PHP

Curs 9/25.04.2023

https://www.php.net/
https://httpd.apache.org/
https://www.apachefriends.org/

Modelul client-server. Pagini web interactive (dinamice)

PHP suportă încărcarea fişierelor de pe calculatorul client, operaţie cunoscută sub numele de *upload* (standard propus de E. Nebel şi L. Masinter de la Xerox) şi oferă suport pentru *cookies* (mecanism de stocare a datelor în navigatorul client pentru identificarea utilizatorilor, propus de Netscape).

Limbajul PHP are deasemenea suport pentru diverse servicii server, utilizând protocoale precum *IMAP, SNMP, NNTP, POP3* şi *HTTP*.
Paginile HTML sunt de tip "static", adică informaţiile prezentate în aceste pagini sunt identice pentru toţi vizitatorii, fiind simple pagini de prezentare. Web-ul este însă dinamic, adică pentru unel pagini este nevoie să fie actualizate "în timp real". **Dacă** doi vizitatori, aflaţi în locuri diferite, accesează acelaşi site de joburi în acelaşi timpasrefel încât unul doreşte afişarea joburilor din domeniul economic, iar al doilea doreşte afişarea joburilor din domeniul securităţii şi sănătăţii în muncă, atunci serverul web va afişa pentru fiecare vizitator al siteului pagina corespunzătoare cerinţelor specificate. Acest lucru este posibil datorită tehnologiilor **de tip client-server**. Acest tip de tehnologii presupun stocarea datelor **pe un server web** şi apoi afişarea acestora la cererea fiecărui vizitator, în forma dorită de acesta.

Un server web reprezinta un calculator conectat permanent la Internet, care trimite către client (care este un calculator pe care rulează un browser) pagini în format HTML. La un server se pot conecta simultan mai mulţi clienţi, care pot avea acces la aceleaşi informaţii. Diferenţa este esenţială comparativ cu paginile simple HTML, care sunt afişate în browserul vizitatorului aşa cum au fost construite. Web-ul este dinamic, iar programele care fac posibil acest lucru sunt numite scripturi CGI sau scripturi server-side, întrucât acestea folosesc o interfaţă standard numită *Common Gateway*. Scripturile sunt scrise in limbajul C sau ajutorul unui limbaj specializat, numit şi limbaj de scripting (cele mai utilizate fiind PHP, ASP şi Perl) şi sunt stocate pe serverul web pe care şi rulează.

Diferenţa dintre limbaj de scripting pe partea de client (ex. JavaScript) şi unul server-side este esenţială doarece JavaScript rulează în browserul clientului, pe când un script **server-side** rulează pe server, având acces la unele informaţii stocate pe server la care un **script clientside** nu are acces. Un exemplu sugestiv este un contor de pagina web. De câte ori

cineva accesează pagina, scriptul server-side va contoriza vizitarea paginii într-o bază de date stocată pe server.

Tipuri de variabile utilizate în limbajul PHP

PHP este un limbaj de programare la fel ca şi BASIC, Pascal, C, C++, Visual Basic, Java etc şi are caracteristicile specifice unui limbaj de programare. În cadrul acestuia se lucrează cu variabile care reprezintă zonă de memorie, căreia i se atribuie un nume pentru a putea fi recunoscută şi apelată ulterior. Variabilele se compun din simbolul $ urmat de numele variabilei $variabilăphp. Numele variabilei poate începe cu o literă sau un underscore, urmat de litere, cifre sau caractere underscore. Principalele tipuri de date sunt cele din tabelul următor.

