

2. CONTROLUL FLUXULUI UNUI PROGRAM

În 1966, Böhm și Jacopini au demonstrat că orice program se poate scrie utilizând numai trei tipuri de structuri de control: *secvențială* (sau *liniară*), *alternativă* și *repetitivă condiționată anterior*. Pe lângă cele trei amintite, oricare limbaj de nivel înalt implementează și alte structuri, introduse în scopul creșterii eficienței activității de programare; acestea sunt *structura repetitivă condiționată posterior* și *structura selectivă* (cea din urmă fiind o *structură alternativă generalizată*). Toate structurile amintite (cu excepția celei secvențiale) pot fi grupate în următoarele două categorii:

- structuri de control *condiționale*: controlează *fluxul* programului, executând sau nu anumite porțiuni de cod, în funcție de îndeplinirea sau neîndeplinirea unor *condiții*;
- structuri de control *repetitive*: execută anumite porțiuni de cod de un număr determinat de ori.

În PHP, ca în alte limbaje de programare, structurile de control enumerate anterior sunt implementate ca *instrucțiuni condiționale* și, respectiv, *instrucțiuni repetitive*, în capitolul curent fiind prezentate ambele categorii. Tot în acest capitol sunt prezentate structuri utilizate pentru includerea codului în scripturi.

2.1. Instrucțiuni condiționale

Structurile condiționale sunt implementate în PHP prin intermediul instrucțiunilor `if` (*structura alternativă*) și `switch` (*structura selectivă*), care permit ramificarea execuției scriptului pe diverse căi, în funcție de deciziile luate în timpul execuției acestuia.

2.1.1. Instrucțiunea `if`

Probabil, cea mai des utilizată instrucțiune condițională este `if`. Aceasta poate fi folosită în variantele `if`, `if...else` și `if...elseif...if`, prezentate în continuare.

Instrucțiunea `if` are următoarea sintaxă:

```
if (expresie) {  
    grup  
}
```

Expresia *expresie* este evaluată la o valoare logică. Dacă aceasta este *true*, va fi executată secvența de cod *grup*, care include una sau mai multe instrucțiuni. Dacă numărul lor este mai mare decât unu, grupul trebuie inclus obligatoriu între acolade, în caz contrar fiind executată numai prima instrucțiune. În mod obișnuit, sunt folosite acoladele chiar dacă grupul conține numai o singură instrucțiune. Iată un exemplu:

```
if ($varsta > 99 && $varsta < 18) {  
    echo "Nu indepliniti conditia de varsta!";  
}
```

Alternativ, se poate utiliza sintaxa următoare (nu mai este necesară prezența acoladelor):

```
if (expresie):  
    grup  
endif;
```

Instrucțiunea *if...else* are următoarea sintaxă:

```
if (expresie) {  
    grup1  
}  
else {  
    grup2  
}
```

Dacă valoarea logică a expresiei *expresie* evaluată de interpretorul PHP este *true*, va fi executat grupul de instrucțiuni *grup1*, în caz contrar fiind executat *grup2*. Iată un exemplu de utilizare:

```
if ($varsta > 99 && $varsta < 18) {  
    echo "Nu indepliniti conditia de varsta!";  
} else  
    echo "Varsta este corespunzatoare!";  
}
```

Alternativ, se poate utiliza sintaxa următoare (în acest caz nu mai sunt necesare acoladele):

```
if (expresie):  
    grup1  
else:  
    grup2  
endif;
```

Instrucțiunea *if...elseif...else* are următoarea sintaxă⁸:

⁸ Este permis ca, în PHP, să se utilizeze și notația *else if* (i.e., în stilul C) în loc de *elseif*.

```

if (expresie1) {
    grup1
}
elseif (expresie2) {
    grup2
}
elseif (expresie3) {
    grup3
}
...
else {
    grup
}

```

Această instrucțiune extinde posibilitățile de execuție ale `if...else`, în cazul în care *expresie1* este evaluată la valoarea logică `false`, situație în care se evaluează *expresie2*. Dacă rezultatul acestei evaluări este `true`, se execută secvența de cod *grup2*, iar în caz contrar *grup3*. Dacă toate expresiile *expresie1*, *expresie2*, ... sunt evaluate la `false`, se va executa *grup*. Dacă este necesar, `elseif` poate fi utilizat de mai multe ori! De asemenea, `else` poate lipsi, ca în exemplul următor:

```

$numar = 10;
if ($numar < 0) {
    echo "Numarul $numar este negativ.";
} elseif ($numar == 0) {
    echo "Numarul $numar este egal cu zero.";
} elseif ($numar > 0) {
    echo "Numarul $numar este pozitiv.";
}

