

Curs TWSS

Informații generale

Module:

- PHP
- WordPress
- Laravel

Informații:

- La fiecare curs (excepție făcând primul curs) se va da o lucrare de control, notată cu maxim **0.1p**. Suma punctelor obținute se va aduna la nota finală

Evaluare:

- La sfârșitul fiecărui modul va fi notată activitatea cu câte o notă. Media acestor note va fi considerată activitatea din timpul laboratoarelor:
- **N1 = 40% din nota finală**
- Examen practic în ultima oră de laborator:
- **N2 = 60% din nota finală**
- Nota obținută la lucrările de control de la curs:
- **N3 = maxim 0.6 puncte bonus la nota finală**
- **Nota finală = N1 + N2 + N3**

Utile:

- [TextWrangler \(OSX\)](#) (editor de text și coduri pe MacOS)
- [Notepad++ \(PC\)](#) (editor de text pe Windows)
- [Sublime Text \(OSX, Windows, Linux\)](#) (editor de coduri pe toate platformele)
- [Atom \(OSX, Windows, Linux\)](#) (editor de coduri pe toate platformele)
- [XAMPP \(OSX, Windows, Linux\)](#) (server/server virtual pentru toate platformele)

Structura materiei:

1. Dezvoltarea aplicațiilor web dinamice
2. Setarea serverului web
3. Introducere în PHP
4. Manipularea formularelor în PHP
5. Expresii și instrucțiuni în PHP
6. Introducere în MySQL
7. Accesarea MySQL din PHP
8. Autentificare și managementul sesiunii în PHP
9. Funcții și obiecte în PHP
10. Șiruri în PHP
11. Lucrul cu fișiere în PHP
12. Validarea și manipularea erorilor în PHP și Javascript
13. Utilizare Ajax și PHP
14. Wordpress
15. Laravel

Dezvoltarea aplicațiilor web dinamice

Noțiuni de bază:

- **WWW – anii 1990 – rețea de întindere planetară**
Termenul World Wide Web, abreviat WWW sau și www, numit scurt și web, care în engleză înseamnă „pânză” (de păianjen) este totalitatea siturilor / documentelor și informațiilor de tip hipertext legate între ele, care pot fi accesate prin rețeaua mondială de Internet. Documentele, care rezidă în diferite locații pe diverse calculatoare server, pot fi regăsite cu ajutorul unui identificator univoc numit URL.
- **HTTP – Hyper Text Transfer Protocol**
este metoda cea mai des utilizată pentru accesarea informațiilor în Internet care sunt păstrate pe servere World Wide Web (WWW). Protocolul HTTP este un protocol de tip text, fiind protocolul "implicit" al WWW. Adică, dacă un URL nu conține partea de protocol, aceasta se consideră ca fiind http. HTTP presupune că pe calculatorul destinație rulează un program care înțelege protocolul. Fișierul trimis la destinație poate fi un document HTML (abreviație de la HyperText Markup Language), un fișier grafic, de sunet, animație sau video, de asemenea un program executabil pe server-ul respectiv sau și un editor de text
- **HTML – Hyper Text Markup Language**
este un limbaj de marcare utilizat pentru crearea paginilor web ce pot fi afișate într-un browser (sau navigator).
- **Browser web + Server web**
Browser web este o aplicație software ce permite utilizatorilor să afișeze text, grafică, video, muzică și alte informații situate pe o pagină din World Wide Web, dar și să comunice cu furnizorul de informații și chiar și ei între ei.
Serverul Web este serverul care stochează (găzduiește) pagini web și le pune la dispoziția solicitanților prin protocolul HTTP.

HTTP – HyperText Transfer Protocol

- Protocolul HTTP este un protocol situat la nivelul aplicație al stivei TCP/IP
- Acest protocol transferă hipertext sau hipermedia pe web
- Paradigma utilizată este cerere/răspuns. Acțiunile întreprinse în urma cererii făcute asupra unei resurse pot fi accesarea și modificarea resursei.
- Un mesaj de tip cerere sau răspuns este format din:
 - Linia de comandă / răspuns;
 - Linia de antet;
 - Linie blank;
 - Corp mesaj.



- Portul specific protocolului HTTP este 80. Acesta este portul implicit, însă se poate modifica. Avantajul utilizării TCP este acela, că nici programele de navigare și nici serverele nu trebuie să se preocupe de mesajele pierdute, mesajele duplicate, mesajele lungi, sau mesajele de confirmare;
- **HTTP/0.9**
Versiune simplă cu numeroase neajunsuri;
- **HTTP /1.0**
- Într-o lume în care pagina tipică Web consta în întregime din text HTML, această metodă era adecvată. În câțiva ani însă, o pagină medie Web conținea un număr mare de iconițe, imagini și alte lucruri plăcute vederii, astfel că stabilirea unei conexiuni TCP pentru a prelua o

singură iconiță a devenit un mod foarte costisitor de a opera.

- Astfel dacă un document HTML cuprinde 10 imagini, vor fi necesare 11 conexiuni TCP, pentru ca pagina să fie afișată complet (în browser).



- **HTTP /1.1**

- O singura conexiune persistentă poate fi folosită pentru mai multe cereri – răspunsuri
- Cererile pot fi transmise și în pipeline (fără a aștepta răspunsurile)
- Astfel pentru documentul HTML cu 10 imagini este necesară doar o singură conexiune TCP. Datorită algoritmului Slow-Start - viteza conexiunii TCP este la început mică, dar acum el e necesar doar o singură dată, se scurtează semnificativ durata totală de încărcare a paginii. La aceasta se adaugă și faptul că versiunea 1.1 poate relua și continua transferuri întrerupte.
- Datorita posibilității de găzduire virtuală, în antetul cererilor trebuie specificat neapărat antetul HOST



HTTPS - Secure Hyper Text Transfer Protocol

- Portul specific protocolului HTTPS este 443.
- Reprezintă protocolul HTTP încapsulat într-un flux SSL/TLS;
- Acest protocol este destinat transferului de informație criptată prin intermediul WWW.
- Datele sunt criptate la server înainte de a fi trimise clientului.
- Pași urmăriți:
 - Prima dată serverul web trimite certificatul digital la client
 - Certificatul digital conține cheia publică a serverului și identitatea serverului
 - Clientul validează certificatul
 - Se negociază o cheie de sesiune (cheie secretă)

HTTP – HyperText Transfer Protocol

Structura mesaj CERERE:

```
METODA/cale-către-resursăHTTP-versiune
Antet-a:valoare
Antet-2:valoare
...
[corpul cererii]
```

Exemplu:

```
GET/HTTP/1.1
HOST:dtic.ubbcluj.ro
```

```
GET/HTTP/1.1
HOST:cci.ubbcluj.ro
```

Structura mesaj RĂSPUNS:

```
HTTP/versiune cod-status mesaj
Antet-1:valoare
Antet-2:valoare
...
[corpul răspunsului]
```

Exemplu:

```
HTTP/1.1200OK
Content-Type:text/html
Content-Lenght:1000
...
<html>
<head>...</head>...
...
</html>
```

- Mesajul de tip cerere este format din una sau mai multe linii de text ASCII;
 - primul cuvânt din prima linie e numele metodei;
 - este semnificativă utilizarea literelor mari (**GET** și nu **get**)
 - Prima linie conține și calea către resursa și versiunea HTTP

- Liniile următoare prezintă antetul cererii HTTP
- Opțional, cererea poate să conțină și un corp prin intermediul căruia se transmite serverului și alte informații
- Mesajul de tip răspuns este format din una sau mai multe linii de text ASCII în care:
 - Prima linie prezintă versiunea HTTP, codul de stare și mesajul corespunzător, prin care se indică dacă cererea a fost satisfăcută
 - Următoarele linii prezintă antetul răspunsului HTTP și eventual corpul răspunsului.

ANTET	TIP	DESCRIERE
User-Agent	Cerere	Informație asupra programului de navigare și a platformei
Accept	Cerere	Tipul de pagini pe care clientul le poate trata
Accept-Charset	Cerere	Seturile de caractere care sunt acceptabile la client
Accept-Encoding	Cerere	Codificările de pagini pe care clientul le poate trata
Accept-Language	Cerere	Limbajele naturale pe care clientul le poate trata
Host	Cerere	Numele DNS al serverului
Authorization	Cerere	O listă a drepturilor clientului
Cookie	Cerere	Trimite un cookie setat anterior înapoi la server
Date	Ambele	Data și ora la care mesajul a fost trimis
Upgrade	Ambele	Protocolul la care tranziția vrea să comute

- Linia de cerere poate fi urmată de linii adiționale cu mai multe informații numite antet de cerere.
- Antetul *User-Agent* permite clientului să informeze serverul asupra programului său de navigare, sistemului de operare și altor proprietăți.
- Antetele *Accept* spun serverului ce este dispus clientul să accepte.
- Primul antet specifică ce tipuri MIME sunt acceptate (text/html).
- Al doilea reprezintă setul de caractere (ISO-8859).
- Al treilea se referă la metode de compresie (gzip).
- Al patrulea indică un limbaj natural (spaniola). Dacă serverul are mai multe pagini din care poate să aleagă, el poate utiliza această informație pentru a furniza clientului pagina pe care o caută. Dacă nu poate satisface cererea, este întors un cod de eroare și cererea eșuează.
- Antetul *Host* denumește serverul. El este luat din URL. Antetul este obligatoriu. Este utilizat deoarece anumite adrese IP pot servi mai multe nume de DNS și serverul are nevoie de o anumită modalitate de a spune cărui calculator să-i trimită cererea.
- Antetul *Authorization* este necesar pentru protecția paginilor. În acest caz, clientul trebuie să demonstreze că are dreptul de a vedea pagina cerută.
- Antetul *Cookie* este utilizat de clienți pentru a întoarce serverului un cookie care a fost anterior trimis de o mașină aflată în domeniul serverului.
- Antetul *Date* poate fi utilizat în ambele sensuri și conține ora și data la care a fost trimis mesajul.
- Antetul *Upgrade* este folosit pentru a face mai ușoară crearea unei tranziții către o viitoare (posibil incompatibilă) versiune a protocolului HTTP. Acesta permite clientului, să anunțe ce anume suportă, și serverului să afirme ceea ce folosește.

