امر با راه اندازی یک استثنا برای بار دوم از داخل بلاک catch امکان پذیر است.

یک اسکریپت خوب، باید خطاهای سیستمی را از کاربران معمولی پنهان کند، چون نمایش این خطاها برای برنامه نویس ها کاربرد دارد و برای کاربران معمولی جالب نیست، یک ایده خوب برای ساخت چنین پیغام هایی، دوباره راه اندازی استثناهاست.

مثال:

```
<?php
class customException extends Exception
  {
  public function errorMessage()
    {
    $errorMsg = $this->getMessage().' is not a valid E-Mail address.';
```

```
    return $errorMsg;
    }
  }

$email = "someone@example.com";

try
  {
  try
    {
    if(strpos($email, "example") !== FALSE)
      {
      throw new Exception($email);
      }
    }
  catch(Exception $e)
    {
    //استثنا دوباره راه اندازی می شود
    throw new customException($email);
    }
  }

catch (customException $e)
  {
  //display custom message
  echo $e->errorMessage();
  }
?>
```

توضیح مثال:

۱. بلاک try بک بلاک try دیگر را در خود جای داده است و این امکان را فراهم می کند تا از بلاکِ try داخلی یک استثنا که قبلاً راه اندازی شده را دوباره راه اندازی کرد.

۲. اگر متغییر ایمیل شامل زیر رشته "example" باشد استثنا برای بار اول راه اندازی می شود.

۳. بلاکِ catch اولی(Exception) استثنا را مهار کرده و در ادامه استثنای customException را راه اندازی می کند.

۴. بلاکِ catch دومی(customException) بلافاصله بعد از مهار کردن استثنا، پیغام مناسب را چاپ می کند.

تعیین یک بررسی کننده استثنای سطح بالا

تابع set_exception_handler()، نام یک تابع تعریف شده توسط کاربر را به عنوان پارامتر ورودی می گیرد و برای کار با همه ی استثناهایی که بلاک catch برای آنها مشخص نشده است، از تابع مذکور استفاده می کند.

مثال:

```php
<?php
function myException($exception)
{
echo "<b>Exception:</b> " , $exception->getMessage();
}

set_exception_handler('myException');

throw new Exception('Uncaught Exception occurred');
?>
```

خروجی کد بالا:

```
Exception: Uncaught Exception occurred
```

در کد بالا بلاک catch وجود ندارد، درعوض یک بررسی کننده استثنای سطح بالا راه اندازی شده، که می تواند برای گرفتن یا مهار کردن استثناهایی که بلاک catch ندارند، استفاده شود.

قوانین استثناها

- با قرار دادن کدهایمان در بلاک try به مهار کردن استثناهای بالقوه کمک کرده ایم.
- هر بلاک try یا throw باید حداقل یک بلاک catch متناظر داشته باشد
- استثناها می توانند درون یک بلاک catch راه اندازی شوند.

یک قانون ساده: چیزی را که راه اندازی(throw) می کنید باید مهار شود(catch).

صفر تا صد PHP -فیلتر ورودی ها

فیلتر ورودی ها در php

تقریباً همه ی برنامه های تحت وب از محیط بیرون دریافت اطلاعات دارند که معمولاً این ورودی ها می تواند از طریق کاربران یا برنامه های دیگر مثل وب سرویس ها باشد، برای اطمینان از صحت اطلاعات ورودی، فیلترهای PHP کمک فراوانی به ما خواهند کرد.

شما باید همیشه داده های ورودی را از فیلترها عبور دهید، در حقیقت فیلتر ورودی ها یکی از مهمترین مسائل امنیت برنامه هاست.

ورودی های برنامه کدامند؟

- داده های دریافت شده از یک فرم HTML
- Cookieها
- داده های دریافت شده از وب سرویس ها

- متغیرهایی که روی سرور ایجاد می شوند مثل sessionها
- اطلاعاتی که از پایگاه داده بازیابی می شود

توابع و فیلترها

برای فیلترکردن یک متغیر، یکی از توابع زیر را استفاده کنید:

- filter_var() : یک متغیر خاص را با یک صافی مخصوص فیلتر می کند.
- filter_var_array() : چندین متغیر را با یک صافی یکسان و یا متفاوت فیلتر می کند.
- filter_input() : یک متغیر ورودی را می گیرد و آن را فیلتر می کند.
- filter_input_array() : چندین متغیر ورودی را می گیرد و آن ها را با یک صافی یکسان و یا متفاوت فیلتر می کند.

مثال: در اینجا با استفاده از تابع filter_var()، صحیح بودن متغیر $int را بررسی می کنیم:(اعشاری نباشد)

```
<?php
$int = 123;

if(!filter_var($int, FILTER_VALIDATE_INT))
  {
  echo("Integer is not valid");
  }
else
  {
  echo("Integer is valid");
  }
?>
```

خروجی کد بالا:

```
Integer is valid
```

در کد بالا از صافی "FILTER_VALIDATE_INT" برای فیلتر کردن متغیر استفاده می کند.

اگر ما مثال بالا را با متغیری که integer نیست مثل "abc۱۲۳" امتحان کنیم، خروجی چنین خواهد بود: "integer is not valid"

برای مشاهده لیست کامل توابع و فیلترها به بخش PHP filter Reference مراجعه کنید.

معتبرسازی و اصولی عمل کردن

به صورت کلی دو نوع فیلتر وجود دارد:

فیلترهای اعتبار سنجی:

- برای معتبر کردن ورودی های کاربر استفاده می شوند
- برای قالب بندی قوانین محض استفاده می شود(مثل معتبرسازی URL یا E-Mail)
- اگر داده ورودی اعتبار نداشته باشد FALSE والا مقدار متغیر را بر می گرداند

فیلترهای اصولی:

- وجود یا عدم وجود کاراکترهایی خاص در یک رشته را بررسی می کند.
- برای قالب بندی داده ها نیست
- همیشه یک رشته برمی گرداند.

انتخاب ها و پرچم ها (optionها و flagها)

Optionها و flagها برای اضافه کردن شرط های خاص به فیلترها استفاده می شوند.

فیلترهای متفاوت optionها و flagهای متفاوتی دارند.

مثال: در اینجا, ما یک متغیر integer را با استفاده از ()filter_var و optionهای "min_range" و "max_range" معتبر می کنیم:

```
<?php
$var=300;

$int_options = array("options"=>array("min_range"=>0,"max_range"=>256));

if(!filter_var($var, FILTER_VALIDATE_INT, $int_options))
  {
  echo("Integer is not valid");
  }
else
  {
  echo("Integer is valid");
  }
?>
```

خروجی کد بالا:

```
Integer Is not valid
```

در کد بالا, optionها باید در یک آرایه ی انجمنی با نام "options" قرار داده شوند.اگر یک flag استفاده می شود نیاز نیست که در آرایه قرار گیرد. (If a flag is used it does not need to be in an array)

برای مشاهده لیست کامل توابع و فیلترها به بخش PHP filter Reference مراجعه کنید. بررسی کنید چه optionها و flagهایی برای هر فیلتر وجود دارد.

ورودی را معتبر کنید

اجازه دهید با یک مثال ورودی یک فرم را معتبر کنیم.

اولین کاری که باید انجام دهیم این است که مطمئن شویم که داده ای که ما به دنبال آن هستیم وجود دارد یا نه.

سپس داده ی ورودی را با استفاده از تابع ()filter_input فیلتر می کنیم.

مثال: در مثال زیر, متغیر ورودی "email" به صفحه php فرستاده می شود:

```
<?php
if(!filter_has_var(INPUT_GET, "email"))
  {
  echo("Input type does not exist");
  }
else
  {
  if (!filter_input(INPUT_GET, "email", FILTER_VALIDATE_EMAIL))
    {
    echo "E-Mail is not valid";
    }
  else
    {
    echo "E-Mail is valid";
    }
  }
?>
```

توضیح مثال:

در مثال بالا یک ورودی email داریم که با استفاده از متد "GET" به آن ارسال شده است:

۱. چک می کند که آیا متغیر ورودی "email" از نوع "GET" موجود هست
۲. اگر متغیر ورودی وجود داشته باشد, چک می کند که آیا یک آدرس ایمیل معتبر هست یا نه

ورودی را اصولی کنید

بیایید پاکسازی یک URL ارسال شده از فرم را امتحان کنیم.

ابتدا مطمئن می شویم که داده ای که ما به دنبال آن هستیم وجود دارد یا نه.

سپس داده ی ورودی را با استفاده از تابع ()filter_input اصولی می کنیم.

مثال: در مثال زیر, متغیر ورودی "URL" به صفحه php فرستاده می شود:

```
<?php
if(!filter_has_var(INPUT_POST, "url"))
  {
  echo("Input type does not exist");
  }
else
  {
  $url = filter_input(INPUT_POST,"url",FILTER_SANITIZE_URL);
  }
?>
```

توضیح مثال:

مثال بالا یک ورودی url دارد که با استفاده از متد "POST" به آن ارسال شده است:

۱. چک می کند که آیا متغیر ورودی "url" از نوع "POST" موجود هست.
۲. اگر متغیر ورودی وجود داشته باشد, آن را اصولی (کاراکترهای نامعتبر را کنار می گذارد) و در متغیر url$ ذخیره می کند.

اگر متغیر ورودی یک رشته مثل این باشد "http://www.W3ååSchøøools.com/" ,متغیر $url بعد از اصولی سازی به این شکل خواهد بود:

```
http://www.W3Schools.com/
```

فیلتر ورودی های چندگانه ی

یک فرم HTML همیشه از بیش از یک فیلد ورودی تشکیل شده است. برای جلوگیری ازچندین بار فراخوانی توابع filter_var یا filter_input , ما می توانیم توابع filter_var_array یا filter_input_array را استفاده کنیم.

مثال: در این مثال ما تابع filter_input_array را برای فیلترکردن سه متغیر که با استفاده از متد GET ارسال شده اند استفاده می کنیم.(نام، سن و آدرس ایمیل)

```
<?php
$filters = array("name" => array("filter"=>FILTER_SANITIZE_STRING),
                 "age" => array("filter"=>FILTER_VALIDATE_INT,
                                "options"=>array("min_range"=>1,"max_range"=>120)),
                 "email"=> FILTER_VALIDATE_EMAIL);
$result = filter_input_array(INPUT_GET, $filters);

if (!$result["age"])
  {
  echo("Age must be a number between 1 and 120.<br />");
  }
elseif(!$result["email"])
```

```
  {
  echo("E-Mail is not valid.<br />");
  }
else
  {
  echo("User input is valid");
  }
?>
```

توضیح مثال:

مثال بالا سه ورودی (name,age,email) دارد که با استفاده از متد "GET" به آن ارسال شده است:

۱. یک آرایه شامل نام متغیرهای ورودی و فیلترهای مصرفی برای متغیرهای ورودی مشخص شده است.
۲. تابع ()filter_input_array را همراه با متغیرهای ورودی GET و آرایه ای که تعیین کردیم فراخوانی می کنیم.
۳. متغیرهای "email" و "age" را برای ورودی های نامعتبر در متغیر $result چک می کند.(اگر هرکدام از متغیرهای ورودی نامعتبر باشند, آن متغیر ورودی پس از تابع ()filter_input_array برابر با FALSE قرار خواهد گرفت)

توجه: پارامتر دوم تابع ()filter_input_array می تواند یک آرایه یا یک صافی خاص باشد. اگر پارامتر یک صافی یا فیلتر خاص باشد تمام مقادیر در آرایه ی ورودی بوسیله ی صافی مشخص شده فیلتر می شوند، اگر پارامتر یک آرایه است باید قواعد زیر را دنبال کند:

- باید یک آرایه ی انجمنی باشد.(PHP Arrays)
- کلید یا ایندکس آرایه باید با نام ورودی ها یکی باشد.
- مقدار آرایه باید یک صافی خاص یا یک آرایه دیگر شامل صافی و flagها و optionها باشد.

صافی های تعریف شده توسط خودمان

امکان فراخوانی یک تابع کاربرنویس و استفاده از آن به عنوان یک فیلتر وجود دارد. با این کار، تمامی فیلترهای ممکن را می توان پوشش داد.

بعد از تعریف تابع خودمان باید با استفاده از FILTER_CALLBACK که به عنوان پارامتر دوم توابع فیلتر است مشخص کنیم که نوع صافی مصرفی ، تابع تعریف شده توسط خودمان است.

مثال: در مثال زیر, همه ی کاراکترهای "_" به خط فاصله تبدیل می شود:

```
<?php
function convertSpace($string)
{
return str_replace("_", " ", $string);
```

```
}

$string = "Peter_is_a_great_guy!";

echo filter_var($string, FILTER_CALLBACK,array("options"=>"convertSpace"));
?>
```

خروجی کد بالا:

```
Peter is a great guy!
```

توضیح مثال:

مثال بالا همه ی کاراکترهای "_" را به خط فاصله تبدیل می کند:

۱. یک تابع برای جایگزین کردن کاراکتر"_" به خط فاصه ایجاد می کند
۲. تابع filter_var() را همراه با فیلتر FILTER_CALLBACK و یک آرایه شامل تابع مان فراخوانی می کند

فصل ۴: آموزش پایگاه داده

صفر تا صد PHP -مقدمه MySQL

با استفاده از PHP به پایگاه داده متصل می شوید و داده ها را دستکاری می کنید.

MySQL، یکی از معروف ترین سیستم های پایگاه داده است که همراه با PHP استفاده می شود.

MySQL چیست؟

- پایگاه داده MySQL، روی وب استفاده می شود.
- پایگاه داده MySQL، روی سرور اجرا می شود.
- پایگاه داده MySQL ، خیلی سریع، قابل اطمیتان و استفاده از آن آسان است.
- پایگاه داده MySQL، از استاندرد SQL استفاده می کند.
- قابلیت حمل بر روی سیستم عامل های مختلف از قبیل Linux و windows و غیره را دارد.
- استفاده و دانلود MySql رایگان است.
- پایگاه داده MySQL، توسط شرکت Oracle توسعه و پشتیبانی می شود.
- بیادبود دختر بنیانگذار MySQL یعنی Monty Widenius در ابتدای نام آن از کلمه My استفاده شده است.

به زبان ساده، در MySQL داده ها در جداول ذخیره می شوند، جدول یک مجموعه ای از داده های مرتبط با هم است و از ستون ها و ردیفهایی تشکیل شده است.

مواقعی که بخواهیم داده ها را به صورت همیشگی ذخیره کنیم، پایگاه های داده مفید واقع می شوند، به طور مثال یک سایت خرید و فروش می تواند شامل جدول های زیر باشد:

۱. جدول کالاها
۲. جدول مشتری ها
۳. جدول سفارشات
۴. جدول کارمندان

PHP + پایگاه داده MySQL

- PHP به همراه MySQl قابلیت cross-platform را داراست. (یعنی پروژه یا سایت تان را می توانید روی Windows پیاده کنید و روی Unix به کاربران ارائه دهید و برعکس ...!)

جداول پایگاه داده

یک پایگاه داده معمولاً از یک یا چند جدول تشکیل شده است، هر جدول را با یک نام مشخص می کنیم، مثلاً برای جدول مشتری ها می توان از نام Customer استفاده کرد. جداول شامل رکوردهایی از داده هاست.

مثال: جدول مشتری ها می تواند شامل ستون ها و ردیف های زیر باشد:

LastName	FirstName	Address	City
Mohammadi	ali	...	Esfahan
Nayeb	Reza	...	Esfahan
Najafi	Sara	...	Tehran

جدول بالا شامل سه رکورد اطلاعاتی و چهار ستون LastName, FirstName, Address, City می باشد.

کوئری ها(Queries)

کوئری را می توان پرسیدن یا درخواست کردن معنی کرد.

با کوئری ها ما می توانیم یک درخواست از پایگاه داده برای بدست آوردن اطلاعاتی خاص داشته باشیم.

مثال: لطفاً به کوئری یا درخواست زیر توجه کنید:

```
SELECT LastName FROM Customer
```

کوئری بالا همه ی داده های ستون LastName از جدول Customer را انتخاب می کند و جدول زیر را نمایش می دهد:

LastName
Mohammadi
Nayeb
Najafi

صفر تا صد PHP -ارتباط با داده ها

در PHP نسخه ۵ به بالا، برای کار با پایگاه داده MySQL می توانید یکی از روش های زیر را استفاده نمایید:

- **افزونه MySQLi** (کاراکتر i مخفف improved بمعنی بهبود یافته است)
- **PDO** که سرنام واژگان PHP Data Objects است

در نسخه های قبلی PHP از افزونه MySQL استفاده می شده، اما استفاده از این افزونه، از ۲۰۱۲ دیگر توصیه نمی شود.

باید از MySQLi استفاده کنم یا PDO ؟

اگر بخواهیم خیلی خلاصه جواب دهیم، این انتخاب بستگی به نیاز شما دارد ...!

هر کدام از گزینه های MySQLi و PDO، مزیت های خودشان را دارند.

گزینه PDO با ۱۲ پایگاه داده مختلف کار می کند، اما MySQLi تنها با پایگاه داده MySQL کار خواهد کرد.

بنابراین اگر بخواهید زمانی به یک پایگاه داده دیگر سوییچ کنید، گزینه PDO کار را آسان تر خواهد کرد و فقط کافی است که ارتباط یا connection به پایگاه داده را تغییر دهید و احتمالاً چند تغییر کوچک در کوئری ها را خواهید داشت.

هر دو گزینه شیء گرا هستند، اما MySQLi رویه های API را نیز ارائه می دهد.

هر دو گزینه، دستورات آماده برای مقابله با تزریقات SQL یا SQL injection را پشتیبانی می کنند، که البته این موضوع، برای حفظ امنیت برنامه های کاربردی وب بسیار حیاتی است.

مثال های MySQL به هر دو روش MySQLi و PDO

در این مطلب و آموزش های بعدی، سه روش مختلف برای کار کردن با پایگاه داده را نشان خواهیم داد:

- MySQLi (شیء گرا یا Object-Oriented)
- MySQLi (رویه ای یا Procedural)
- PDO

نصب MySQLi

برای سیستم عامل Linux و Windows، افزونه MySQLi در اغلب موارد بصورت اتوماتیک بعد از نصب پکیج php5 mysql نصب خواهد شد.

برای کسب اطلاعات بیشتر به لینک روبرو مراجعه فرمایید: http://php.net/manual/en/mysqli.installation.php

---

نصب PDO

برای کسب اطلاعات بیشتر به لینک روبرو مراجعه فرمایید:http://php.net/manual/en/pdo.installation.php

---

ارتباط یا Connection به پایگاه داده MySQL

قبل از اینکه به داده های پایگاه داده دسترسی داشته باشید، باید یک ارتباط یا Connection تعریف کنید:

مثال (MySQLi Object-Oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

// Create connection
$conn = new mysqli($servername, $username, $password);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
?>
```

نکته: توجه داشته باشید که در مثال شیء گرای بالا، $connect_error تا PHP 5.2.9 و PHP ۵.۳.۰ کار نمی کند، بنابراین اگر می خواهید که کدتان با نسخه های قدیمی PHP نیز سازگار باشد، بجای آن از کد زیر استفاده نمایید:

```
 // Check connection
if (mysqli_connect_error()) {
    die("Database connection failed: " . mysqli_connect_error());
}
```

در مثال زیر با استفاده از تابع ()mysqli_connect یک connection به پایگاه داده تعریف شده است:

مثال (MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

// Create connection
$conn = mysqli_connect($servername, $username, $password);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
?>
```

مثال(PDO)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

try {
    $conn = new PDO("mysql:host=$servername;dbname=myDB", $username, $password);
    // set the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    echo "Connected successfully";
    }
catch(PDOException $e)
    {
    echo "Connection failed: " . $e->getMessage();
    }
?>
```

نکته: توجه داشته باشید که در مثال PDO بالا، با تنظیم گزینه dbname با مقدار "myDB" نام پایگاه داده را نیز مشخص کرده ایم. اگر

PDO موفق به اتصال به پایگاه داده نشود یک استثنا یا exception پرتاب (thrown) می شود.

نکته: یکی از بزرگترین مزیت های PDO وجود کلاس exception برای مدیریت خطاها است. همان طور که می دانید این خطاها ممکن است در کوئری های پایگاه داده رخ دهد و اگر یک استثنا یا exception از داخل بلاک try{ } پرتاب (thrown) شود، اجرای اسکریپت متوقف شده و جریان کار به اولین بلاک catch{ } منتقل می شود. در واقع استثنای که در قسمت try رخ داده در قسمت catch به دام می افتد.

بستن یک ارتباط

ارتباط یا Connection به صورت اتوماتیک وقتی که اسکریپت پایان یافت، بسته می شود، اما اگر مایل باشید می توانید این کار را قبل از پایان اسکریپت انجام دهید:

مثال(MySQLi Object-Oriented)

```
$conn->close();
```

مثال(MySQLi Procedural)

```
mysqli_close($conn);
```

مثال(PDO)

```
$conn = null;
```

آموزش MySQL-ایجاد پایگاه داده

پایگاه داده شامل یک یا جند جدول است.

باید بدانید که برای ایجاد یا حذف پایگاه داده، باید دسترسی لازم را داشته باشید.

ایجاد پایگاه داده با استفاده از MySQLi و PDO

دستور CREATE DATABASE برای ایجاد یک پایگاه داده در MySQL استفاده می شود.

نحوه استفاده:

```
CREATE DATABASE database_name
```

در مثال زیر یک پایگاه داده به نام "myDB" ایجاد کرده ایم:

مثال (MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
```

```
// Create connection
$conn = new mysqli($servername, $username, $password);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Create database
$sql = "CREATE DATABASE myDB";
if ($conn->query($sql) === TRUE) {
    echo "Database created successfully";
} else {
    echo "Error creating database: " . $conn->error;
}

$conn->close();
?>
```

توجه: زمانی که می خواهید یک پایگاه داده جدید ایجاد کنید، در شیء mysqli تنها سه آرگومان اول را باید مشخص نمایید.
(servername و username و password)

نکته: اگر برای اتصال به پایگاه داده باید از یک Port مشخص استفاده نمایید، برای تنظیم این آرگومان در شیء mysqli باید آرگومان چهارم یعنی database-name را با یک رشته خالی تنظیم نمایید و سپس آرگومان پنجم را برای تنظیم Port استفاده نمایید:

```
mysqli("localhost", "username", "password", "", port)
```

مثال (MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

// Create connection
$conn = mysqli_connect($servername, $username, $password);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// Create database
$sql = "CREATE DATABASE myDB";
if (mysqli_query($conn, $sql)) {
    echo "Database created successfully";
} else {
    echo "Error creating database: " . mysqli_error($conn);
}
```

```
mysqli_close($conn);
?>
```

در مثال زیر یک پایگاه داده به نام "myDBPDO" ایجاد کرده ایم:

مثال(PDO)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

try {
    $conn = new PDO("mysql:host=$servername;dbname=myDB", $username, $password);
    // set the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $sql = "CREATE DATABASE myDBPDO";
    // use exec() because no results are returned
    $conn->exec($sql);
    echo "Database created successfully<br>";
    }
catch(PDOException $e)
    {
    echo $sql . "<br>" . $e->getMessage();
    }

$conn = null;
?>
```

نکته: یکی از بزرگترین مزیت های PDO وجود کلاس exception برای مدیریت خطاها است. همان طور که می دانید این خطاها ممکن است در کوئری های پایگاه داده رخ دهد و اگر یک استثنا یا exception از داخل بلاک { }try پرتاب (thrown) شود، اجرای اسکریپت متوقف شده و جریان کار به اولین بلاک { }catch منتقل می شود. در واقع استثنای که در قسمت try رخ داده در قسمت catch به دام می افتد. در بلاک catch، با استفاده از دستور echo، دستور SQL و متن خطای تولید شده را چاپ می کنیم.

