**PHP-NİN YARANMA TARİXİ VƏ İMKANLARI**

PHP dili 1994-cü ildə yaradılıb. Onun yaradıcısı Rasmus Lerdorf, ilk olaraq, bu dilin vasitəsilə sayta daxil olan ziyarətçilərin sayını müəyyən etməyi bacarıb.  
Yaradıcının sözünə görə, dil demək olar ki, çox qısa zamanda yazılıb. Əvvəlcə proqram Perl dilində yazılaraq, sayt əlavəsinin interfeysindən ibarət olub.   
Proqram saytda daxil olan informasiyanın dinamik dəyişməsi funksiyasını yerinə yetirirdi. Ancaq yaradılmış proqram həddindən artıq yavaş işlədiyi üçün Rasmus Lerdorf proqramı yenidən işləmiş və əlavə olaraq dilə C/C++, Fortran, Perl, Java, Visual Basic imkanlarını daxil etmişdir.   
Bundan sonra, serverdən istifadə edənlər bu yeni dil ilə maraqlanmışdır. Beləliklə, PHP dilinin birinci versiyası yayılmağa başlamışdır. Artıq 1995-ci ildə bu dil rəsmi olaraq yayılmağa başlamış və PHP (Personal Home Page) adlandırılmışdır.   
Bununla belə, o vaxt bu dilin imkanları həddindən artıq aşağı olmuşdur. 1995–ci ilin ortalarında PHP-nin, inkşaf etdirilərək, ikinci versiyası yaradıldı. Bu versiyaya əlavə olaraq verilənlər bazası ilə işləmək imkanları daxil edildi. Beləliklə, PHP daim inkşaf etdirilərək 3-cü, 4-cü və hal-hazırda 5-ci versiyası yaradıldı.   
Bu gün PHP dili güclü imkanlara malik HTML səhifəsində xüsusi kodları emal edən bir proqramlaşdırma dilidir. Məhz bu dilin köməyi ilə saytları dinamik etmək olur. Bu dilin köməyi ilə yaradılmış fayllar serverdə yerləşir və orada emal olunur. İstifadəçi PHP kodlu səhifələrə müraciət edərsə, informasiya emal olunaraq istifadəçinin brauzerində görünür.   
Yuxarıda qeyd etdiyimiz kimi, PHP dilinin böyük imkanları ilə yanaşı, bir çox çatışmayan cəhətləri də mövcuddur. Ən böyük çatışmayan cəhət yaradılmış proqramın yavaş işləməsidir. C dili ilə müqayisədə bu dilin surəti dəfələrlə aşağıdır. Ancaq HTML səhifəsində bu dildən script-lər şəklində istifadə olduqda bu problem hiss olunmur.   
Bundan başqa, artıq PHP–nin 4.1 versiyasında yaradılmış kompilyator surətin kifayət qədər artırılmasına imkan verir. Ona görə bu versiyadan sonra PHP dili kompilyasiyası olan dillər qrupuna daxil edildi. Bununla yanaşı, dildə qismən interpretasiya etmək imkanı da saxlanılıb.

### PHP-NİN İMKANLARI

- Verilənlər bazası ilə işləyən Web səhifələrin yaradılması üçün PHP ən əlverişli proqramlaşdırma dilidir. PHP Oracle, Adabas D, Sybase, FilePro, MySQL, Informix, Solid, dBase, ODBC və s.  
- Proqramlara girişi təşkil etmək üçün xüsusi funksiyanın yaradılması və login/parolun qoyulması imkanları;  
- PHP nəinki HTML-in kodunu, mətnini, eyni zamanda, müxtəlif formatlı (GIF, JPEG və ya PNG) şəkilləri emal edə bilir;  
- PHP serverdən faylların daşınmasını təşkil edir. Bundan başqa, PHP yüklənən fayllara tam nəzarət edir və onun ölçülərinə məhdudiyyət qoya bilir, eyni zamanda, həmin faylları tam idarə edir;  
- İstifadəçinin kompüterində brauzer vasitəsilə baxılmış fayllar haqqında (4kb) informasiya saxlayır. Belə informasiya "Cookie" adlanır. Cookie HTTP-nin başlığıdır. İstənilən göndərilən Cookie avtomatik olaraq PHP dəyişəninə çevrilir;  
- İstənilən mətn sətirləri üzərində hər cür manipulyasiya aparmağa imkan verir.  
- Lokal və ya Qlobal səviyyədə səhvləri müəyyən edə bilir və bu səhvlərin istifadəçiyə göstərə və ya göstərməyə bilər. Eyni zamanda, bu səhvləri elektron poçt vasitəsilə istənilən ünvana göndərə bilər.  
- Elektron ünvanları idarə edir və istənilən məktubu istənilən ünvanlara göndərmək imkanı vardır.  
- Sonda onu qeyd etmək olar ki, tam PHP vasitəsilə yaradılmış proqramlar yükləmə zamanı heç bir gecikmələrə yol vermir və normal fəaliyyət göstərir.