Reprezentări ale resursei cerute

Resursa solicitată este definită de câteva caracteristici:

- Codificarea setului de caractere:
 - ISO-8859-1, UTF-8, ... ;
- Codificarea mesajelor:

- Comprimarea și asigurarea identității și integrității:
 - gzip;
- Formatul reprezentării:
 - Text (HTML, CSS, JS, XML, text);
 - Binar (JPEG, PNG, PDF);
- Tipul conținutului resursei
 - MIME type.

Grupuri de răspunsuri ale codurilor de stare

Erorile de HTTP sunt clasificate în 5 clase.

- 1xx - erori informaționale: această clasă a status-ului indică un răspuns provizoriu al serverului
Nu sunt aplicații necesare pentru această clasă de răspuns/status. Aceste status-uri pot fi ignorate.
- 2xx - răspuns reușit: clasa de răspuns ce indică utilizatorului că cererea a fost primită, înțeleasă și acceptată cu succes.
- 3xx - redirectări: această clasă de răspuns indică faptul că acțiunile următoare vor trebui făcute de browser pentru a putea fi îndeplinită cererea.
- 4xx - erori ale utilizatorilor: această clasă de mesaje este folosită în cazurile în care utilizatorul ar putea greși formularea cererii.
- 5xx - erori de server: răspunsurile ce încep cu cifra 5 indică cazurile în care serverul e conștient de greșelile produse sau e incapabil să execute cererea.

COD	SEMNIFICAȚIE	EXEMPLE
1xx	Informare	100 = serverul acceptă tratarea cererii de la client; 101 = schimbare de protocol;
2xx	Succes	200 = cerere reușită; 202 = cerere acceptată; 204 = nu există conținut;
3xx	Redirectare	301 = pagină mutată definitiv; 302 = pagină mutată temporar; 304 = pagina din memoria ascunsă este încă validă;
4xx	Eroare la client	401 = cererea necesită autentificare; 403 = pagină interzisă; 404 = pagina nu a fost găsită;
5xx	Eroare la server	500 = eroare internă la server; 503 = încearcă mai târziu;

Antete de mesaje

ANTET	TIP	DESCRIERE
Server	Răspuns	Informație despre server
Content-Encoding	Răspuns	Cum este codat conținutul (de exemplu gzip)
Content-Language	Răspuns	Limbajul natural utilizat în pagină
Content-Length	Răspuns	Lungimea paginii în octeți
Content-Type	Răspuns	Răspunsul MIME al paginii
Last-Modified	Răspuns	Ora și data la care pagina a fost ultima dată modificată
Location	Răspuns	O comandă pentru client pentru a trimite cererea în altă parte
Accept-Ranges	Răspuns	Serverul va accepta cereri în anumite limite de octeți
Set-Cookie	Răspuns	Serverul vrea să salveze un cookie la client

- Similar cu cererile si raspunsul poate fi urmat de antet de raspuns.
- Antetul *Server* permite serverului să spună cine este și câteva proprietăți, dacă dorește.
- Antetele *Content* permit serverului să descrie proprietățile paginii pe care o transmite.
- Antetul *Last-Modified* spune când a fost modificată ultima dată pagina. Acest antet joacă un rol important în mecanismul de memorie ascunsă.
- Antetul *Location* este utilizat de server pentru a informa clientul că ar trebui să utilizeze un alt URL.
- Este utilizată pentru companiile care au o pagină de Web principală în domeniul *com*, dar care redirecționează clienții la o pagină națională sau regională în funcție de adresa lor IP sau limba preferată.
- Dacă o pagină este foarte mare, un client mic poate nu o dorește dintr-o dată. Unele servere acceptă cereri în anumite intervale de octeți, astfel că pagina poate fi citită în mai multe unități mai mici. Antetul *Accept-Ranges* anunță asentimentul severului de a trata acest tip de cerere parțială de pagini.
- Al doilea antet pentru cookie, *Set-Cookie*, se referă la modul în care serverele trimit cookie-uri la clienți. Este de așteptat salvarea cookie-ului de către client și returnarea acestuia la cereri ulterioare ale serverului.

Content – Type:

- Permite transferul datelor de orice tip:
 - text/plain;
 - text/html;
 - text/css;
 - image/png;
 - audio/mpegapplication/javascript;
 - application/json.

Location:

- http:// domeniu [: port] [cale absolută];
- Redirecționează clientul spre o altă reprezentare a resursei;
- [Exemplu Redirect.](#)

Referer:

- Desemnează URL-ul resursei Web care a referit resursa curentă;
- [Exemplu Referer.](#)

Host:

- Specifică adresa IP a mașinii de pe care se solicită accesul la o resursă;
- [Exemplu Host.](#)

Metode de cerere standard pentru HTTP

METODA	DESCRIERE
GET	Tere de citire a unei pagini Web
Post	Trimite date de intrare către server
ÎN CONTEXTUL SERVICIILOR WEB	
HEAD	Cerere de trimitere a antetului unei pagini Web
PUT	Cerere de depunere a unei pagini Web
DELETE	Ștergerea unei pagini de Web
TRACE	Transmite în ecou cererea care a sosit
CONNECT	Rezervat pentru o utilizare în viitor
OPTIONS	Interogarea anumitor opțiuni

- Uzual, browser-ul Web permite doar folosirea metodelor GET și POST.
- Metodele GET și POST sunt metodele utilizate de obicei de către browserele web.
- Există și metode care inițial nu au avut o utilizare, iar ulterior au primit utilitate în contextul serviciilor web.
- Acestea sunt HEAD, PUT, DELETE, TRACE, CONNECT și OPTIONS.
-

Metode sigure	Metode nesigure
GET	POST
HEAD	PUT
	DELETE

- Metodele de cerere HTTP sunt clasificate în metode care sunt sigure și metode care nu sunt sigure, în funcție de modificările pe care le fac asupra server-ului.
- Metodele GET și HEAD sunt sigure, iar metodele POST, PUT și DELETE nu sunt sigure.

Exemplu de utilizare HTML

```
[diana@cc ~/concurs]$ telnet cs.ubbcluj.ro 80
Trying 193.0.225.34...
Connected to cs.ubbcluj.ro.
Escape character is '^]'.
GET / HTTP/1.1
HOST: cs.ubbcluj.ro

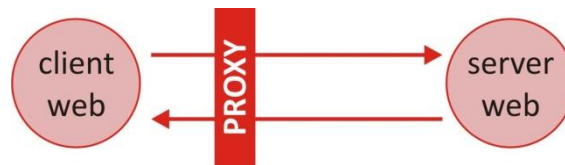
HTTP/1.1 301 Moved Permanently
Date: Mon, 30 Jan 2017 10:43:30 GMT
Server: Apache/2.2.15 (CentOS)
Location: http://www.cs.ubbcluj.ro/
Content-Length: 233
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>301 Moved Permanently</title>
</head><body>
<h1>Moved Permanently</h1>
<p>The document has moved <a href="http://www.cs.ubbcluj.ro/">here</a>.</p>
</body></html>
Connection closed by foreign host.
```

- Această secvență de comenzi pornește o conexiune telnet (adică TCP) pe portul 80 al serverului Web *cs.ubbcluj.ro*.
- Apoi urmează comanda *GET* denumind fișierul și protocolul.
- Următoarea linie este antetul obligatoriu *Host*.
- Linia rămasă liberă este de asemenea cerută.
- Ea semnalează serverului că nu mai sunt antete de cerere. Comanda *close* indică programului de telnet să întrerupă conexiunea.
- Primele trei linii reprezintă ieșirea programului telnet, și nu de la serverul la distanță.
- Linia ce începe cu HTTP/1.1 este răspunsul prin care serverul spune că este dispus să vorbească HTTP/1.1 cu tine.
- Apoi urmează un număr de antete și apoi conținutul.

Proxy

- Localizat în proximitatea clientului/serverului are rol atât de server, cât și de client.
-



- Un proxy HTTP este un program care joacă rolul unui intermediar între client și server.
- El primește cereri de la clienți și le trimite mai departe serverului.
- Răspunsurile ajung înapoi la client prin același mecanism. Prin urmare, un proxy funcționează atât ca și client cât și ca server.

Cache

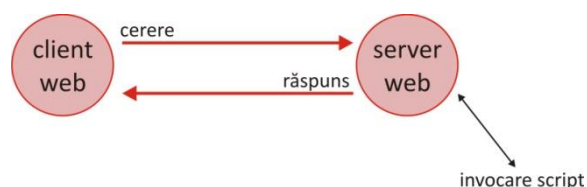
- Memoria ascunsă este o zonă locală de stocare a datelor la nivel de server, respectiv client.
- Scopul acesteia este de a asigura performanța aplicațiilor web.
- În cazul existenței unui proxy atunci salvarea sau aducerea resursei din cache poate fi făcută la nivelul acestuia.
 - Dacă proxy-ul are pagina o returnează imediat;
 - Dacă nu, aduce pagina de la server, o adaugă în cache pentru utilizarea ulterioară și o întoarce clientului care a cerut-o.

Instrumente

- [Google Chrome Developer Tools](#)
- [Firefox Developer Tools](#)
- [Extensia Firebug](#) (la nivel de client; JavaScript)

Dezvoltarea de aplicații web

- Necesitate:
 - generarea dinamică la nivel de server de reprezentări ale unor resurse solicitate de clienții Web;
- Soluții:
 - CGI – interfață de programare, independentă de limbaj, facilitând interacțiunea dintre clienți și programe invocate la nivel de server;
 - Serverul va invoca scriptul CGI pasându-i datele la intrarea standard sau via variabile de mediu;
- Cererile adresate serverului Web sunt logate.
-



- Pentru a exemplifica structura cererilor și răspunsurilor vom utiliza CGI, care deși este învechit, este din punct de vedere didactic un instrument foarte util.
- În exemplele noastre clientul va adresa o cerere serverului, iar serverul va invoca scriptul cgi pasându-i datele la intrarea standard sau prin intermediul variabilelor de mediu.
- Toate cererile adresate serverului web sunt logate.

Exemplu antet cerere

```
#include <stdio.h>

int
main ()
{
    printf ("Content-type: text/html\n");
    printf ("\n");
    printf ("Hello world!");
}
```

- Exemplu este scris în C, și compilat ca un script cgi.