Tipuri Scalare	**Integer –** este o valoare din mulţimea nuinformaticalor întregi
	Float - o astfel de variabilă poate fi specificată fie prin forma zecimală, fie prin cea ştiinţifică (cu exponent)
	String - este un şir de caractere şi poate fi specificat în 2 moduri: o folosind ghilimele simple ‘’ o folosing ghilimele duble “”
Tipuri compuse	**Array** **Object**
Tipuri speciale	**Resource** - variabilă de tip Resource este o variabilă specială, folosită pentru păstrarea unor referinţe către anumite resurse externe
	NULL - valoarea specială NULL este atribuită oricărei variabile care nu a fost iniţializată. Aceasta este singura valoare pe care o pot avea variabilele de tip NULL. Se consideră că o variabilă este de tip NULL în următoarele situaţii: i s-a atribuit constanta NULL; nu a fost iniţializată; a fost deziniţializată (prin intermediul funcţiei unset()).
Tipul BOOLEAN	False - se poate converti o variabilă de orice tip la tipul Boolean. Valorile care în urma conversiei se transformă în FALSE sunt: • Nr întreg 0 • Nr real 0.0 • Şirul vid • Şirul “0” • Un vector fără nici un element • Un obiect fără nici o variabilă membru • O variabilă de tipul NULL • O variabilă nedefinită
	True - orice altă valoare se converteşte în TRUE

Tipul ARRAY (vectori) pot fi considerate mulţimi formate din chei. Fiecărei chei i se ataşează o valoare. Acest tip este optimizat, astfel încât să poată fi folosit în locul următoarelor structuri de date: liste, tabele de dispersie, colecţii, stive, cozi etc. Vectori pot fi :

- vectori numerici
- vectori asociativi
- vectori multidimensionali

VARIABILE GLOBALE sunt disponibile în orice zona a codului sursă. Exemple:

$GLOBALS	cuprinde referinţe spre orice variabilă globală care este accesibilă scriptului PHP curent;
$_SERVER	conţine o serie de variabile ale căror valori sunt setate de server-ul web. (majoritatea depind de mediul de execuţie al script-ului curent);
$_GET şi $_POST	sunt variabile array globale. $_GET poate fi folosită pentru a trimite variabile cu valori prin intermediul link-urilor. $_POST poate fi
	folosită pentru a trimite variabile cu valori prin intermediul formularelor.
$_COOKIE	conţine valorile variabilelor care cuprind informaţii referitoare la cookie-urile păstrate pe calculatorul utilizatorului ce accesează pagina web.
$_FILES	conţine variabile primite de script prin intermediul încărcărilor de fişiere prin metoda post.
$_ENV	conţine variabile disponibile prin intermediul mediului în care este executat.
$_REQUEST	variabile disponibile prin intermediul oricărui tip de mecanism cu ajutorul căruia utilizatorul poate introduce date.
$_SESSION	variabile care corespund sesiunii curente a script-ului.

Article III.

Crearea scripturilor PHP

Fiecare script PHP arata ca in figurile de mai jos si indică serverului că textul cuprins între acestea este format din instrucţiuni PHP.

```
<?php
// continutul scriptului
?>
```

sau, echivalent:

```
<?
// continutul scriptului
?>
```

Scripturile PHP execută următoarele de operaţii elementare:

- preluarea datelor de la utilizator,
- prelucrarea acestor date,
- obţinerea accesului la datele stocate pe server, - prelucrarea datelor stocate pe server şi
- afişarea datelor.

Scripturile PHP sunt formate din instrucţiuni iar cea mai simplă instrucţiune PHP este cea de afişare a unui text în browser.

echo "Informaţii afişate în browser";

Tag-urile HTML se introduc în codul PHP astfel:

echo " Introducerea codului HTML într-un script ";

Un comentariu se introduce într-un script PHP cu ajutorul marcajului // (când comentariul este scris pe o singură linie) sau /* conţinutul comentariului */ (atunci când comentariul este scris pe mai multe linii).

```
<?php
// comentariu pe o linie
//continutul scriptului
?>
```

```
<?php
/* comentariu pe mai
multe linii */
// continutul scriptului
?>
```

Instrucţiunea **continue** este utilizată într-o buclă iar execuţia iteraţiei curente este întrerupă şi se trece la execuţia iteraţiei următoare. Instrucţiunea continue acceptă un argument numeric opţional care va indică câte bucle imbricate vor fi ignorate.

Instrucţiunea **foreach** – versiunea PHP4 (nu şi PHP3) dispune de comanda foreach, ca Perl sau alte limbaje. Instrucţiunea oferă un mod simplu de a parcurge un tablou.

Caracterizarea formularelor HTML

- Permit introducerea datelor de către vizitatorul paginii web.
- Crearea se face începând cu matematicachea de etichete **<form>…</form>.**
- Atributele lui <form> sunt: **action** şi **method**.