### PHP PROQRAMLAŞDIRMA

PHP dilində proqramlaşdırma 2 üsul ilə yerinə yetirilə bilər: Web server əlavəsi kimi və ya Konsol proqramı kimi.  
Biz əsasən birinci üsul üzrə proqramlara baxacağıq. Yəni brauzerin serverə müraciət edərək, yerinə yetirilən proqramın icrasına baxaq.  
Beləliklə, brauzer .php tipli faylı serverdən yükləyir və nəticəni HTML kodu şəklində verir.  
PHP kodunu HTML səhifəsinə daxil etmək üçün teqlərdən istifadə olunur və 4 formada, aşağıdakı şəkildə yazıla bilər:  
  
**1. XML stili:**

<?php

...

?>

**2. SGML stili:**

<?

...

?>

**3. HTML stili:**

<script language="php">

...

</script>

**4. ASP stili:**

<%

...

%>

PHP dilində kodları HTML səhifəsində yazmaq üçün kodları aşağıdakı şəkildə daxil etmək lazımdır:

<?php

.....

?>

Qeyd etmək lazımdır ki, bu yazılışda "php" sözünü yazmamaq da olar.  
Mətnləri və ya dəyişənlərin qiymətini ekrana vermək üçün "echo" əmrindən istifadə olunur. Burada əmrin böyük və ya kiçik hərflərlə yazılmasının heç bir fərqi yoxdur.  
Nümunə:

<?php

echo "Salam, xoş gəlmisiniz.";

?>

PHP dilində dəyişənləri istənilən simvolla və ya simvol birləşmələri ilə işarə etmək olar. Ancaq birinci simvol rəqəm və ya altdan xətt işarəsi ola bilməz.   
Bu zaman dəyişən qarşısında $ işarəsi qoymaq lazımdır. Nəzərə almaq lazımdır ki, C dilində olduğu kimi, dəyişənin böyük və ya kiçik hərflərlə işarə edilməsində fərq vardır.

<?php

$a=5;

$b=6;

$y=$a+$b;

echo $y;

?>

### ŞƏRHİN VERİLMƏSİ

PHP dilində şərhlərin verilməsinin bir çox üsulu vardır. Ən sadə üsul ikili (//) sləşdir.  
Bu işarədən sonra PHP kompilyatoru bütün sözləri əmr kimi qəbul etmir və yerinə yetirmir.  
Bununla yanaşı çoxsətirli şərhdən də (/\*...\*/) istifadə etmək olar. Bir sətirlik şərh vermək üçün (#) işarəsi tətbiq olunur.

<?php

echo("<p>Hello</p>"); // Şərh

echo("<p>Hello</p>"); # Şərh

/\*

Şərh

\*/

?>

### SABİTLƏR

PHP dilində sabit ədədlər define() funksiyası vasitəsilə elan edilir.