-

×	Headers	Preview	Response	Cookies	Timing
▼	General				
	Request URL:	http://www.scs.ubbcluj.ro/~diana.sotropa/cgi-bin/dl.cgi			
	Request Method:	GET			
	Status Code:	200 OK			
	Remote Address:	193.226.40.130:80			
▼	Response Headers	view source			
	Connection:	close			
	Content-Type:	text/html; charset=UTF-8			
	Date:	Mon, 30 Jan 2017 18:58:03 GMT			
	Server:	Apache/2.2.15 (CentOS)			
	Transfer-Encoding:	chunked			
▼	Request Headers	view source			
	Accept:	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8			
	Accept-Encoding:	gzip, deflate, sdch			
	Accept-Language:	ro-RO,ro;q=0.8,en-US;q=0.6,en;q=0.4,es;q=0.2,id;q=0.2,ms;q=0.2,de;q=0.2,fr;q=0.2			
	Connection:	keep-alive			
	Cookie:	_ym_uid=1470139771198424854; kvcd=1470139772630; km_ai=UrH3gO7w6DeYzOR0Iyx%2BpiYs4F4%3D; km_lv=x; km_uq=; __unam=9a23bed-1540a0b0a7f-532e6880-82; __utma=223269727.2103271345.1457420654.1476165956.1478181808.205; __utmc=223269727; __utmz=223269727.1475072397.197.8.utmcsr=testadmitere.ubbcluj.ro utmccn=(referral) utmcmd=referral utmcct=/errors/404.php; _gat=1; _ga=GA1.2.2103271345.1457420654			
	Host:	www.scs.ubbcluj.ro			
	Upgrade-Insecure-Requests:	1			
	User-Agent:	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/55.0.2883.87 Safari/537.36			

- În urma apelului acestui script din browser, se poate observa ca ceea ce s-a transmis în antetul cererii server side poate fi vizualizat folosind Developer Tools client-side.
- Deoarece nu a fost specificată dimensiunea corpului de text ce va fi trimis, antetul Transfer-Encoding a fost setat la chunked.

```
#include <stdio.h>

int
main ()
{
    printf ("Content-type: text/html\n");
    printf ("Content-length: 12\n");
    printf ("\n");
    printf ("Hello world!");
}
```

```

▼ General
Request URL: http://www.scs.ubbcluj.ro/~diana.sotropa/cgi-bin/d1-2.cgi
Request Method: GET
Status Code: 200 OK
Remote Address: 193.226.40.130:80

▼ Response Headers view source
Connection: close
Content-length: 12
Content-Type: text/html; charset=UTF-8
Date: Tue, 31 Jan 2017 17:19:08 GMT
Server: Apache/2.2.15 (CentOS)

▼ Request Headers view source
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp
Accept-Encoding: gzip, deflate, sdch
Accept-Language: ro-RO,ro;q=0.8,en-US;q=0.6,en;q=0.4,es;q=0.2,id;q=0.2
Connection: keep-alive
Cookie: __utma=223269727.325436979.1459326301.1465895101.1467880149.6;
26301
Host: www.scs.ubbcluj.ro
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
Chrome/55.0.2883.87 Safari/537.36

```

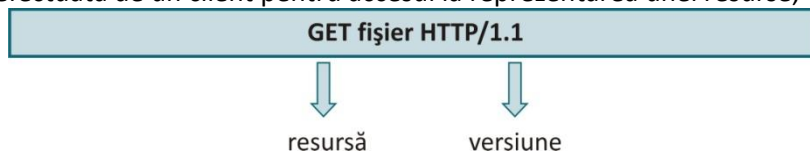
- În mod similar, odata cu setarea lungimii conținutului trimis, antetul Transfer-Encoding dispăre.

Variabile

- Variabile de mediu (*specifice cererilor transmise spre programul CGI*):
 - REQUEST_METHOD;
 - QUERY_STRING;
 - REMOTE_HOST, REMOTE_ADDR;
 - CONTENT_TYPE;
 - CONTENT_LENGTH;
- Variabile suplimentare (*generate uzual de serverul web*):
 - HTTP_ACCEPT;
 - HTTP_COOKIE;
 - HTTP_HOST;
 - HTTP_USER_AGENT.
 - HTTP_REFERER

Metode de cerere standard pentru HTTP: GET

- Cerere efectuată de un client pentru accesul la reprezentarea unei resurse;



- Fișier:
 - HTML, CSS, JS, XML, JSON, PDF, PNG, SVG, ...
- Procesarea datelor:
 - Date disponibile în variabila de mediu QUERY_STRING.

Încapsularea parametrilor trimiși din browser prin GET

Pentru fiecare câmp al formularului, se generează o pereche nume_câmp=valoare delimitată de & ce va fi adăugată URL-ului unde e stocat programul

```
<!DOCTYPE html>
<html>
<head>
<title>Query String Demo</title>
</head>
<body>
<form method="GET" action="d2.cgi">
<input type="text" name="nume" placeholder="Nume"><br>
<input type="text" name="telefon" placeholder="Telefon"><br>
<input type="text" name="varsta" placeholder="Varsta"><br>
<input type="submit" value="Trimite">
</form>
</body>
</html>
```

▼ General

Request URL: http://www.scs.ubbcluj.ro/~diana.sotropa/cgi-bin/d2.cgi?nume=Diana&telefon=1234&varsta=25
Request Method: GET
Status Code: 200 OK
Remote Address: 193.226.40.130:80

▼ Response Headers [view source](#)

Connection: close
Content-Type: text/html; charset=UTF-8
Date: Mon, 30 Jan 2017 19:05:43 GMT
Server: Apache/2.2.15 (CentOS)
Transfer-Encoding: chunked

▼ Request Headers [view source](#)

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp
Accept-Encoding: gzip, deflate, sdch
Accept-Language: ro-RO,ro;q=0.8,en-US;q=0.6,en;q=0.4,es;q=0.2,id;q=0.2
Connection: keep-alive
Cookie: _ym_uid=1470139771198424854; kvcd=1470139772630; km_ai=UrH3gO7=x; km_uq=; __unam=9a23bed-1540a0b0a7f-532e6880-82; __utma=223269727.6.1478181808.205; __utmc=223269727; __utms=223269727.1475072397.197.8 utmccn=(referral)|utmcmd=referral|utmctt=/errors/404.php; _gat=1; _ga
Host: www.scs.ubbcluj.ro
Referer: http://www.scs.ubbcluj.ro/~diana.sotropa/cgi-bin/d2.html
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.5.0.2883.87 Safari/537.36

▼ Query String Parameters [view source](#) [view URL encoded](#)

nume: Diana
telefon: 1234
varsta: 25

Metode de cerere standard pentru HTTP: POST

- Similară metodei PUT;
- Se utilizează atunci când datele transmise serverului au dimensiuni mari sau sunt delicate;
- Creează o resursă, trimițând uzual entități (date, acțiuni) spre server;
- Procesarea datelor:
 - Datele vor fi preluate de la INTRAREA STANDARD, lungimea în octeți a acestora fiind specificată de variabila CONTENT_LENGTH .

Metode de cerere standard pentru HTTP: HEAD

- Similară cu GET;
- Cere doar antetul mesajului, fără să ceară și pagina propriu-zisă;
- SCOP:
 - Aflarea informațiilor despre ultima modificare;

- Obținerea informațiilor pentru indexare;
- Verificarea corectitudinii unui URL.
-

```
[diana.sotropa@linux cgi-bin]$ telnet dtic.ubbcluj.ro 80
Trying 193.231.20.40...
Connected to dtic.ubbcluj.ro.
Escape character is '^]'.
HEAD /test.html HTTP/1.1
Host: dtic.ubbcluj.ro

HTTP/1.1 200 OK
Date: Tue, 31 Jan 2017 14:01:26 GMT
Server: Apache
Last-Modified: Tue, 31 Jan 2017 14:01:11 GMT
ETag: "19-547645d4a37c0"
Accept-Ranges: bytes
Content-Length: 25
Content-Type: text/html
```

Metode de cerere standard pentru HTTP: PUT

- Inversă metodei GET – în loc să citească o pagină, o scrie;
- Actualizează o reprezentare de resursă sau eventual creează o resursă la nivel de server Web;
- Autentificarea și drepturile de acces joacă un rol important;
- Corpul cererii conține pagina.

Metode de cerere standard pentru HTTP: DELETE

- Opusul metodei PUT;
- Șterge o resursă/ reprezentarea ei de pe server;
- Autentificarea și drepturile de acces joacă un rol important;
- Nu există nici o garanție asupra succesului operației *DELETE*, deoarece chiar dacă serverul dorește să execute ștergerea, fișierul poate să aibă attribute care să interzică serverului HTTP să îl modifice sau să îl șteargă;

Metode de cerere standard pentru HTTP: TRACE

- Verificarea corectitudinii;
- Cere serverului să trimită înapoi cererea;
- SCOP:
 - Dacă cererile nu sunt procesate corect și clientul vrea să știe ce fel de cerere a ajuns de fapt la server;
- Este o metodă folosită de obicei pentru diagnosticare, putând da mai multe informații despre traseul urmat de legătura HTTP, fiecare server proxy adăugându-și semnătura în antetul Via.

Caracterul state-less al protocolului HTTP

- Deservește cereri multiple provenite de la clienți pe baza protocolului HTTP;
- Fiecare cerere e considerată independentă de alta, chiar dacă provine de la același client Web:
 - Nu e păstrată starea conexiunii: **stateless**;
- Necesitate:
 - Păstrarea datelor de-a lungul mai multor accesări succesive ale aceluiași client;
 - Asocierea vizitatorilor site-ului cu un identificator unic (**session ID**).



- HTTP nu poate oferi informații dacă anumite cereri succesive provin de la același client. exemple: starea coșului de cumpărături, formulare Web completate în mai mulți pași, paginarea conținutului, starea autentificării utilizatorului etc.
- Session id stocat într-un cookie (e.g., ASP.NET_SessionId, PHPSESSID, session-id, _wp_session) ori propagat via URL , astfel se pot identifica vizite (cereri) consecutive realizate de același utilizator
- Sau https – sesiunea se mentine pe baza cheii de sesiune negociate de cei doi parteneri

Cookie. Definiție

- O înregistrare de descriere a unui fișier UNIX sau un identificator al unui obiect Windows;
- Un *fișier* (sau *șir de caractere*) de dimensiune mică (cel mult 4 KB);
- Mecanism standard ce permite ca un server Web să plaseze date pe calculatorul-client, prin intermediul browser-ului, pentru ca, ulterior, navigatorul să returneze acele date aceluiași server;
- Mijloc persistent de stocare a datelor pe mașina clientului Web cu scopul de a fi apoi accesate de un program rulând pe server;
- Memorarea preferințelor fiecărui utilizator.

