

Capitolo 2 10/02/09

IPSIA “Antonio Pacinotti” - Pistoia

Corso sul linguaggio PHP

Gestione Web di un Magazzino



III Area

Classe V - Gestione Web di un
magazzino di materiale elettronico

Gualtiero Lapini

Corso sul linguaggio PHP

Le Strutture

Dopo gli esercizi visti nel capitolo precedente, che introducevano solo alcuni dei comandi del linguaggio PHP e solo alcune funzioni, vediamo questa volta i principali comandi che permettono di realizzare alcune strutture di controllo nel linguaggio PHP. Queste strutture sono assolutamente necessarie in qualsiasi linguaggio di programmazione e sono quelle che permettono di realizzare un *programma* degno di questo nome, non quindi una semplice lista di istruzioni ma bensì qualcosa di più sofisticato.

Il ciclo FOR

Iniziamo da una struttura che permette di ripetere uno o più comandi per un numero predeterminato di volte. Questo comando prende il nome di **ciclo** perché si ripete diverse volte.

<pre><?php echo "Ciclo FOR
"; echo "Esecuzione di un numero fisso di volte
"; for(\$n=1; \$n <= 10; \$n=\$n+1) { echo "@"; } echo "
"; ?></pre>	<p>Il testo che compare tra le due linee sottostanti</p> <hr/> <p>Ciclo FOR Esecuzione di un numero fisso di volte @@@@@@@@@@@@</p> <hr/> <p>Questo invece è di nuovo testo in HTML.</p>
--	---

Abbiamo una variabile **\$n** detta variabile di ciclo, che è quella variabile che tiene il conto di quante volte è stato ripetuto l'insieme di istruzioni racchiuse fra le parentesi graffe.

Anche questo comando in realtà è una **funzione** ovvero subito dopo il comando troviamo una coppia di parentesi tonde che racchiudono tre parametri.

In questo caso nell'ordine sono: *valore della variabile ad inizio ciclo*, *condizione che provoca il mantenimento del loop* (ciclo di ripetizione) ed il *comando che viene eseguito a fine di ogni ripetizione* del ciclo stesso.

A seguire ci sono uno o più comandi racchiusi fra parentesi graffe. Questo *blocco di istruzioni* è quello che viene ripetuto ad ogni esecuzione del loop.

Nel nostro caso l'effetto finale è quello di scrivere a video per 10 volte il carattere @ (*chiocciola*).