<?php

define(CONSTANT, value)

?>

Birinci sabitin adı, ikinci isə onun qiymətidir. Sabitlərdən istifadə etdikdə onların adına əsalanılır:

<?php

define(CONSTANT1,15);

define(CONSTANT2,"\x20");

define(CONSTANT3,"Hello");

echo(CONSTANT1);

echo(CONSTANT2);

echo(CONSTANT3);

?>

### PHP DİLİNDƏ VERİLƏNLƏRİN TİPİ. TİPLƏRİN DƏYİŞDİRİLMƏSİ

Yuxarıda qeyd edildiyi kimi, PHP dilində tipləri elan etmək mümkündür. Bu tipləri proqram boyu sətirlər və ya rəqəmlər üçün istifadə etmək olar. Bir çox tip toplusu mövcuddur:  
integer; string; boolean; double; array; object;  
  
**gettype()** funksiyası dəyişilmiş tipi əvvəlki vəziyyətə qaytarır.

<?php

$var = "5";

$var1 = "5";

echo(gettype($var));

echo "<br />";

echo(gettype($var1));

?>

Birinci halda PHP string tipini, iknici halda integer tipini qaytarır.  
  
**settype()** funksiyası tipi müəyyən edir:

<?php

$var = "5";

echo(gettype($var));

settype($var,integer);

echo "<br />";

echo(gettype($var));

?>

### OPERATORLAR

**Riyazi operatorlar**

|  |  |
| --- | --- |
| + | Toplama |
| - | Çıxma |
| \* | Vurma |
| / | Bölmə |
| % | Qalığın modul qiyməti (nümunə: 5 % 2=1) |

**Məntiq operatorları**

|  |  |
| --- | --- |
| & | (and) |
| | | (or) |
| ^ | və ya (xor) |
| ~ | (not) |
| << | Qüvvətə yüksəltmə |
| >> | Qüvvətə azaltma |

<?php

echo(4<<2); // bərabər 16

echo"<br />";

echo(5>>1); // bərabər 2

echo"<br />";

echo(6&5); // bərabər 4

echo"<br />";

echo(6|5); // bərabər 7

echo"<br />";

echo(6^5); // bərabər 3

?>

**print()** (çapa vermək) funksiya sayılmır (dilin konstruksiyasıdır) və ona görə də, yanında mötərizələr qeyd olunur. Bu komandadan sonra müxtəlif formalara əmrlər vermək olar.  
Ümumi qeyd: Bütün funksiyaların sətri bitdikdən sonra ; (nöqtəli vergül) qoyulmalıdır. Digər halda ondan sonra gələn yeni funksiyanız işləməyəcək və nəticədə error (səhv) verəcək.  
Burda diqqətli olun. Bu funksiya echo ilə oxşar olsa da, onunla eyni formada yazılmır. Ümumilikdə aşağıda göstərdiklərimi siz də yoxlayın.

<?php

print("Hello World"); // Hello World

$z="Nazim";

print $z; // Nazim

print "\" Rüstəmov Nazim\".";

// eynilə echo-dakı kimi, dırnaqlar slash işarəsi ilə görünür " Rüstəmov Nazim".

$i="Rüstəmov";

print "Nazim $i"; // Nazim Rüstəmov

?>

### SADƏ HESAB ƏMƏLİYYATLARI

Digər proqramlaşdırma dillərində olduğu kimi, burada da hesab əməlləri sadə məntiqlə yerinə yetirilir.  
PHP bütün riyazi funksiyaları, mötərizələri, məntiqi funksiyaları, bir vahid artmağa, azalmağa və başqaları.  
  
Məsələn:  
**$x++;** - **x** dəyişəninin bir vahid artması;  
**$x=$y=4;** - **x** və **y** dəyişənlərinin hər ikisi 4-ə bərabərdir;  
**$b=$a=5;**  
**$c=$a++;**  
  
Burada **c** dəyişəni **a** dəyişəninə bərabər olur və bir vahid artır.  
$x=2\*($a++);  
  
İndi ardıcıl 5 ədədin ekrana yazılması proqramına baxaq:

<?php

$i = 1;

echo $i;

$i++;

echo "".$i;

$i++;

echo "".$i;

$i++;

echo "".$i;

$i++;

echo "".$i;

?>

### GET VƏ POST METODLARI

Bu metod vasitəsilə dəyişənlərin qiymətlərini daxil etmək olar. Bu zaman lokal kompüterdən serverə müraciət olunur və dəyişənlərin qiymətini serverə çağırır. Bu metodun yazılış qaydası aşağıdakı kimidir:

<form action="hesabla2.php" method="get">

Daxil et x: (bura istənilən sözü yazmaq olar) <input type="text" name="x" /><br />

Daxil et y: <input type="text" name="y" /><br />

<input type="submit" value="Hesabla" />

</form>

Burada **x** və **y** dəyişənlərinin qiymətini daxil edib "Hesabla" düyməsini basan kimi "hesabla2.php" faylı yüklənir.