Exemple tipice: opțiuni vizând interacțiunea – temă vizuală (e.g., cromatică), preferințe lingvistice etc. localizare geografică, interese privind cumpărăturile

Cookie. Utilizări

- Completarea automată a formularelor;
- Monitorizarea accesului la o resursă Web;
- Stocarea informațiilor de autentificare;
- Starea tranzacțiilor în cadrul unei aplicații Web;
- Managementul sesiunilor web.

Chiar înainte ca un program de navigare să transmită cererea pentru o pagină către un sit Web, el verifică directorul de cookie-uri pentru a vedea dacă există vreun cookie care a fost stocat de către domeniul la care se duce cererea. În caz afirmativ, toate cookie-urile stocate de către acel domeniu sunt incluse în mesajul ce conține cererea. Atunci când serverul le obține, le poate interpreta în orice mod dorește.

Aspect de interes: Web analytics colectarea de informații despre clienți (platformă hardware, browser, rezoluție etc.)

Aspect de interes: user tracking monitorizarea comportamentului utilizatorului

Reținerea datelor privitoare la contul utilizatorului în contextul comerțului electronic

Starea coșului de cumpărături în cadrul unui magazin virtual (e-shop)

Cookie. Clasificare

- Cookie-uri persistente:
 - Nu vor fi distruse la închiderea navigatorului Web, ci vor fi memorate într-un fișier, perioada lor de viață fiind stabilită de creatorul cookie-urilor;
- Cookie-uri nepersistente:
 - Dispar la închiderea browser-ului.

Cookie

Set-Cookie: nume=valoare;expires=data;path=cale;domain=domeniu;secure

- Trimitere Cookie:
 - Folosind câmpul **Set-Cookie** dintr-un antet al unui mesaj de răspuns HTTP;

DOMAIN	PATH	CONTENT	EXPIRES	SECURE
Domeniu 1	/	CustomerID=497793521	15-10-02 17:00	DA
Domeniu 2	/	Cart1=1-00501;1-07031;2-13721	11-10-02 14:22	NU
Domeniu 3	/	Prefs=Stk:SUNW+ORCL;Spt:Jets	31-12-10 23:59	NU
Domeniu 4	/	UserID=3627239101	31-12-12 23:59	NU

- **DOMAIN:**
 - Semnifică numele simbolic al serverului Web care a generat cookie-ul;
- **PATH:**
 - Specifică un subset de URL-uri din domeniul corespunzător unui cookie diferențiază aplicații multiple existente pe același server;
- **CONTENT**
 - Câmpul în care se stochează conținutul unui cookie;
- **EXPIRE:**
 - Indică data și timpul când cookie-ul va expira, iar clientul Web îl va distruge;
- **SECURE:**
 - Acest cookie va fi transmis doar în cazul în care canalul de comunicație este „sigur” (via HTTPS).

Set-Cookie: nume=valoare;expires=data;path=cale;domain=domeniu;secure

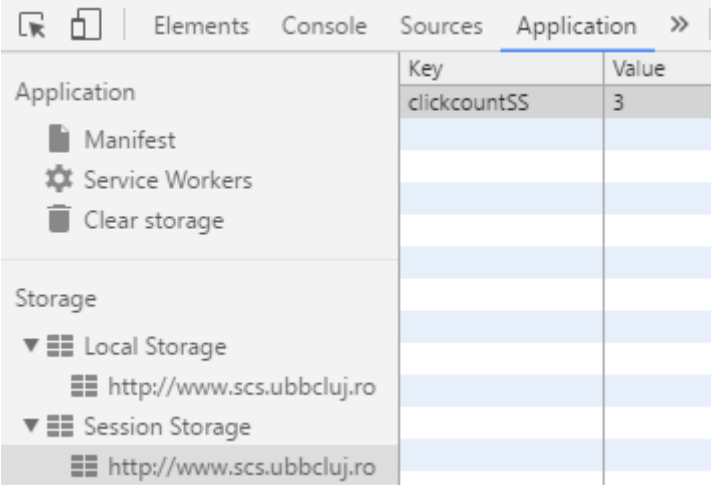
1. Câmpul *Domeniu* spune de unde a sosit cookie-ul;
2. Câmpul *Cale* reprezintă o cale în structura de directoare a serverului care identifică ce parte a arborelui de fișiere de pe server poate utiliza cookie-ul respectiv;
“/” = întregul arbore
3. Câmpul *SECURE* poate indica dacă programul de navigare poate transmite cookie-ul numai unui server sigur.
Acest element este utilizat pentru comerțul electronic, aplicații bancare și alte aplicații sigure.
4. Câmpul *Expiră* arată când expiră un cookie. Dacă acest câmp este absent, programul de navigare șterge cookie-ul la terminarea execuției. Un astfel de cookie se numește **cookie ne-persistent**. Dacă se oferă o dată și o oră, cookie-ul se numește **persistent** și este păstrat până la expirare. Pentru a șterge un cookie de pe discul unui client, un server retransmite cookie-ul cu timpul de expirare în trecut.

Cookie: nume1=valoare1; nume2=valoare2;...

- Transmitere cookie de la client:
 - Doar dacă îndeplinește toate condițiile de validitate, antetul mesajului HTTP va conține o linie de forma:

Web Storage

- **HTML5 oferă Web Storage**
 - **stocare la nivel de browser a unor liste de perechi de forma cheie - valoare:**
 - **sessionStorage:**
 - **Stocare nepersistentă (suport pentru sesiuni);**
 - **localStorage:**
 - **Alternativă la cookie-uri;**
 - **Nu au timp de viață stabilit a-priori;**
 - **Datele memorate sunt disponibile numai la nivel local (browser);**



	Key	Value
Application	clickcountSS	3
Manifest		
Service Workers		
Clear storage		
Storage		
▼ Local Storage		
http://www.scs.ubbcluj.ro		
▼ Session Storage		
http://www.scs.ubbcluj.ro		

Setarea serverului web

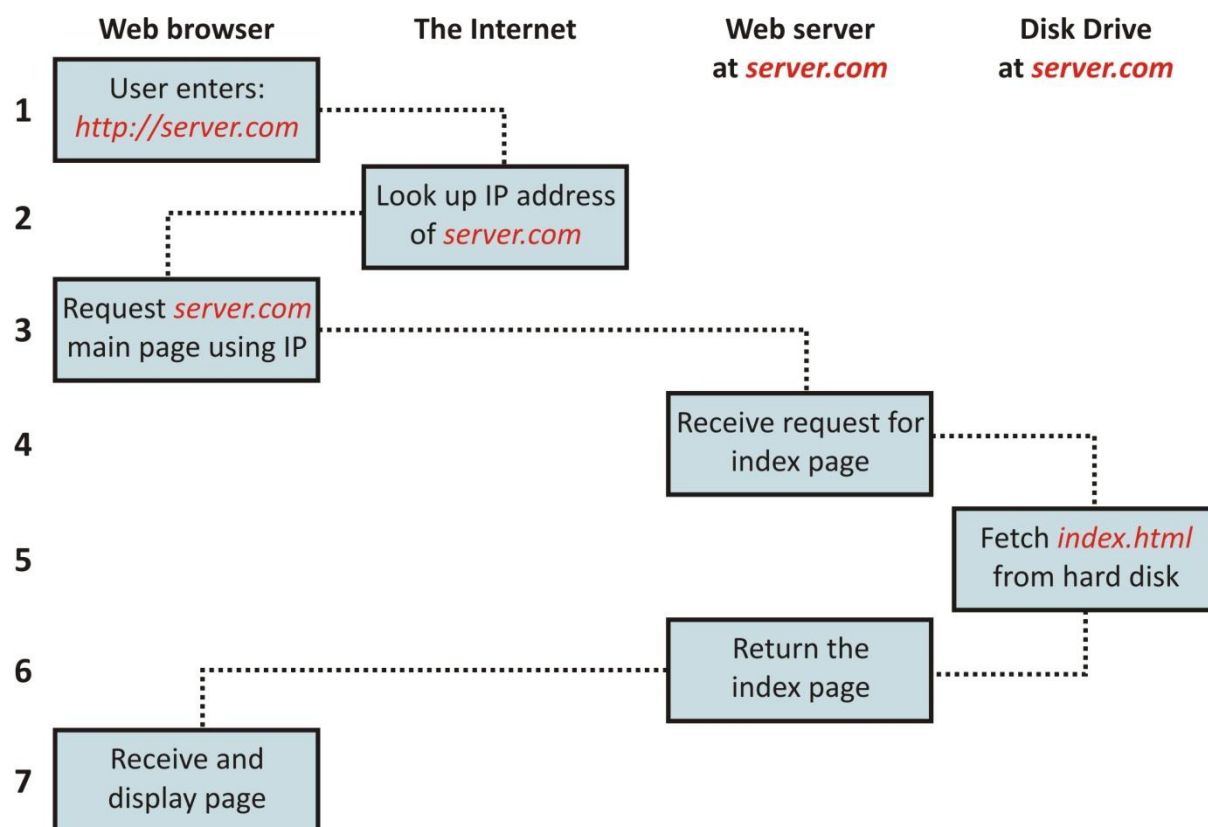
Server Web

- Dezvoltarea unei aplicații web PHP necesită:
 - Instalare Apache (sau IIS), instalare PHP și MySQL
 - Instalare Wampserver (PHP, Apache, MySql server) pe mașina locală
- Download
 - PHP: <http://www.php.net/downloads.php>
 - Mysql: <https://www.mysql.com/downloads/>
 - Apache: <http://httpd.apache.org/download.cgi>

Serverul web APACHE

- Manipulează diferite tipuri de fișiere: HTML, imagini, fișiere Flash, fișiere MP3, RSS, etc.