آموزش MySQL-ایجاد جدول

یک جدول در پایگاه داده شامل یک نام منحصر بفرد می باشد و شامل تعدادی ستون و ردیف است.

ایجاد یک جدول در MySQL با استفاده از MySQLi و PDO

دستور CREATE TABLE برای ایجاد یک جدول در MySQL استفاده می شود.

در اینجا قصد داریم یک جدول با نام "MyGuests" که شامل ۵ فیلد "id" و "firstname" و "lastname" و "email" و "reg_date" است را ایجاد نماییم:

```
CREATE TABLE MyGuests (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP
)
```

نکاتی درباره جدول بالا:

نوع داده مشخص می کند که چه داده هایی در فیلدها نگهداری شوند، اگر مایلید در این زمینه اطلاعات بیشتری کسب کنید به لینک آموزش SQL-انواع داده مراجعه نمایید. در مثال بالا، موقعی که نوع فیلدها را varchar در نظر می گیرید باید حداکثر طول آنرا در پرانتز مشخص کنید(تا ۲۵۵ کاراکتر)، اگر به اندازه ای بزرگتر از ۲۵۵ نیاز دارید از نوع text استفاده کنید(تا ۶۵,۵۳۵ کاراکتر).

بعد از مشخص کردن نوع داده یا Data Type، گزینه های اختیاری دیگری نیز برای هر ستون وجود دارد:

- NOT NULL - محدودیت NOT NULL یک ستون را مجبور می کند که مقدار خالی را قبول نکند.
- مقدار DEFAULT - محدودیت DEFAULT برای وارد کردن مقداری به صورت پیش فرض در یک ستون استفاده می شود.
- UNSIGNED - این محدودیت تنها برای نوع عددی استفاده می شود و زمان اضافه کردن رکورد تنها اجازه وارد کردن صفر و اعداد مثبت وجود دارد.
- AUTO INCREMENT - مقدار فیلد بصورت اتوماتیک به ازای هر رکورد جدید ۱ واحد اضافه می شود. (خود افزا)
- PRIMARY KEY - هر جدول می تواند یک کلید اصلی (PRIMARY KEY) داشته باشد، کلید اصلی برای منحصر به فرد کردن ردیفهای یک جدول استفاده می شود، بنابراین مقدار فیلدی را که کلید در نظر می گیرد در کل نباید تکراری باشد. کلید اصلی معمولاً از نوع عددی و خودافزا است، همچنین باید با مقدار NOT NULL تنظیم شود.

مثال های زیر، نشان می دهد که چگونه می تون در PHP یک جدول را ایجاد کرد:

مثال(MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
```

```
// sql to create table
$sql = "CREATE TABLE MyGuests (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP
)";

if ($conn->query($sql) === TRUE) {
    echo "Table MyGuests created successfully";
} else {
    echo "Error creating table: " . $conn->error;
}

$conn->close();
?>
```

مثال(MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// sql to create table
$sql = "CREATE TABLE MyGuests (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP
)";

if (mysqli_query($conn, $sql)) {
    echo "Table MyGuests created successfully";
} else {
```

```
    echo "Error creating table: " . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

مثال(PDO)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";

try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
    // set the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    // sql to create table
    $sql = "CREATE TABLE MyGuests (
    id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    firstname VARCHAR(30) NOT NULL,
    lastname VARCHAR(30) NOT NULL,
    email VARCHAR(50),
    reg_date TIMESTAMP
    )";

    // use exec() because no results are returned
    $conn->exec($sql);
    echo "Table MyGuests created successfully";
    }
catch(PDOException $e)
    {
    echo $sql . "<br>" . $e->getMessage();
    }

$conn = null;
?>
```

آموزش MySQL-دستور Insert

درج کردن اطلاعات در MySQL با استفاده از MySQLi و PDO

بعد از اینکه پایگاه داده و جداول را ایجاد کردیم، حالا نوبت به درج اطلاعات می رسد.

دستور INSERT INTO برای درج کردن یک رکورد جدید در جدول استفاده می شود.

در اینجا چند قانون ساختاری وجود دارد که باید پیروی کنید:

- کوئری های SQL در PHP باید با علامت کوتیشن محصور شوند.
- مقادیر رشته ای استفاده شده در کوئری ها باید با علامت کوتیشن محصور شوند.
- اعداد را نباید با کوتیشن محصور کنید.
- کلمه NULL نباید با کوتیشن محصور شود.

به دو صورت می توان از دستور INSERT INTO استفاده کرد:

۱- در این روش نیازی نیست که نام ستونها ذکر شود:

```
INSERT INTO table_name
VALUES (value1, value2, value3,...)
```

۲- در این روش باید نام ستون ها و مقادیر متناظرشان ذکر شود:

```
INSERT INTO table_name (column1, column2, column3,...)
VALUES (value1, value2, value3,...)
```

برای کسب اطلاعات بیشتر در مورد SQL، به لینک روبرو مراجعه فرمایید: آموزش SQL-مقدمه

در مطلب قبل، یک جدول با نام "MyGuests" با ۵ فیلد "id" و "firstname" و "lastname" و "email" و "reg_date" ایجاد کردیم،

حالا اجازه دهید تا جدول را با اطلاعات پر کنیم.

> **توجه:** اگر هنگام تعریف جدول، برای یک ستون، گزینه AUTO_INCREMENT را تنظیم کرده باشیم (مانند فیلد id) و یا فیلد مورد نظر از نوع TIMESTAMP باشد (مانند فیلد reg_date) هنگام درج اطلاعات نیازی به مشخص کردن مقدار نیست و MySQL بصورت اتوماتیک آنها را پر خواهد کرد.

مثال های زیر یک رکورد جدید به جدول "MyGuests" اضافه خواهند کرد:

مثال (MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
```

```
$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if ($conn->query($sql) === TRUE) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

مثال(MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if (mysqli_query($conn, $sql)) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

توجه: همانطور که قبلاً گفته شد در روش MySQLi Procedural برای اجرای کوئری ها، باید از تابع ()mysqli_query استفاده کرد، این تابع توسط Connectionی که برقرار شده برای ارسال کوئری به MySQL استفاده می شود.

مثال(PDO)

```
<?php
$servername = "localhost";
```

```
$username = "username";
$password = "password";
$dbname = "myDBPDO";

try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
    // set the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $sql = "INSERT INTO MyGuests (firstname, lastname, email)
    VALUES ('John', 'Doe', 'john@example.com')";
    // use exec() because no results are returned
    $conn->exec($sql);
    echo "New record created successfully";
    }
catch(PDOException $e)
    {
    echo $sql . "<br>" . $e->getMessage();
    }

$conn = null;
?>
```

آموزش MySQL-آخرین ID درج شده

برگرداندن ID آخرین رکورد درج شده یا ویرایش شده در MySQL

اگر روی جدولی که دارای فیلد خودافزا یا AUTO_INCREMENT است دستور INSERT یا UPDATE را اجرا کنیم، می توانیم بلافاصله ID آخرین رکورد درج/ویرایش شده را بدست آوریم.

فیلد "id" در جدول "MyGuests"، خودافزا یا AUTO_INCREMENT است:

```
CREATE TABLE MyGuests (
id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
firstname VARCHAR(30) NOT NULL,
lastname VARCHAR(30) NOT NULL,
email VARCHAR(50),
reg_date TIMESTAMP
)
```

مثال زیر، همان مثال آموزش قبلی است (آموزش MySQL-ایجاد جدول) با این تفاوت که برای چاپ ID آخرین رکورد درج شده، دو خط کد به آن اضافه شده است:

مثال(MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
```

```
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if ($conn->query($sql) === TRUE) {
    $last_id = $conn->insert_id;
    echo "New record created successfully. Last inserted ID is: " . $last_id;
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

مثال(MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com')";

if (mysqli_query($conn, $sql)) {
    $last_id = mysqli_insert_id($conn);
    echo "New record created successfully. Last inserted ID is: " . $last_id;
} else {
```

```
    echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

مثال (PDO)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";

try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
    // set the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $sql = "INSERT INTO MyGuests (firstname, lastname, email)
    VALUES ('John', 'Doe', 'john@example.com')";
    // use exec() because no results are returned
    $conn->exec($sql);
    $last_id = $conn->lastInsertId();
    echo "New record created successfully. Last inserted ID is: " . $last_id;
    }
catch(PDOException $e)
    {
    echo $sql . "<br>" . $e->getMessage();
    }

$conn = null;
?>
```

آموزش MySQL-درج چند رکورد

درج چندین رکورد در MySQL با استفاده از MySQLi و PDO

با استفاده از تابع ()mysqli_multi_query می توان چندین دستور SQL را اجرا نمود.

در مثال زیر، سه رکورد جدید در جدول "MyGuests" درج می شود:

مثال (MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";
```

```
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com');";
$sql .= "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('Mary', 'Moe', 'mary@example.com');";
$sql .= "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('Julie', 'Dooley', 'julie@example.com')";

if ($conn->multi_query($sql) === TRUE) {
    echo "New records created successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

نکته: توجه داشته باشید که دستورات SQL باید با علامت سمیکالن (;) جدا شوند. (در کد بالا با رنگ قرمز متمایز شده است)

مثال (MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('John', 'Doe', 'john@example.com');";
$sql .= "INSERT INTO MyGuests (firstname, lastname, email)
```

```
VALUES ('Mary', 'Moe', 'mary@example.com');";
$sql .= "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('Julie', 'Dooley', 'julie@example.com')";

if (mysqli_multi_query($conn, $sql)) {
    echo "New records created successfully";
} else {
    echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

درج چند رکورد، در روش PDO کمی متفاوت است:

مثال(PDO)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";

try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
    // set the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    // begin the transaction
    $conn->beginTransaction();
    // our SQL statememtns
    $conn->exec("INSERT INTO MyGuests (firstname, lastname, email)
    VALUES ('John', 'Doe', 'john@example.com')");
    $conn->exec("INSERT INTO MyGuests (firstname, lastname, email)
    VALUES ('Mary', 'Moe', 'mary@example.com')");
    $conn->exec("INSERT INTO MyGuests (firstname, lastname, email)
    VALUES ('Julie', 'Dooley', 'julie@example.com')");

    // commit the transaction
    $conn->commit();
    echo "New records created successfully";
    }
catch(PDOException $e)
    {
    // roll back the transaction if something failed
    $conn->rollback();
    echo "Error: " . $e->getMessage();
```

```
    }

$conn = null;
?>
```

آموزش MySQL-دستورات آماده

دستورات آماده برای مقابله با SQL injection ها بسیار کارآمداند.

دستورات آماده (Prepared) و Bind کردن پارامترها

با استفاده از دستورات آماده یا Prepared، می توانید یک کوئری یکسان را به تعداد دفعات دلخواه با راندمان بالا اجرا نمایید.

نحوه عملکرد دستورات آماده یا Prepared:

۱. **Prepare**: با استفاده از این دستور می توانید قالبِ یک کوئری SQL را ایجاد کرده و به پایگاه داده ارسال نمایید. در واقع دلیل استفاده از کلمه قالب در اینجا این است که بجای پارامترهای لازم در کوئری، از علامت سوال (؟) استفاده می کنیم.

مثال

```
INSERT INTO MyGuests VALUES(?, ?, ?)
```

۲. حالا قالب SQL ارسال شده، توسط پایگاه داده، تجزیه و کامپایل می شود و در ادامه بهینه ساز کوئری (query optimization) روی آن اجرا می شود و در نهایت نتیجه کار بدون اجرا در پایگاه داده ذخیره می شود.

۳. **Execute**: در ادامه ی اسکریپت، مقادیر لازم را به پارامترهای کوئری مان Bind می کنیم و با استفاده از دستور Execute، کوئری را اجرا می کنیم. اما نکته ای که وجود دارد این است که کوئری ما همچنان در پایگاه داده وجود دارد و می توانیم مجدداً آنرا با پارامترهای جدید اجرا کنیم ...!

روش استفاده از دستورات آماده در مقایسه با اجرای مستقیم کوئری ها دو مزیت اصلی دارد:

- دستورات آماده، زمان تجزیه (parsing) کوئری را کاهش می دهند و این در حالی است که برای مواقعی که قصد داریم یک کوئری را به دفعات اجرا کنیم، این زمان فقط یکبار محاسبه خواهد شد.
- از آنجایی که برای هر بار اجرای کوئری، فقط باید پارامترها را ارسال کنیم، بنابراین پهنای باند کمتری مصرف خواهد شد.
- دستورات آماده برای مقابله با SQL injection ها بسیار کارآمد هستند و این بخاطر استفاده از یک پروتوکول متفاوت برای انتقال مقادیر پارامترهاست.

تزریق SQL یا SQL injection چیست؟

SQL injection زمانی رخ می دهد که شما از کاربر یک ورودی می خواهید (مثلا نام خانوادگی ، سن و ...) و کاربر بجای نام، یک دستور MySQL را وارد می نماید و این دستور بدون اطلاع شما بر روی دیتابیس اجرا می شود.

مثال

```
$username="'; Delete From users";
$query="Select From users where usernamr=$username";
```

```
echo $query;
```

نتیجه کد بالا

```
Select From users where usernamr= ''; Delete From users
```

توضیح: همان طور که در کد بالا مشاهده می کنید، ورودی مخرب، باعث می شود که دو کوئری روی پایگاه داده اجرا شود و از آن مهمتر، کوئری دوم است که باعث می شود تا تمام اطلاعات جدول users حذف شود...!

دستورات آماده یا Prepared در MySQLi

در مثال زیر،نحوه ی استفاده از دستورات آماده یا Prepared و همچنین نحوه ی Bind کردن پارامترها در MySQLi نشان داده شده است:

مثال(MySQLi with Prepared Statements)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// ایجاد ارتباط به پایگاه داده
$conn = new mysqli($servername, $username, $password, $dbname);

// اطمینان از صحت ارتباط
if ($conn->connect_error) {
die("Connection failed: " . $conn->connect_error);
}

// تنظیم کوئری و باینند کردن پارامترها با استفاده از دستورات آماده
$stmt = $conn->prepare("INSERT INTO MyGuests (firstname, lastname, email) VALUES (?, ?,
?)");
$stmt->bind_param("sss", $firstname, $lastname, $email);

// تنظیم پارامترها و اجرای کوئری
$firstname = "John";
$lastname = "Doe";
$email = "john@example.com";
$stmt->execute();

$firstname = "Mary";
$lastname = "Moe";
$email = "mary@example.com";
$stmt->execute();

$firstname = "Julie";
$lastname = "Dooley";
$email = "julie@example.com";
```

```
$stmt->execute();

echo "New records created successfully";

$stmt->close();
$conn->close();
?>
```

توضیح مثال بالا:

```
"INSERT INTO MyGuests (firstname, lastname, email) VALUES (?, ?, ?)"
```

در کوئری مان می توانیم از علامت سوال (؟) استفاده کنیم، این علامت در ادامه با یک عدد صحیح (integer)، رشته (string)، عدد اعشاری (double) یا داده های با اندازه بزرگ (**OB**ject **L**arge **B**inary - BLOB) جایگزین خواهد شد.

در ادامه نگاهی به تابع ()bind_param خواهیم انداخت:

```
$stmt->bind_param("sss", $firstname, $lastname, $email);
```

با استفاده از تابع ()bind_param، می توانیم پارامترها را به کوئری SQL مان Bind کنیم و به پایگاه داده بگوییم که چه پارامترهایی داریم. همچنین با استفاده از آرگومان اول یعنی "sss"، نوع پارامترها را تعیین می کنیم. کاراکتر s به MySQL می گوید که پارمتر ما از نوع رشته (string) است.

نوع آرگومان ها می تواند یکی از چهار گزینه زیر باشد:

- i - integer
- d - double
- s - string
- b - BLOB

برای هر کدام از آرگومان ها، باید یکی از چهار مقدار بالا را تنظیم کنیم.

زمانی که نوع پارامترها را مشخص می کنیم، ریسک تزریقات SQL یا SQL injections توسط هکرها را به حداقل می رسانیم.

توجه: اگر قصد دارید که داده ی دریافتی از کاربران را در پایگاه داده ذخیره کنید، اعتبارسنجی داده ها یک امر بسیار مهم است.

دستورات آماده یا Prepared در PDO

در مثال زیر،نحوه ی استفاده از دستورات آماده یا Prepared و همچنین نحوه ی Bind کردن پارامترها در PDO نشان داده شده است:

مثال(PDO with Prepared Statements)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";

try {
//ایجاد ارتباط
$conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
// تنظیم حالت برخورد با خطاها
$conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

// تنظیم کوئری و باینند کردن پارامترها با استفاده از دستورات آماده
$stmt = $conn->prepare("INSERT INTO MyGuests (firstname, lastname, email)
VALUES (:firstname, :lastname, :email)");
$stmt->bindParam(':firstname', $firstname);
$stmt->bindParam(':lastname', $lastname);
$stmt->bindParam(':email', $email);

// تنظیم پارامترها و اجرای کوئری
$firstname = "John";
$lastname = "Doe";
$email = "john@example.com";
$stmt->execute();

// تنظیم پارامترها و اجرای کوئری
$firstname = "Mary";
$lastname = "Moe";
$email = "mary@example.com";
$stmt->execute();

// تنظیم پارامترها و اجرای کوئری
$firstname = "Julie";
$lastname = "Dooley";
$email = "julie@example.com";
$stmt->execute();

echo "New records created successfully";
}
catch(PDOException $e)
{
echo "Error: " . $e->getMessage();
}
```

```
$conn = null;
?>
```

آموزش MySQL-دستور Select

انتخاب داده ها از پایگاه داده MySQL

دستور SELECT برای انتخاب داده از یک جدول استفاده می شود.