```

Alternativ, se poate utiliza sintaxa următoare (în care nu mai sunt necesare acoladele):

```

if (expresie1):
    grup1
elseif (expresie2):
    grup2
elseif (expresie3):
    grup3
...
else:
    grup
endif;

```

V-247

MANUALUL tău de PHP

Traian Anghel

PARTEA I-A

Bazele PHP

1. VOCABULAR, TIPURI DE DATE ȘI OPERATORI

- 1.1. PHP, limbaj de programare Web pentru server
 - 1.1.1. Programare Web pentru client și server
 - 1.1.2. Limbaje de programare Web pentru server
 - 1.1.3. Caracteristici ale limbajului PHP
- 1.2. Vocabular, simboluri, expresii, instrucțiuni și comentarii
 - 1.2.1. Vocabular
 - 1.2.2. Simboluri
 - 1.2.3. Expresii
 - 1.2.4. Instrucțiuni
 - 1.2.5. Comentarii
- 1.3. Tipuri de date, variabile și constante
 - 1.3.1. Tipuri de date
 - 1.3.2. Variabile
 - 1.3.3. Constante
- 1.4. Operatori
 - 1.4.1. Operatori unari
 - 1.4.2. Operatori binari
 - 1.4.3. Operatori ternari
 - 1.4.4. Precedența operatorilor
- 1.5. Utilizarea datelor incluse în formulare
 - 1.5.1. Atribute specifice elementului form
 - 1.5.2. Categori de câmpuri
 - 1.5.3. Utilizarea datelor incluse în formularele XHTML
- 1.6. Generarea dinamică a conținutului
 - 1.6.1. Generarea formatelor textuale
 - 1.6.2. Generarea formatelor grafice
 - 1.6.3. Utilizarea mecanismului output buffering
- 1.7. Expresii regulate
 - 1.7.1. Introducere în expresii regulate
 - 1.7.2. Suport PHP pentru expresii regulate

2. CONTROLUL FLUXULUI UNUI PROGRAM

- 2.1. Instrucțiuni condiționale
 - 2.1.1. Instrucțiunea if
 - 2.1.2. Instrucțiunea switch

- 2.2. Instrucțiuni repetitive
- 2.2.1. Instrucțiunile while și do...while
- 2.2.2. Instrucțiunea for
- 2.2.3. Instrucțiuni de control suplimentare
- 2.3. Instrucțiuni folosite pentru includerea codului

3. TABLOURI

- 3.1. Crearea tablourilor
- 3.1.1. Atribuire directă
- 3.1.2. Utilizarea funcției array()
- 3.1.3. Utilizarea cheilor
- 3.2. Parcurgerea și sortarea tablourilor
- 3.2.1. Parcurgerea tablourilor
- 3.2.2. Sortarea tablourilor
- 3.3. Tablourile tratate ca stive și cozi
- 3.4. Transformarea tablourilor
- 3.4.1. Unirea tablourilor
- 3.4.2. Extragerea unei porțiuni dintr-un tablou
- 3.4.3. Eliminarea unei porțiuni dintr-un tablou

4. FUNCȚII ȘI CLASE

- 4.1. Funcții
- 4.1.1. Funcții definite de utilizator
- 4.1.2. Funcții predefinite
- 4.2. Clase
- 4.2.1. Introducere în POO
- 4.2.2. Definirea claselor
- 4.2.3. Constructorul și destructorul
- 4.2.4. Moștenire
- 4.2.5. Proprietăți și metode statice
- 4.2.6. Polimorfism
- 4.2.7. Clase abstracte

PARTEA II-A

LUCRUL CU DATE STOCATE

5. LUCRUL CU FIȘIERE ȘI DIRECTOARE

- 5.1. Modificarea permisiunilor de acces
- 5.2. Funcții de lucru cu directoare
- 5.2.1. Obținerea și modificarea directorului curent de lucru
- 5.2.2. Deschiderea, citirea și închiderea directoarelor
- 5.2.3. Crearea, redenumirea și ștergerea directoarelor

- 5.2.4. Utilizarea funcțiilor de lucru cu directoare
- 5.3. Funcții de lucru cu fișiere
 - 5.3.1. Copierea, redenumirea și ștergerea fișierelor
 - 5.3.2. Deschiderea și închiderea fișierelor
 - 5.3.3. Citirea și scrierea în fișiere
 - 5.3.4. Determinarea permisiunilor, lungimii și tipului fișierelor
 - 5.3.5. Fișiere temporare
 - 5.3.6. Suma de control
 - 5.4. Încărcarea fișierelor pe server (upload)