HTTP & HTML



Server web

- WAMP – “Windows, Apache, MySQL and PHP”
- MAMP – “Mac, Apache, MySQL and PHP”
- LAMP – “Linux, Apache, MySQL and PHP”

<http://127.0.0.1>

<http://localhost>

- C:/xampp/htdocs– folderul în care se pun informațiile care se doresc a fi accesate

- De exemplu, pentru primul laborator se creeaza folderul php si in el se pun fisierele asociate

localhost/php

- **VIRTUAL HOSTS**
 - Editați fișierul C:/xampp/apache/conf/extra/httpd-vhosts.conf

```
<VirtualHost *:80>
DocumentRoot "c:/xampp/htdocs/php"
ServerName php.twss
<Directory "c:/xampp/htdocs/php">
</Directory>
</VirtualHost>
```

- **VirtualHost:** Majoritatea serverelor web au setat portul 80 ca port default.
- **DocumentRoot:** Acesta definește folderul unde se vor afla fișierele corespunzătoare site-ului.
- **ServerName:** Reprezintă URL-ul asociat host-ului virtual
- **Directory:** Reprezintă directorul host-ului virtual

Informații:

<https://www.cloudways.com/blog/configure-virtual-host-on-windows-10-for-wordpress/>

- **VIRTUAL HOSTS**
Editați fișierul C:\Windows\System32\drivers\etc\hosts

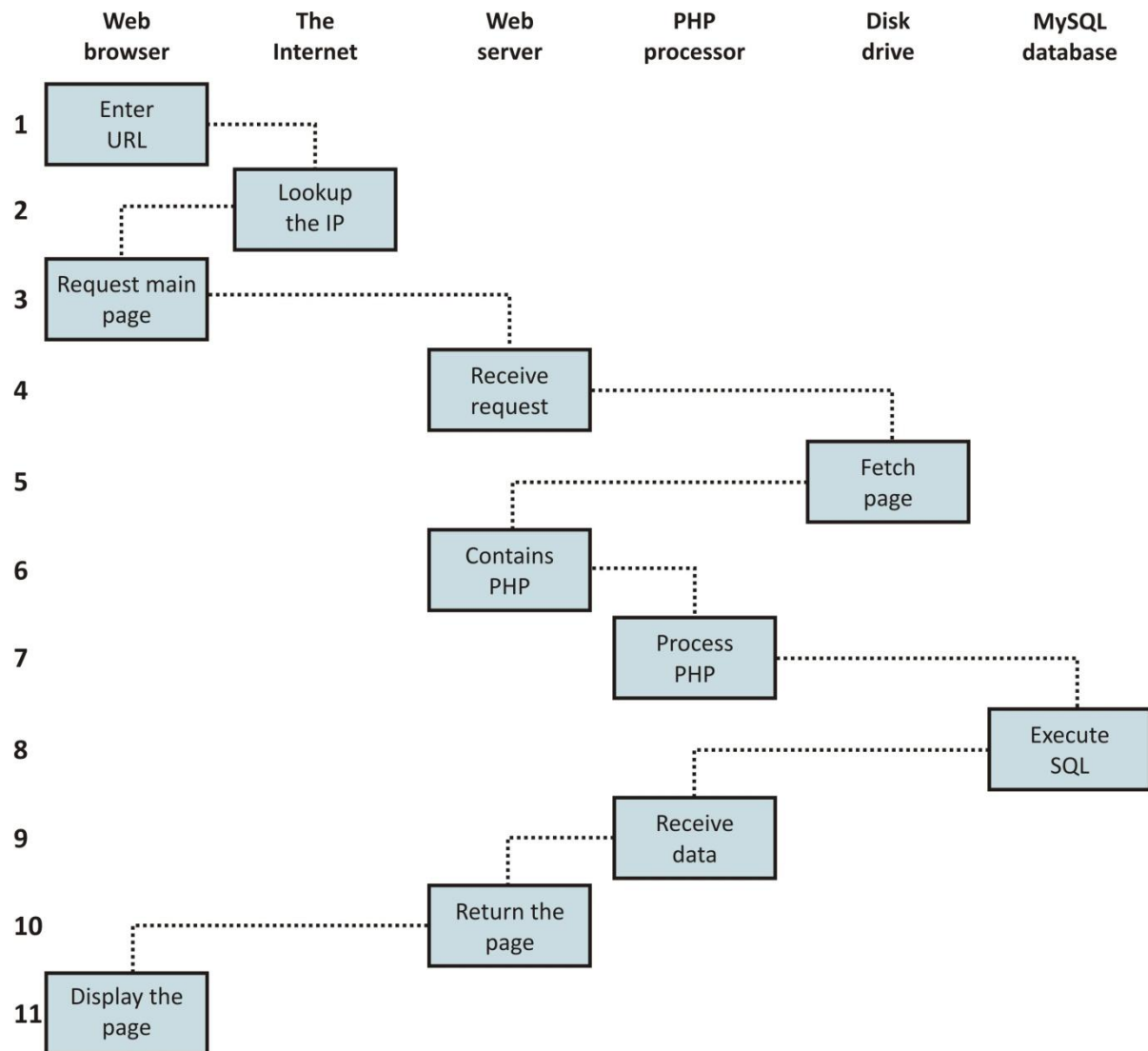
127.0.0.1 php.twss

- **Restart la Apache**

http://php.twss

Introducere în PHP

HTTP & PHP & MySQL



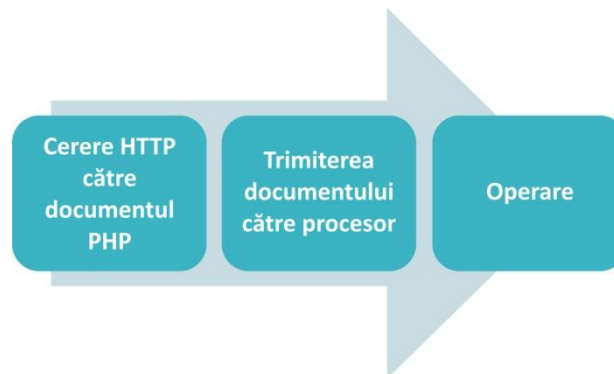
PHP script

- Script – colecție de programe sau secvență de instrucțiuni care sunt interpretate sau executate de către un alt program, altul decât procesorul calculatorului
- Script Client Side: scriptul se rulează în browser (ex.: JavaScript)
- Script Server Side: scriptul se rulează pe server (ex.: PHP, ASP)

Avantaje script Server Side:

- Acces la baze de date și sistem de fișiere
- Control asupra compatibilității platformelor
- Acces la funcții și librării
- Acces la rețea
- Conținut dinamic
- Securitate

Cum funcționează PHP-ul?



Operare

- Copierea codului HTML + Afișare
- Interpretarea codului PHP

Avantaj

- Clientul nu vede niciodată php-ul, ci numai pagina generată de cod

Ce este PHP-ul?

- PHP: Hypertext Preprocessor
- Limbaj de programare server side open source
- Script-urile PHP se execută pe server
- Suportă utilizarea multiplelor baze de date
- Extensie: .php, .php3, .phtml
- Codul se încapsulează între <?php și ?>

Exemple:

```
<html>
  <head>
  </head>
  <body>
    <p>
      <?php
        echo "My first line of PHP!";
      ?>
    </p>
  </body>
</html>
```

My first line of PHP!

```
<html>
  <head>
  </head>
  <body>
    <p>
      <?php
        echo "Hello," . " " . "world" . "!";
      ?>
    </p>
  </body>
</html>
```

Hello, world!

Comentarii in PHP

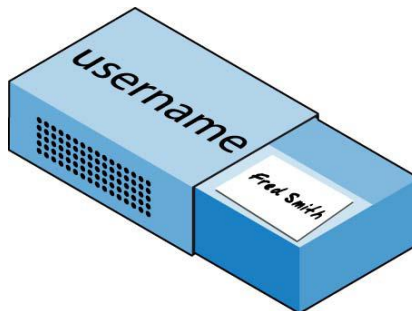
```
<?php // This is a comment?>
<?php # This is a comment?>
<?php
/* This is a section
of multiline comments
which will not be
interpreted */
?>
```

Rădăcini: C, Perl, Java

- Reguli:
 - ; – la finalul fiecărei comenzi
 - \$ – în fața numelor de variabile
 - { } – încapsularea mai multor comenzi
 -

```
<?php
$mycounter = 1;
$mystring = "Hello";
$myarray = array("One", "Two", "Three");
?>
```

Variabile PHP



Închipuiți că aveți o cutie de chibrituri pe care este scris *username*. Luați o bucată de hârtie, scrieți pe el Fred Smith, și puneți hârtia în cutie. Așa este și procesul de a aloca o valoare tip string unei variabile.

```
<?php
$username = "Fred Smith";
echo $username;
$current_user = $username;
?>
```

Fred Smith

```
<?php
$username = "Fred Smith";
echo $username;
echo "<br />";
$current_user = $username;
```

```
echo $current_user;  
?>  
Fred Smith  
Fred Smith
```

Reguli:

- Numele variabilelor trebuie să înceapă cu o literă sau _
- Numele variabilelor pot conține: a-z, A-Z, 0-9 sau _
- Numele variabilelor nu pot conține spații
- Numele variabilelor sunt case sensitive

Exemple de nume de variabile corecte:

- \$count
- \$_Obj
- \$A123

Exemple de nume de variabile incorecte:

- \$123
- \$*ABC

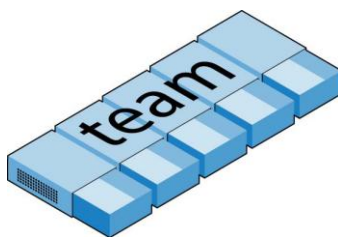
Managementul variabilelor

- **isset()** – verifică dacă o variabilă a fost declarată
- **unset()** – golește memoria ocupată prin declararea și folosirea variabilei
- **empty()** – verifică dacă o variabilă nu a fost declarată sau nu conține informații

Variabilele pot fi accesate prin referențiere indirectă :

```
<?php  
$name = "John";  
$$name = "Registered user";  
print $John;  
?>  
Registered user
```

Șiruri de variabile



Închipuiți la exemplul de mai înainte, că luați cinci cutii de chibrite, și le lipiți una de alta. Puneți o hârtie cu un nume în fiecare cutie. Așa funcționează șirurile în PHP.

```
<?php  
$team = array('Bill', 'Joe', 'Mike', 'Chris', 'Jim');  
echo $team[3];  
?>  
Chris
```

Matrici de variabile



Matricea este de fapt varianta bidimensională a șirului. Închipuiți, că vreți să țineți evidența la jocul X-O. Luați nouă cutii de chibrituri, lipiți-le laolaltă, câte trei pe un rând, trei rânduri una-peste alta. Așa funcționează matricile în PHP.

```
<?php
$oxo = array( array('x', '', 'o'),
               array('o', 'o', 'x'),
               array('x', 'o', '' ));
echo $oxo[1][2];
?>
x
```