Action precizează ce se va întâmpla cu datele introduse în formular. În general i se asociază adresa unui script aflat pe server (în cazul nostru numele fişierului PHP) care va prelucra datele.

Method se referă la modul în care vor fi trimise datele spre scriptul de pe server. Valorile posibile sunt: **GET şi POST**.

- GET permite trimiterea cantităţi restrânse de date prin adăugarea lor la URL
- POST permite trimiterea cantităţilor mari de date iar acestea sunt expediate separat.

O regulă empirică privind alegerea între GET şi POST ar fi următoare: mulţi programatori utilizează GET pentru formularele care execută o căutare sau interogare şi POST pentru formularele care actualizează o bază de date sau un fişier. Astfel, datele trimise prin metoda GET pot fi vizualizate de catre utilizator.

Pentru a realiza un formular se utilizează eticheta FORM exemplificată în capitolul HTML:

<FORM name =”*numeformular” METHOD=”metoda” ACTION=”url”* >

Atributul ACTION specifică adresa URL a scriptului PHP care prelucrează datele transmise prin intermediul formularului. Adresa URL poate fi o adresa completă, care include protocolul, numele gazdei şi calea de acces, respectiv o adresă parţială, care specifică o locaţie relativă la locaţia paginii curente.

Exemplu: <FORM method=”POST” action=”test1.php”>

Pricipalele controale care pot fi incluse în formular (vezi capitolul de HTML) cât şi principalele evenimente ataşate acestora sunt:

Listă controale

- textbox (text), password textbox (password), butoane radio (radio), casete de validare (checkbox), butoane (button, submit, reset), lista derulantă (select), editbox (textarea), caseta de fişier (file).

Listă evenimente

- onclick, ondblclick, onmousedown, onmouseup, onmouseover onmousemove, onmouseout, onkeypress, onkeydown, onkeyup

Section 6.01 Prin intermediul formularului se trimite la server un fişier astfel:

<input type=“file” accept=“tip_mime” name=“nume” value=“text”>
unde:

- **accept** este un atribut care specifică tipul fişierului,
- **name** este numele casetei,
- **value** se foloseşte pentru un nume prestabilit de fişier,
- **MIME** (Multipurpose Internet Mail Extensions).

În eticheta form trebuie adăugat şi atributul enctype=“multipart/form-data ”.

Formatele cele mai frecvent folosite pt MIME sunt:

TIP MIME	TIP DATE	EXTENSII DE FIŞIER
application/msexcel	Microsoft Excel	xls
application/msword	Microsoft Word	doc, dot
application/octet-stream	Binara	exe
application/pdf	Adobe Acrobat	pdf
application/ppt	Microsoft PowerPoint	ppt
application/zip	Date comprimate în format ZIP	zip
audio/midi	Musical Instrument Digital Interface (MIDI)	mid,midi
audio/x-wav	Windows Audio Format (WAV)	wav
image/gif	Compuserve Graphics Interchange Format (GIF)	gif
image/jpeg	Joint Photographics Expert Group (JPEG)	jpeg, jpg, jpe
image/TIFF	Tagged Image Format (TIF)	tif, tiff
text/HTML	HTML	htm, html
text/plain	Text simplu	txt
text/richtext	Rich Text Format (RTF)	rtf
video/mpeg	Secventa video	mpg, mpv, mpe, mpeg
video/quicktime	Secventa video	qt, mov
video/x-msvideo	Secventa video	avi

În general prelucrarea datelor transmise la server se face într-un fişier distinct decât cel care conţine formularul, caz în care **action="nume_fişier.php".**

O variabilă se preia dintr-un formular prin una din modalităţile :

$_POST["nume_variabilă"] sau cu $_GET["nume_variabilă"] (în funcţie de atributul method al formului, acesta poate fi post sau get).