نحوه استفاده:

می توانیم با استفاده از نام ستون ها، تعداد محدودی از ستون های یک جدول را انتخاب کنیم:

```
SELECT column_name(s) FROM table_name
```

یا می توانیم با استفاده از کاراکتر *، تمام ستون های یک جدول را انتخاب کنیم:

```
SELECT * FROM table_name
```

برای کسب اطلاعات بیشتر در مورد SQL، به لینک روبرو مراجعه فرمایید: آموزش SQL-مقدمه

انتخاب داده ها با استفاده از MySQLi

در مثال زیر، ستون های id و firstname و lastname از جدول MyGuests در صفحه نمایش داده خواهند شد:

مثال(MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// ارتباط به پایگاه داده
$conn = new mysqli($servername, $username, $password, $dbname);
// مطمئن شدن از صحت ارتباط
if ($conn->connect_error) {
die("Connection failed: " . $conn->connect_error);
}
//تنظیم کوئری
$sql = "SELECT id, firstname, lastname FROM MyGuests";
//اجرای کوئری و قرار دادن نتیجه در متغیر
$result = $conn->query($sql);

if ($result->num_rows > 0) {
// نمایش داده ها به ازای هر ردیف
while($row = $result->fetch_assoc()) {
echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"].
"<br>";
}
```

```
} else {
echo "0 results";
}
//بستن ارتباط با پایگاه داده
$conn->close();
?>
```

خروجی کد بالا:

```
id: 1 - Name: John Doe
id: 2 - Name: Mary Moe
id: 3 - Name: Julie Dooley
```

توضیح مثال:

۱. ابتدا یک کوئری SQL را تنظیم کردیم که ستون های id و firstname و lastname از جدول MyGuests را انتخاب می کند. نتیجه اجرای کوئری توسط تابع query() در متغیر $result ذخیره می شود.

۲. حالا نتیجه کوئری در متغیر $result است و می خواهیم که آنها را در خروجی نمایش دهیم، اما قبل از آن باید مطمئن شویم که کوئری ما نتیجه ای را در بر داشته است، بنابراین با استفاده از تابع num_rows() چک می کنیم که ردیف ها بزرگتر از صفر باشد.

۳. با استفاده از تابع fetch_assoc() ردیف اول داده ها برگردانده می شود، در یک اسکریپت با هر بار فراخوانی این تابع ردیف های بعدی بر می گردد.

۴. با استفاده از حلقه while و با هر بار فراخوانی تابع fetch_assoc() نتیجه در متغیر $row ذخیره می شود و این کار تا آخرین رکورد ادامه می یابد و اطلاعات ستون ها چاپ می شود.

در مثال زیر نیز با استفاده از روش MySQLi procedural ستون های id و firstname و lastname از جدول MyGuests در صفحه نمایش داده خواهند شد:

مثال(MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// ایجاد ارتباط با پایگاه داده
$conn = mysqli_connect($servername, $username, $password, $dbname);
// اطمینان از صحت ارتباط
if (!$conn) {
die("Connection failed: " . mysqli_connect_error());
}
//تنظیم کوئری
$sql = "SELECT id, firstname, lastname FROM MyGuests";
```

```
$result = mysqli_query($conn, $sql);

if (mysqli_num_rows($result) > 0) {
// نمایش داده ها به ازای هر ردیف
while($row = mysqli_fetch_assoc($result)) {
echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"].
"<br>";
}
} else {
echo "0 results";
}

mysqli_close($conn);
?>
```

خروجی کد بالا:

```
id: 1 - Name: John Doe
id: 2 - Name: Mary Moe
id: 3 - Name: Julie Dooley
```

همچنین می توانید خروجی را در یک جدول HTML نمایش دهید:

مثال(MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// ایجاد ارتباط با پایگاه داده
$conn = new mysqli($servername, $username, $password, $dbname);
// اطمینان از صحت ارتباط
if ($conn->connect_error) {
die("Connection failed: " . $conn->connect_error);
}
//تنظیم کوئری
$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
echo "<table><tr><th>ID</th><th>Name</th></tr>";
// نمایش داده ها به ازای هر ردیف
while($row = $result->fetch_assoc()) {
echo "<tr><td>".$row["id"]."</td><td>".$row["firstname"]."
".$row["lastname"]."</td></tr>";
}
```

```
echo "</table>";
} else {
echo "0 results";
}
$conn->close();
?>
```

خروجی کد بالا، البته با استایل زیر:

```
<style>
table, th, td {
border: 1px solid black;
}
</style>
```

خروجی:

ID	Name
۱	John Doe
۲	Mary Moe
۳	Julie Dooley

انتخاب داده ها با استفاده از PDO + دستورات آماده

در مثال زیر، از دستورات آماده (prepared statements) استفاده شده است.

در مثال زیر نیز با استفاده از روش PDO ستون های id و firstname و lastname از جدول MyGuests در یک جدول HTMLی نمایش داده خواهند شد:

مثال(PDO)

```
<?php
echo "<table style='border: solid 1px black;'>";
echo "<tr><th>Id</th><th>Firstname</th><th>Lastname</th></tr>";

class TableRows extends RecursiveIteratorIterator {
function __construct($it) {
parent::__construct($it, self::LEAVES_ONLY);
}

function current() {
return "<td style='width:150px;border:1px solid black;'>" . parent::current(). "</td>";
}
```

```
function beginChildren() {
echo "<tr>";
}

function endChildren() {
echo "</tr>" . "\n";
}
}

$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";

try {
$conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
$conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
$stmt = $conn->prepare("SELECT id, firstname, lastname FROM MyGuests");
$stmt->execute();

// set the resulting array to associative
$result = $stmt->setFetchMode(PDO::FETCH_ASSOC);
foreach(new TableRows(new RecursiveArrayIterator($stmt->fetchAll())) as $k=>$v) {
echo $v;
}
}
catch(PDOException $e) {
echo "Error: " . $e->getMessage();
}
$conn = null;
echo "</table>";
?>
```

خروجی کد بالا:

Id	Firstname	Lastname
۱	John	Doe
۲	Mary	Moe
۳	Julie	Dooley

آموزش MySQL-فیلتر کردن

فیلتر کردن رکوردها

قسمت WHERE در انتهای دستورات SELECT، UPDATE و یا DELETE قرار می گیرد و رکوردها را فیلتر می کند(مثلاً در دستور DELETE مشخص می کند که عمل حذف روی کدام رکوردها اتفاق بیافتد).

نحوه استفاده:

```
SELECT column_name(s)
FROM table_name
WHERE column_name operator value
```

برای کسب اطلاعات بیشتر در مورد SQL، به لینک روبرو مراجعه فرمایید: آموزش SQL-مقدمه

در مثال زیر، بشرطی ستون های id و firstname و lastname از جدول MyGuests انتخاب می شوند که مقدار ستون lastname آنها برابر "Doe" باشد:

مثال(MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// ارتباط به پایگاه داده
$conn = new mysqli($servername, $username, $password, $dbname);
// مطمئن شدن از صحت ارتباط
if ($conn->connect_error) {
die("Connection failed: " . $conn->connect_error);
}
//تنظیم کوئری
$sql = "SELECT id, firstname, lastname FROM MyGuests WHERE lastname='Doe'";
//اجرای کوئری و قرار دادن نتیجه در متغیر
$result = $conn->query($sql);

if ($result->num_rows > 0) {
// نمایش داده ها به ازای هر ردیف
while($row = $result->fetch_assoc()) {
echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"].
"<br>";
}
} else {
echo "0 results";
}
//بستن ارتباط با پایگاه داده
$conn->close();
?>
```

خروجی کد بالا:

```
id: 1 - Name: John Doe
```

مثال(MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// ایجاد ارتباط با پایگاه داده
$conn = mysqli_connect($servername, $username, $password, $dbname);
// اطمینان از صحت ارتباط
if (!$conn) {
die("Connection failed: " . mysqli_connect_error());
}
//تنظیم کوئری
$sql = "SELECT id, firstname, lastname FROM MyGuests WHERE lastname='Doe'";
$result = mysqli_query($conn, $sql);

if (mysqli_num_rows($result) > 0) {
// نمایش داده ها به ازای هر ردیف
while($row = mysqli_fetch_assoc($result)) {
echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"].
"<br>";
}
} else {
echo "0 results";
}

mysqli_close($conn);
?>
```

خروجی کد بالا:

```
id: 1 - Name: John Doe
```

آموزش MySQL-مرتب کردن

مرتب کردن داده ها

با استفاده از کلمه کلیدی ORDER BY می توانید داده ها را مرتب کنید.

به صورت پیشفرض داده ها به صورت صعودی مرتب می شوند. (ASC)

اگر مایلید که اطلاعات به صورت نزولی مرتب شوند باید از کلمه کلیدی DESC استفاده کنید.

نحوه استفاده:

```
SELECT column_name(s)
FROM table_name
ORDER BY column_name(s) ASC|DESC
```

برای کسب اطلاعات بیشتر در مورد SQL، به لینک روبرو مراجعه فرمایید: آموزش SQL-مقدمه

در مثال زیر ستون های id و firstname و lastname از جدول MyGuests انتخاب شده و بر اساس ستون lastname نمایش داده می شود (ASC):

مثال(MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// ارتباط به پایگاه داده
$conn = new mysqli($servername, $username, $password, $dbname);
// مطمئن شدن از صحت ارتباط
if ($conn->connect_error) {
die("Connection failed: " . $conn->connect_error);
}
//تنظیم کوئری
$sql = "SELECT id, firstname, lastname FROM MyGuests ORDER BY lastname";
//اجرای کوئری و قرار دادن نتیجه در متغیر
$result = $conn->query($sql);

if ($result->num_rows > 0) {
// نمایش داده ها به ازای هر ردیف
while($row = $result->fetch_assoc()) {
echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"].
"<br>";
}
} else {
echo "0 results";
}
//بستن ارتباط با پایگاه داده
$conn->close();
?>
```

خروجی کد بالا:

```
id: 1 - Name: John Doe
id: 3 - Name: Julie Dooley
id: 2 - Name: Mary Moe
```

در مثال زیر داده ها بصورت DESC (بزرگ به کوچک) نمایش داده می شود:

مثال(MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// ایجاد ارتباط با پایگاه داده
$conn = mysqli_connect($servername, $username, $password, $dbname);
// اطمینان از صحت ارتباط
if (!$conn) {
die("Connection failed: " . mysqli_connect_error());
}
//تنظیم کوئری
$sql = "SELECT id, firstname, lastname FROM MyGuests ORDER BY lastname DESC";
$result = mysqli_query($conn, $sql);

if (mysqli_num_rows($result) > 0) {
// نمایش داده ها به ازای هر ردیف
while($row = mysqli_fetch_assoc($result)) {
echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"].
"<br>";
}
} else {
echo "0 results";
}

mysqli_close($conn);
?>
```

خروجی کد بالا:

```
id: 2 - Name: Mary Moe
id: 3 - Name: Julie Dooley
id: 1 - Name: John Doe
```

مرتب سازی براساس دو ستون

هنگام مرتب سازی براساس بیشتر از یک ستون، ستون دوم تنها موقعی استفاده می شود که مقادیر در ستون اول مساوی باشند.

در واقع ستون دوم موقعی در مرتب سازی دخالت می کند که مقادیر ردیفها در ستون اول مساوی باشند و در بقیه موارد اثری ندارد.

نحوه استفاده:

```
SELECT column_name(s)
FROM table_name
ORDER BY column1, column2
```

آموزش MySQL-دستور Update

ویرایش اطلاعات یک جدول با استفاده از MySQLi و PDO

دستور UPDATE برای ویرایش اطلاعات یک جدول استفاده می شود.

نحوه استفاده:

```
UPDATE table_name
SET column1=value, column2=value2,...
WHERE some_column=some_value
```

نکته: اگر در دستور UPDATE، قسمت WHERE را در نظر نگیرید، کلیه ستونهایی که مشخص کرده اید، ویرایش می شوند، در حقیقت قسمت WHERE مشخص می کند که چه ردیف هایی باید ویرایش شوند.

برای کسب اطلاعات بیشتر در مورد SQL، به لینک روبرو مراجعه فرمایید: آموزش SQL-مقدمه

به جدول "MyGuests" که در فصل های قبلی از آن استفاده شده است توجه کنید:

id	firstname	lastname	email	reg_date
۱	John	Doe	john@example.com	۲۳-۱۰-۲۰۱۴ ۱۴:۲۶:۱۵
۲	Mary	Moe	mary@example.com	۲۳-۱۰-۲۰۱۴ ۱۰:۲۲:۳۰

در مثال زیر، رکوردی که id آن برابر با ۲ است در جدول "MyGuests" ویرایش خواهد شد:

مثال (MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// ایجاد ارتباط
$conn = new mysqli($servername, $username, $password, $dbname);
// اطمینان از صحت ارتباط
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "UPDATE MyGuests SET lastname='Doe' WHERE id=2";

if ($conn->query($sql) === TRUE) {
```

```
    echo "Record updated successfully";
} else {
    echo "Error updating record: " . $conn->error;
}

$conn->close();
?>
```

مثال(MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// ایجاد ارتباط
$conn = mysqli_connect($servername, $username, $password, $dbname);
// اطمینان از صحت ارتباط
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

$sql = "UPDATE MyGuests SET lastname='Doe' WHERE id=2";

if (mysqli_query($conn, $sql)) {
    echo "Record updated successfully";
} else {
    echo "Error updating record: " . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

مثال(PDO)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";

try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
    // تنظیم حالت برخورد با خطاها
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

```
    // تنظیم کوئری ویرایش
    $sql = "UPDATE MyGuests SET lastname='Doe' WHERE id=2";

    // دستورات آماده
    $stmt = $conn->prepare($sql);

    // اجرای کوئری
    $stmt->execute();

    // چاپ پیغام موفقیت آمیز بودن عملیات ویرایش
    echo $stmt->rowCount() . " records UPDATED successfully";
    }
catch(PDOException $e)
    {
    echo $sql . "<br>" . $e->getMessage();
    }

$conn = null;
?>
```

نتیجه کار بعد از ویرایش رکورد:

reg_date	Email	lastname	firstname	id
۱۴:۲۶:۱۵ ۲۳-۱۰-۲۰۱۴	john@example.com	Doe	John	۱
۱۰:۲۲:۳۰ ۲۳-۱۰-۲۰۱۴	mary@example.com	Doe	Mary	۲

آموزش MySQL-دستور Delete

حذف اطلاعات یک جدول با استفاده از MySQLi و PDO

دستور DELETE FROM برای حذف رکورد از یک جدول استفاده می شود.

نحوه استفاده:

```
DELETE FROM table_name
WHERE some_column = some_value
```

نکته: قسمت WHERE مشخص می کند که چه رکورد یا رکوردهایی باید حذف بشوند، توجه داشته باشید که اگر این قسمت را در نظر نگیرید کلیه رکوردهای جدول حذف می شوند.

برای کسب اطلاعات بیشتر در مورد SQL، به لینک روبرو مراجعه فرمایید: آموزش SQL-مقدمه

به جدول "MyGuests" که در فصل های قبلی همین آموزش ایجاد شده است توجه کنید:

reg_date	email	lastname	firstname	Id
۲۳-۱۰-۲۰۱۴ ۱۴:۲۶:۱۵	john@example.com	Doe	John	۱
۲۳-۱۰-۲۰۱۴ ۱۰:۲۲:۳۰	mary@example.com	Moe	Mary	۲
۲۶-۱۰-۲۰۱۴ ۱۰:۴۸:۲۳	julie@example.com	Dooley	Julie	۳

در مثال زیر، رکوردی که id آن برابر با ۳ است در جدول "MyGuests" حذف خواهد شد:

مثال(MySQLi Object-oriented)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// ایجاد ارتباط
$conn = new mysqli($servername, $username, $password, $dbname);
// اطمینان از صحت ارتباط
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// تنظیم کوئری حذف
$sql = "DELETE FROM MyGuests WHERE id=3";

if ($conn->query($sql) === TRUE) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record: " . $conn->error;
}

$conn->close();
?>
```

مثال(MySQLi Procedural)

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// ایجاد ارتباط
$conn = mysqli_connect($servername, $username, $password, $dbname);
```

```php
// اطمینان از صحت ارتباط
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// تنظیم کوئری حذف
$sql = "DELETE FROM MyGuests WHERE id=3";

if (mysqli_query($conn, $sql)) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record: " . mysqli_error($conn);
}

mysqli_close($conn);
?>
```

مثال(PDO)

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";

try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
    // تنظیم حالت برخورد با خطاها
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    // تنظیم کوئری حذف
    $sql = "DELETE FROM MyGuests WHERE id=3";

    // بدلیل اینکه کوئری نتیجه ای را برنمی گرداند از تابع زیر استفاده می کنیم
    $conn->exec($sql);
    echo "Record deleted successfully";
    }
catch(PDOException $e)
    {
    echo $sql . "<br>" . $e->getMessage();
    }

$conn = null;
?>
```

نتیجه کار بعد از حذف رکورد:

reg_date	email	lastname	Firstname	Id
۱۴:۲۶:۱۵ ۲۳-۱۰-۲۰۱۴	john@example.com	Doe	John	۱
۱۰:۲۳:۳۰ ۲۳-۱۰-۲۰۱۴	mary@example.com	Moe	Mary	۲

آموزش MySQL-صفحه بندی

محدود کردن رکوردهای انتخاب شده در پایگاه داده MySQL

در پایگاه داده MySQL، با استفاده از قسمت LIMIT می توانید تعداد رکوردهای برگشتی را محدود کنید.

با وجود امکان LIMIT دیگر بسادگی می توانید **صفحه بندی** یا pagination را روی جداول بزرگ پیاده نمایید. همان طور که می دانید در چنین جدول هایی با تعداد رکوردهای بالا، نمایش تمام رکوردها می تواند کارایی برنامه را بسیار کاهش دهد.

فرض کنید بخواهیم تمام رکوردهای ۱ تا ۳۰ از جدول "Orders" را انتخاب کنیم، بنابراین کد زیر را خواهیم داشت:

```
$sql = "SELECT * FROM Orders LIMIT 30";
```

زمانی که کوئری بالا اجرا می شود، تنها ۳۰ رکورد اول انتخاب خواهد شد.

اگر بخواهیم رکوردهای ۱۶ تا ۲۵ را انتخاب کنیم، چه باید کرد؟

با استفاده از دستور OFFSET در پایگاه داده MySQL می توانید مشکل بالا را حل کنید ...!

در مثال زیر، تنها ۱۰ رکورد انتخاب می شود (LIMIT 10) اما از رکورد ۱۶ به بعد این انتخاب صورت خواهد گرفت (OFFSET 15):

```
$sql = "SELECT * FROM Orders LIMIT 10 OFFSET 15";
```

برای کوتاه تر شدن کد می توانید بصورت زیر عمل کنید:

```
$sql = "SELECT * FROM Orders LIMIT 15, 10";
```

توجه فرمایید که زمانی که از کاما (,) استفاده می کنید، گزینه های "تعداد رکوردهای برگشتی" و "نقطه شروع" برعکس خواهد شد.

آموزش MySQL-ارتباط ODBC

ایجاد یک ارتباط ODBC

با ODBC یا Open Database Connectivity می توان از هر کامپیوتری که روی شبکه قرار دارد به هر پایگاه داده ای متصل شد البته تازمانی که ارتباط ODBC برقرار باشد.

در اصل ODBC فراهم کننده یک رابط نرم افزاری (API) می باشد که بدین وسیله می توان از DBMS های مختلف استفاده کرد. هدف طراحان ODBC بوجود آوردن یک بستر مستقل از زبان های برنامه نویسی، سیستم عامل ها و DBMS ها بوده است.

در زیر چگونگی ایجاد یک ارتباط ODBC به پایگاه داده Access توضیح داده شده است:

۱. در کنترل پانل ویندوز روی آیکون Administrative Tools کلیک کنید.

۲. آیکون Data Sources را باز کنید. (ODBC)

۳. تب System DSN را انتخاب کنید.

۴. روی دکمه Add در تب System DNS کلیک کنید.

۵. Microsoft Access Driver را انتخاب کنید و دکمه Finish را کلیک کنید.

۶. در صفحه باز شده محل پایگاه داده را انتخاب کنید.

۷. یک نام مناسب در قسمت Data Source Name وارد کنید.

۸. دکمه OK را کلیک کنید.

توجه داشته باشید که تنظیمات بالا باید روی کامپیوتری انجام شود که وب سایتتان روی آن قرار دارد و در ادامه اگر Internet Information Server یا IIS روی کامپیوترتان نصب باشد، سایتتان به درستی کار خواهد کرد، اما اگر وب سایتتان روی یک server راه دور قرار دارد، برای تنظیم کردن موارد بالا باید دسترسی فیزیکی به آن سرور داشته باشید و یا می توانید از میزبان وبتان در این مورد سوال کنید.

ارتباط به یک پایگاه داده

تابع ()odbc_connect برای ارتباط به پایگاه داده ODBC استفاده می شود. این تابع چهار پارامتر دارد:

- data source name
- username
- password
- optional cursor type

می توان از تابع ()odbc_exec برای اجرای یک کوئری روی پایگاه داده استفاده نمود.

مثال: در مثال زیر یک ارتباط به پایگاه داده northwind ایجاد کرده ایم و سپس یک کوئری اجرا شده است:

```
$conn=odbc_connect('northwind','','');
$sql="SELECT * FROM customers";
$rs=odbc_exec($conn,$sql);
```

بازیابی رکودها

تابع ()odbc_fetch_row برای بازیابی رکوردهای، کوئری های اجرا شده استفاده می شود، این تابع اگر ردیفی برای بازیابی باشد، مقدار true والا مقدار false را برمی گرداند. این تابع دو پارامتر دارد:

- نتیجه اجرای یک کوئری
- پارامتر اختیاری شماره ردیف

```
odbc_fetch_row($rs)
```

بازیابی یک فیلد

تابع odbc_result() برای خواندن یک فیلد از یک رکورد استفاده می شود، این تابع دو پارامتر می گیرد:

- نتیجه کوئری اجرا شده
- نام فیلد یا شماره فیلد

کد زیر مقدار فیلد اول از رکورد اول را در متغیر $compname قرار می دهد:

```
$compname=odbc_result($rs,1);
```

کد زیر مقدار فیلدی که نام آن CompanyName است را در متغیر $compname قرار می دهد:

```
$compname=odbc_result($rs,"CompanyName");
```

بستن ارتباط

تابع odbc_close() برای بستن یک ارتباط ODBC استفاده می شود.

```
odbc_close($conn);
```

یک مثال ODBC:

در مثال زیر ابتدا یک Connection ایجاد می کنیم و سپس با اجرای کوئری، نتایج آنرا در یک جدول HTMLی نشان می دهیم:

```
<html>
<body>

<?php
$conn=odbc_connect('northwind','','');
if (!$conn)
  {exit("Connection Failed: " . $conn);}
$sql="SELECT * FROM customers";
$rs=odbc_exec($conn,$sql);
```

```
if (!$rs)
  {exit("Error in SQL");}
echo "<table><tr>";
echo "<th>Companyname</th>";
echo "<th>Contactname</th></tr>";
while (odbc_fetch_row($rs))
  {
  $compname=odbc_result($rs,"CompanyName");
  $conname=odbc_result($rs,"ContactName");
  echo "<tr><td>$compname</td>";
  echo "<td>$conname</td></tr>";
  }
odbc_close($conn);
echo "</table>";
?>

</body>
</html>
```

فصل ۵: آموزش XML در PHP

صفر تا صد PHP -معرفی DOM

تجزیه کننده DOM چیست؟

یک سری اشیاء استاندارد برای دسترسی و دستگاری فایل های HTML و XML فراهم می کند و در سه بخش زیر مجزا شده است:

- هسته DOM: یک سری اشیاء استاندارد برای فایل های ساخت یافته تعریف می کند.
- XML DOM: یک سری اشیاء استاندارد برای فایل های XML تعریف می کند.
- HTML DOM: یک سری اشیاء استاندارد برای فایل های HTML تعریف می کند.

تجزیه کننده XML

برای خواندن، ویرایش کردن، ایجاد و خلاصه دستگاری یک فایل XML، به یک تجزیه کننده XML نیاز داریم.

به صورت کلی دو نوع تجزیه کننده XML وجود دارد:

۱. **تجزیه کننده درختی:** این تجزیه کننده فایل XML را به یک ساختار درختی منتقل می کند و تمام فایل را تحلیل کرده و دسترسی به عناصر درخت را فراهم می کند.
۲. **تجزیه کننده رویدادی:** یک فایل XML را به صورت یک سری از رویدادها می بیند و زمانی که یک رویداد خاص اتفاق می افتد، یک تابع را برای بررسی آن صدا می زند.

تجزیه کننده DOM از نوع درختی است.

به فایل XML زیر توجه کنید:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<from>Amir</from>
```

تجزیه کننده DOM فایل XML بالا را به صورت زیر می بیند(سه سطح):

- سطح اول درخت: یک فایل XML را مشخص می کند.
- سطح دوم درخت: ریشه عنصر را مشخص می کند که تگ <from> است.
- سطح سوم درخت: متن عنصر که کلمه "Amir" است.