PHP – tipuri de date

- **Numere întregi**
 - 32 biți, cu semn
 - -2147483648, +2147483647
 - Întregii pot fi scriși în baza 16 (prefixare cu 0x) sau în baza 8 (prefixare cu 0) și pot include unul din semnele +/-
- *Exemple*
 - 240000
 - 0xABCD
 - 007
 - -100
- **Numere reale**
 - 8 octeți
 - 2.2E-308, 1.8E+308
 - Numere reale pot include ., unul din semnele +/- și o valoare exponențială
- *Exemple*
 - 3.14
 - +0.9e-2 (0.09 = 9*10[^](-2))
 - -170000.5
 - 54.6E42
- **String**
 - Șir de caractere
- *Exemple*
 - "PHP: Hypertext Pre-processor"
 - "GET / HTTP/1.0\n"
 - "1234567890"
 - 'Hello, World'
 - 'Today\'s the day'

- **Caractere speciale**
 - \n, \t, \", \', \\
 - \0 (null)
 - \r (line feed)
 - \\$ (caracterul \$)

PHP - String

Pentru concatenarea șirurilor de caractere se folosește .

```
<?php
$msgs = 5;
echo "You have " . $msgs . " messages.";
?>
```

You have 5 messages

Funcții pentru prelucrarea șirurilor de caractere

strlen() - lungimea unui șir de caractere

```
<?php
echo strlen("Hello world!");
?>
```

12

strpos() - căutarea unui subșir sau a unui caracter într-un șir de caractere

```
<?php
echo strpos("Hello world!", "world");
?>
```

6

Două tipuri:

- ‘ ‘
- “ ”

```
<?php
$variable="qwertyuiopasdfghjklzxcvbnm";
$info = 'Preface variables with a $ like this: $variable';
$info = "Preface variables with a $ like this: $variable";
?>
```

Preface variables with a \$ like this: \$variable
Preface variables with a \$ like this: qwertyuiopasdfghjklzxcvbnm

PHP - Escaping characters

```
<?php
$text = 'My sister's car is a Ford'; // Eroare de sintaxă
?>
```

```
<?php
$text = 'My sister\'s car is a Ford';
$text = "My Mother always said \"Eat your greens\".";
$heading = "Date\tName\tPayment";
?>
```

```
My sister's car is a Ford
My Mother always said "Eat your greens".
Date Name Payment
```

PHP - Multi-line commands

```
<?php
$author = "Alfred E Newman";
echo "This is a Headline
This is the first line.
This is the second.
Written by $author.";
?>
```

```
This is a Headline This is the first line. This is the second. Written by Alfred E Newman.
```

```
<?php
$author = "Alfred E Newman";
$text = "This is a Headline
This is the first line.
This is the second.
Written by $author.";
echo $text;
?>
```

```
This is a Headline This is the first line. This is the second. Written by Alfred E Newman.
```

```
<?php
$author = "Alfred E Newman";
echo <<<_END
This is a Headline
This is the first line.
This is the second.
- Written by $author.
_END;
?>
```

```
This is a Headline This is the first line. This is the second. - Written by Alfred E Newman.
```

```
<?php
$author = "Alfred E Newman";
$out = <<<_END
This is a Headline
This is the first line.
This is the second.
- Written by $author.
_END;
echo $out;
?>
```


This is a Headline This is the first line. This is the second. - Written by Alfred E Newman.

PHP - Constante

`define("CONSTANT_NAME", value [,case_sensitivity = false])`

```
<?php
define("ROOT_LOCATION", "/usr/local/www/");
$directory = ROOT_LOCATION;
echo $directory;
?>
/usr/local/www/
```

PHP - Constante predefinite

__LINE__	Numărul liniei curente
__FILE__	Calea absolută și numele fișierului
__DIR__	Directorul în care se află fișierul <code>dirname(__FILE__)</code> .
__FUNCTION__	Numele funcției
__CLASS__	Numele clasei
__METHOD__	Numele metodei

```
<style>
*{box-sizing:border-box;}
.right,.left{width:calc(50% - 22px);float:left;border:1px solid #ccc;margin:0 10px;padding:10px;}
.right{background:#00bcd4;color:#fff;}
</style>
<h1>Exemplu</h1>
<div class="left">
<h2>Cod</h2>
<pre><code>&#x3C;?php
echo &#x22;This is line &#x22; . __LINE__ . &#x22; of file &#x22; . __FILE__;
?&#x3E;

</code></pre>
</div>
<div class="right">
<h2>Efect</h2>
<?php
echo "This is line " . __LINE__ . " of file " . __FILE__;
?>

</div>
This is line 35 of file /home/info/diana.sotropa/public_html/files/TWSS/0exemple/ex (24).php
```

PHP – ECHO vs PRINT

- ECHO e mai rapid decât PRINT
- PRINT e funcție, ECHO nu

Deoarece ECHO nu e funcție nu poate fi folosit ca parte a unor comenzi mai complexe

```
<?php
$b=1;
$b ? print "TRUE " : print "FALSE ";
$b=0;
$b ? print "TRUE " : print "FALSE ";
?>
```

TRUE FALSE

PHP - Funcții

- Se utilizează:
 - Pentru a separa secțiuni din cod care execută o cerință particulară
 - Pentru a refolosi porțiuni de cod
 - Modificare cod într-un singur loc

```
<?php
function longdate($timestamp)
{
    return date("l F jS Y", $timestamp);
}
echo longdate(time());
echo longdate(time() - 17 * 24 * 60 * 60);
?>
```

Saturday December 8th 2018
Wednesday November 21st 2018

PHP - Variabile

PHP – Variabile locale

- Variabile create în cadrul unei funcții
- Pot fi accesate doar din interiorul funcției care le-a creat

```
<?php
function longdate($timestamp)
{
    $temp = date("l F jS Y", $timestamp);
    return "The date is $temp";
}
echo longdate(time());
?>
```

The date is Saturday December 8th 2018

```
<?php
$temp = "The date is ";
echo longdate(time());
function longdate($timestamp)
{
    return $temp . date("l F jS Y", $timestamp);
}
```

```
}  
?>
```

Saturday December 8th 2018

```
<?php  
$temp = "The date is ";  
echo $temp . longdate(time());  
function longdate($timestamp)  
{  
    return date("l F jS Y", $timestamp);  
}  
?>
```

The date is Saturday December 8th 2018

```
<?php  
$temp = "The date is ";  
echo longdate($temp, time());  
function longdate($text, $timestamp)  
{  
    return $text . date("l F jS Y", $timestamp);  
}  
?>
```

The date is Saturday December 8th 2018

PHP - Variabile globale

- Poate fi accesată oriunde în cod
- Dacă variabilele au un conținut complex și se preferă ca acesta să nu fie transmis ca parametru al funcției

```
<?php  
global $is_logged_in;  
...  
$is_logged_in = 1;  
...  
$is_logged_in = 0;  
?>
```

PHP - Variabile statice

- O variabilă locală care își păstrează valoarea și după ce funcția în care este setată se termină

```
<?php  
function test()  
{  
    static $count = 0;  
    echo $count;  
    $count++;  
}  
test();
```

```
test();  
test();  
test();  
?>
```

0123

PHP – Variabile super globale

Variabilă	Descriere
\$GLOBALS	Variabilele definite în scopul global
\$_SERVER	Informații referitoare la locații, antete, script-uri
\$_GET	Variabile trimise către server prin metoda HTTP GET
\$_POST	Variabile trimise către server prin metoda HTTP POST
\$_FILES	Fișierele încărcate pe server prin metoda HTTP POST
\$_COOKIE	Variabilele trimise către server prin HTTP cookies
\$_SESSION	Variabile de sesiune
\$_REQUEST	\$_GET, \$_POST și \$_COOKIE
&_ENV	Variabile de mediu

- Securitate:
 - Sanitizarea variabilelor super globale
 -

```
<?php  
$came_from = htmlentities($_SERVER['HTTP_REFERER']);  
echo $came_from;  
?>
```

[http://www.cs.ubbcluj.ro/~diana.sotropa/files/TWSS/0exemple/ex%20\(32\).php](http://www.cs.ubbcluj.ro/~diana.sotropa/files/TWSS/0exemple/ex%20(32).php)

Manipularea formularelor în PHP

- Crearea unui formular folosind tag-urile: **<form>** și **</form>**
- Specificarea metodei de transmitere a datelor: **GET** sau **POST**
- Unul sau mai multe **input**-uri
- **URL**-ul spre care sunt transmise datele odată cu submit-ul formularului

PHP - Input-uri

Text

```
<input type="text" name="name" size="size" maxlength="length" value="value" />
```

```
<textarea name="name" cols="width" rows="height" wrap="type">  
</textarea>
```

```
<textarea name="name" cols="width" rows="height" wrap="type">
```

This is some default text.

```
</textarea>
```

* *wrap* = {*off, soft, hard*}

Checkbox

```
<input type="checkbox" name="name" value="value" checked="checked" />
```

Butoane radio

```
<input type="radio" name="name" value="value" checked="checked" />
```

Checkbox

```
<input type="checkbox" name="name" value="value" checked="checked" />
```

Butoane ascunse

```
<input type="hidden" name="name" value="value" />
```

Select

```
<select name="name" size="size" multiple="multiple">
```

```
<option selected="selected" value="value">Text for value</option>
```

Etichete

```
<label>8am-Noon<input type="radio" name="time" value="1" /></label>
```

Buton tip submit

```
<input type="submit" value="Search" />
```

```
<input type="image" name="submit" src="image.gif" />
```

Expresii și instrucțiuni în PHP

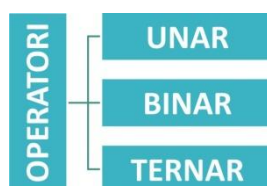
PHP - Expresii

- Expresie = o combinație de valori, variabile, operatori și funcții
- Valoarea returnată: număr, string, boolean (TRUE și FALSE)

$$y = 3(|2x| + 4)$$

```
$y = 3 * (abs(2*$x) + 4);
```

<pre><?php echo "a: [" . (20 > 9) . "]
"; echo "b: [" . (5 == 6) . "]
"; echo "c: [" . (1 == 0) . "]
"; echo "d: [" . (1 == 1) . "]
"; ?></pre>	<pre>a: [1] b: [] c: [] d: [1]</pre>
<pre><?php echo "a: [" . TRUE . "]
"; echo "b: [" . FALSE . "]
"; ?></pre>	<pre>a: [1] b: []</pre>
<pre><?php \$myname = "Brian"; \$myage = 37; echo "a: " . 73 . "
"; // Numeric echo "b: " . "Hello" . "
"; // String echo "c: " . FALSE . "
"; // Constant echo "d: " . \$myname . "
"; // Variable string echo "e: " . \$myage . "
"; // Variable numeric ?></pre>	<pre>a: 73 b: Hello c: d: Brian e: 37</pre>
<pre><?php \$day_number=350; \$days_to_new_year = 366 - \$day_number; // Expression if (\$days_to_new_year < 30) { echo "Not long now till new year"; // Statement } ?></pre>	<pre>Not long till new year</pre>