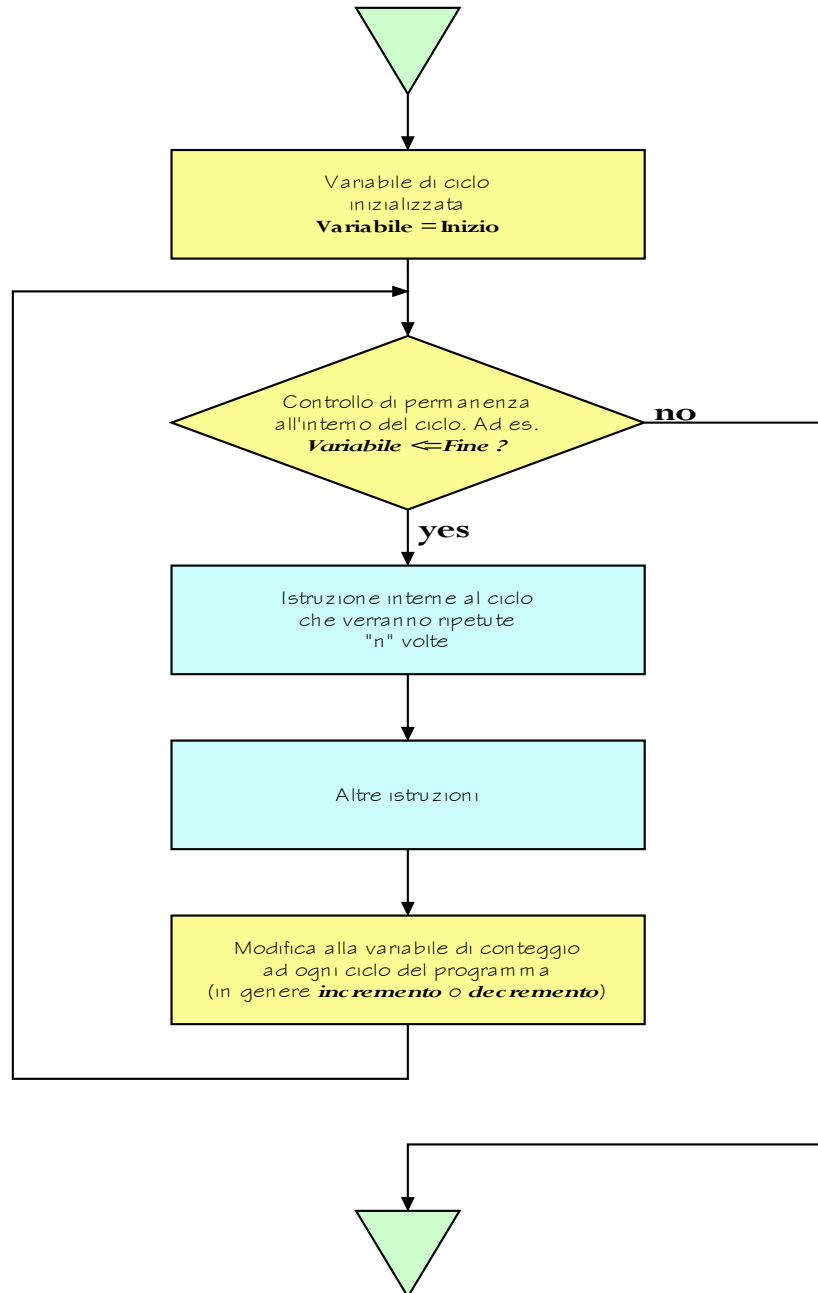
Da notare che lo stesso effetto a video poteva essere ottenuto con questo programma:

```
<?php
echo "<b>Ciclo FOR</b><br>";
echo "Esecuzione di un numero fisso di volte<br>";
echo "@";
echo "@";
echo "@";
echo "@";
echo "@";
echo "@";
echo "@";
echo "@";
echo "@";
echo "@";
echo "@";
echo "<br>";
?>
```

Con la differenza che la prima versione è molto più sintetica. Provate inoltre a pensare se si doveva **ripetere cento** volte lo stesso carattere !!!

Nella figura seguente si può vedere il diagramma di flusso di questa struttura di programma.

Ciclo FOR (ripetizioni predefinite)



Il ciclo DO/WHILE

Questa struttura invece permette di ripetere uno o più comandi per un numero non predeterminato di volte. Questo non significa che il ciclo di istruzioni venga ripetuto per un numero *casuale* di volte, ma solo che il numero di esecuzioni non è fisso e predeterminato.

In sostanza il ciclo si ripete fino a quando la situazione che determina il controllo di validità non si modifica, questa situazione cambierà, molto probabilmente, in base a condizioni esterne al programma, ad esempio in base alla risposta dell'utente, in base al contenuto dei dati di un archivio, ecc. ecc.

Per chiarire meglio, se devo controllare tutti le abitazioni sul lato destro di una strada dovrò eseguire un controllo di questo tipo:

1. Partire dall'abitazione contrassegnata dal n° 2 (cioè la prima a destra) *INIZIO PREDETERMINATO*.
2. Controllare tutte le abitazioni presenti, eseguendo il compito assegnato, ad esempio lasciare un messaggio nella casella delle lettere.
3. Terminare con l'ultima abitazione, contrassegnata dal n° 164 (sappiamo già fin dall'inizio quante abitazioni sono, non cambiano durante il mio turno di lavoro, anzi ho portato con me il numero esatto di volantini da consegnare). *TERMINE PREDETERMINATO E CONOSCIUTO DALL'INIZIO*.

Per questo compito userò una struttura del tipo **FOR**.

Mentre, ad esempio, sono un tecnico riparatore di lavatrici e devo eseguire questo compito:

1. Mi presento in sede la mattina, ritiro tutte le chiamate di assistenza, parto per eseguire il mio giro di riparazioni;
2. Eseguo una riparazione presso un cliente;
3. Depenno il cliente dalla lista;
4. Ho terminato la lista dei clienti: **VERO** = passo al punto 5, **FALSO** = torno al punto 2;
5. Fine del lavoro.
- 6.