نصب تجزیه کننده XML

توابع تجزیه کننده DOM قسمتی از هسته PHP هستند و نیازی به نصب وجود ندارد.

فایل XML زیر در مثال های آینده استفاده خواهد شد:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

لود و چاپ یک فایل XML

مثال: در مثال زیر می خواهیم یک تجزیه کننده XML تعریف کرده و فایل XML بالا را داخل آن لود کنیم و سپس در خروجی چاپ کنیم:

```
<?php
$xmlDoc = new DOMDocument();
$xmlDoc->load("note.xml");
print $xmlDoc->saveXML();
?>
```

خروجی کد بالا:

```
ToveJaniReminderDon't forget me this weekend!
```

اگر در Browser ویندوزتان گزینه "View source" را انتخاب کنید، فایل HTML زیر را خواهید دید:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

مثال بالا یک شی از DOMDocument ایجاد می کند و فایل "note.xml" را داخل آن لود می کند.

سپس تابع ()saveXML فایل XML لود شده را داخل یک رشته قرار می دهد، حالا می توانیم آنرا در خروجی چاپ کنیم.

حلقه زدن در یک فایل XML

مثال: در مثال زیر می خواهیم یک تجزیه کننده XML تعریف کنیم و فایل XML قبلی را در آن لود کرده و بین عناصر آن یک حلقه بزنیم:

```
<?php
$xmlDoc = new DOMDocument();
$xmlDoc->load("note.xml");

$x = $xmlDoc->documentElement;
foreach ($x->childNodes AS $item)
  {
  print $item->nodeName . " = " . $item->nodeValue . "<br />";
  }
?>
```

خروجی کد بالا:

```
#text =
to = Tove
#text =
```

```
from = Jani
#text =
heading = Reminder
#text =
body = Don't forget me this weekend!
#text =
```

در مثال بالا می بینید که بین هر عنصر یک نود خالی وجود دارد.

در زمان ایجاد یک فایل XML، در بین نودهای آن فضاهای خالی قرار می گیرد، تجزیه کننده XML با این فضاهای خالی مانند یک عنصر معمولی برخورد می کند و اگر مراقب آنها نباشید، برای شما مشکل ایجاد خواهند کرد.

صفر تا صد PHP -معرفی Expat

XML چیست؟

XML برای شرح و بیان اطلاعات طراحی شده است، در واقع ساختار اطلاعات را شرح می دهد.

در XML هیچ تگ از پیش تعریف شده ای وجود ندارد، و باید خودمان تگ ها را تعریف کنیم.

تجزیه کننده Expat چیست؟

برای خواندن، ویرایش کردن، ایجاد و خلاصه دستگاری یک فایل XML، به یک تجزیه کننده XML نیاز داریم.

به صورت کلی دو نوع تجزیه کننده XML وجود دارد:

۱. **تجزیه کننده درختی:** این تجزیه کننده فایل XML را به یک ساختار درختی منتقل می کند و تمام فایل را تحلیل کرده و دسترسی به عناصر درخت را فراهم می کند.(همان طور که قبلاً گفته شد تجزیه کننده DOM از نوع درختی است)
۲. **تجزیه کننده رویدادی:** یک فایل XML را به صورت یک سری از رویدادها می بیند و زمانی که یک رویداد خاص اتفاق می افتد، یک تابع را برای بررسی آن صدا می زند.

تجزیه کننده Expat از نوع رویدادی است.

نکته: تجزیه کننده رویدادی، به جای اینکه روی ساختار تمرکز کند روی محتوای یک فایل XML تمرکز می کند، به همین خاطر دسترسی به اطلاعات در این نوع تجزیه کننده سریع تر است.

به فایل XML زیر توجه کنید:

```
<from>Amir</from>
```

یک تجزیه کننده رویدادی فایل XML بالا را به صورت سه رویداد زیر گزارش می دهد:

۱. شروع عنصر: from

۲. شروع بخص اطلاعات با مقدار: "Amir"

۳. بستن عنصر: form

مثال XML بالا با یک فرمت درست نوشته شده است، اما به خاطر اینکه DTD ندارد، معتبر نیست.

نکته: DTD یا Document Type Definition مشخص کننده قوانین به کار رفته در یک سند است، به این قوانین الگو می گویند، الگوها ابزارهای مهمی برای نگهداری پیوستگی اسناد محسوب می گردند. با مقایسه سند معینی با الگوی آن اعتبار آن سند تعیین می شود. اگر یک سند با تمام قوانین موجود در الگوی آن مطابقت داشته باشد به آن یک سند معتبر می گویند. معتبر بودن یک سند نشانه مطلوب بودن داده های آن است.

تجزیه کننده Expat یک تجزیه کننده بدون اعتبار است، و برای آن فرقی نمی کند که سند شنا از DTD استفاده کرده یا نه!

سند XML شما باید در یک فرمت درست نوشته شده باشد والا تجزیه کننده Expat خطایی نمایش خواهد داد.

نصب تجزیه کننده Expat

توابع تجزیه کننده Expat قسمتی از هسته PHP هستند و نیازی به نصب وجود ندارد.

فایل XML زیر در مثال های آینده استفاده خواهد شد:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

اعلان یک تجزیه کننده XML

در PHP می توانیم یک تجزیه کننده XML را اعلان کنیم، برای رویدادهای مختلف، تعدادی رسیدگی کننده (Handler) تعریف می شود و سپس فایل XML را تجزیه می کنیم.

مثال:

```
<?php
//Initialize the XML parser
$parser=xml_parser_create();
```

```
//Function to use at the start of an element
function start($parser,$element_name,$element_attrs)
  {
  switch($element_name)
    {
    case "NOTE": echo "-- Note --<br />"; break;
    case "TO": echo "To: "; break;
    case "FROM": echo "From: "; break;
    case "HEADING": echo "Heading: "; break;
    case "BODY": echo "Message: ";
    }
  }

//Function to use at the end of an element
function stop($parser,$element_name)
  {
  echo "<br />";
  }

//Function to use when finding character data
function char($parser,$data)
  {
  echo $data;
  }

//Specify element handler
xml_set_element_handler($parser,"start","stop");

//Specify data handler
xml_set_character_data_handler($parser,"char");

//Open XML file
$fp=fopen("test.xml","r");

//Read data
while ($data=fread($fp,4096))
  {
  xml_parse($parser,$data,feof($fp)) or die (sprintf("XML Error: %s at line %d",
  xml_error_string(xml_get_error_code($parser)),
  xml_get_current_line_number($parser)));
  }

//Free the XML parser
xml_parser_free($parser);
?>
```

خروجی کد بالا:

```
-- Note --
To: Tove
From: Jani
Heading: Reminder
Message: Don't forget me this weekend!
```

توضیح مثال بالا:

۱. تابع ()xml_parser_create یک تجزیه کننده XML را اعلان می کند.

۲. تابع start به عنوان رسیدگی کننده به رویدادهای مختلف ایجاد شده است.

۳. تابع ()xml_set_element_handler مشخص می کند، زمانی که تجزیه کننده با تگ باز و بسته مواجه شد، کدام تابع اجرا شود.

۴. تابع ()xml_set_character_data_handler مشخص می کند، زمانی که تجزیه کننده با اطلاعات مواجه شد، کدام تابع اجرا شود.

۵. تابع ()xml_parse فایل "test.xml" را تجزیه می کند.

۶. در صورتی که خطایی رخ دهد، تابع ()xml_error_string خطای گزارش شده را به شرحی مناسب تبدیل می کند.

۷. در آخر تابع ()xml_parser_free حافظه اختاص داده شده به تجزه کننده را آزاد می کند.

صفر تا صد PHP -معرفی SimpleXML

SimpleXML چیست؟

SimpleXML یک روش جدید برای خواندن خصوصیات و متن یک عنصر در PHP5 فراهم می کند. (البته اگر طرح بندی سند XML را بدانید)

SimpleXML در مقایسه با تجزیه کننده DOM یا Expat با کد کمتری به اطلاعات عناصر دسترسی پیدا می کند.

SimpleXML یک سند XML را به یک شی تبدیل می کند:

- **عناصر:** بوسیله شیء SimpleXMLElement به یک سری خصوصیات تنها تبدیل می شوند، زمانی که در یک سطح، بیش از یک عنصر وجود دارد، از آرایه ها استفاده می شود.
- **خصوصیات:** خصوصیات عناصر در یک آرایه انجمنی(PHP Arrays) قابل دسترسی است، نام خصوصیت ایندکس آرایه است.
- **اطلاعات عنصر:** متن اطلاعات عنصر به یک رشته تبدیل می شود، اگر یک عنصر بیش از یک متن داشت، آنها را به ترتیبی که پیدا کرده می چیند.

برای موارد پایه ای شبیه زیر، استفاده از SimpleXML بسیار ساده و سریع است:

- خواندن یک فایل XML
- استخراج اطلاعات از یک رشته XMLی

- ویرایش متن یا مشخصه های یک عنصر

نکته: زمانی که با یک فایل XML پیشرفته شبیه namespaceها سروکار دارید بهتر است که از تجزه کننده های Expat یا DOM استفاده کنید.

نصب SimpleXML

توابع تجزیه کننده SimpleXML قسمتی از هسته PHP 5.0 هستند و نیازی به نصب وجود ندارد.

چگونه از SimpleXML استفاه کنیم

لطفاً به فایل XML زیر توجه کنید:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

می خواهیم نام و متن اطلاعات هر عنصر را در خروجی چاپ کنیم.

راه حل؟

۱. فایل XML را لود کنید.
۲. نام اولین عنصر را بخوانید.
۳. یک حلقه روی هر نود(فرزند) ایجاد کنید و برای خواندن اطلاعات هر نود(فرزند) از تابع ()children استفاده کنید.
۴. برای هر نود(فرزند) نام عنصر و متن اطلاعات آنرا چاپ کنید.

مثال:

```
<?php
$xml = simplexml_load_file("test.xml");

echo $xml->getName() . "<br />";

foreach($xml->children() as $child)
  {
  echo $child->getName() . ": " . $child . "<br />";
```

```
  }
?>
```

خروجی کد بالا:

```
note
to: Tove
from: Jani
heading: Reminder
body: Don't forget me this weekend!
```

فصل ۶: آموزش AJAX در PHP

صفر تا صد PHP -مقدمه AJAX

AJAX (آجاکس)، هنر تبادل داده ها با سرور و بروز رسانی بخش هایی از یک صفحه وب، بدون بارگذاری مجدد کل صفحه است.

AJAX (آجاکس) چیست؟

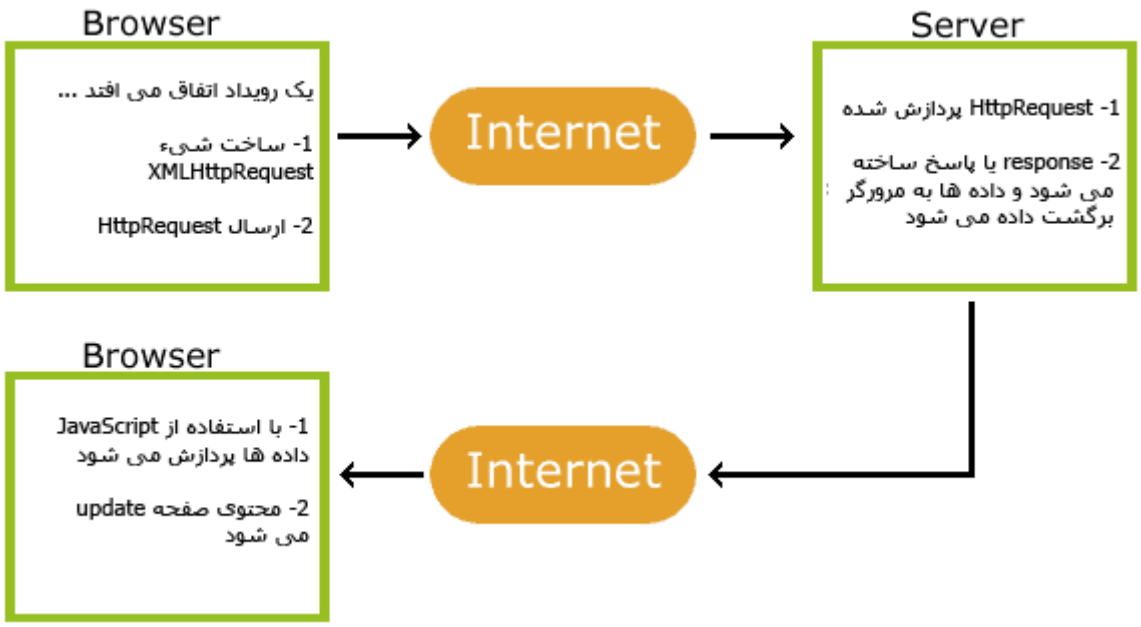
AJAX مخفف کلمات **A**synchronous **J**avaScript **X**ML **A**nd است.

به طور خلاصه، AJAX در مورد بارگذاری داده ها در پس زمینه و نمایش آن بر روی صفحه وب بدون بارگذاری مجدد کل صفحه است.

مثال هایی از استفاده AJAX در برنامه ها:

- Gmail
- Google Maps
- Youtube
- Facebook

AJAX چگونه کار می کند



AJAX براساس استانداردهای اینترنت

AJAX براساس استانداردهای اینترنت است و ترکیبی از موارد زیر را استفاده می کند:

- شیء XMLHttpRequest - برای تبادل داده با سرور، بصورت غیرهمزمان (asynchronously) استفاده می شود.
- JavaScript/DOM - برای نمایش اطلاعات و یا تعامل با داده ها
- CSS - برای مشخص کردن نحوه نمایش داده ها
- XML - اغلب بعنوان یک فرمت برای انتقال داده ها استفاده می شود.

نکته: برنامه های کاربردی AJAX مرورگرند و مستقل از پلتفرم عمل می کنند ...!

امکان Google Suggest

AJAX در سال ۲۰۰۵ توسط Google با امکان Google Suggest این سایت معروف شد.

این امکان (Google Suggest) با استفاده از AJAX یک واسط کاربری بسیار پویا ایجاد کرده است. زمانی که در کادر جستجوی Google شروع به تایپ می کنید، یک کد JavaScript حروف را به سرور ارسال می کند و سرور لیستی از پیشنهادات برای کلمه مورد جستجوی شما را برمی گرداند.

شروع به استفاده از AJAX

در کتاب

PHP، ما نحوه update کردن بخشی از صفحه وب، بدون بارگذاری کل صفحه را با استفاده از AJAX آموزش خواهیم داد.

اسکریپت های سمت سرور، با PHP نوشته خواهد شد.

صفر تا صد PHP -آجاکس در php

با استفاده از AJAX (آجاکس)، می توانید برنامه ها کاربری خود را تعاملی تر کنید.

مثال AJAX (آجاکس) در PHP

مثال زیر نشان خواهد داد شد که چگونه یک صفحه وب می تواند با یک وب سرور گفتگو کند (ارتباط داشته باشد). تا زمانی که کاربر در کادر ورودی، کلمه ای را تایپ می کند، این ارتباط وجود دارد:

مثال

لطفاً نامی را در کادر زیر تایپ نمایید:

نام:

پیشنهادها:

توضیح مثال بالا:

زمانی که کاربر در کادر ورودی بالا، کلمه ای را تایپ کند، تابع "showHint()" اجرا می شود. این تابع زمانی که رویداد "onkeyup" فیلد ورودی رخ دهد، فراخوانی می شود:

```
<html>
<head>
<script>
function showHint(str)
{
if (str.length==0)
  {
  document.getElementById("txtHint").innerHTML="";
  return;
  }
var xmlhttp=new XMLHttpRequest();
xmlhttp.onreadystatechange=function()
  {
  if (xmlhttp.readyState==4 && xmlhttp.status==200)
    {
    document.getElementById("txtHint").innerHTML=xmlhttp.responseText;
    }
  }
xmlhttp.open("GET","gethint.php?q="+str,true);
xmlhttp.send();
}
</script>
</head>
<body>
```

```
<p>< b >لطفاً نامی را در کادر زیر تایپ نمایید:</b></p>
<form>
نام : <input type="text" onkeyup="showHint(this.value)">
</form>
<p>پیشنهادها: <span id="txtHint"></span></p>

</body>
</html>
```

توضیح کد Javascript:

اگر فیلد ورودی خالی باشد (str.length==0)، محتوای عنصر با شناسه "txtHint" خالی و از تابع خارج می شود.

اگر فیلد ورودی خالی نباشد، تابع ()showHint کارهای زیر را انجام می دهد:

- شیء XMLHttpRequest ایجاد می شود.
- زمانی که پاسخ سرور آماده باشد، یک تابع ایجاد و اجرا می شود. (وظیفه این تابع این است که پاسخ دریافت شده از سرور را در عنصر با شناسه "txtHint" قرار دهد)
- درخواست یک فایل به سرور ارسال می شود.
- توجه داشته باشید که یک پارامتر (q) به URL اضافه شده است (با محتوای فیلد ورودی تنظیم می شود)

فایل PHP مثال بالا:

فایلی که توسط کد JavaScript بالا صدا زده می شود یک فایل PHP بنام "gethint.php" است.

در فایل "gethint.php"، یک آرایه از نام ها چک می شود، و آنهایی که با پارامتر (q) متناظر است، چاپ می شود: (پاسخ ارسال می شود)

```
<?php
// Fill up array with names
$a[]="Anna";
$a[]="Brittany";
$a[]="Cinderella";
$a[]="Diana";
$a[]="Eva";
$a[]="Fiona";
$a[]="Gunda";
$a[]="Hege";
$a[]="Inga";
$a[]="Johanna";
$a[]="Kitty";
$a[]="Linda";
$a[]="Nina";
$a[]="Ophelia";
$a[]="Petunia";
$a[]="Amanda";
$a[]="Raquel";
$a[]="Cindy";
$a[]="Doris";
```

```
$a[]="Eve";
$a[]="Evita";
$a[]="Sunniva";
$a[]="Tove";
$a[]="Unni";
$a[]="Violet";
$a[]="Liza";
$a[]="Elizabeth";
$a[]="Ellen";
$a[]="Wenche";
$a[]="Vicky";

// get the q parameter from URL
$q=$_REQUEST["q"]; $hint="";

// lookup all hints from array if $q is different from ""
if ($q !== "")
  { $q=strtolower($q); $len=strlen($q);
    foreach($a as $name)
    { if (stristr($q, substr($name,0,$len)))
      { if ($hint==="")
        { $hint=$name; }
        else
        { $hint .= ", $name"; }
      }
    }
  }

// Output "no suggestion" if no hint were found
// or output the correct values
echo $hint==="" ? "no suggestion" : $hint;
?>
```

صفر تا صد PHP -واکشی اطلاعات

با استفاده از AJAX، می توانید با پایگاه داده ارتباط متقابل داشته باشید.

واکشی اطلاعات از پایگاه داده با استفاده از AJAX

در مثال زیر، نحوه واکشی اطلاعات از پایگاه داده، با استفاده از AJAX نشان داده شده است:

مثال

توضیح مثال - پایگاه داده MySQL

جدولی که در مثال بالا از آن استفاده شده است:

Job	Hometown	Age	LastName	FirstName	id
Developer	Isfahan	۳۰	Pahlavansadegh	Amir	۱
Developer	Isfahan	۳۵	Karimi	Nader	۲
Developer	Isfahan	۲۵	Aslani	Farshid	۳
Developer	Isfahan	۲۸	Ahmadi	Sara	۴
Developer	Isfahan	۳۸	Masomi	Reza	۵

توضیح مثال - صفحه HTML

زمانی که کاربر، یکی از آیتم های لیست کشویی بالا را انتخاب می کند، تابع "showUser()" فراخوانی و اجرا می شود. این تابع توسط رویداد "onchange" راه اندازی می شود:

```
<html>
<head>
<script>
function showUser(str)
{
if (str=="")
  {
  document.getElementById("txtHint").innerHTML="";
  return;
  }
if (window.XMLHttpRequest)
  {// code for IE7+, Firefox, Chrome, Opera, Safari
  xmlhttp=new XMLHttpRequest();
  }
else
  {// code for IE6, IE5
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
xmlhttp.onreadystatechange=function()
  {
  if (xmlhttp.readyState==4 && xmlhttp.status==200)
    {
    document.getElementById("txtHint").innerHTML=xmlhttp.responseText;
    }
  }
xmlhttp.open("GET","getuser.php?q="+str,true);
xmlhttp.send();
}
</script>
</head>
<body>
```

```
<form>
<select name="users" onchange="showUser(this.value)">
</option>انتخاب کنید<option value="">
<option value="1">Amir Pahlavansadegh</option>
<option value="2">Nader karimi</option>
<option value="3">farshad Aslani</option>
<option value="4">Sara Ahmadi</option>
<option value="5">Reza Masomi</option>
</select>
</form>
<br>
 <div id="txtHint"><b>اطلاعات شخص انتخاب شده، اینجا نمایش داده می شود...</b></div>

</body>
</html>
```

تابع showUser() کارهای زیر را انجام می دهد:

- چک می کند که یک شخص انتخاب شده باشد. (اگر گزینه ای انتخاب نشده باشد، محتوی عنصر با شناسه "txtHint" خالی می شود)
- یک شیء XMLHttpRequest ایجاد می شود.
- زمانی که پاسخ سرور آماده باشد، یک تابع ایجاد و اجرا می شود. (وظیفه این تابع این است که پاسخ دریافت شده از سرور را در عنصر با شناسه "txtHint" قرار دهد)
- درخواست یک فایل به سرور ارسال می شود.
- توجه داشته باشید که یک پارامتر (q) به URL اضافه شده است (با محتوای گزینه انتخاب شده)

توضیح مثال - فایل PHP

فایلی که توسط کد JavaScript بالا صدا زده می شود یک فایل PHP بنام "getuser.php" است.