Tip operand 1	Tip operand 2	Conversie
Integer	Real	Integer => Real
Integer	String	String => Integer
Real	String	String => Real

- Toți operatorii (în afară de operatorul de concatenare) sunt operatori numerici

Tipuri de operatori:

- Aritmetici
- de Atribuire
- de Comparare
- Logici
- Pe biți

Operatori aritmetici		
Operator	Descriere	Exemplu
+	Adunare	\$j + 1
-	Scădere	\$j -6
*	Înmulțire	\$j * 11
/	Împărțire	\$j / 4
%	Rest	\$j % 9
++	Incrementare	++\$j
++	Incrementare	\$j++
--	Decrementare	--\$j
--	Decrementare	\$j--

Operatori de atribuire		
Operator	Exemplu	Echivalent cu
=	\$j = 15	\$j = 15
+=	\$j += 5	\$j = \$j +5
-=	\$j -= 3	\$j = \$j - 3
*=	\$j *= 8	\$j = \$j * 8
/=	\$j /=7	\$j = \$j / 7
.=	\$j .= \$k	\$j = \$j . \$k
%=	\$j %= \$k	\$j = \$j % 4
^=	\$j ^= \$k	\$j = \$j ^ \$k
&=	\$j &= \$k	\$j = \$j & \$k
=	\$j = \$k	\$j = \$j \$k
<<=	\$j <<= \$k	\$j = \$j << \$k
>>=	\$j >>= \$k	\$j = \$j >> \$k

Operatori de comparare		
Operator	Descriere	Exemplu
==	Este egal	\$j == 4
!=	Este diferit	\$j != 21
>	Este mai mare	\$j > 3
<	Este mai mic	\$j < 100
>=	Este mai mare sau egal	\$j >= 15
<=	Este mai mic sau egal	\$j <= 8
===	Este identic	1 === "1" False 1 === 1 True
!==	Nu este identic	1 !== "1" True 1 !== 1 False

Operatori logici		
Operator	Descriere	Exemplu
&&	AND	\$j == 3 && \$k == 2
and	AND	\$j == 3 and \$k == 2
	OR	\$j < 5 \$j > 10
or	OR	\$j < 5 or \$j > 10
!	NOT	! (\$j == \$k)
xor	Exclusive OR	\$j xor \$sk

A	B	AND	OR	XOR
1	1	1	1	0
1	0	0	1	1
0	1	0	1	1
0	0	0	0	0

Operatori pe biți		
Operator	Descriere	Exemplu
&	AND	\$a & \$b
	OR	\$a \$b
^	XOR	\$a ^ \$b
!	Logical Negation	TRUE (operand = false) FALSE (operand = true)
~	NOT	~ \$a
<<	Shift left	\$a << \$b
>>	Shift right	\$a >> \$b

PHP – Operatori Precedență

- Fiecare operator are un număr diferit de operanzi:
 - Unar (ex. incrementarea)
 - Binar (ex. adunarea)
 - Ternar (ex. **Expression ? Operand1 : Operand2** – single line if)
- Precedența operatorilor
 - Operatorii care au aceeași precedență vor fi procesați în ordinea apariției

```
<?php
```



```

$no1=10;
$no2=20;
$result=($no1>$no2)?"no1 is greater":"no2 is greater";
echo $result;
// no2 is greater
?>
no2 is greater

```

Operatori	Tip
()	Paranteze
++ --	Incrementare / Decrementare
!	Operatori logici
* / %	Operatori aritmetici
+ - .	Operatori aritmetici + String
<< >>	Operatori pe biți
<<= >>= < >	Operatori de comparare
== != === !==	Operatori de comparare
&	Operatori pe biți
^	Operatori pe biți
	Operatori pe biți
&&	Operatori logici
	Operatori logici
? :	Operatori logici
= += -= *= /=	Operatori de atribuire
.= %= &= != ^=	
<<= >>=	
and	Operatori logici
xor	Operatori logici
or	Operatori logici

PHP – Operatori Asociativitate

- Proprietatea de a efectua operațiile de la dreapta la stânga

Operatori	Descriere
New	Creează un obiect nou
!	NOT – operator logic
~	NOT – operator pe biți
++ --	Incrementare/Decrementare
+ -	Operatori aritmetici
(int)	Cast la integer
(double)	Cast la float
(string)	Cast la string
(array)	Cast la array
(object)	Cast la obiect
@	Înhibarea reportării erorilor
? :	Operator condițional
=	Operator de atribuire

```
<?php
$level = $score = $time = 0;
?>
000
```

PHP – Operatori Egalitate și Identitate

```
<?php
$a = "1000";
$b = "+1000";
if ($a == $b) echo "1"; // 1
if ($a === $b) echo "2"; // nu intră
?>
1
```

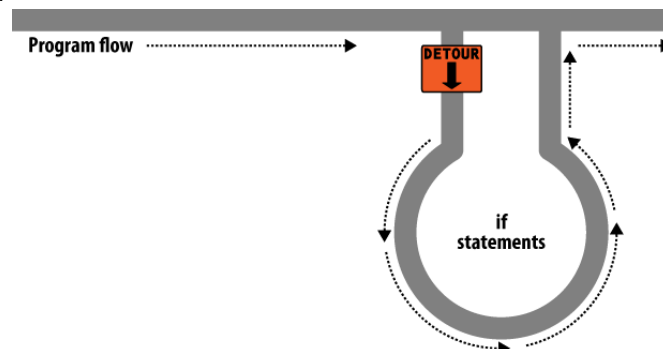
```
<?php
$a = "1000";
$b = "+1000";
if ($a != $b) echo "1"; // nu intră
if ($a !== $b) echo "2"; // 2
?>
2
```

PHP – Condiționale

- If
- Switch
- Operatorul ?

Condiționale IF:

- Expresie PHP validă: egalitate, comparație, test 0 sau NULL, rezultat returnat de funcții

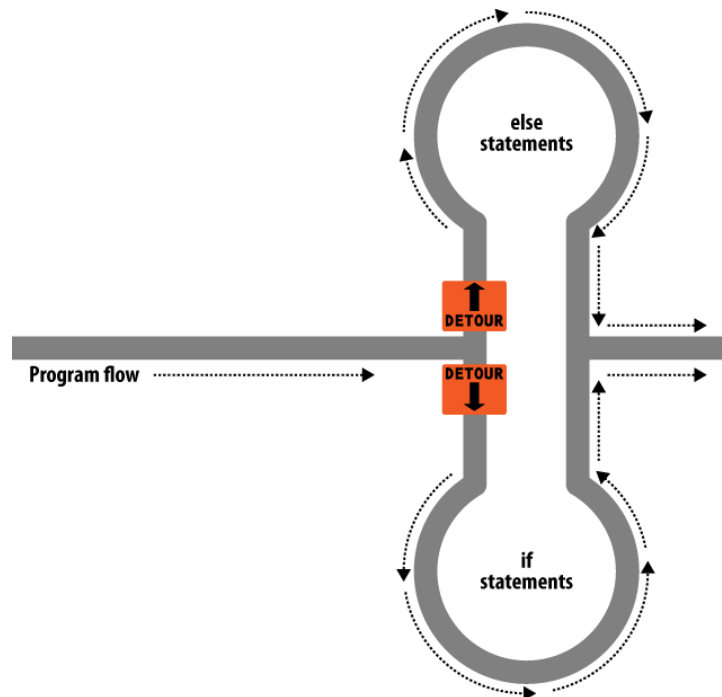


```
<?php
$money=500;
$bank_balance=20;
if ($bank_balance < 100)
{
```

```
$money += 1000;
$bank_balance += $money;
}
?>
```

Money: 1500Bank balance: 1520

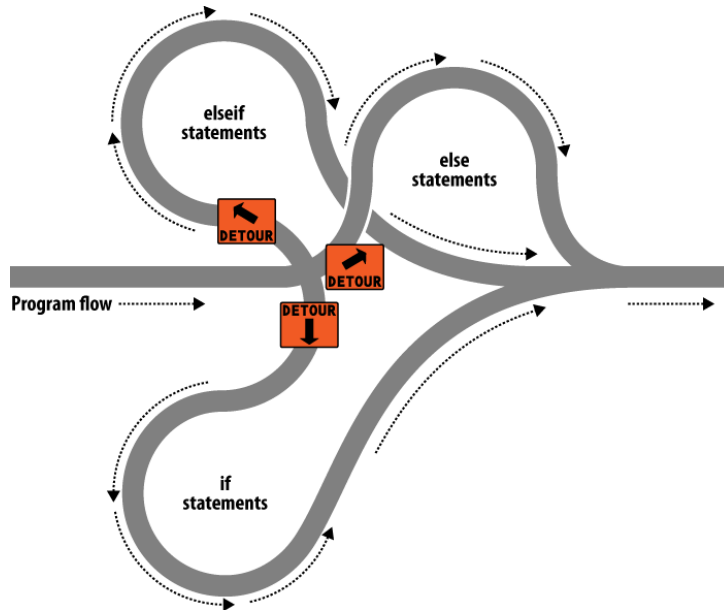
Condiționale ELSE:



```
<?php
$money=500;
$savings=500;
$bank_balance=20;
if ($bank_balance < 100)
{
    $money += 1000;
    $bank_balance += $money;
}
else
{
    $savings += 50;
    $bank_balance -= 50;
}
echo "Money: ".$money;
echo "Savings: ".$savings;
echo "Bank balance: ".$bank_balance;
?>
```

Money: 1500Savings: 500Bank balance: 1520

Condiționale ELSEIF:



```
<?php
$bank_balance=300;
if ($bank_balance < 100)
    $bank_balance += $money;
elseif ($bank_balance > 200)
    $bank_balance -= 100;
else
    $bank_balance -= 50;
echo $bank_balance;
?>
```

200

Condiționale SWITCH:

```
<?php
$page="News";
switch ($page)
{
case "Home":
    echo "You selected Home";
    break;
case "About":
    echo "You selected About";
    break;
case "News":
    echo "You selected News";
    break;
case "Login":
    echo "You selected Login";
    break;
case "Links":
```

```
<?php
$page="News";
if ($page == "Home")
    echo "You selected Home";

elseif ($page == "About")
    echo "You selected About";

elseif ($page == "News")
    echo "You selected News";

elseif ($page == "Login")
    echo "You selected Login";

elseif ($page == "Links")
    echo "You selected Links";
```