Aplicatii

1.

```
<html>
<head></head>
<body>
Bun venit
<?php
$nume="Popescu";//preluam dintr-un input...$_GET; $_POST
echo "<br /><b><i>Bun venit Domnule </i></b>".$nume;
?>
<h2> MESAJ</h2>
<?php
echo "<br />terminat selectia";
?>
</body>
</html>
```

Aplicatia 2

```
<head></head>
<body>
Bun venit
<?php
$nume="Popescu";
echo "<br />Bun venit Domnule ".$nume;
?>
<h2> Alegeti data [zi/luna/an]</h2>
<b>Ziua</b>
<select name="s">
<?php
for($i=1;$i<=31;$i++)
{
echo "<option value=".$i.">".$i."</option>";
}
?>
</select>
<b>Luna</b>
<select name="s">
<script>

for(i=1;i<=12;i++)
{
document.write("<option value="+i+">"+i+"</option>");
}
</script>
</select>
<?php
echo "<br />terminat selectia";
?>
</body>
<html>
```

Aplicatia 3

```php
<html>
<head>
<title>explode, implode, print_r</title>
</head>
<body>
<h1>Exemplificare utilizare implode, explode, print_r</h1>
<p>Cerinta : Realizati un script php care utilizează funcțiile print_r, explode și implode.</p>
<?php
$array_var=array();
$string_var="";
for ($i=1; $i<5; $i++) {
    $array_var []= "Element_$i";
}
?>
<h3>Scrie print_r($array_var)</h3>
<p><?php print_r($array_var); ?></p>

<h3>Implode array -> string, cu separator ':', print_r($string_var)</h3>
<p>$string_var =
<?php $string_var = implode($array_var, ":"); print_r($string_var); ?></p>
<hr />
<h3>Explode string into array, print_r($array_var2)</h3>
<pre><?php $array_var2 = explode(":", $string_var); print_r($array_var2); ?></pre>

</body>
</html>
```

Aplicatia 4

```php
<html>
<body>
<form name="f" action="ex3.php" method="post">
Nume= <input type="text" name="t1"><br/>
Alegeti o culoare:
<br/>
<input type="checkbox" name="c1" value="rosu"> Rosu <br>
<input type="checkbox" name="c2" value="verde"> Verde <br>
<input type="checkbox" name="c3" value="galben"> Galben <br>
<input type="checkbox" name="c4" value="albastru"> Albastru <br>
Alegeti o culoare: <br/>
<input type="radio" name="r" value="rosu"> Rosu <br>
<input type="radio" name="r"  value="verde"> Verde <br>
<input type="radio" name="r"  value="galben"> Galben <br>
<input type="radio" name="r"  value="albastru"> Albastru <br>
Alegeti o culoare: <br/>
<select name=s[] size="3" multiple>
<option value="rosu"> Rosu </option>
<option value="verde"> Verde </option>
<option value="galben"> Galben </option>
<option value="albastru"> Albastru </option>
<option value="portocaliu"> Portocaliu </option>
<option value="violet"> Violet </option>
</select>

<br>
<input type="reset" value="sterge">
<input type="submit">
</form>
</body>
</html>
```

Fisierul php

```
<?php
$filename="afisare.txt";
$handle=fopen($filename, "a");//preluare
$a1=$_POST["t1"];
$x="".$a1." culoare checkbox: ";
if(isset($_POST["c1"]))
	$a1=$_POST["c1"];
else
	$a1="";
if(isset($_POST["c2"]))
	$a2=$_POST["c2"];
else
	$a2="";
if(isset($_POST["c3"]))
	$a3=$_POST["c3"];
else
	$a3="";
if(isset($_POST["c4"]))
	$a4=$_POST["c4"];
else
	$a4="";
if($a1)
	$x=$x." ".$a1;
if($a2)
	$x=$x." ".$a2;
if($a3)
	$x=$x." ".$a3;
if($a4)
	$x=$x." ".$a4;
// radio
$culoare_radio="";
	for($i=0;$i<4;$i++)
	if(isset($_POST["r"]))
		$culoare_radio=$_POST["r"];
if($culoare_radio)
	$x=$x." coloare button radio ".$culoare_radio." ";
$culoare_lista="";
if(isset($_POST["s"]))
	 foreach ($_POST['s'] as $culoare)
		$culoare_lista=$culoare_lista." ".$culoare;
$x=$x." coloare lista selectie ".$culoare_lista." ";
echo "<br />".$x;
fwrite($handle, $x);
echo "
<form name='a' method='post' action ='raspuns.php'>
continuati:<input type='text' name='ras' size='1' value='d' /><b>da/nu</b>
<input type='submit' value='trimite' />
</form>
";
?>
```

```php
<?php
$rasp=$_POST['ras'];
//echo $rasp;
if((strtoupper($rasp))=='D')
{include "ex3.html";}
else
    echo "sfarsit";//select
?>
```