$x=$\_GET['x'];

$y=$\_GET['y'];

Bu zaman GET metodu vasitəsilə **x** və **y** qiymətləri oxunaraq serverin yaddaşına yüklənir.  
  
Misal:  
**Birinci fayl:**

<html>

<head>

<meta http-equiv="content-type" content="text/html; charset=iso-8859-1">

<title>Hesabla1</title>

</head>

<body>

<form action="hesabla2.php" method="get">

Daxil et x: (bura istənilən sözü yazmaq olar) <input type="text" name="x" /><br />

Daxil et y: <input type="text" name="y" /><br />

<input type="submit" value="Hesabla" />

</form>

</body>

</html>

**İkinci fayl:**

<html>

<head>

<meta http-equiv="content-type" content="text/html; charset=iso-8859-1">

<title>Hesabla2</title>

</head>

<body>

<?php

$x=$\_GET['x'];

$y=$\_GET['y'];

$result=$x\*$y;

echo "y=$result<br>";

$result=sqrt($x);

echo $result

?>

</body>

</html>

### PRİNTF() И SPRİNTF() FUNKSİYALARI

Qeyd etmək lazımdır ki, hər iki funksiya nəticələrin verilməsi formatını müəyyən etmək məqsədi üçündür. **printf()** funksiyası nəticə selinin brauzerdə verilməsi, **sprintf()** funksiyası isə sətirin veriməsi üçün uyğun formatın seçilməsi üçündür. Yazılış qaydası belədir:

int printf(string format [, mixed args])

string sprintf(string format [, mixed args])

Ümumiyyətlə, bu funksiyaların yazılışı belə olur:

<?php

printf("Hello!"); // "Hello!" yazılır.

sprintf("Hello!"); // Heç nə verilmir.

$str = sprintf("Hello!"); // Həmin sətri dəyişənə mənimsədir.

printf($str); // Həmin söz ekrana verilir.

?>

**Request metodu**  
Bu metod da GET və ya POST metodu kimi dəyişənləri serverə göndərmək üçündür.  
  
**Şərti keçid operatoru (if)**  
Bir çox hallarda qoyulmuş şərtdən asılı olaraq, müxtəlif kodları yerinə yetirmək tələb olunur. Bu halda **if** əmrindən istifadə edilir. Yazılış qaydası aşağıdakı kimidir:

if (şərt)

ifadə\_1;

else

ifadə\_2;

<?php

$i = 10;

$j = 5\*2;

if ($i == $j)

echo '$i və $j bərabərdir';

else

echo '$i və $j bərabər deyil';

?>

Göründüyü kimi, **$i** və **$j** arasındakı şərt == (iki bərabərlik) işarəsi vasitəsilə yerinə yetirilir. Beləliklə, şərtlərin yoxlanması aşağıdakı işarələr vasitəsilə göstərilir:  
Burada **if/else** operatoru şərti operator kimi istifadə olunur. Ümumilikdə bu bele görünür:  
!= - bərabər deyil;  
< - kiçik;  
> - böyük;  
<= - kiçik bərabər;  
>= - böyük bərabər.

<?php

$i = 10;

$j = 5\*2;

if ($i == $j)

echo '$i və $j dəyişənlərinin eyni qiyməti var';

?>

<?php

$i = 10;

$j = 11;

if ($i > $j)

$diff = $j - $i;

echo '$j $i -dən böyükdür; $j və $i arasındakı fərq: '. $diff; // Səhvdir!

?>

Bir çox hallarda qoyulmuş şərtə əsasən, bir neçə əmrlər ardıcıllığı yerinə yetirmək tələb olunur. Bu zaman həmin əmrlər fiqurlu mötərizə daxilində yazılır:

<?php

$i = 10;

$j = 11;

if ($i > $j) {

$diff = $j - $i;

echo '$j $i -dən böyükdür; $j və $i arasındakı fərq: '. $diff;

}

?>

Qeyd etmək lazımdır ki, hətta təkcə bir əmri yerinə yetirmək lazımdırsa, onda da bir sətri fiqurlu mötərizə daxilində yazmaq məsləhətdir. Bu zaman təsadüfi səhvlərə yol vermək şansı azalır.  
Bundan başqa, şərt daxilində yeni şərt vermək olar və bu zaman yazılış qaydası bu cür olur: if... else if... else.