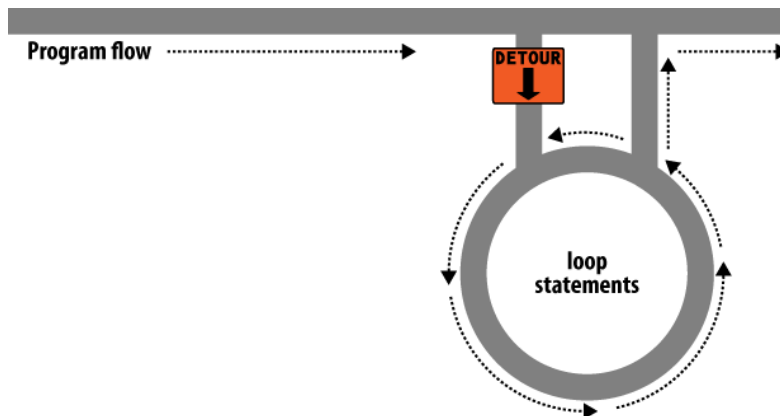
<pre> echo "You selected Links"; break; default: echo "Unrecognized selection"; break; } ?> </pre>	<pre> else echo "Unrecognized selection"; ?> </pre>
You selected News	You selected News

<pre> <?php \$page="Links"; switch (\$page): case "Home": echo "You selected Home"; break; // etc... case "Links": echo "You selected Links"; break; endswitch; ?> </pre>
You selected Links

Condiționale Operatorul ?

<pre> <?php \$fuel=0.5; echo \$fuel <= 1 ? "Fill tank now" : "There's enough fuel"; ?> </pre>
Fill tank now
<pre> <?php \$fuel=2; \$enough = \$fuel <= 1 ? FALSE : TRUE; echo \$enough; ?> </pre>
1
<pre> <?php \$saved=100; \$new=120; \$saved = \$saved >= \$new ? \$saved : \$new; echo \$saved; ?> </pre>
120

Condiționale Bucle



PHP - Condiționale Bucle WHILE

- Buclele WHILE se repetă până ce condiția din while este TRUE, și se oprește atunci când este FALSE

```
<?php
$fuel = 10;
while ($fuel > 1)
{
    // Keep driving ...
    echo "There's enough fuel";
    $fuel--;
}
?>
```

There's enough fuelThere's enough fuelThere's enough fuelThere's enough fuelThere's enough fuel
There's enough fuelThere's enough fuelThere's enough fuelThere's enough fuel

```
<?php
$count = 1;
while ($count <= 12)
{
    echo "$count times 12 is " . $count * 12 . "<br />";
    ++$count;
}
?>
```

1 times 12 is 12
2 times 12 is 24
3 times 12 is 36
4 times 12 is 48
5 times 12 is 60
6 times 12 is 72
7 times 12 is 84
8 times 12 is 96
9 times 12 is 108
10 times 12 is 120
11 times 12 is 132
12 times 12 is 144

```
<?php
$count = 0;
while (++$count <= 12)
echo "$count times 12 is " . $count * 12 . "<br />";
?>
```

```
1 times 12 is 12
2 times 12 is 24
3 times 12 is 36
4 times 12 is 48
5 times 12 is 60
6 times 12 is 72
7 times 12 is 84
8 times 12 is 96
9 times 12 is 108
10 times 12 is 120
11 times 12 is 132
12 times 12 is 144
```

PHP - Condiționale Bucle DO WHILE

- Buclele DO WHILE este ca și WHILE, cu diferența, că măcar o dată este executată bucla.

```
<?php
$count = 1;
do
echo "$count times 12 is " . $count * 12 . "<br />";
while (++$count <= 12);
?>
```

```
1 times 12 is 12
2 times 12 is 24
3 times 12 is 36
4 times 12 is 48
5 times 12 is 60
6 times 12 is 72
7 times 12 is 84
8 times 12 is 96
9 times 12 is 108
10 times 12 is 120
11 times 12 is 132
12 times 12 is 144
```

```
<?php
$count = 1;
do {
echo "$count times 12 is " . $count * 12;
echo "<br />";
} while (++$count <= 12);
?>
```

```
1 times 12 is 12
2 times 12 is 24
```

```
3 times 12 is 36
4 times 12 is 48
5 times 12 is 60
6 times 12 is 72
7 times 12 is 84
8 times 12 is 96
9 times 12 is 108
10 times 12 is 120
11 times 12 is 132
12 times 12 is 144
```

PHP - Condiționale Bucle FOR

- Buclele WHILE se repetă până ce condiția din while este TRUE, și se oprește atunci când este FALSE

```
<?php
for ($count = 1 ; $count <= 12 ; ++$count)
echo "$count times 12 is " . $count * 12 . "<br />";
?>
```

```
1 times 12 is 12
2 times 12 is 24
3 times 12 is 36
4 times 12 is 48
5 times 12 is 60
6 times 12 is 72
7 times 12 is 84
8 times 12 is 96
9 times 12 is 108
10 times 12 is 120
11 times 12 is 132
12 times 12 is 144
```

```
<?php
for ($count = 1 ; $count <= 12 ; ++$count)
{
echo "$count times 12 is " . $count * 12;
echo "<br />";
}
?>
```

```
1 times 12 is 12
2 times 12 is 24
3 times 12 is 36
4 times 12 is 48
5 times 12 is 60
6 times 12 is 72
7 times 12 is 84
8 times 12 is 96
9 times 12 is 108
10 times 12 is 120
11 times 12 is 132
12 times 12 is 144
```



```
<?php
for ($i = 1, $j = 1 ; $i + $j < 10 ; $i++ , $j++)
{
    echo $i." - ".$j."<br>";
}
?>
```

```
1 - 1
2 - 2
3 - 3
4 - 4
```

PHP - Condiționale Bucle BREAK

```
<?php
$fp = fopen("text.txt", 'wb');
for ($j = 0 ; $j < 100 ; ++$j)
{
    $written = fwrite($fp, "data");
    if ($written == FALSE) break;
}
fclose($fp);
?>
```

```
data0
```

PHP - Condiționale Bucle CONTINUE

```
<?php
$j = 10;
while ($j > -10)
{
    $j--;
    if ($j == 0) continue;
    echo (10 / $j) . "<br />";
}
?>
```

```
1.1111111111111
1.25
1.4285714285714
1.6666666666667
2
2.5
3.3333333333333
5
10
Nu se poate
-10
-5
-3.3333333333333
-2.5
```

```
-2  
-1.66666666666667  
-1.4285714285714  
-1.25  
-1.11111111111111  
-1
```

PHP – Conversii de tip

Conversii de tip explicite

```
<?php  
$a = 56;  
$b = 12;  
$c = $a / $b; // 4.66  
echo $c;  
$c = (int) ($a / $b); // 4  
echo $c;  
?>
```

Tip de conversie	Descriere
(int) (integer)	Conversie la întreg prin renunțarea la zecimale
(bool) (Boolean)	Conversie la Boolean
(float) (double) (real)	Conversie la real
(string)	Conversie la șir de caractere
(array)	Conversie la șis
(object)	Conversie la obiect

Cursul următor:

- Funcții și obiecte în PHP
- Șiruri în PHP
- Lucrul cu fișiere în PHP
- Introducere în MySQL
- Accesarea MySQL din PHP

Cuprins

Curs TWSS	1
Informații generale	1
Module:.....	1
Informații:	1
Evaluare:	1
Utile:.....	1
Structura materiei:	1
Dezvoltarea aplicațiilor web dinamice.....	2
Noțiuni de bază:	2
HTTP – HyperText Transfer Protocol.....	2
HTTPS - Secure Hyper Text Transfer Protocol.....	3
HTTP – HyperText Transfer Protocol.....	3
Reprezentări ale resursei cerute.....	4
Grupuri de răspunsuri ale codurilor de stare.....	5
Antete de mesaje	5
Metode de cerere standard pentru HTTP	6
Exemplu de utilizare HTML	7
Proxy	8
Cache.....	8
Instrumente	8
Dezvoltarea de aplicații web.....	8
Exemplu antet cerere.....	9
Variabile	10
Metode de cerere standard pentru HTTP: GET	10
Încapsularea parametrilor trimiși din browser prin GET	11
Metode de cerere standard pentru HTTP: POST	11
Metode de cerere standard pentru HTTP: HEAD.....	11
Metode de cerere standard pentru HTTP: PUT	12
Metode de cerere standard pentru HTTP: DELETE	12
Metode de cerere standard pentru HTTP: TRACE	12
Caracterul state-less al protocolului HTTP.....	12
Cookie. Definiție.....	13
Cookie. Utilizări	13

Cookie. Clasificare	13
Cookie	13
Web Storage.....	15
Setarea serverului web	16
Server Web.....	16
Serverul web APACHE	16
HTTP & HTML	16
Server web	16
Introducere în PHP	18
HTTP & PHP & MySQL	18
PHP script	18
Avantaje script Server Side:	18
Comentarii in PHP	20
Rădăcini: C, Perl, Java.....	20
Variabile PHP.....	20
PHP – tipuri de date	22
PHP - String	23
PHP - Multi-line commands	24
PHP - Constante	25
PHP – ECHO vs PRINT	25
PHP - Funcții	26
PHP - Variabile.....	26
PHP – Variabile locale	26
PHP - Variabile globale	27
PHP - Variabile statice	27
PHP – Variabile super globale	28
Manipularea formularelor în PHP	29
PHP - Input-uri.....	29
Expresii și instrucțiuni în PHP.....	30
PHP - Expresii	30
PHP – Operatori	31
PHP – Operatori Precedență	32
PHP – Operatori Asociativitate	33
PHP – Operatori Egalitate și Identitate	34
PHP – Condiționale.....	34
Condiționale IF:	34

Condiționale ELSE:.....	35
Condiționale ELSEIF:.....	36
Condiționale SWITCH:	36
Condiționale Operatorul ?	37
Condiționale Bucle	38
PHP - Condiționale Bucle WHILE	38
PHP - Condiționale Bucle DO WHILE	39
PHP - Condiționale Bucle FOR	40
PHP - Condiționale Bucle BREAK	41
PHP - Condiționale Bucle CONTINUE	41
PHP – Conversii de tip	42
Cursul următor:	45