2. CONTROLUL FLUXULUI UNUI PROGRAM

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

- structuri de control *condiţionale*: controlează *fluxul* programului, executând sau nu anumite porţiuni de cod, în funcţie de îndeplinirea sau neîndeplinirea unor *condiţii*;
- structuri de control *repetitive*: execută anumite porţiuni de cod de un număr determinat de ori.

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

2.1. Instrucţiuni condiţionale

Structurile condiţionale sunt implementate în PHP prin intermediul instrucţiunilor `if` (*structura alternativă*) şi `switch` (*structura selectivă*), care permit ramificarea execuţiei scriptului pe diverse căi, în funcţie de deciziile luate în timpul execuţiei acestuia.

2.1.1. Instrucţiunea `if`

Probabil, cea mai des utilizată instrucţiune condiţională este `if`. Aceasta poate fi folosită în variantele `if`, `if...else` şi `if...elseif...if`, prezentate în continuare.

Instrucţiunea `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 instructiuni. 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 sintaxă[8]:

[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

7.2.6. Instrucțiuni de procesare
7.3. Spații de nume
7.4. Validarea documentelor XML
7.4.1. Documente XML bine formatate şi 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 ŞI 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
9.2.4. Adăugarea variabilelor în sesiune
8.2.5. Eliminarea variabilelor şi 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. Şabloane utilizate în dezvoltarea aplicațiilor bazate pe AJAX
9.2.1. Şabloane de interacțiune Web
9.2.2. Şabloane arhitecturale
9.2.3. Şabloane 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