<?php

$i = 10;

$j = 11;

if ($i > $j) {

echo '$i $j -dən böyükdür';

} else if ($i < $j) {

echo '$i $j -dən kiçikdir';

} else { // Bu halda bərabərlikdən başqa bir şey qalmır :)

echo '$i və $j bərabərdirlər';

}

?>

Dəyişənlərin qiymətini serverə göndərərkən ilkin olaraq, həmin dəyişənin elan olunmamasına görə ekrana bu haqda sorğu çıxacaq. Belə vəziyyəti aradan götürmək üçün şərti keçid əmrindən istifadə olunur.

<html>

<head>

<meta http-equiv="content-type" content="text/html; charset=iso-8859-1">

<title>Untitled Document</title>

</head>

<body>

<form action="vurmaq.php" method="post">

<input type="text" name="a" size="20" maxlength="8" /><br />

<input type="text" name="b" size="20" maxlength="8" /><br />

<input type="submit" value="Vurmaq"></form>

<?php

if (isset($\_POST["a"])) {$a=$\_POST["a"]; }

else {$a=0;};

if (isset($\_POST["b"])) {$b=$\_POST["b"];

} else {$b=0;};

echo "<br> hasil=".$a\*$b;

?>

</body>

</html>

Bu yazılış qaydası hər üç - GET, POST və REQUEST metodları üçün doğrudur.

<html>

<head>

<meta http-equiv="content-type" content="text/html; charset=iso-8859-1">

<title>Mühazirə 3</title>

</head>

<body>

<form method="post" action="muhaz3.php">

<input name="i" type="text" size="30" />

<input name="j" type="text" size="20" />

<input type="image" src="../photos/search.jpg" border="0" />

</form>

<?php

if (isset($\_POST["i"])) {$i=$\_POST["i"];}

else {$i=0;};

if (isset($\_POST["j"])) {$j=$\_POST["j"];}

else {$j=0;};

$i=$\_POST["i"];

$j=$\_POST["j"];

if ($i > $j)

{

$diff=$j+$i;

echo "Cəm".$diff;

}

else {

$diff=$j-$i;

echo "Fərq".$diff;

}

?>

</body>

</html>

### SWİTCH

Bir çox hallarda eyni dəyişənin qiymətindən asılı olaraq, proqramın icrasını şaxələndirmək lazım gəlir. Bu zaman **switch** əmrindən istifadə olunur.  
Məsələn, aşağıdakı əmrlər toplusunun əvəzinə

if ($i==1) {

ifadə 1

} else if ($i==2) {

ifadə 2

} else if ($i==3) {

ifadə 3

}

Daha əhəmiyyətli yazılış qaydasından istifadə etmək olar:

<form action="switch.php" method="post">

<input type="text" name="i" />

<input type ="submit" value="Ok" />

</form>

<?php

if (isset($\_POST["i"])) {$i=$\_POST["i"];}

else {$i=1;};

switch ($i) {

case 1:

echo 'Bir';

break;

case 2:

echo 'İki';

break;

case 3:

echo 'Üç';

break;

default:

echo '$i > 3';

};

?>

**switch** əmrindən sonra yazılmış dəyişənin qiymətindən asılı olaraq, ekranda **case** əmrindən sonra göstərilmiş qiymətə uyğun sətir yerinə yetirilir. Burada yazılan ifadəni fiqurlu mötərizənin içərisinə almaq lazım deyil. Burada proqramın icrası **break** əmri vasitəsilə yekunlaşır.  
**default** əmri qalan bütün qiymətlərdə doğruluğunu göstərir. Bu əmr sonda olduğu üçün **break** əmrinə ehtiyac yoxdur. Bu əmrə ehtiyac yoxdursa, onda yazmamaq olar.