در فایل "getuser.php"، یک query در پایگاه داده MySQL اجرا می شود و نتیجه در یک جدول HTML برگردانده می شود.

```
<?php
$q = intval($_GET['q']);

$con = mysqli_connect('localhost','peter','abc123','my_db');
if (!$con)
  {
  die('Could not connect: ' . mysqli_error($con));
  }

mysqli_select_db($con,"ajax_demo");
$sql="SELECT * FROM user WHERE id = '".$q."'";

$result = mysqli_query($con,$sql);

echo "<table border='1'>
```

```
<tr>
<th>Firstname</th>
<th>Lastname</th>
<th>Age</th>
<th>Hometown</th>
<th>Job</th>
</tr>";

while($row = mysqli_fetch_array($result))
 {
 echo "<tr>";
 echo "<td>" . $row['FirstName'] . "</td>";
 echo "<td>" . $row['LastName'] . "</td>";
 echo "<td>" . $row['Age'] . "</td>";
 echo "<td>" . $row['Hometown'] . "</td>";
 echo "<td>" . $row['Job'] . "</td>";
 echo "</tr>";
 }
echo "</table>";

mysqli_close($con);
?>
```

توضیح: زمانی که درخواست از JavaScript به فایل PHP ارسال می شود، کارهای زیر اتفاق می افتد:

۱. یک ارتباط به پایگاه داده MySQL ایجاد می شود.

۲. شخص مورد نظر پیدا می شود.

۳. یک جدول HTML ایجاد و با داده های مناسب پر می شود و در نهایت داده ها برگشت می شود و در عنصر با شناسه "txtHint" قرار می گیرد.

صفر تا صد PHP -خواندن فایل XML

با استفاده از AJAX، می توانید با یک فایل XML، ارتباط متقابل داشته باشید.

واکشی اطلاعات از یک فایل XML با استفاده از AJAX

در مثال زیر، نحوه واکشی اطلاعات از یک فایل XML، با استفاده از AJAX نشان داده شده است:

مثال

اطلاعات گزینه انتخاب شده، اینجا نمایش داده می شود

توضیح مثال - صفحه HTML

زمانی که کاربر، یکی از آیتم های لیست کشویی بالا را انتخب می کند، تابع "showCD()" فراخوانی و اجرا می شود. این تابع توسط رویداد "onchange" راه اندازی می شود:

```
<html>
<head>
<script>
function showCD(str)
{
if (str=="")
  {
  document.getElementById("txtHint").innerHTML="";
  return;
  }
if (window.XMLHttpRequest)
  {// code for IE7+, Firefox, Chrome, Opera, Safari
  xmlhttp=new XMLHttpRequest();
  }
else
  {// code for IE6, IE5
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
xmlhttp.onreadystatechange=function()
  {
  if (xmlhttp.readyState==4 && xmlhttp.status==200)
    {
    document.getElementById("txtHint").innerHTML=xmlhttp.responseText;
    }
  }
xmlhttp.open("GET","getcd.php?q="+str,true);
xmlhttp.send();
}
</script>
</head>
<body>

<form>
Select a CD:
<select name="cds" onchange="showCD(this.value)">
<option value="">Select a CD:</option>
<option value="Bob Dylan">Bob Dylan</option>
<option value="Bonnie Tyler">Bonnie Tyler</option>
<option value="Dolly Parton">Dolly Parton</option>
</select>
</form>
 <div id="txtHint"><b>اطلاعات گزینه انتخاب شده، اینجا نمایش داده می شود</b></div>

</body>
</html>
```

تابع showCD() کارهای زیر را انجام می دهد:

- چک می کند که یک گزینه انتخاب شده باشد. (اگر گزینه ای انتخاب نشده باشد، محتوی عنصر با شناسه "txtHint" خالی می شود)
- یک شیء XMLHttpRequest ایجاد می شود.
- زمانی که پاسخ سرور، آماده شده باشد یک تابع ایجاد و اجرا می شود. (وظیفه این تابع این است که پاسخ دریافت شده از سرور را در عنصر با شناسه "txtHint" قرار دهد)

- درخواست یک فایل به سرور ارسال می شود.
- توجه داشته باشید که یک پارامتر (q) به URL اضافه شده است (با محتوای گزینه انتخاب شده)

توضیح مثال - فایل PHP

فایلی که توسط کد JavaScript بالا صدا زده می شود یک فایل PHP بنام "getcd.php" است.

در فایل "getcd.php"، یک سند XML بنام "cd_catalog.xml" بارگذاری می شود. در این فایل یک جستجو انجام شده و نتیجه بصورت HTML برگردانده می شود.

```
<?php
$q=$_GET["q"];

$xmlDoc = new DOMDocument();
$xmlDoc->load("cd_catalog.xml");

$x=$xmlDoc->getElementsByTagName('ARTIST');

for ($i=0; $i<=($x->length-1); $i++)
{
//Process only element nodes
if ($x->item($i)->nodeType==1)
  {
  if ($x->item($i)->childNodes->item(0)->nodeValue == $q)
    {
    $y=($x->item($i)->parentNode);
    }
  }
}

$cd=($y->childNodes);

for ($i=0;$i<$cd->length;$i++)
{
//Process only element nodes
if ($cd->item($i)->nodeType==1)
  {
  echo("<b>" . $cd->item($i)->nodeName . ":</b> ");
  echo($cd->item($i)->childNodes->item(0)->nodeValue);
  echo("<br>");
  }
}
?>
```

توضیح: زمانی که درخواست از JavaScript به فایل PHP ارسال می شود، کارهای زیر اتفاق می افتد:

۱. یک شیء XML DOM ایجاد می شود.

۲. تمام عناصر <artist> که با نام ارسال شده از JavaScript تطابق داشته باشند، پیدا می شود.

۳. یک خروجی مناسب چاپ می شود و در نهاییت در عنصر با شناسه "txtHint" قرار می گیرد.

صفر تا صد PHP -جستجوی پیشرفته

با استفاده از AJAX، می توانید جستجوهای کاربر پسندتر و تعاملی تری ایجاد نمایید.

آموزش ساخت جستجوی پیشرفته با استفاده از AJAX در PHP

مزیت های جستجوی پیشرفته در مقایسه با جستجوی سنتی:

- در لحظه ی تایپ، نتیجه نمایش داده خواهد شد.
- اگر تایپ کردن را ادامه دهید، نتیجه محدود تر می شود.
- اگر نتیجه بیش از حد محدود شده است، می توانید کاراکترهای پایانی را پاک کنید تا نتیجه مناسب تری داشته باشید.

مثال زیر، نحوه ساخت یک جستجوی پیشرفته را نشان می دهد، نتیجه در لحظه ای که تایپ می کنید، نمایش داده می شود:

مثال

جستجوی پیشرفته:

نتیجه جستجوی مثال بالا، با استفاده از یک سند XML بنام "links.xml" تولید می شود. بدلیل اینکه می خواستیم یک مثال ساده و کوچک داشته باشیم، تنها شش نتیجه قابل نمایش است.

توضیح مثال - صفحه HTML

زمانی که کاربر در کادر ورودی بالا، کلمه ای را تایپ کند، تابع "showResult()" اجرا می شود. این تابع زمانی فراخوانی می شود که رویداد "onkeyup" فیلد ورودی رخ دهد:

```
<html>
<head>
<script>
function showResult(str)
{
if (str.length==0)
  {
  document.getElementById("livesearch").innerHTML="";
  document.getElementById("livesearch").style.border="0px";
  return;
  }
if (window.XMLHttpRequest)
  {// code for IE7+, Firefox, Chrome, Opera, Safari
  xmlhttp=new XMLHttpRequest();
  }
else
  {// code for IE6, IE5
```

```
 xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
 }
xmlhttp.onreadystatechange=function()
 {
 if (xmlhttp.readyState==4 && xmlhttp.status==200)
  {
  document.getElementById("livesearch").innerHTML=xmlhttp.responseText;
  document.getElementById("livesearch").style.border="1px solid #A5ACB2";
  }
 }
xmlhttp.open("GET","livesearch.php?q="+str,true);
xmlhttp.send();
}
</script>
</head>
<body>

<form>
 <p>جستجوی پیشرفته:</p>
<input type="text" size="30" onkeyup="showResult(this.value)">
<div id="livesearch"></div>
</form>

</body>
</html>
```

توضیح کد Javascript:

اگر فیلد ورودی خالی باشد (str.length==0)، محتوای عنصر با شناسه "livesearch" خالی و خصوصیت border آن با مقدار "px۰" تنظیم می شود و در نهایت، از تابع خارج می شود.

اگر فیلد ورودی خالی نباشد، تابع showResult() کارهای زیر را انجام می دهد:

- شیء XMLHttpRequest ایجاد می شود.
- زمانی که پاسخ سرور آماده باشد، یک تابع ایجاد و اجرا می شود. (وظیفه این تابع این است که پاسخ دریافت شده از سرور را در عنصر با شناسه "livesearch" قرار دهد)
- درخواست یک فایل به سرور ارسال می شود.
- توجه داشته باشید که یک پارامتر (q) به URL اضافه شده است (با محتوای فیلد ورودی تنظیم می شود)

فایل PHP

فایلی که توسط کد JavaScript بالا صدا زده می شود، یک فایل PHP بنام "livesearch.php" است.

در فایل "livesearch.php"، یک سند XML بنام "links.xml" بارگذاری می شود. در این فایل یک جستجو انجام شده و نتیجه بصورت HTML برگردانده می شود:

```
<?php
$xmlDoc=new DOMDocument();
```

```
$xmlDoc->load("links.xml");

$x=$xmlDoc->getElementsByTagName('link');

//get the q parameter from URL
$q=$_GET["q"];

//lookup all links from the xml file if length of q>0
if (strlen($q)>0)
{
$hint="";
for($i=0; $i<($x->length); $i++)
  {
  $y=$x->item($i)->getElementsByTagName('title');
  $z=$x->item($i)->getElementsByTagName('url');
  if ($y->item(0)->nodeType==1)
    {
    //find a link matching the search text
    if (stristr($y->item(0)->childNodes->item(0)->nodeValue,$q))
      {
      if ($hint=="")
        {
        $hint="<a href='" .
        $z->item(0)->childNodes->item(0)->nodeValue .
        "' target='_blank'>" .
        $y->item(0)->childNodes->item(0)->nodeValue . "</a>";
        }
      else
        {
        $hint=$hint . "<br /><a href='" .
        $z->item(0)->childNodes->item(0)->nodeValue .
        "' target='_blank'>" .
        $y->item(0)->childNodes->item(0)->nodeValue . "</a>";
        }
      }
    }
  }
}

// Set output to "no suggestion" if no hint were found
// or to the correct values
if ($hint=="")
  {
  $response="no suggestion";
  }
else
  {
  $response=$hint;
  }

//output the response
echo $response;
?>
```

توضیح: زمانی که درخواست از JavaScript به فایل PHP ارسال می شود، کارهای زیر اتفاق می افتد:

۱. یک شیء XML DOM ایجاد و فایل "links.xml" درون آن load می شود.

۲. تمام عناصر <title> که با نام ارسال شده از JavaScript تطابق داشته باشند، پیدا می شود.

۳. URL صحیح بهمراه عنوان لینک در متغیر "$response" ذخیره می شود. اگر بیشتر از یک مورد پیدا شود، با استفاده از تگ
 آنها را پشت سر هم در متغیر ذکر شده ذخیره می کنیم.

۴. اگر هیچ موردی پیدا نشود متغیر "$response" با مقدار "no suggestion" تنظیم می شود.

صفر تا صد PHP -سیستم نظرسنجی

سیستم نظرسنجی با استفاده از AJAX

مثال زیر، یک سیستم نظر سنجی را نشان می دهد که نتیجه آن بدون بارگذاری مجدد صفحه نمایش داده می شود:

مثال

آیا تا اینجا به PHP و AJAX علاقه مند شده اید؟

خیر:

توضیح مثال - صفحه HTML

زمانی که کاربر، یکی از آیتم های مثال بالا را انتخاب می کند، تابع "getVote()" فراخوانی و اجرا می شود. این تابع توسط رویداد "onclick" راه اندازی می شود:

```
<html>
<head>
<script>
function getVote(int)
{
if (window.XMLHttpRequest)
  {// code for IE7+, Firefox, Chrome, Opera, Safari
  xmlhttp=new XMLHttpRequest();
  }
else
  {// code for IE6, IE5
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
xmlhttp.onreadystatechange=function()
  {
  if (xmlhttp.readyState==4 && xmlhttp.status==200)
    {
    document.getElementById("poll").innerHTML=xmlhttp.responseText;
    }
  }
xmlhttp.open("GET","poll_vote.php?vote="+int,true);
xmlhttp.send();
}
</script>
</head>
<body>
```

```
<div id="poll">
<h3>Do you like PHP and AJAX so far?</h3>
<form>
Yes:
<input type="radio" name="vote" value="0" onclick="getVote(this.value)">
<br>No:
<input type="radio" name="vote" value="1" onclick="getVote(this.value)">
</form>
</div>

</body>
</html>
```

تابع ()getVote کارهای زیر را انجام می دهد:

- یک شیء XMLHttpRequest ایجاد می شود.
- زمانی که پاسخ سرور آماده باشد، یک تابع ایجاد و اجرا می شود. (وظیفه این تابع این است که پاسخ دریافت شده از سرور را در عنصر با شناسه "poll" قرار دهد)
- درخواست یک فایل به سرور ارسال می شود.
- توجه داشته باشید که یک پارامتر (vote) به URL اضافه شده است (با محتوای گزینه انتخاب شده)

توضیح مثال - فایل PHP

فایلی که توسط کد JavaScript بالا صدا زده می شود یک فایل PHP بنام "poll_vote.php" است.

```
<?php
$vote = $_REQUEST['vote'];

//get content of textfile
$filename = "poll_result.txt";
$content = file($filename);

//put content in array
$array = explode("||", $content[0]);
$yes = $array[0];
$no = $array[1];

if ($vote == 0)
  {
  $yes = $yes + 1;
  }
if ($vote == 1)
  {
  $no = $no + 1;
  }

//insert votes to txt file
$insertvote = $yes."||".$no;
$fp = fopen($filename,"w");
fputs($fp,$insertvote);
```

```
fclose($fp);
?>

<h2>Result:</h2>
<table>
<tr>
<td>Yes:</td>
<td>
<img src="/poll.gif"
width='<?php echo(100*round($yes/($no+$yes),2)); ?>'
height='20'>
<?php echo(100*round($yes/($no+$yes),2)); ?>%
</td>
</tr>
<tr>
<td>No:</td>
<td>
<img src="/poll.gif"
width='<?php echo(100*round($no/($no+$yes),2)); ?>'
height='20'>
<?php echo(100*round($no/($no+$yes),2)); ?>%
</td>
</tr>
</table>
```

توضیح: زمانی که درخواست از JavaScript به فایل PHP ارسال می شود، کارهای زیر اتفاق می افتد:

۱. محتوای فایل "poll_result.txt" خوانده می شود.
۲. با استفاده از تابع ()explode، محتوای فایل "poll_result.txt" که شامل یک رشته مثل (۲۰||۵۰) است را در یک آرایه ذخیره می کنیم، سلول اول شامل تعداد رأی های "آری" و سلول دوم تعداد "خیرها" را نشان می دهد.
۳. براساس انتخاب کاربر، یک واحد به تعداد آراء اضافه می شود.
۴. نتیجه در فایل "poll_result.txt" نوشته می شود.
۵. نتیجه نظرسنجی بصورت گرافیکی ایجاد می شود.

فایل متنی poll_result.txt

فایل "poll_result.txt" جایی است که نتیجه نظرسنجی در آن ذخیره می شود.

فایل "poll_result.txt"، می تواند شامل رشته زیر باشد:

```
۰||۰
```

عدد اول نشان دهنده ی رأی "مثب" و عدد دوم رأی "منفی" را نشان می دهد.

توجه: به یاد داشته باشید که مجوز ویرایش این فایل فقط باید برای وب سرور (PHP) وجود داشته باشد. اجازه ی دسترسی کاربران به این فایل نباید وجود داشته باشد.

فصل ۷: آموزش شیء گرایی

صفر تا صد PHP -کلاس ها

کلاس ها در PHP

در این فصل به بررسی یکی از زیباترین، و در عین حال خطرناکترین مباحث برنامه نویسی می پردازیم، این مبحث از این جهت خطرناك 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 در دنیای واقعی کنار خواهید گذاشت.

در اینجا قصد نداریم شروع کنیم به آوردن مثالهایی از Object در دنیای واقعی و فرض می کنیم 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 می گویند.

دلیل استفاده از اشیاء در زبان های برنامه نویسی: در برنامه های حجیم، استفاده از کلاس ها موجب می شود تا کدهای تکراری نوشته نشوند لذا مقدار کدها کمتر می شود و در نتیجه فشار کمتری به سیستم اجرا کننده دستورات می آید و همچنین سرعت پردازش کد ها مقدار زیادی افزایش می یابد.

به عبارت دیگر ویژگی کلاسها در این است که می توان یک کلاس را بعداً با نام های مختلف و متغیرهای مختلف برای استفاده های مشابه استفاده نمود.

---

object یا شیء چیست؟

مجموعه ای از متغیرها و توابع است که از یک الگوی خاص به نام کلاس ساخته شده است. اما کلاس ها چه هستند؟ فرض کنید ما یک شرکت داریم که این شرکت از بخش های مختلفی تشکیل شده است حال ما در هر بخش 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 بدهد و گزارش مالی خود را دریافت کند 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.

نحوه تعریف کلاس:

```
<?php
```

```
//تعریف کلاس
class class_name
{
//Properties
//Method
}

//ساختن یک شی از کلاس
$obj=new class_name();
?>
```

توضیح:

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

۲. شما در هر جای کلاس قادر به تعریف متغیرها یا Propertyهای کلاس هستید، اما بهتر است که آنها را در ابتدای کلاس تعریف کنید.

۳. بعد از تعریف متغیرها، توابع یا Methodها را تعریف می کنیم، توابع درون کلاس همانند توابع معمولی تعریف می شوند.

۴. تمام موارد بالا بین دو علامت "{}" قرار می گیرد.

۵. در برنامه نویسی شئ گرا یک شئ چندین بار با چندین اسم متفاوت و ویژگی های متفاوت (متغیرهای مختلف) می تواند مورد استفاده قرار گیرد، برای ساختن یک شئ از کلمه کلیدی new استفاده می کنیم.

توجه: با بزرگتر شدن سیستم ها، ضروری خواهد بود که از یک ساختار دایرکتوری درخت مانند، برای نگهداری تمامی کلاسهایی که در برنامه وبتان بکار برده اید، استفاده کنید. در ادامه شما می توانید با استفاده از تابع include_once یا require_once برای اضافه کردن کلاسهای تعریف شده به صفحات دلخواهتان استفاده کنید.

صفر تا صد PHP -متغییرهای کلاس

متغییرها یا Properties

اشیاء یا Objectها به متغییرهای خاصی دسترسی دارند که به آنها Property می گویند این Propertyها می توانند در هرجای بدنه کلاس تعریف شوند اما برای اینکه اسکریپتمان مرتب باشد بهتر است که در بالای کلاس تعریف شوند.

مثال: در my_class یک property به نام name با مقدار اولیه "Amir" تعریف شده است:

```
<?php
class my_class
{
  public $name="Amir";
}

$obj1=new my_class();
$obj1->name="Reza";
```

```
$obj2=new my_class();
print "$obj1->name"."<br />";
print "$obj2->name"."<br />";
?>
```

خروجی کد بالا:

```
Reza
Amir
```

توضیح مثال بالا:

۱. در مثال بالا متغییر یا Property تعریف شده، در ابتدا با مقدار "Amir" تنظیم شده است، ولی در ادامه با ساختن شیء obj1 و دسترسی به این متغییر مقدار آنرا به "Reza" تغییر می دهیم.

۲. به این نکته توجه داشته باشید که بعد از اسم کلاس از دو پرانتز خالی استفاده نمودیم چون کلاس را ساده و بدون سازنده (Constructor) ساختیم.

(بعداً به سازنده (Constructor) و مخرب (Destructor) می پردازیم.)

۳. علامت "->" به شما اجازه می دهد تا به متغییر های یک کلاس دسترسی داشته باشید.

آموزش PHP-متدهای کلاس

تعریف Methodها

Methodها در واقع توابعی هستند که داخل کلاس وجود دارند و با علامت "->" صدا زده می شوند، مهمتر اینکه متدها به اعضای متغیرهای یک کلاس دسترسی دارند.

مثال۱: در مثال زیر یک تابع ساده برای چاپ یک عبارت تعریف شده است:

```
<?php
class My_class
{
  private $name="Amir";
  function SayHello()
  {
   print "Hello My Name is $this->name";
  }
}

$obj=new My_Class();
$obj->SayHello();
?>
```

خروجی کد بالا:

```
Hello My Name is Amir
```

توجه: درون خود تابع برای دسترسی به تمامی متغیر ها و توابع دیگر با <-$this و سپس نام تابع یا متغیر می توانیم عمل کنیم.

مثال۲: در مثال زیر ابتدا اسممان، امیر است و بعد از صدا زدن تابع ()SetName به رضا تغییر می یابد:

```
<?php
class My_class
{
  private $name="Amir";
  function SetName($Param)
  {
   $this->name=$Param;
  }
  function SayHello()
  {
   print "Hello My Name is $this->name";
  }
}

$obj=new My_Class();
$obj->SetName("Reza");
$obj->SayHello();
?>
```

خروجی کد بالا:

```
Hello My Name is Reza
```

در مثال بالا یک تابع به نام ()SetName تعریف کردیم که یک پارامتر دارد و می توانیم اسم را در همه جا تغییر بدهیم.

صفر تا صد PHP -سازنده کلاس

سازنده یا Constructor

سازنده تابعی است که در هنگام ایجاد کلاس به صورت اتوماتیک فراخوانی می شود و می توان توسط آن تنظیمات اولیه همانند ایجاد ارتباط با دیتابیس و یا کارهای مشابه را انجام داد.