In effetti questo ciclo fino a qui è identico al ciclo FOR: ha un inizio, una fine ed un numero fissato di esecuzioni (le chiamate prelevate la mattina). Però se aggiungo questa riga, tra la numero 3 e la numero 4...

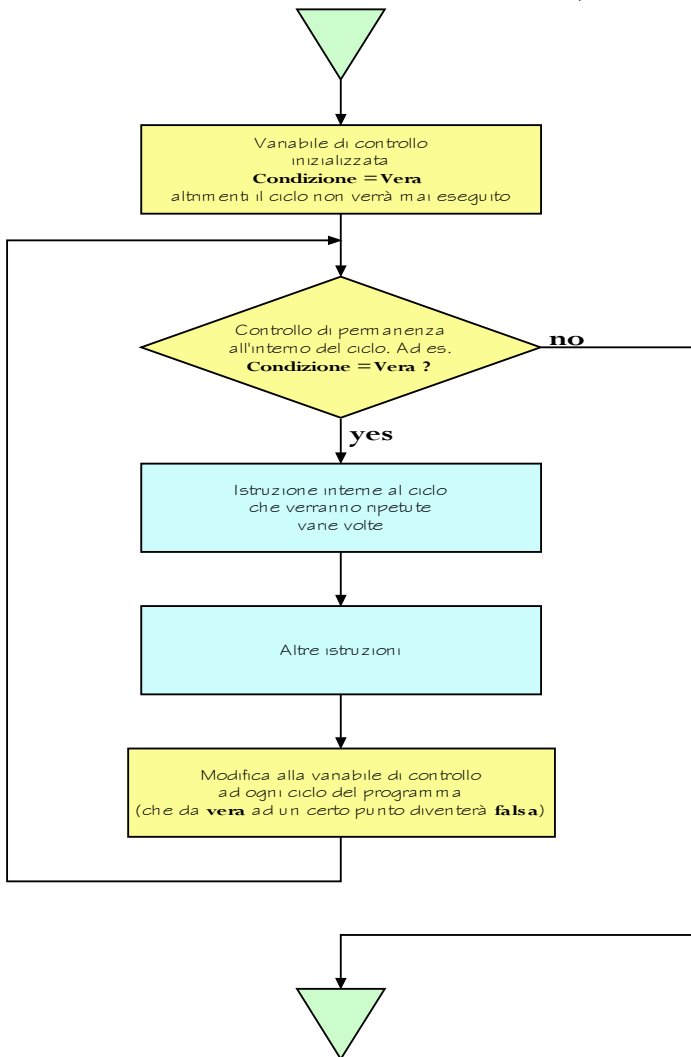
3bis. Ricevo una telefonata di un nuovo cliente e allora lo aggiungo alla lista.

In questo caso non ho idea, all'inizio del ciclo, di quanti clienti visiterò perché il numero può cambiare durante l'esecuzione stessa, al limite se ricevo chiamate in continuazione non tornerò mai più a casa!!! Oppure se la mattina non ho avuto chiamate allora non partirò neanche per eseguire il giro di riparazioni.