<?php

$i = 1;

switch ($i) {

case 0: // "break" qəsdən yazılmayıb!

case 1:

echo 'Sıfır və ya bir';

break;

case 2:

echo 'İki';

break;

case 3:

echo 'Üç';

break;

}

?>

#### ****Dövrlər****

İstənilən proqramlaşdırma dilində bu və digər şəkildə dövrlərdən istifadə olunur. Bu baxımdan PHP dilində 3 cür dövr vardır:  
  
**while**  
Bu əmrin yazılış qaydası aşağıdakı kimidir:

<?php

$i = 1;

while($i < 10) {

echo $i . "<br>\n";

$i++;

}

?>

Dövr **while** əmrindən sonra gələn şərti yoxlayır və ondan asılı olaraq, növbəti ifadə və ya fiqurlu mötərizənin içərisindəki ifadələr yerinə yetirilir.  
  
**do..while**

<?php

$i = 1;

do {

echo $i . "<br>\n";

} while ($i++ < 10);

?>

### POST METODU

Bu dərsimizdə sizinlə birlikdə PHP-də **post** funksiyasını öyrənəcəyik. PHP-də **post** funksiyası müəyyən məlumatın daşınması üçün istifadə olunur. Bunun üçün HTML kod ilə qutucuq yaradırıq:

<?php

echo "<input name=\"php\">";

?>

İndi isə, **post** metodunu daha dəqiq öyrənək. Kodlara baxaq:

<?php

// **post** əmri gəldikdə görsənməsi üçün

if($\_POST) {

echo "$\_POST['php']";

}

elseif"<form action=\"hansı səhifəyə yönlənməsi üçün biz bu səhifədə iş gördüyümüz üçün buranı boş saxlayın\" method=\"POST\">"; {

echo "<input name=\"php\">"; // Qutucuq ("php" sözünü istədiyiniz sözlə əvəz edə bilərsiniz.)

echo "<input type=\"submit\" value=\"Göndər\">";

}

?>

Gördüyünuz kimi, əsas tərəfdə kodları yazdıq, ikinci tərəfdə isə, echo $\_POST['postun adı']; ilə ekrana çıxardıq.

### WHİLE FUNKSİYASI

**while** funksiyası funksiyanın təkrarlanması üçün istifadə olunur. Məsələn, bir dəyişənin hər təkrarda artması funksiyasını yazaq.  
Kodlar:

<?php

$a=0;

while($a<100) {

$a++; // Dəyərin artması üçün

echo "$a<br/>";

?>

Bu funksiya əsasən, məntiqi proqramların yazılmasında istifadə olunur. Məsələn, sıfır və yüz arasında beşə bölünən ədədləri ekrana çıxaraq:

<?php

$a=0;

while($a<100)

$a=$a+5;

echo "$a<br/>";

?>

### FOREACH

**foreach** funksiyası PHP-də gələn məlumatları seçdirmək üçün istifadə olunur. Məsələn, bir array var və içində 100-dən artıq məlumat var. Əlbəttə, onu tək-tək seçib yazmaq çətin olur, bu vaxt **foreach** işə yarayır.  
Koda baxaq:

<?php

$ar=array("facebook.com","ilkaddimlar.com","mobiz.az","google.az");

echo "Dəyərin içindəki məlumatlar:<br/>";

$a=0;

foreach($ar as $murik=>$site) {

$a++;

echo "$a)$site<br/>";

?>

### PHP-Nİ YÜKLƏMƏ YOLU

PHP dinamik dil sayılır, yəni belə proqramlar məlumatların sorğu şəklində müəyyən bir serverə göndərilib və oradan sorğuya uyğun cavabların alınması sistemi ilə işləyir. Bu sistemin işləməsi üçün bizə baza məlumatlarına girə bilən serverlə bağlı proqramlar lazımdır. Apache, MYSQL kimi. Mən Apache-ni uyğun bilirəm (daha ətraflı izah olunacaq).  
  