Funcţii în PHP

```
function nume_functie(param1, param2,…,paramN){ instrucţiuni;    }

$var_returnată=nume_funcţie(param1,param2,..,paramN);
```

O funcţie poate fi definită oriunde în cadrul script-ului, iar în interiorul unei funcţii poate să apară orice secvenţă validă de cod (poate cuprinde definirea altor funcţii, clase etc.). Pentru ca funcţia să returneze un rezultat se foloseşte construcţia return urmată de un parametru ce reprezintă valoarea funcţiei.

Funcţii de afişare

PHP include doua functii utile pentru generarea datelor de iesire formatate.

printf() si **sprintf()**.

Functia **printf()** afiseaza datele sale de iesire, iar **sprintf()** returneaza datele sale de iesire sub forma unei valori sir.

string **sprintf** (string $format [, mixed $args [, mixed $...]])

Returns a string produced according to the formatting string `format`.

Parametri

format

The format string is composed of zero or more directives: ordinary characters (excluding *%*) that are copied directly to the result, and *conversion specifications*, each of which results in fetching its own parameter. This applies to both **sprintf()** and printf().

Each conversion specification consists of a percent sign (*%*), followed by one or more of these elements, in order:

1. An optional *sign specifier* that forces a sign (- or +) to be used on a number. By default, only the - sign is used on a number if it's negative. This specifier forces positive numbers to have the + sign attached as well, and was added in PHP 4.3.0.
2. An optional *padding specifier* that says what character will be used for padding the results to the right string size. This may be a space character or a *0* (zero character). The default is to pad with spaces. An alternate padding character can be specified by prefixing it with a single quote (*'*). See the examples below.
3. An optional *alignment specifier* that says if the result should be left-justified or right-justified. The default is right-justified; a - character here will make it left-justified.

4. An optional number, a *width specifier* that says how many characters (minimum) this conversion should result in.
5. An optional *precision specifier* in the form of a period (`.') followed by an optional decimal digit string that says how many decimal digits should be displayed for floating-point numbers. When using this specifier on a string, it acts as a cutoff point, setting a maximum character limit to the string.
6. A *type specifier* that says what type the argument data should be treated as. Possible types:
    - *%* - a literal percent character. No argument is required.
    - *b* - the argument is treated as an integer, and presented as a binary number.
    - *c* - the argument is treated as an integer, and presented as the character with that ASCII value.
    - *d* - the argument is treated as an integer, and presented as a (signed) decimal number.
    - *e* - the argument is treated as scientific notation (e.g. 1.2e+2). The precision specifier stands for the number of digits after the decimal point since PHP 5.2.1. In earlier versions, it was taken as number of significant digits (one less).
    - *E* - like *%e* but uses uppercase letter (e.g. 1.2E+2).
    - *u* - the argument is treated as an integer, and presented as an unsigned decimal number.
    - *f* - the argument is treated as a float, and presented as a floating-point number (locale aware).
    - *F* - the argument is treated as a float, and presented as a floating-point number (non-locale aware). Available since PHP 4.3.10 and PHP 5.0.3.
    - *g* - shorter of *%e* and *%f*.
    - *G* - shorter of *%E* and *%f*.
    - *o* - the argument is treated as an integer, and presented as an octal number.
    - *s* - the argument is treated as and presented as a string.
    - *x* - the argument is treated as an integer and presented as a hexadecimal number (with lowercase letters).
    - *X* - the argument is treated as an integer and presented as a hexadecimal number (with uppercase letters).