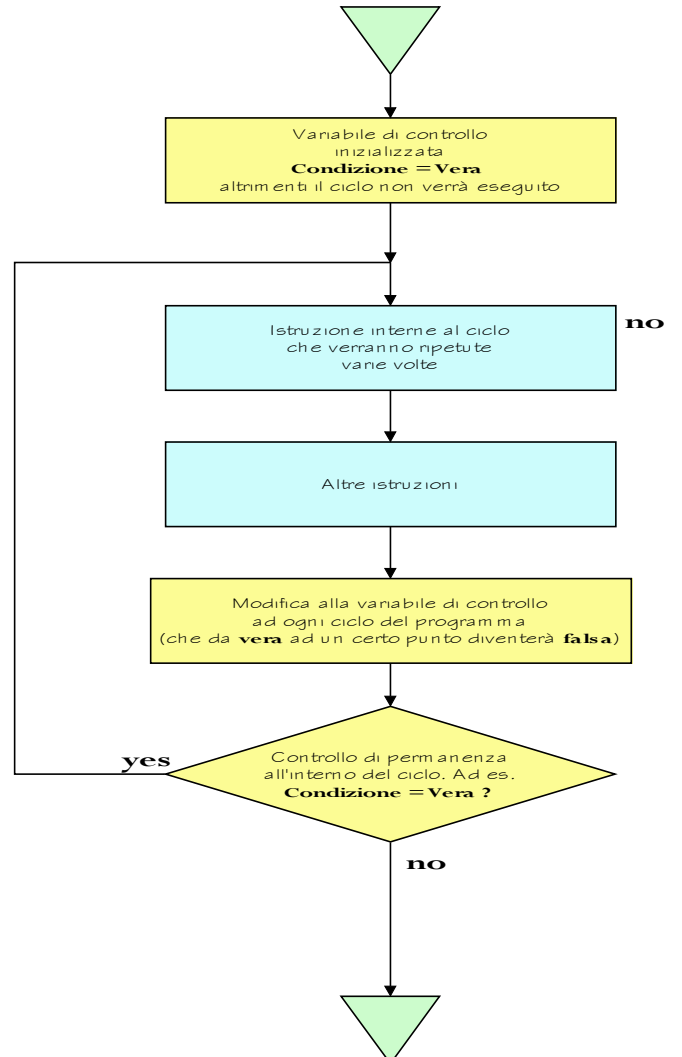
L'uso di una struttura oppure di un'altra va visto caso per caso, sarà poi l'esperienza che mi farà scegliere quale è l'impostazione migliore. Inoltre la differenza fra il tipo DO/WHILE ed il tipo WHILE/DO è questa: nel primo caso (**DO/WHILE**) il controllo viene fatto al termine o in coda al ciclo (quindi anche nel caso in cui il controllo dia come risultato un valore FALSO viene eseguito almeno una volta), mentre nel secondo caso (**WHILE/DO**) il controllo viene effettuato all'inizio o in testa al ciclo, quindi le istruzioni contenute nel ciclo potrebbero anche non essere mai eseguite.

Nelle due figure successive possiamo vedere il diagramma di flusso di queste due strutture e la piccola ma sostanziale differenza fra eseguire il controllo in testa oppure in coda al ciclo.

Ciclo WHILE DO
 (ripetizioni fino a che una situazione rimane vera)



Ciclo DO WHILE
 (ripetizioni fino a che una situazione rimane vera)



La scelta semplice IF/ELSE

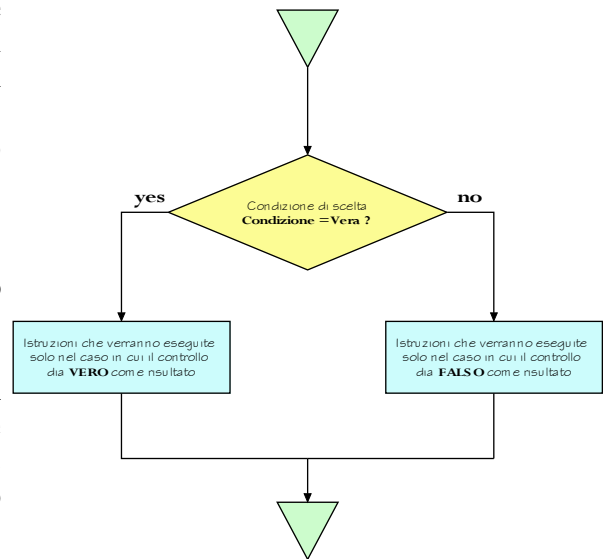
Vediamo adesso la struttura di scelta semplice:
la struttura IF/ELSE.

La comprensione di questa struttura dovrebbe essere facile, si tratta semplicemente di una diramazione nel flusso ordinato delle istruzioni di un programma. In base alla verifica di una condizione (detta di scelta) si hanno due soli casi possibili risultanti: VERO o FALSO.

In un caso vengono eseguite le istruzioni di un ramo, nell'altro caso viene scelto il ramo opposto. Chiaramente il percorso di un ramo esclude l'altro percorso.