در PHP 4 سازنده تابعی می باشد که نام آن همان نام کلاس می باشد و در PHP 5 تابعی است که با نام"__construct" می باشد.

مثال۱: در مثال زیر به دو روش، تابع سازنده تعریف شده است:

```
<?php
class className{
    function __construct()
    {
        echo 'className created from PHP5 format';
```

```
    }
    function className()
    {
        echo 'className created from PHP4 format';
    }
}
$a=new className();
?>
```

خروجی کد بالا:

اگر در PHP 5 اجرا شود خروجی آن className created from PHP5 format خواهد بود و اگر در PHP 4 اجرا شود خروجی آن className created from PHP4 format خواهد بود.

توجه: در PHP 5 چنانچه تابع __construct وجود نداشته باشد مدل PHP 4 آن اجرا خواهد شد.

مثال۲:

```
<?php
class My_class
{
  public $name;
  function __construct($Param="Amir")
  {
   $this->name=$Param;
  }
  function SayHello()
  {
   print "Hello My Name is $this->name"."<br />";
  }
}

$obj1=new My_Class();
$obj1->SayHello();
$obj2=new My_Class("Reza");
$obj2->SayHello();
?>
```

خروجی کد بالا:

```
Amir
Reza
```

همانطور که دیدید در مثال بالا یک سازنده ساختیم و مقدار دیفالتش را روی "Amir" گذاشتیم که اگر هیچی وارد نشد این عبارت چاپ بشود.

مخرب یا Destructors

درکنار سازنده مخرب نیز وجود دارد که معکوس سازنده عمل می کند، قبل از، از بین رفتن کلاس اجرا می شود و برای خالی کردن حافظه و یا قطع ارتباط دیتابیس و کارهای مشابه به کار می رود.

در واقع مخرب، زمانی فراخوانی می شود که تمام رفرنس های کلاس مورد نظر رفته اند.

مثال:

```
<?php
class className{
    funcrion __construct()
    {
        echo 'Start of className';
    }
    function __destruct()
    {
        echo 'End of className';
    }
}
$a=new className();
?>
```

خروجی کد بالا:

```
Start of className
End of className
```

صفر تا صد PHP -ارث بری کلاس

ارث بری یک کلاس از کلاس دیگر

یکی از مزیت های برنامه نویسی شی گرا، ارث بری یا Inheritance می باشد. ارث بری قابلیت توسعه کلاس را به برنامه نویس می دهد که توسط آن براحتی می توان کلاس نوشته شده را update کرد بدون آنکه بخواهیم در تعریف اصلی کلاس تغییری ایجاد نماییم.

با استفاده از کلمه کلیدی extends در جلوی نام کلاس و در ادامه نام کلاس والد، مشخص می کنیم که کلاسمان از کلاس والد خود ارثبری داشته باشد.

کلاس توسعه دهنده یک کلاس، تمام توابع و متغیرهای کلاس اول را شامل می شود.

مثال:

```php
<?php
class a{
    function test()
    {
        echo "Test in class a"."<br />";
    }
    function show($var)
    {
        echo "in class a: the varible is $var"."<br />";
    }
}
class b extends a {
    function test()
    {
        echo "Test in class b"."<br />"; ;
    }
}
class c extends b {
    function test()
    {
        parent::test();
    }
}
class d extends c {
    function test()
    {
        a::test();
    }
}
$a = new a();
$b = new b();
$c = new c();
$d = new d();
$a->test();
$b->test();
$b->show('Iran');
$c->test();
$d->test();
?>
```

خروجی کد بالا:

```
Test in class a
Test in class b
in class a: the varible is Iran
Test in class b
Test in class a
```

توضیح مثال بالا:

۱. در مثال بالا کلاس a را تعریف کردیم و a را در b توسعه داده و b را در کلاس c و کلاس c را نیز در d توسعه دادیم.

۲. برای اجرای یک تابع یا یک متغیر در یک کلاس دیگر می توانیم از نام کلاس و :: استفاده نماییم.(نام تابع یا متغیر کلاس والد::نام والد کلاس فعلی)

۳. توجه به این نکته نیز ضروری است که با استفاده از کلمه parent میتوانیم کلاس توسعه داده شده یا به اصطلاح والد را فراخوانی کنیم.

صفر تا صد PHP -متغییر static

متغیر ها و توابع Static

به متغیر ها و توابع static برخلاف متغیر ها و توابع معمولی می توانند بدون ایجاد شئ از کلاس، دسترسی داشت.

نحوه دسترسی:

نام متغیر یا تابعی که static تعریف شده :: نام کلاس

چهار محدوده نمایش وجود دارد:

۱. **public** : منبع مورد نظر در هر قسمتی قابل استفاده می باشد.

۲. **protected** : فقط در کلاسی که تعریف شده و کلاس هایی که آن را توسعه می دهند قابل استفاده می باشد.

۳. **private** : فقط در کلاسی که تعریف شده قابل استفاده می باشد.

۴. **final** : در هر جایی قابل استفاده می باشد، اما در کلاس های توسعه دهنده نمی تواند دوباره تعریف شود.(final به کلاس ها نیز میتواند تخصیص یابد، کلاس هایی که به صورت final تعریف شده اند، نمی توانند توسعه یابند.)

مثال: در مثال زیر متغیر name و تابع ()action به صورت static تعریف شده است:

```
<?php
class My_Class{
    static $name="Example.com is for tutorial\n";
    static function action()
    {
        echo "PHP is easy!!\n";
    }
}
My_Class::action();
echo My_Class::$name;
$obj=new My_Class();
$obj->action();
//echo $obj->name;  Undefined "Property" name
echo $obj::$name;
?>
```

خروجی کد بالا:

```
PHP is easy!!
Example.com is for tutorial
PHP is easy!!
Example.com is for tutorial
```

توجه: به متغیرهای Static یک کلاس، نمی توان از طریق شیء و عملگر فلش (<-) دسترسی داشت (با خطای `Undefined "Property name` مواجه خواهید شد.)

ثوابت (const)

ثابت ها یا همان const ها در کلاس ها همانند متغیر ها هستند با این تفاوت که مقدار آنها قابل تغییر نمی باشد و با این مزیت که در هرجایی میتوانند مورد استفاده قرار گیرند!

مثال:

```
<?php
class My_Class{
    const name="www.example.com<br />";
    function test()
    {
        echo self::name;
    }
}
echo My_Class::name;
$obj=new My_Class;
$obj->test();
?>
```

خروجی کد بالا:

```
www.example.com
www.example.com
```

صفر تا صد PHP -تجرد یا Abstract

تجرد یا Abstract:

تجرد یا Abstract موقعی استفاده می شود که بخواهیم کلاس یا تابعی را بدون داشتن بدنه تعریف کنیم و بخواهیم بدنه آن را بعداً تعریف کنیم.

به عنوان مثال اگر بخواهیم چندین کلاس مختلف برای کار با دیتابیس های مختلف در برنامه تعریف کنیم که در آن برخی کارها یکسان می باشد؛ میتوانیم یک کلاس abstract برای تعریف اولیه و توابع یکسان آنها تعریف کنیم و سپس به نسبت هر دیتابیس توابع مخصوص آن را فراخوانی کرد.

مزیت های استفاده از abstarct:

۱. کم تر شدن کدهای نوشته شده

۲. افزایش سرعت برنامه

۳. روان تر و خواناتر بودن کدها

مثال:

```
<?php
abstract class Base_DB{
    private $id;
    abstract function update($data,$id);
    abstract function insert($data);
    function save($data){
        if(is_null($this->id)){
            $this->insert($data);
        }else{
            $this->update($data,$this->id);
        }
    }
}
class MySQL_DB extends Base_DB{
    function update($data,$id)
    {
        // کدهای مربوط به ویرایش
    }
    function insert($data)
    {
        // کدهای مربوط به درج کردن
    }
}
class Oracle_DB extends Base_DB{
    function update($data,$id)
    {
        // کدهای مربوط به ویرایش
    }
    function insert($data)
    {
        // کدهای مربوط به درج کردن
    }
}
?>
```

با کمک کلاس های MySQL_DB و Oracle_DB که در بالا تعریف شد می توان با تابع save و یا هر تابع دیگری کارهای مربوطه را انجام داد؛ توجه داشته باشید که توابع update و insert که در کلاس Base_DB به صورت abstract تعریف شده اند باید در کلاسهایی که از آن سرچشمه می گیرند تعریف شده باشند.

interface

کلاسهایی که به صورت interface تعریف می شوند تقریباً همانند کلاس هایی هستند که به صورت abstract تعریف می شوند اما نحوه تعریف و استفاده آنها متفاوت می باشد.

مثال:

```
<?php
interface class Base_DB {
    public function update($data,$id);
    public function insert($data);
}

class MySQL_DB implements Base_DB{
    function update($data,$id)
    {
        // کدهای مربوط به ویرایش
    }
    function insert($data)
    {
        // کدهای مربوط به درج کردن
    }
}
class Oracle_DB implements Base_DB{
    function update($data,$id)
    {
        // کدهای مربوط به ویرایش
    }
    function insert($data)
    {
        // کدهای مربوط به درج کردن
    }
}
?>
```

توجه داشته باشید که کلاسهایی که به صورت interface تعریف می شوند نباید شامل بدنه یک تابع باشند؛ در صورتی که کلاس های abstract می توانستند ادغامی از توابع معمولی و توابع abstract شوند.

به این نکته نیز توجه داشته باشید که کلاسهای interface بجای extends با implements توسعه می یابند.

تشخیص کلاس یک شئ

در برنامه نویسی توجه به این نکته که آیا شئ مورد نظر از کلاس مد نظر ما ساخته شده است یا نه نکته ای است که کاربرد های زیادی دارد.

برای این منظور می توان از دستور instanceof استفاده کرد؛

instanceof همچنین برای کلاس هایی که از یک کلاس دیگر ارث بری کرده اند نیز استفاده می شود؛

مثال:

```
<?php
class Parent
{
    // ...
}

class PLUS extends Parent
{
    // ...
}

$test = new PLUS();
if($test instanceof PLUS)
{
    echo "test comes from PLUS....<br />";
}
if($test instanceof Parent)
{
    echo "test comes from Parent....<br />";
}
?>
```

خروجی کد بالا:

```
test comes from PLUS....
test comes from Parent....
```

همانطور که در مثال بالا دیده میشود $test هم از نوع PLUS است و هم از نوع Parent!

فصل ۸: آموزش متفرقه PHP

آموزش ساخت خوراک مطالب سایت (RSS) با PHP و MYSQL

دوستان سلام،توی این مطلب قصد داریم یکی قسمت های مهم وب سایت های امروزی یعنی RSS رو با PHP و MYSQL آموزش بدیم،این آموزش خیلی ساده است و ما فقط قصد داریم یه ایده برای کد نویسی این قسمت از سایتون بهتون بدیم.

خوب اول از همه جدول اخبار یا مطالب سایت مون رو ایجاد میکنیم(فرض ما اینه که شما این جدول رو ندارید و گرنه باید از همون جدول اصلی مطالب یا اخبار خودتون استفاده کنید).

کد SQL

```
CREATE TABLE IF NOT EXISTS `news` (
`id` int(11) NOT NULL AUTO_INCREMENT,
`title` text COLLATE utf8_bin NOT NULL,
`sm` text COLLATE utf8_bin NOT NULL,
`body` text COLLATE utf8_bin NOT NULL,
`author` varchar(250) COLLATE utf8_bin NOT NULL,
`date` datetime NOT NULL,
PRIMARY KEY (`id`)
)
```

خوب جدول که توضیح خاصی نداره،ستون عنوان،خلاصه،متن،نویسنده و تاریخ خبر رو توش درج میکنیم(البته فرض ما اینجا اینه که این جدول شما مطلب داره و ما قرار نیست اینجا توش مطلبی ثبت کنیم).

ایجاد صفحه RSS

خوب کد های لازم برای ایجاد صفحه خوراک مطالب یا همون RSS رو به این صورت می نویسیم.

```
require_once 'config/database.php';
header("Content-type:text/xml;charset=utf-8");
global $db;
$sql="SELECT * FROM news ORDER BY id DESC LIMIT 10";
$db->connect();
$result=$db->query($sql);

function removeillegalchar($string){
$illegal_chars=array("&","\"","'","<",">");
$sanitize_chars=array("&amp;","&quot;","&apos;","&lt;","&gt;");
return str_replace($illegal_chars,$sanitize_chars,$string);
}
```

خوب ما مثل همیشه از یه کلاس که خودمون نوشتیم و تو اسکریپت های قبلی هم ازش استفاده کردیم،اینجا هم برای پردازش های بانک اطلاعات استفاده میکنیم(فایلش توی فایل های پروژه موجوده)،اول از همه اون کلاس رو به صفحه اضافه

کردیم،یه هدر قرار دادیم تا نوع محتوا رو XML مشخص کنیم (RSS بر پایه XML کار میکنه)،بعدشم یه کوئری نوشتیم که اطلاعات لازم رو واکشی کنیم،بعدشم اتصال به دیتابیس و ذخیره اطلاعات دریافتی در یک متغییر،ما بعد از این یه تابع تعریف کردیم که کار این تابع اینه که کاراکتر های غیر مجاز توی سند XML رو به کاراکتر های مجاز اون تبدیل میکنه،حالا خودتون نحوه استفاده شو می بینید.

```
<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
<channel>
<title>مرجع تخصصی طراحی وب</title>
<link>http://www.iranmabna.com</link>
<description>
مرجع تخصصی طراحی وب -ایران مبنا
</description>
<?php while($row=$db->fetch_array($result)):?>
<item>
<title><?php echo removeillegalchar($row['title']) ?></title>
<link>http://www.yourdomain.com/news.php?id=<?php echo $row['id']; ?></link>
<description><?php echo removeillegalchar($row['sm']); ?></description>
</item>
<?php endwhile; ?>

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

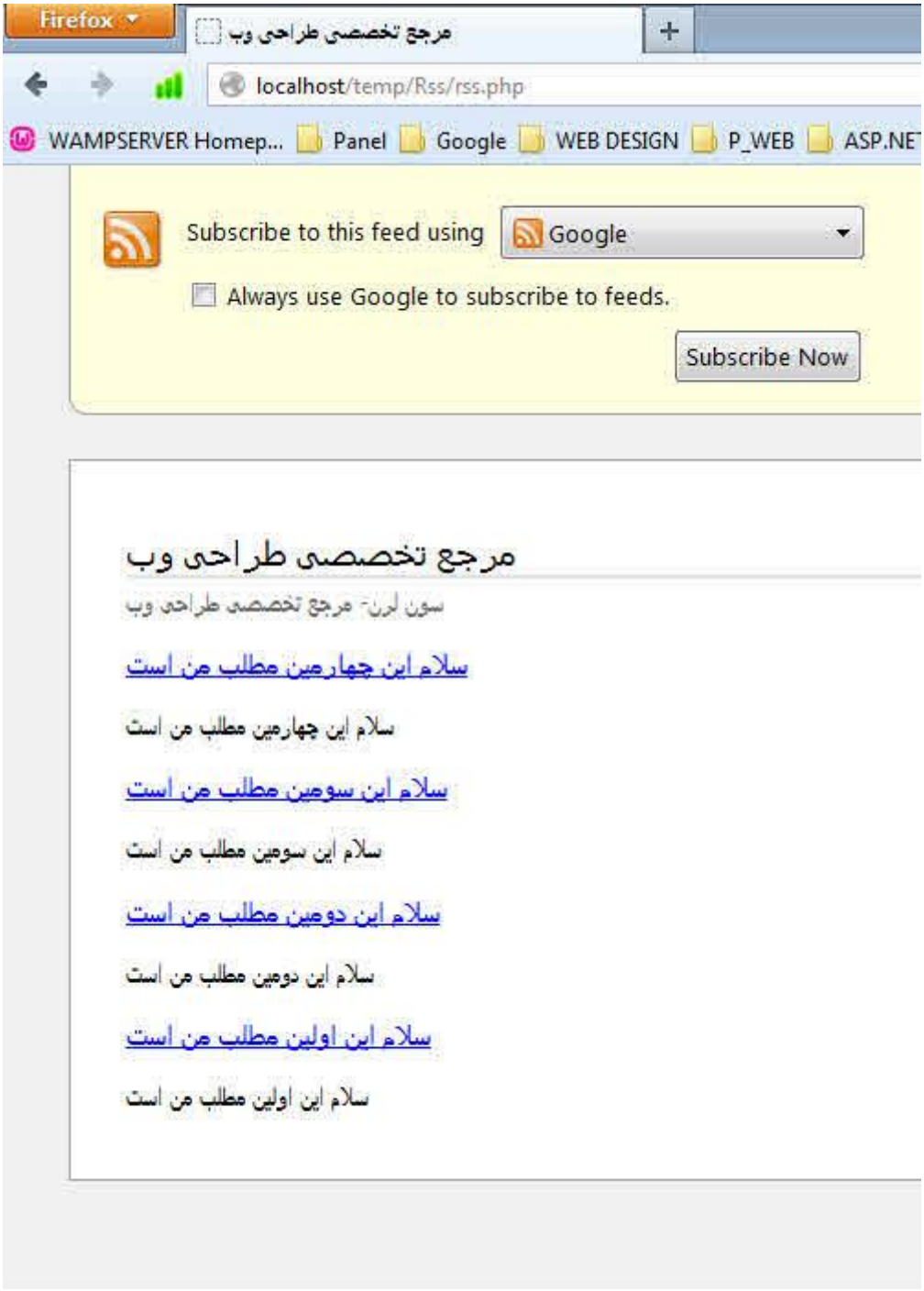
خوب اینم از کد های اصلی تولید محتوای خوراک،اولش که سند XML و RSS رو تعریف کردیم،توی XML هم شبیه HTML هر گره باید آغاز و پایانش مشخص باشه،با گره channel ما بدنه اصلی رو تعریف کردیم برای قسمت های عنوان و توضیحات از عنصر های title,link,description استفاده کردیم که توضیحات آدرس و عنوان وب سایت خودتون رو باید داخلش قرار بدید،خوب حالا نوبت به قسمت مطالب میرسه،اینجا باید از عنصر item استفاده کنیم و داخل خود عنصر item هم سه عنصر عنوان،لینک و خلاصه مطلب قرار میگیره،ما از یه حلقه استفاده کردیم و مطالبی که تو قسمت قبلی واکشی کردیم رو این جا به نمایش در میاریم.توی قسمت title عنوان مطلب، توی قسمت link هم لینک مطلب رو قرار دادیم (شما باید آدرس صفحه نمایش مطلب خودتون رو به همراه اطلاعات لازم بنویسید) و توی قسمت description هم خلاصه مطلب رو می نویسیم،می بینید که از اون تابع که کاراکتر های غیر مجاز رو حذف میکرد اینجا استفاده کردیم،خوب آخر کار هم ما اتصال به دیتابیس رو قطع کردیم.

کد های کامل برای صفحه RSS

```
<?php
require_once 'config/database.php';
header("Content-type:text/xml;charset=utf-8");
global $db;
$sql="SELECT * FROM news ORDER BY id DESC LIMIT 10";
$db->connect();
$result=$db->query($sql);
```

```
function removeillegalchar($string){
$illegal_chars=array("&","\"","'","<",">");
$sanitize_chars=array("&amp;","&quot;","&apos;","&lt;","&gt;");
return str_replace($illegal_chars,$sanitize_chars,$string);
}
?>
<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
<channel>
<title>مرجع تخصصی طراحی وب</title>
<link>http://www.iranmabna.com</link>
<description>
مرجع تخصصی طراحی وب -ایران مبنا
</description>
<?php while($row=$db->fetch_array($result)):?>
<item>
<title><?php echo removeillegalchar($row['title']) ?></title>
<link>http://www.yourdomain.com/news.php?id=<?php echo $row['id']; ?></link>
<description><?php echo removeillegalchar($row['sm']); ?></description>
</item>
<?php endwhile; ?>

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

خروجی کار شما توی FF تقریبا باید چیزی شبیه به این باشه.



اجرای PHP در IIS

بهترین راهکار برای اجرای PHP در IIS ویندوز

تا قبل از آشنایی با امکانات **IIS 7** به نظر می رسید که بهترین جا برای میزبانی یک وب سایت PHP فقط و فقط یک سرور Linux می تواند باشد، چون IIS 6 خیلی خوب PHP را اجرا نمی کند و خبری از URL rewriting هم در IIS 6 نیست، اما تغییر زاویه حرکت مایکروسافت در جهتِ گرفتن بیشتر سهم بازار باعث شده امکانات جالبی به IIS7 اضافه بشود و امروز می شود گفت ویندوز به خوبی لینوکس می تواند یک وب سایت PHP را میزبانی کند.

سه دلیل اصلی که ویندوز می تواند میزبان خوبی برای PHP باشد:

- اولین و بهترین دلیل این است که، IIS7 خیلی خوب URL rewriting را پشتیبانی می کند، شما به راحتی می توانید از امکان Pretty URLs سایت وردپرسی خودتان در سرور های ویندوز که IIS7 دارن استفاده کنید.

- دلیل دوم هم اینه که مایکروسافت وقت زیادی را صرف کرده تا عملکرد Fast CGI را تا حد ممکن در IIS 7 بهبود بدهد، و کاملاً مشکل Performance پایین PHP در ویندوز حل شده.

- دلیل سوم Module اختصاصی مایکروسافت برای PHP هست که کار Output Caching صفحات PHP رو بسیار آسان می کند و در نتیجه باز هم Performance بهتر...

و حالا یک نکته مهم: تمام Performance خوبی که تا اینجا ازش صحبت شد در گرو خوب Config کردن IIS 7 برای اجرای PHP هست.

در گذشته ? راه برای اجرای PHP در IIS وجود داشت، یعنی PHP را می شد در دو Mode مختلف روی IIS ویندوز اجرا کرد:

۱. **CGI Application**
۲. **ISAPI Managed Handler**

اما هر کدام از این دو راه حل، خوبی ها و مشکلات فراوانی را به همراه داشت.