Exemple:

```
<?php
$num = 5.7;
echo $num;
$num = sprintf("%05.2f", $num);
echo '<br>'. $num;
?>      Rezultatul afisat va fi:  5.7  05.70
```

Cu ajutorul expresiei "%05.2f", sprintf() formateaza numarul din $num astfel incat acesta sa fie format din 5 caractere dintre care unul punct (.) si 2 zecimale.
exemplu cu **printf()**:

```
<?php
$n = 8;
printf("Valoarea lui n este: %d", $n);
?>  Va afisa "Valoarea lui n este: 8".
```

Funcţii pentru şiruri

- *str_repeat($şir, $n)* – repetă şirul $şir de un număr de n ori.
- *strrchr($şir, $caracter)* –returnează parte a unui $şir, începând cu ultima apariţie a caracterului $caracter în şirul $şir.
- *trim($şir)* – elimină spaţiile din stânga şi din dreapta unui şir.
- *explode($separator, $şir)* – "rupe" valorile dintr-un şir în care ele sunt delimitate de un separator, şi le plasează într-un vector
- *implode($şir, $vector)* - preia valorile dintr-un vector şi le reuneşte într-un şir
- *number_format($număr)* – afişează valoarea numerică folosind separatorul de mii.
- *strpos($sir_princip,$sir_căutat)* - returnează poziţia în care se regăseşte şirul căutat în şirul principal.
- *substr($şir,$start,$end)* – extrage parte dintr-un şir începând din poziţia $start şi până în poziţia $end.
- *int strlen(string str)* –returnează lungimea unui şir de caractere;
- *string strstr(sirul de baza, sirul cautat)* – returnează subşirul din şirul de bază care începe cu şirul căutat (exemplu*: $email = 'abc@utm.com'; $domain = strstr($email, '@'); print $domain; // tipareste @utm.com.*).
- *string strtolower(string str)* – converteşte un şir la litere mici.
- *string strtoupper(string str)* – converteşte un şir la litere mari.
- *string ucwords(string str)* – converteşte un şir astfel încat va avea fiecare iniţiala a fiecarui cuvant scrisă cu majusculă. Restul literelor rămân neschimbate.
- *string ucfirst(string str)* – converteşte un şir astfel încât va fi scris cu iniţiala majuscula. Restul literelor ramân neschimbate.
- *int strcmp(string str1, string str2)* – compară şirul str1 cu şirul str2 din punct de vedere al codului ASCII, şi returnează urmatoarele valori întregi:<0 dacă *str1* este mai mic decat *str2*, > 0 dacă *str1* este mai mare decat *str2* şi 0 dacă sirurile sunt egale.
- *trim()* - funcţie care elimina spatiile goale de al inceputul şi sfarsitul unui sir de caractere specificat ca parametru (asemanator funcţie standard în C);

Unele funcţii PHP au argumente opţionale, care pot fi specificate sau omise, în conformitate cu intenţiile programatorului.

Când se produce o eroare în timpul execuţiei unei funcţii, PHP generează mesaje de eroare. Uneori, asemenea mesaje de eroare sunt nedorite. În acest caz, puteti suprima generarea mesajelor de eroare prin prefixarea numelui funcţie invocate cu ajutorul caracterului @. De exemplu, pentru a suprima mesajele de eroare care pot aparea în timpul execuţiei funcţie f( ), invocati aceasta funcţie dupa cum urmează: Y = @f(x);

Definirea argumentelor prestabilite

PHP va permite să definiţi funcţii cu argumente prestabilite. Dacă invocati o funcţie care are un argument prestabilit, dar nu furnizaţi nici o valoare pentru argumentul respectiv, argumentul ia o valoare prestabilită specificată. Exemplu:

```
function calcul($val , $rata = 0.01)
{
    echo "<BR>cant=$val";
    echo "<BR>rata=$rata";
    return $val * $rata;
}
$cumparaturi = 123.45;
echo "<BR>cumparaturi = $cumparaturi";
$impozit = calcul($cumparaturi,0.1);
echo "<BR>impozit = $impozit";
$cumparaturi = 123.45;
echo "<BR>cumparaturi = $cumparaturi";
$impozit = calcul($cumparaturi);
echo "<BR>impozit = $impozit";
```

Valoarea lui n este: 8
cumparaturi = 123.45
cant=123.45
rata=0.1
impozit = 12.345
cumparaturi = 123.45
cant=123.45
rata=0.01
impozit = 1.2345