6. LUCRUL CU BAZE DE DATE

- 6.1. Sisteme de gestiune a bazelor de date relaționale
 - 6.1.1. Modelul relațional
 - 6.1.2. Limbajul SQL
 - 6.1.3. Interfața ODBC
- 6.2. Implementarea MySQL a limbajului SQL
 - 6.2.1. Administrarea conturilor utilizatorilor
 - 6.2.2. Crearea și ștergerea bazelor de date
 - 6.2.3. Actualizarea caracteristicilor bazelor de date
 - 6.2.4. Crearea, redenumirea și ștergerea tabelelor
 - 6.2.5. Crearea și ștergerea indexurilor
 - 6.2.6. Actualizarea structurii tabelelor
 - 6.2.7. Actualizarea conținutului tabelelor
 - 6.2.8. Regăsirea datelor din tabele
- 6.3. Accesarea bazelor de date MySQL folosind PHP
 - 6.3.1. Conectarea la serverul MySQL
 - 6.3.2. Selectarea bazei de date
 - 6.3.3. Trimiterea interogărilor
 - 6.3.4. Prelucrarea setului de rezultate
 - 6.3.5. Determinarea erorilor
 - 6.3.6. Utilizarea extensiei mysqli
 - 6.3.7. Utilizarea interfeței ODBC

7. LUCRUL CU DATE XML

- 7.1. Introducere în limbajul XML
 - 7.1.1. Caracteristici ale limbajului XML
 - 7.1.2. Familia de limbaje XML
- 7.2. Componentele unui document XML
 - 7.2.1. Declarația XML
 - 7.2.2. Elemente XML
 - 7.2.3. Atribute
 - 7.2.4. Entități
 - 7.2.5. Secțiuni CDATA

- 7.2.6. Instrucțiuni de procesare
- 7.3. Spații de nume
- 7.4. Validarea documentelor XML
 - 7.4.1. Documente XML bine formate și valide
 - 7.4.2. Validarea documentelor XML folosind DTD
 - 7.4.3. Validarea documentelor XML via XML Schema
- 7.5. Tipuri MIME. Utilizarea limbajului XML
 - 7.5.1. Tipuri MIME asociate documentelor XML
 - 7.5.2. Utilizarea limbajului XML
- 7.6. Procesarea documentelor XML folosind PHP
 - 7.6.1. Metode de procesare
 - 7.6.2. Procesoare XML
 - 7.6.3. Procesarea documentelor XML utilizând API-ul DOM
 - 7.6.4. Procesarea documentelor XML utilizând SimpleXML
 - 7.6.5. Procesarea documentelor XML utilizând XMLReader
 - 7.6.6. Transformarea XSLT a documentelor XML

8. LUCRUL CU COOKIE-URI ȘI SESIUNI

- 8.1. Cookie-uri
 - 8.1.1. Crearea cookie-urilor
 - 8.1.2. Utilizarea cookie-urilor
- 8.2. Sesiuni
 - 8.2.1. Identificator de sesiune
 - 8.2.2. Crearea sesiunilor
 - 8.2.3. Durata de viață a unei sesiuni
 - 8.2.4. Adăugarea variabilelor în sesiune
 - 8.2.5. Eliminarea variabilelor și distrugerea sesiunii
 - 8.2.6. Utilizarea sesiunilor

9. TEHNICA AJAX

- 9.1. Introducere în AJAX
 - 9.1.1. Caracteristici ale aplicațiilor Web clasice
 - 9.1.2. Ce este AJAX?
 - 9.1.3. Obiectul XMLHttpRequest
 - 9.1.4. Instrumente de dezvoltare
 - 9.1.5. Neajunsuri ale aplicațiilor Web bazate pe AJAX
- 9.2. Șabloane utilizate în dezvoltarea aplicațiilor bazate pe AJAX
 - 9.2.1. Șabloane de interacțiune Web
 - 9.2.2. Șabloane arhitecturale
 - 9.2.3. Șabloane de prezentare a datelor
- 9.3. Utilizarea tehnicii AJAX
 - 9.3.1. Validarea datelor introduse de utilizatori în formulare
 - 9.3.2. Actualizare fără reîncărcarea întregii pagini