Bizə lazım olacaq ləvazimatlar (proqramlar):   
**PHP 5.1.2** [www.php.net/downloads.php](http://www.php.net/downloads.php)  
**Apache 1.3.34** [www.php.net/downloads.php](http://www.php.net/downloads.php)  
  
PHP 5.1.2 arxivini (\*.rar arxivi) C:\php direktoriyasına boşaltdıqdan sonra C:\php\php5ts.dll faylını C:\WINDOWS\system32 direktoriyasına köçürürük (copy). C:\php qovluğunda php.ini-dist faylını tapacaqsınız. Bu faylı C:\WINDOWS direktoriyasına köçürür və adını dəyişdiririk (php.ini kimi). Php.ini faylını notepad vasitəsi ilə açırsınız və Ctrl+F düymələrini birlikdə basaraq, axtarışa "globals", "magic\_quotes" (ayrı-ayrılıqda) yazın və aşağıdakı formada olmasından əmin olun (əks halda, böyük ehtimalla işləməyəcək). Bunun səbəbini irəlidə izah edəcəyəm.   
registry\_globals= on  
magic\_quotes\_gpc= off  
  
Sonra, 2-ci əsas məsələ, Apache 1.3.34 yükləməyə çalışaq.  
İki dəfə üzərinə basın və "Next" düyməsi ilə irəliləyin, sonra Apache License qəbul edərək (I accept) işarə edin, "Next" basıb irəliləyin və belə bir ekran çıxanda şəkildəki kimi doldurun. (localhost, öz e-poçtunu yaz )



Yüklədikdən sonra C:\Program Files\Apache Group\Apache\conf\httpd daxil olub, orada Ctrl+F vasitəsi ilə "LoadModule" yazıb axtarışa verin və aşağıda yazdığım formada yazı tapsanız, deməli, doğru yoldasınız. Mənim qırmızı ilə işarələdiyim yazını (LoadModule php5\_module C:/php/php5apache.dll) eyni ilə LoadModule-ların sonunda yaz. Səhvə yol verməyin və aşağıdakı ilə eyni olmasını yoxla.  
  
#LoadModule vhost\_alias\_module modules/mod\_vhost\_alias.so  
#LoadModule mime\_magic\_module modules/mod\_mime\_magic.so  
#LoadModule status\_module modules/mod\_status.so  
#LoadModule info\_module modules/mod\_info.so  
#LoadModule speling\_module modules/mod\_speling.so  
#LoadModule rewrite\_module modules/mod\_rewrite.so  
#LoadModule anon\_auth\_module modules/mod\_auth\_anon.so  
#LoadModule dbm\_auth\_module modules/mod\_auth\_dbm.so  
#LoadModule digest\_auth\_module modules/mod\_auth\_digest.so  
#LoadModule digest\_module modules/mod\_digest.so  
#LoadModule proxy\_module modules/mod\_proxy.so  
#LoadModule cern\_meta\_module modules/mod\_cern\_meta.so  
#LoadModule expires\_module modules/mod\_expires.so  
#LoadModule headers\_module modules/mod\_headers.so  
#LoadModule usertrack\_module modules/mod\_usertrack.so  
#LoadModule unique\_id\_module modules/mod\_unique\_id.so  
  
LoadModule php5\_module C:/php/php5apache.dll  
  
#  
# Reconstruction of the complete module list from all available modules  
  
2-ci addım. Yenə axtarış verin və bu dəfə "AddModule" axtarın. Eynilə yuxarıda izah elədiyim kimi, AddModule-ların axırında AddModule mod\_php5.c yazın. Yaddaşa verin, faylı bağlayıb çıxın.  
  
#AddModule mod\_vhost\_alias.c  
AddModule mod\_env.c  
AddModule mod\_log\_config.c  
#AddModule mod\_mime\_magic.c  
AddModule mod\_mime.c  
AddModule mod\_negotiation.c  
#AddModule mod\_status.c  
#AddModule mod\_info.c  
AddModule mod\_include.c  
AddModule mod\_autoindex.c  
#AddModule mod\_unique\_id.c  
AddModule mod\_so.c  
AddModule mod\_setenvif.c  
AddModule mod\_php5.c  
  
#  
# ExtendedStatus controls whether Apache will generate "full" status  
  
Sonra Start-dan "Run" düyməsinə basın və şəkildəki kimi yazıb "OK" düyməsinə basın. (net stop apache)



Sonra, bir neçə saniyədən sonra aşağıdakı kimi yazıb OK düyməsinə basın.(net start apache)