- خوبی **CGI** این است که PHP رو خوب اجرا می کند و کمتر Crash اتفاق می افتد، اما از طرفی خیلی مصرف CPU بالایی دارد چون CGI برای اجرای هر کار کوچکی یک Process جدید ایجاد می کند و سیستم باید برای اجرای یک پردازش کوچیک صفحه PHP بهای راه اندازی و بستن یک Process رو پرداخت کند.
سرورهای شلوغ هرگز نمی تونن از CGI برای اجرای PHP استفاده کنند. چون هم Ram به زودی پر می شود و هم CPU از پردازش باز می ماند و نهایتاً کار به Crash کردن ویندوز می کشه.

- اما **ISAPI** این مشکل رو حل کرده، PHP در ISAPI Mode به روشی اجرا میشه که ASP و ASP.Net اجرا میشوند، به این صورت که همه صفحات توسط یک Process پرازش میشوند، اما چطوری؟ IIS توسط سرویس word wide web ویندوز همه پردازش ها را انجام می دهد. و برای اینکه بفهمد فایل های مثلا با پسوند php را چه جوری اجرا کند، از یک فایل dll کمک می گیرد. IIS برای اجرای php یا aspx یا asp از dll های مخصوص همون زبان، که بهش **Managed Handler** میگن، کمک می گیره و با سرویس world wide web ویندوز اجرا می شود.
ISAPI مشکل Performance اجرای PHP رو حل کرده اما مشکلات دیگه ای با خودش همراه داره. از جمله اینکه PHP در حالت ISAPI گاهی Crash می کنه و یه جورایی سایت هایی که با این روش کار می کند اعصاب صاحباشون رو خورد می کنن.

ورود Fast CGI و پایان مشکلات

مشکلات PHP در CGI و ISAPI موجب به وجود آمدن **Fast CGI** شد. از اسمش هم معلومه که با چه هدفی ایجاد شده.

هدف Fast CGI کنار هم آوردن **سرعت** ISAPI و **کارایی** CGI هست. در واقع روش کار Fast CGI مثل روش کار CGI هست. اما با تفاوتی کوچک اما کلیدی که سرعت بالایی بهش میده.

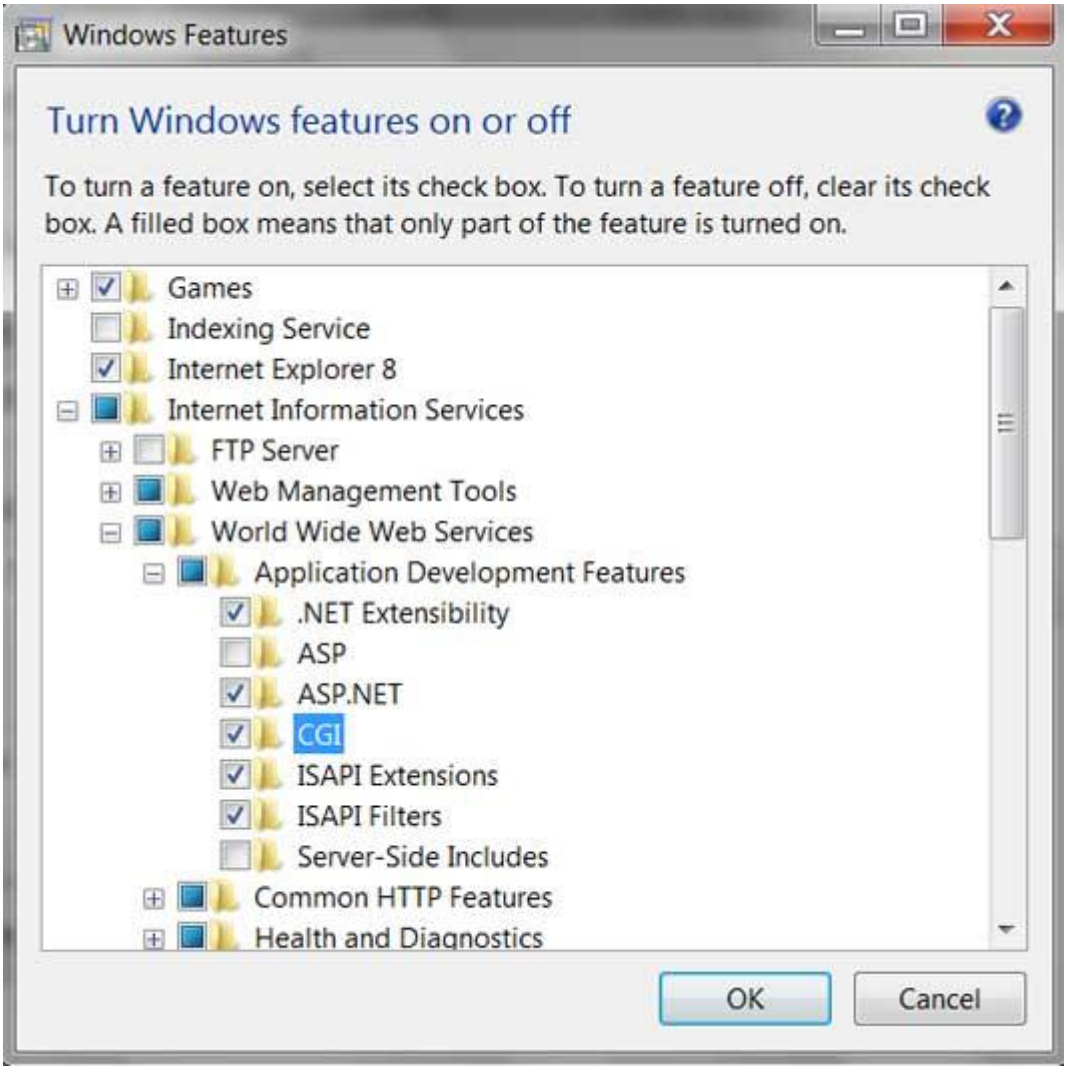
Fast CGI برای انجام هر پردازشی یک Process جدید ایجاد نمی کنه. بلکه **یک Process بعد از ایجاد باز می مونه و تعدادی پردازش رو انجام میده و بعد بسته میشه.** همین تغییر کوچک موجب پایین اومدن باور نکردنی مصرف منابع سیستم توسط PHP میشه.

با این حال Config کردن Fast CGI کمی سخت تر از CGI و ISAPI هست. اما بعد از اون مثل بنز کار می کنه.

خوب پس بهترین راه کار برای اجرای PHP در IIS مشخص شد:

۱. قبل از همه مطمئن شوید که CGI روی IISتون نصب هست، این رو می‌تونید از اینجا چك کنید:

Control Panel -> Programs and Features -> Turn Windows features on or off



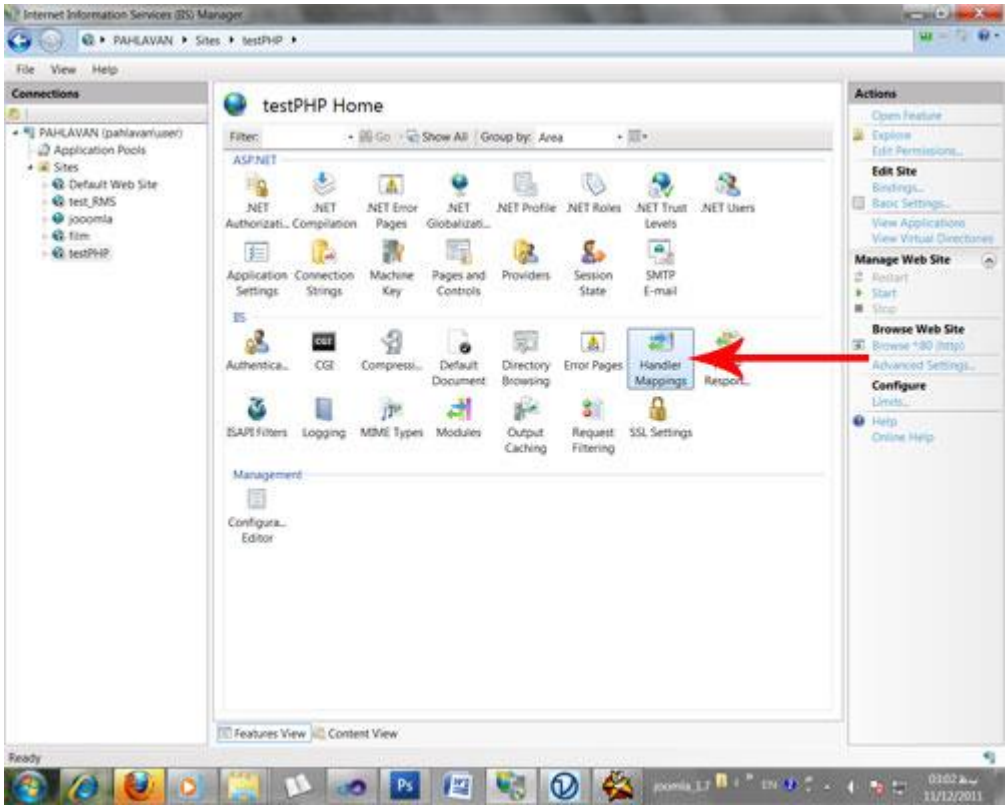
۲. نسخه غیرنصبی PHP رو دانلود کنید و در مسیر دلخواه (مثلاً C:\PHP) کپی کنید.

۳. فایل php.ini-recommended رو به php.ini تغییر نام بدید و این تغییرات رو در اون ایجاد کنید:

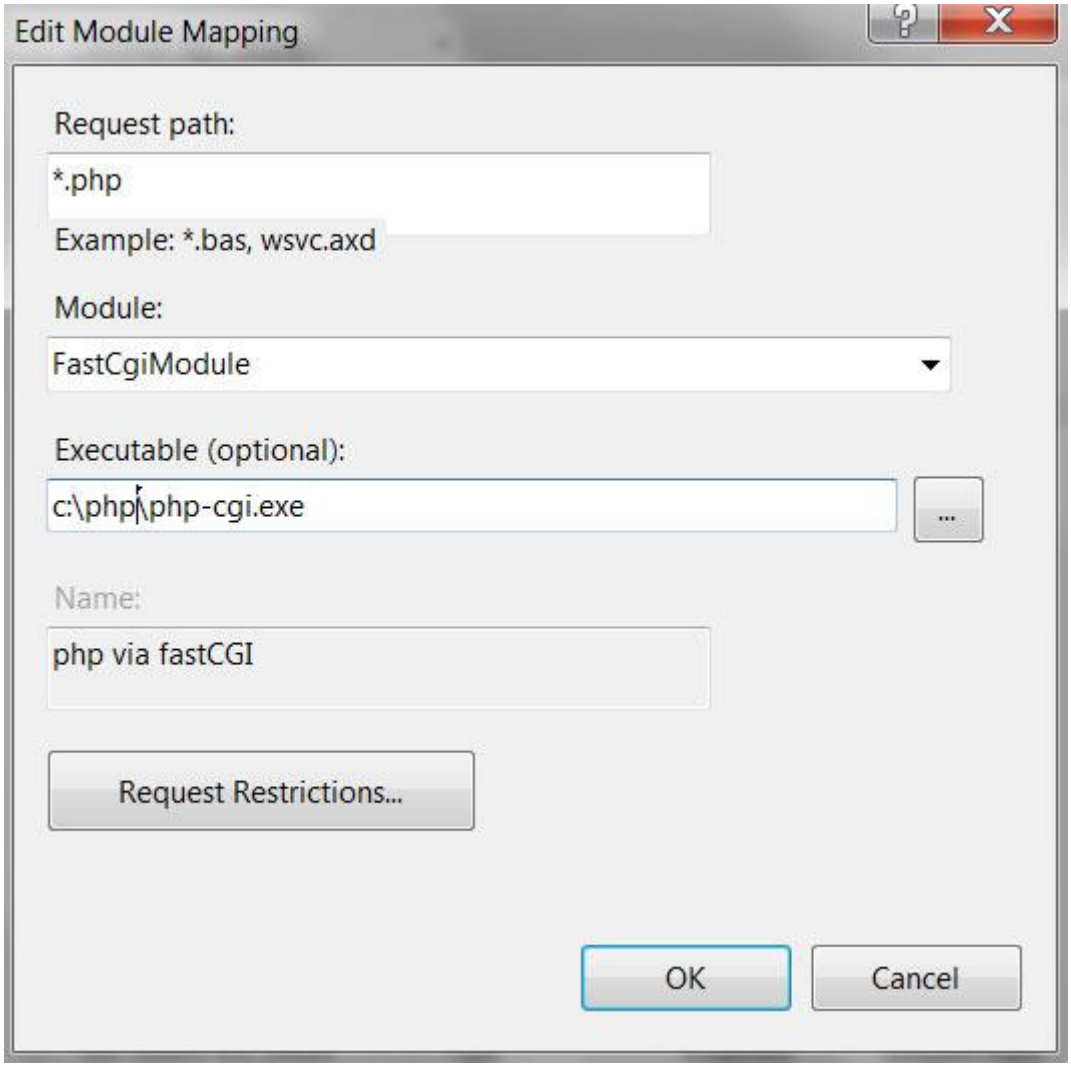
- کاراکتر ; رو از کنار خط fastcgi.impersonate = 1 بردارید. (اگر مقدارش یك نیست اون رو به یك تغییر بدید).
- برای خط cgi.fix_pathinfo = 1 هم همون کار رو انجام بدید.
- این بار خط cgi.force_redirect=0 (حتماً مقدارش صفر باشد).
- مقدار خط open_basedir رو برابر با مسیر c:\inetpub\wwwroot قرار بدید.

حالا این فایل رو Save کنید و برای تست صحت نصب در اعلان داس (Command Prompt) دستور php -info رو اجرا کنید، اگه اطلاعاتی رو در مورد php نشون داد، یعنی کار تا اینجا خوب پیش رفته.

۴. نوبت به تنظیمات در IIS می‌رسه. ابتدا IIS Manager رو اجرا کنید. سرور اصلی رو از لیست Connections سمت چپ انتخاب کنید و از قسمت میانی گزینه Handler Mapping رو دوبار کلیك کنید. این هم عکس:



از ستون سمت راست گزینه Add Module Mapping رو انتخاب کنید و اون رو مثل عکس زیر پر کنید و OK که یه پنجره دیگه بعدش باز می‌شه اون رو هم Yes کنید:



۶. کار تمومه اما برای تست تو پوشه c:\inetpub\wwwroot یه فایل به نام phpinfo.php بسازید با این محتوا:

```
<?php
echo 'Hello';
?>
```

و اون را با این آدرس باز کنید: http://localhost/phpinfo.php

اجرای ایمن اسکریپت های PHP

استفاده از مخفی کننده اسکریپت

برای اتصال به یک پایگاه داده نیاز است که رمزهای عبور آن در برخی قسمتهای اسکریپت وجود داشته باشد اسکریپت های دیگر شما ممکن است اطلاعات حساس خود را داشته باشند این اطلاعات اگر شما اسکریپت های PHP خود را پنهان نکنید، ممکن است افشا شوند . حفاظت کردن اسکریپتهایتان از چشمهای کنجکاو wrapping گفته می شود هنگامی که شما یک اسکریپت را می پوشانید توسط اسکریپت دیگری که برنامه اصلی شما را پنهان کرده و آزمایشهای ایمنی را برای حصول اطمینان از دستیابی مجاز انجام میدهد فراخوانی می گردد.

دیگر کاربران رمز عبورها را چگونه می بینند؟

دو راه وجود دارد؟

۱- اگر شما به اندازه کافی دقت به خرج ندهید و رمز عبورهای خود را در ریشه اصلی وب تان و در درون فایلی که توسط مفسر PHP تجزیه و تحلیل نمی شود قرار دهید (مانند http://www.mysite.com.pass.txt) هر کسی می تواند آنرا دریافت کند و محتویاتش را به دست آورد.

۲- اگر شما رمز عبورهای خود را در فایلی قرار دهید که می تواند توسط وب سرور به طور مستقیم یا بوسیله دیگر کاربرانی که در گروه شما و در همان سرور هستند خوانده شود بکارگیری این روش بجز توسط افرادی که می توانند یک وب سرور اختصاصی داشته باشند توصیه نمی شود.

اگر شما رمز عبورها را در یک فایل که توسط دیگر کاربران قابل خواندن است قرار دهید آنها میتوانید بوسیله FTP و یا Telnet به شاخه شما رفته و فایلها را مستقیماً بخوانند. برخی میزبانهای وب از این رفتار توسط chrooting جلوگیری می کنند.(Usr/home/username راهنگامی که شما وارد می شوید برابر با می کند Chroot دستور UNIX برای تغییر ریشه است) اما بسیاری از میزبانها در مورد فایلهایی که صرفاً اجازه خواندن آن به وب سرور داده شده است اجازه دسترسی را به دیگر کاربران نمی دهند. در این حالت نیز دیگران می توانند اسکریپتی بنویسند که با مشخصه کاربری nobady کار کند و این اسکریپت می تواند به تمام فایلهایی که در سرور وجود دارد وصرفاً توسط آن خوانده می شود دسترسی پیدا کند.

Wrapping چگونه از این امر جلوگیری می کند؟

Wrapping نام کاربری کسی که به فایل می تواند دسترسی داشته باشد را برابر نام کاربری شما قرار می دهد. یک مخفی کننده ایمن، چندین بررسی و آزمون امنیتی بر روی درخواستها، قبل از اجرای آنها انجام می دهد.

Php- cgiwrap چیست؟

یک اسکریپت مخفی کننده برای PHP است. Cgiwrap یک اسکریپت مخفی کننده بر ای برنامه های Prel است و Php- cgiwrap نسخه مربوطه به PHP انست که توسط Pair Netwoks تهیه شده و صرفاً بر روی سرورهای آنان کار می کند. اگر از میزبان دیگری استفاده می کنید روشهای دیگری برای مخفی کردن وجود دارد که می توانید از آنها استفاده کنید.

پوشش شفاف

آخرین بحث ما درباره php مربوط به پوشش شفاف برای ایمنی صفحات است. برخی از میزبانان وب پوشش شفاف لاینفکی برای صفحات PHP ارائه کرده اند. اگر صفحات از پسوند php3. استفاده نمایند و در این صورت شما اصلا متوجه نخواهید شد که صفحات شما از پوشش استفاده می کنند اما نکته مهم آنست که از فراهم کننده خدمات میزبانی وب خود در این باره پرسش می نماید.

راههای مختلفی برای اجرای این کار وجود دارد. شما می توانید از mod- rewrite استفاده کنید تا به طور خودکار تمام درخواستهای PHP از طریق اسکریپت مخفی کننده ارسال شود. یا از خدمات سرور Apache suExec استفاده نمایید. برای اطلاعات بیشتر از نشانی http:// www. Apache. Org/ docs/ suexec. Htm استفاده کنید.

اخطار: اگر میزبان وب شما راهی برای اجرای ایمن اسکریپت هایتان پیش بینی نکرده، و راهی برای مخفی کردن رمز عبورها از دید کاربران کنجکاو ندارید. بایستی به میزبانی مراجعه کنید که بیشتر نسبت به امنیت کاربران اهمیت قایل است.

بخش ۹: MVC در PHP

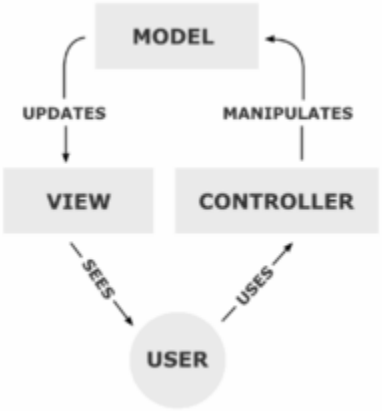
MVC در PHP - معرفی

اولین سوالی که پیش میاد اینه که MVC چیست و چه کاربردی داره؟

MVC یک معماری سه لایه است که در سال ۱۹۷۰ ایجاد شد.این معماری ابتدا برای زبان smalltalk ایجاد شد. هدف از ایجاد این معماری این بود که قسمت کد نویسی یا منطق برنامه رو از قسمت طراحی اون جدا کنن. مزایایی که این کار در بر داشت این بود که نگهداری و تغییر در کدها در آینده بسیار راحت بود ، کدهایی که نوشته میشدن قابل استفاده مجدد بودن و مهمترین قابلیت این بود که یک طراح و یک برنامه نویس میتونستن روی یک پروژه در آن واحد کار کنن بدون اینکه اختلالی توی کار هم به وجود بیارن.

با فراگیر تر شدن وب برنامه نویسان وب تصمیم به استفاده از این معماری در طراحی و برنامه نویسی وب کردند.

معماری سه لایه از سه قسمت Model , View , Controller تشکیل میشه. که در زیر به شرح مختصری در مورد هر کدوم از لایه ها میپردازیم.



لایه Model

" این لایه برای ارتباط با دیتابیس استفاده میشه. این لایه باید اجازه دسترسی ، تغییر یا اضافه کردن داده ها رو بده . این لایه در واقع یک پل بین لایه View و لایه Controller ـه. یکی از مهمترین خاصیت های این لایه اینه که " نابیناست " به این معنی که مدل نمیدونه وقتی که داد هارو به View یا کنترلر ارسال کرد قرار چه اتفاقی براشون بیافته و به دنبال پاسخی از

Controller یا View نیست.تنها هدفش اینه که داده هارو ذخیره کنه یا زمانی که درخواستی از بقیه لایه ها ارسال شد تغییرشون بده. " (منبع)

در خیلی از جاها گفته شده که لایه مدل نباید با لایه ویو ارتباط داشته و فقط و فقط باید با لایه کنترلر در ارتباط باشه.ولی چیزی که در مورد این لایه مهمه اینه که این لایه مسئول ارتباط با دیتابیسه . حذف ، اضافه و ویرایش داده های دیتابیس توی این لایه انجام میشه.

لایه View

این لایه جاییه که داده ها از مدل گرفته میشن و به صورت خروجی به کاربر نمایش داده میشن.در برنامه های وب این لایه جاییه که کدهای HTML ساخته و نمایش داده میشن.توی این لایه ارتباط با کاربر انجام میشه و با کنترلر درخواست کاربر رو انجام میده.به عنوان مثال یک button رو درنظر بگیرید که وقتی روش کلیک شد یکی از action های کنترلر رو صدا میزنه.