10. UTILIZAREA POȘTEI ELECTRONICE

- 10.1. Serviciul de poștă electronică
- 10.2. Standarde pentru formatul mesajelor de e-mail
 - 10.2.1. Standardul RFC 822
 - 10.2.2. Standardul MIME
- 10.3. Setări și funcții
 - 10.3.1. Setări în fișierul php.ini
 - 10.3.2. Funcția mail()
- 10.4. Tipuri de mesaje de e-mail
 - 10.4.1. Mesaje de e-mail bazate pe text
 - 10.4.2. Mesaje de e-mail în format XHTML
 - 10.4.3. Mesaje de e-mail cu fișiere atașate

11. INSTRUMENTE PENTRU DEZVOLTAREA APLICAȚIILOR

- 11.1. Editoare de text
- 11.2. Sisteme de management al conținutului
- 11.3. Cadre de lucru. Modelul MVC
- 11.4. Medii integrate de dezvoltare

PARTEA III-A

ERORI ȘI SECURITATE

12. MANIPULAREA ERORILOR

- 12.1. Tipuri de erori
 - 12.1.1. Erori de sintaxă
 - 12.1.2. Erori semantic
 - 12.1.3. Erori semantice
 - 12.1.4. Erori logice
- 12.2. Nivelul de raportare a erorilor
- 12.3. Accesarea manualului PHP
- 12.4. Portabilitatea codului
- 12.5. Optimizarea codului PHP
 - 12.5.1. Creșterea vitezei de execuție a scriptului
 - 12.5.2. Economisirea resurselor

13. SECURITATEA ÎN PHP

- 13.1. Principii și exemple de bune practice
 - 13.1.1. Principii
 - 13.1.2. Exemple de bune practici
- 13.2. Formulare și URL-uri
 - 13.2.1. Formulare și date
 - 13.2.2. Cross-Site Scripting

- 13.3. Baze de date și SQL
 - 13.3.1. Securizarea datelor și a interogărilor
 - 13.3.2. Expunerea datelor de acces
 - 13.3.3. SQL Injection
- 13.4. Cookie-uri și sesiuni
 - 13.4.1 Furtul cookie-urilor
 - 13.4.2 Expunerea sesiuni de date
- 13.5. Includerea fișierelor
 - 13.5.1. Expunerea codului sursă
 - 13.5.2. Manipularea numelor de fișiere
 - 13.5.3. Injectarea codului
- 13.6. Autentificare și autorizare
 - 13.6.1. Atacuri de tip “forță brută”
 - 13.6.2. Detectarea parolelor
- 13.7. Criptarea datelor
 - 13.7.1. Stocarea parolelor
 - 13.7.2. Utilizarea extensiei mcrypt

PARTEA IV-A

Evaluare și îndrumare

14. EXERCITII PROPUSE ȘI EXERCITII REZOLVATE

- 14.1. Exerciții propuse
- 14.2. Exerciții rezolvate

ANEXA A

INSTALAREA ȘI CONFIGURAREA SERVERELOR

- A.1. Instalarea serverelor pe platforma Windows
 - A.1.1. Instalarea serverului Web Apache
 - A.1.2. Instalarea serverului de aplicații PHP
 - A.1.3. Instalarea SGBDR MySQL
- A.2. Instalarea serverelor pe platforma Linux
 - A.2.1. Instalarea serverului Web Apache
 - A.2.2. Instalarea serverului de aplicații PHP
 - A.2.3. Instalarea SGBDR MySQL
- A.3. Configurarea serverului Web Apache
 - A.3.1. Fișierul *httpd.conf*
 - A.3.2. Directive de configurare
 - A.3.3. Directive orientate pe server
 - A.3.4. Directive orientate pe utilizator
 - A.3.5. Controlul accesului
 - A.3.6. Securizarea accesului prin controlul metodelor HTTP

- A.3.7. Rescrierea URL-urilor
- A.4. Configurarea serverului de aplicații PHP
 - A.4.1. Stiluri de tag-uri
 - A.4.2. Preluarea datelor
 - A.4.3. Utilizarea ghilimelelor magice
 - A.4.4. Utilizarea fișierelor situate pe alte servere
 - A.4.5. Încărcarea extensiilor
 - A.4.6. Alte directive

ANEXA B

INSTALAREA UNOR INSTRUMENTE DE LUCRU

- B.1. phpMyAdmin
- B.2. PSPad
- B.3. Alte instrumente

ANEXA C

EXECUȚIA AUTOMATĂ A SCRIPTURILOR PHP

- C.1. Fișiere crontab
- C.2. Comanda crontab
- C.3. Intrări cron
- C.4. Planificarea execuției scripturilor PHP