یک سری تصورات غلط در مورد این لایه در برنامه نویسان وب وجود داره . یکی از این تصورات اشتباه اینه که لایه View نباید هیچ ارتباطی با لایه Model داشته باشه و باید همه اطلاعات رو از لایه Controller بگیره.مثلا در فریم ورک Cakephp و اکثر فریم ورکهای PHP این اشتباه وجود داره. در واقع این تفکر نادیده گرفتن کامل تئوریه پشت معماری سه لایه است.

قسمت بالا از Callum Hopkins بود که در سایت sitepoint مقاله معماری سه لایه رو نوشته بود. دراین مقاله ذکر شده که لایه مدل و ویو میتونن باهم ارتباط داشته باشن و درواقع لایه Model یک پل بین لایه ویو و کنترلره . در همین سایت و در این مقاله ذکر شده که :

این نکته خیلی مهمه که توجه داشته باشید برای پیاده سازی درست معماری MVC لایه View نباید با لایه Model ارتباط داشته باشه و منطق برنامه باید فقط در لایه Controller انجام بشه.

از اونجا که بین علما اختلاف نظر وجود داره تحقیق بشتری کردم و در این مقاله به نقل از گروه Gang of four در مورد معماری MVC اینطور نوشته :

MVC شامل مدل ها و ویو هاییه که میتونن با هم ارتباط داشته باشن. یک View باید مطمئن باشه که ظاهری که الان میخواد به خودش بگیره باید حالتی از model باشه.هر وقت که دیتا در لایه model تغییر کرد یک پیغام به ویو میفرسته که بسته به اون تغییر کنه.

بازم چیزی که در مورد این لایه مهمه اینه که این لایه وظیفه نمایش داده ها و گرفتن و ارسال اون رو به لایه Controller داره.ما هم زیاد سخت نمیگیریم و هر طوری که راحتیم فریم ورک خودمون رو مینویسم چون قرار نیست کسی به خاطر اینکه آیا این دولایه باید باهم ارتباط داشته باشن یا نه ، مارو مجازات کنه.

لایه Controller

این لایه منطق برنامه رو کنترل میکنه.این لایه ورودی ها رو میگیره و درخواست های کاربر رو انجام میده ، اگر نیاز باشه از دیتابیس مقادیری برای کاربر ارسال بشه رو از لایه Model میگیره و به لایه View ارسال میکنه.

به هر حال در باره کارکرد این سه لایه باهم نقل قول های زیادی شده که هر کی هرجور دلش خواسته باهاش بخورد کرده.شما با هر کدوم راحت ترید کار کنید ولی اینو بدونید که اصل موضوع چیه.

خب این سه لایه رو به شکل مختصر توضیح دادیم بریم ببنیم این معماری چطوری کار میکنه.

ایده این معماری بسیار ساده است.شما یک فایل اصلی دارین مثلا index.php که تمام درخواست های کاربر به اون فرستاده میشه.بعد از اینکه درخواست رو ازکاربر دریافت کرد کلاس ها و توابع مورد نظر رو اجرا میکنه و View مورد نظر رو به کاربر نمایش میده.

برای درخواست زیر :

```
index.php?page=user&action=delete&param=2,3,5
```

مثلا این کدها اجرا میشن :

```
if(isset($_GET['page'])){
		$page = $_GET['page'];
}else
{
		$page = 'index';
}

include "lib/$page.php";
if(isset($_GET['action'])){
		$action = $_GET['action'];
}else{
		$action = 'default';
}

if(isset($_GET['param'])){
```

```
        $param = explode(',',$_GET['param']);
}else{
        $param = array();
}
$obj = new $page();
$obj->$action($param);
```

کدهای بالا مستقیم در فایل index.php اجرا میشن. اگه میبینید سرو ته نداره نگران نباشید این فقط یک مثال بود.

توی این مقاله سعی میکنم تا با معماری سه لایه که در فریم ورکهای PHP مثل : Laravel , Yii , cakePHP , Zend و ... پیاده سازی شده توضیح بدم و در کنارش سعی میکنیم باهم یک فریم ورک ساده رو پیاده سازی کنیم تا با نحوه کار این فریم ورکها آشنا بشیم و اگر دوست داشتید بعدا از یکیشون استفاده کنید !!

به امید خدا توی قسمتهای بعدی در مورد پیاده سازی یک فریم ورک بحث میکنیم

MVC در Router - PHP

یکی از مهمترین قسمتهای پیاده سازی MVC در وب مسیریابی یا Routing میباشد.این یعنی اینکه با استفاده از درخواستی که کاربر داره بتونیم کنترلر ، مدل و ویو مورد نظر رو لود و اجرا کنیم.

درخواست کاربر به وسیله URL ارسال میشه.به عنوان مثال اگر کاربر روی لینک زیر کلیک کنه :

```
www.domail.com/article/view/2-mvc-in-php
```

در این معماری باید به این شکل باهاش بخورد کنیم :

article نام کنترلر

view نام اکشن یا متدی که در کنترلر article قرار داره

و قسمت بعدی پارامترهایی که به متد view ارسال میشه رو مشخص میکنه.

برای مسیریابی یا routing در فریم ورک ها کلاس با نام Routing وجود داره که این کار رو انجام میده.مسیریابی در فریم ورکهای مختلف به روشهای متفاوتی پیاده سازی شده.یکی از مسیریابی دستی استفاده کرده یکی از مسیریابی اتوماتیک و بیشتر فریم ورکها هردو امکان رو به کاربر میدن

ما برای فریم ورکمون از مسیریابی اتوماتیک! استفاده میکنیم. برای پیاده سازی مسیریابی باید در فایل htaccess یه سری خطوط رو اضافه کنیم :

```
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
 RewriteRule ^(.*)$ index.php?url=$1 [PT,L]
 </IfModule>
```

در کد بالا همه درخواستها به فایل index.php ارسال میشن و پارامترهای ارسالی به این فایل در متغیر url ذخیره میشن.

(برای اطلاعات بیشتر میتونید این مقاله رو بخونید)

یک فایل با نام index.php بسازید و کدهای زیر رو داخلش بنویسید :

```
<?php

define('DS', DIRECTORY_SEPARATOR);
define('ROOT', dirname(__FILE__));

error_reporting(E_ALL - E_NOTICE);

$url = $_GET['url'];

require_once (ROOT . DS . 'library' . DS . 'Router.php');

Router::route($url);
```

یک ثابت برای نگهداری " جداکننده پوشه ها " به این خاطر که در لینوکس و ویندوز این کاراکتر فرق داره. در لینوکس از اسلش (/) و در ویندوز از بک اسلش (\ ) استفاده میشه.

ثابت ROOT برای نگهداری مسیر اصلی فریم ورک

یک فایل در پوشه library با نام Router.php بسازید و محتویات زیر رو داخلش قرار بدید :

```
<?php


class Router {
    static function route($url){


        $urlparts = @explode('/',$url);


        $controller = ($urlparts[0] == '' ) ? 'index' : $urlparts[0];
        array_shift($urlparts);
        $action = ($urlparts[0] == '' ) ? 'index' : $urlparts[0];
        array_shift($urlparts);
        $param = $urlparts;


        if(file_exists($cFile = ROOT.'/app/controller/'.$controller.'.php')){
           include  $cFile;
        }else{
            die ('Controller ' . $controller .' Not found');
        }
        $controller = ucwords($controller).'Controller';
        $controllerObj = new $controller();


        if(method_exists($controllerObj, $action)){
            call_user_func_array(array($controllerObj,$action), $param);
        }else{
            die("Action $action not found in $controller Class");
        }


    }
```

```
}
```

این کلاس برای مسیر یابی به کار میره.یک متد با نام route که یک پارامتر میگیره.پارامتر ارسالی همون url ی هست که کاربر درخواست کرده.

ابتدا url رو به وسیله تابع explode به آرایه تبدیل میکنیم.

قسمت اولش میشه نام کنترلرمون . قسمت دومش میشه نام تابع یا متدی که کاربر درخواست کرده. اگر هرکدوم وجود نداشته باشن به صورت پیش فرض کنترلر Index و متد Index اجرا میشه.

بقیه قسمتها هم به عنوان پارامتر در نظر گرفته میشن.

بعدش کلاس کنترلر مورد نظر رو از پوشه app/controller/ لود میکنیم. ازش یک شی میسازیم و با تابع call_user_func_array متد مورد نظر رو با پارامترهایی که داریم اجرا میکنیم.

دقت کنید که نام کلاس کنترلر باید به این صورت باشه :

IndexController

و نام فایلش باید اینجوری باشه :

index.php

خی حالا تغریبا قسمت اول کارمون تمومه و برای تست یک پوشه با نام app بسازید و توی این پوشه یکی دیگه با نام controller بسازید.

یک فایل با نام article.php ایجاد کنید و محتویات زیر رو داخلش بنویسید :

```
<?php

class ArticleController{

    function index(){
        echo 'IndexController -> index action';
    }
```

```
    function view($id){
        echo "ID =  $id";
    }
}
```

حالا آدرس زیر رو باز کنید :

```
۱۲۷,۰,۰,۱/article/view/32
```

خب اولین قسمت کارمون تموم شد در پستهای بعدی لایه مدل و ویو رو هم باهم میسازیم

MVC در PHP – Model

در این قسمت از سری مقالات MVC در PHP به لایه Model میپردازیم.همینطور که در قسمت اول اشاره شد لایه Model وظیفه ارتباط با پایگاه داده رو داره. در معماری MVC و کلا سیستم هایی که به صورت شیء گرا نوشته میشن معمولا برای هر کدوم از جدولهای دیتابیس یک کلاس جدا باید داشته باشیم.

این به این خاطره که کار نگهداری و گسترش در آینده به راحتی انجام بشه .حالا ما هم در این فریم ورک از این روش استفاده میکنیم.

لایه Model (در فریم ورک ما) برای ارتباط با دیتابیس باید قابلیت های زیر رو داشته باشه :

- ارتباط با پایگاه داده
- انجام عملیات اضافه / حذف و ویرایش روی جداول
- اجرای کوئری های کاربر

برای داشتن یک فریم ورک خوب و کاربردی باید یک سری پیش نیازها رو قبل از پیاده سازی لایه Model انجام بدیم.پس قبل از پیاده سازی لایه مدل ما به کلاس های زیرنیاز داریم :

کلاس Config برای نگهداری تنظیمات

کلاس SqlQuery برای ارتباط با پایگاه داده و اجرای عملیات رو جدول مورد نظر

خوب با هم اولین قسمت رو که نوشتن کلاس Config هست انجام میدیم،این کلاس وظیفه نگهداری تنظیمات کلی فریم ورک رو بر عهده داره.برای اینکه تنظیمات رو یه جایی نگهداری کنیم راههای مختلفی وجود داره.میتونه این تنظیمات به صورت یک کلاس باشه یا میتونه در یک فایل با پسوند ini ذخیره بشه. برای اینکار ما از یک فایل با نام config.ini و یک کلاس برای دسترسی به تنظیمات ذخیره شده توی این فایل نیاز داریم.

یک فایل با نام config.ini توی پوشه library بسازید با این محتویات :

```
[database]
driver = "Mysql"
dbName = "phpromvc"
dbUsername = "root"
dbPassword = ""
```

خب یک کلاس با نام Config.php در پوشه library بسازید :

```
<?php

class Config {

    static function get($key) {
        $config = parse_ini_file(LIB_DIR.DS. 'config.ini');
        return $config[$key];
    }

}
```

توی این کلاس با استفاده از تابع parse_ini_file محتویات فایل config.ini رو به صورت آرایه در اختیار داریم و سپس با متد Config::get میتونیم به تنظیماتمون دسترسی داشته باشیم.

شاید بپرسید چرا از یک فایل ini برای تنظیمات استفاده کردم.جوابم اینه که میخواستم توی این آموزش یک چیز جدید رو باهم تجربه کنیم.شما از هر روشی که دوست دارید میتونید استفاده کنید.

مشکلی که این روش داره اینه که اگر کاربر آدرس فایل config.ini رو مستقیم توی مرورگر وارد کنه به تنظیمات ما دستری پیدا میکنه که از لحاظ امنیتی مشکل داره.برای رفع این مشکل خط زیر رو به فایل htaccess اضافه میکنیم :

```
<Files *.ini>
    Order deny,allow
    Deny from all
</Files>
```

که دسترسی مستقیم به این فایل رو غیرممکن میکنه.

بریم به ادامه مطلب که این بار باید کلاسی بسازیم برای ارتباط و انجام عملیات روی دیتابیس.

برای اینکه فریم ورکمون انعطاف بیشتری داشته باشه باید کلاسی بنویسیم که بتونه با انواع دیتابیس ها ارتباط برقرار کنه.ما کلاس PDO رو انتخاب میکنیم که از هر لحاظ میتونه به ما کمک کنه :

```
<?php

class Model {

    private $_pdo ;
    private $_query = '';
    protected $table = '';
    protected $pk = '';

    function __construct() {

        $dns = Config::get("driver");
        $dbName = Config::get("dbName");
        $username = Config::get("dbUsername");
        $password = Config::get("dbPassword");

        if($this->table == ''){
```

```
            $this->table = get_class($this);
        }

        $this->_pdo= new PDO($dns . ':host=localhost;dbname=' . $dbName, $username, $password);
    }

    function get_rows($fields = '*', $where = ' 1=1 ', $order = 'ASC', $limit = 10) {
        $this->_query = "select $fields from {$this->table} where {$where} ORDER BY {$this->pk} $order LIMIT
$limit";
        $stm = $this->_pdo->query($this->_query);
        return $stm->fetchAll();
    }

    function get_row($fields = '*', $where = ' 1=1 ', $order = '', $limit = 10) {
        $this->_query = "select $fields from {$this->table} where {$where} ORDER BY {$this->pk} $order LIMIT
$limit";
        $stm = $this->_pdo->query($this->_query);
        return $stm->fetch();
    }

    function delete($id) {
        $this->_query = "delete from $this->table where id = '$id'";
        $this->_pdo->exec($this->_query);
    }

    function update($data, $where = ' 1 = 1 ') {
        $this->_query = " update {$this->table} set $data where $where";
        $this->_pdo->exec($this->_query);
    }
```

```
    function insert($fields, $data) {
        $this->_query = " insert into $this->table ($fields) VALUES ($data)";
        $this->_pdo->exec($this->_query);
    }

    function run($query) {
        $this->query = $query;
        $this->_pdo->exec($query);
    }

    function last_query(){
        return $this->_query;
    }
}
```

این کلاس خیلی ساده است که امکان اضافه / حذف / ویرایش و انجام کوئری های مورد نظرمون رو میده.

یک دیتابیس با یک جدول بسازید. اسم جدول رو بذارید "articles" و فیلدهای زیر رو بهش اضافه کنید :

- id
- title
- body

حالا چندتا ردیف با مقادیر دلخواه به این جدول اضافه کنید.

یک پوشه با نام model در پوشه ی app بسازید. یک کلاس در این پوشه با نام articles.php ایجاد کنید و کدهای زیررو داخلش قرار بدید :

```
<?php

class articles extends Model{
    protected $table = 'articles';
```

```
    protected $pk = 'id';
}
```

همینطور که قبلا هم اشاره کردم نام کلاس باید با نام جدول همنام باشن. کلاس articles از کلاس Model ارث بری کرده و به تمام متدهای public و protected اون دسترسی داره. متغیر $table نام جدول رو میگیره که اگر خالی باشه نام کلاس به عنوان نام جدول در نظر گرفته میشه و متغیر $pk نام کلید اصلی رو نگهداری میکنه.

قبل از اینکه کاری انجام بدید یک نکته دیگه رو هم باید اشاره کنم. در سیستم های بزرگ برای جلوگیری از include های پی در پی از تابع autoload استفاده میکنن.

با استفاده از کلاس Load ما دیگه نیازی به include کردن کلاسهامون نداریم.

کدهای زیر رو به فایل index.php اضافه کنید :

```
define("LIB_DIR" , ROOT.DS.'library');
define("APP_DIR" , ROOT.DS.'app');
include LIB_DIR.DS.'Load.php';
spl_autoload_register(array('Load', 'autoload'));
```

و کلاس Load در پوشه library :

```
<?php

class Load {

    static function autoload($class) {
            include LIB_DIR . DS . $class . '.php';
    }

    static function model($modelName) {
        if(file_exists($file = APP_DIR.DS.'model'.DS.$modelName.'.php')){
            include $file;
            return new  $modelName();
```

```
    }
  }
}
```

با استفاده از متد model میتونیم model مورد نظرمون رو load کنیم.

توی کنترلر ArticleController متد view رو اضافه کنید :

```
<?php

class ArticleController{

  function index(){
    echo 'IndexController -> index action';
  }

  function view($id){
    $article = Load::model("articles");
    $row = $article->get_row('*' , " id = $id ");
    print_r($row);
    echo $article->last_query();
  }
}
```

با رفتن به آدرس زیر میتونید نتیجه رو مشاهده کنید :

۱۲۷,۰,۰,۱/article/view/2

خب این قسمت هم به پایان رسید.امیدوارم که مفید بوده باشه.اگر نظر یا سوالی در مورد آموزشها دارن میتونن زیر همین مقاله نظر خودشون رو بگن.اگر هم مشکلی توی آموزشها میبینید میتونید بازهم همینجا بگید خوشحال میشم.

در قسمتهای بعدی به بررسی لایه های Controller و View میپردازیم.

MVC در PHP – View و Controller

با آخرین قسمت از قسمتهای MVC در PHP در خدمتتون هستم.در این قسمت به لایه View و Controller میپردازیم. همینطور که در قسمت اول گفتم لایه View وظیفه نمایش دادن خروجی و گرفتن ورودی هارو از کاربر داره.و لایه Controller به پردازش ورودی و خروجی ها میپردازه .میتونید فایلهای این پروژه رو از لینک زیر دریافت کنید :

https://github.com/PHProir/PHProMVC

لایه View

این لایه باید طوری طراحی بشه که بتونه اطلاعات رو از کاربر بگیره و به کنترلر بفرسته و در مقابل بتونه داده هایی که از سمت کنترلر میاد رو به کاربر نمایش بده.برای اینکار ما به یک کلاس نیاز داریم.یک کلاس در مسیر library بسازید و اسمش رو view بزارید :

```
<?php

class View {

    private $vars = array();

    function set($var , $data) {
        $this->vars[$var] = $data;
    }

    function render($view) {
        extract($this->vars);
        include APP_DIR.DS.'view'.DS.$view.'.php';
    }

}
```

این کلاس دوتا متد داره.یکی متد set که کارش اینه که متغیرهایی رو که میخواید بفرستید به view مورد نظر رو مدیریت میکنه.یکی متد render که نام view مورد نظر رو از کاربر میگیره و نمایش میده.

دقت کنید که این کلاس خیلی ساده است و برای داشتن یک ویو انعطاف پذیر باید بیشتر روش کار بشه.

حالا یک کلاس دیگه با نام Controller در مسیر library بسازید :

```
<?php

class Controller {

    protected $view;

    function __construct() {
        $this->view = new View();
    }

}
```

و کنترلر ArticleController رو از کلاس Controller ارث بری کنید و متد view رو به شکل زیر تغییر بدید :

```
<?php

class ArticleController extends Controller{

    function index(){
        echo 'IndexController -> index action';
    }

    function view($id){
        $article = Load::model("articles");
        $row = $article->get_row('*' , " pro_id = $id ");
        $this->view->set('data',$row);
        $this->view->render('view');
    }
```

```
}
```

تا اینجا ما اطلاعات رو از دیتابیس گرفتیم و به view ارسال کردیم.حالا باید یک فایل در مسیر app/view با نام view.php بسازید و به شکل زیر میتونید با مقادیر ارسالی کار کنید :

```
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title></title>
    </head>
    <body>
        <?php
            echo $data['id'] , ' - ';
            echo $data['body'] , '<br>';
            echo $data['title'] , '<br><hr>';
        ?>
    </body>
</html>
```

تا اینجای کار ما یک فریم ورک سه لایه داریم.که قسمتهای View و Model و Controller از هم جدا هستن. شیوه کار اکثر فریم ورکها بر همین مبناست اما هر چی بخواید فریم ورک کاربردی تری داشته باشن باید رنگ و لعاب بیشتری بهش بدید.مثلا در فریم ورکهای دیگه در کنار این سه قسمت قسمتهای اضافی مثل plugin و helper و ... داریم که به منعطف بودن و کاربردی بودن فریم ورکمون کمک میکنه.

مثلا در لایه ویو میتونید از موتور قالب های مختلف مثل smarty , twig و ... استفاده کنید. در کنار این باید فریم ورکی که مینویسید کتابخانه ی بزرگی داشته باشه که تقریبا همه نیازهای کاربر رو جواب بده و بتونه با کمترین کد نویسی از طرف کاربر بهترین نتیجه رو بگیره.

حرف آخر هم اینکه این دوره از آموزشهای MVC در PHP تموم شد.در این دوره با شیوه کارکرد معماری سه لایه آشنا شدیم و تونستیم در کنار هم یک فریم ورک بنویسیم.هدف از این دوره این بود که اگر از فریم ورکهایی مثل zend , cakephp ,

laravel , Yii و ... استفاده میکنید بفهمید که پشت این فریم ورکها چه اتفاقی داره میافته. حالا که دیدیم این فریم ورکها چطور کار میکنن شاید علاقه داشته باشید که با یکیشون شروع به کار کنید یا حتی شروع به نوشتن یک فریم ورک کنید.

چیزی که مهمه اینه که از یادگیری چیزای جدید نترسید و خودتون رو توی چالش بندازید چون فقط در این صورته که میتونید چیزای جدید یاد بگیرید.در کشور ما متاسفانه زیاد به مقوله برنامه نویسی اهمیت داده نمیشه و توی هیچ موسسه ای برنامه نویسی رو به صورت اصولی آموزش نمیدن . اما این بستگی به خودتون داره که تا چه حد عاشق برنامه نویسی باشید و با تلاش و پشتکار بتونید پله های پیشرفت رو طی کنید.