Un esempio potrebbe essere quello illustrato nel programma seguente: il confronto fra le età di due individui per far comparire a video una scritta che reciti "Giorgio è più anziano di Luca" oppure il suo contrario "Luca è più anziano di Giorgio".

Struttura IF-ELSE (Scelta semplice)



```
<?php
$giorgio = 38;
$luca = 33;
// Confronta le età delle due persone
if( $giorgio > $luca ) {
    echo "Giorgio è più anziano di Luca";
}
else {
    echo "Luca è più anziano di Giorgio";
}
// Ritorno a capo
echo "<br>";
?>
```

La scelta multipla SWITCH/CASE

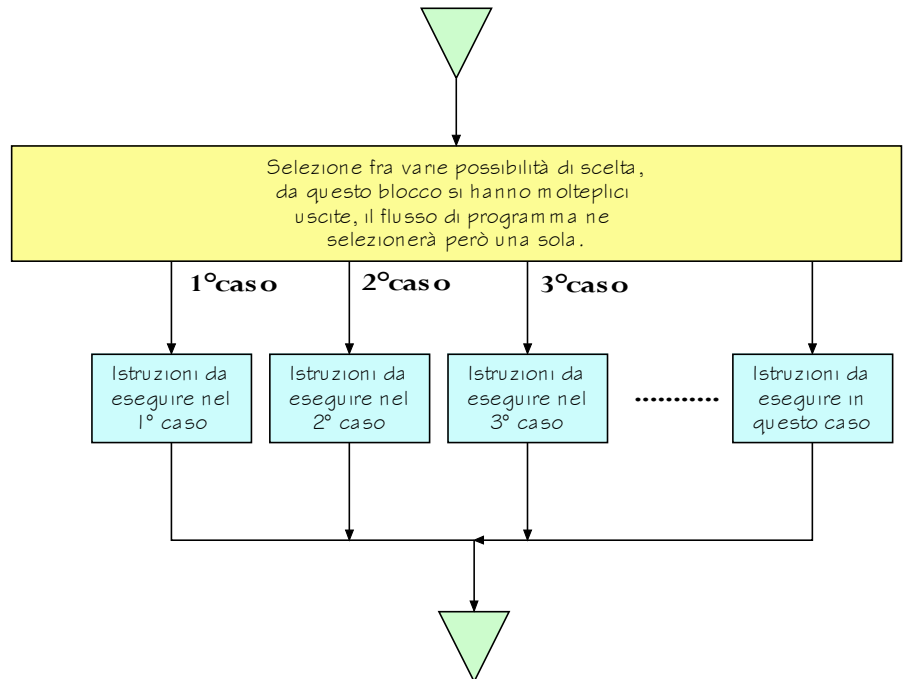
Vediamo adesso la struttura di scelta multipla:
la struttura SWITCH/CASE.

Questa struttura torna utile per gestire quei casi in cui non si abbia una diramazione semplice del tipo Vero/Falso, Si/No, Alto/Basso, ecc.

Ad esempio supponiamo di dover gestire una scelta di questo tipo: in base al mese corrente si deve visualizzare una scritta differente per ciascun mese, abbiamo quindi 12 casi diversi.

Vediamo l'esempio scritto in PHP che serve per scrivere in lingua italiana il nome del giorno, visto che il linguaggio PHP gestisce e visualizza solo il nome in lingua inglese.

Struttura SWITCH/CASE (Scelta Multipla)



```
<?php
// Con il simbolo "w" viene visualizzato il giorno della settimana
// 0=domenica, 1=lunedì, .... , 6=sabato
$giorno = date("w");
// Controlla il giorno della settimana
switch( $giorno ) {
    case 0:
        echo "Oggi è Domenica";
        break;
    case 1:
        echo "Oggi è Lunedì";
        break;
    case 2:
        echo "Oggi è Martedì";
        break;
    case 3:
        echo "Oggi è Mercoledì";
        break;
    case 4:
        echo "Oggi è Giovedì";
        break;
    case 5:
        echo "Oggi è Venerdì";
        break;
    case 6:
        echo "Oggi è Sabato";
        break;
    default:
        echo "Errore nella data";
        break;
}
// Ritorno a capo
echo "<br>";
?>
```

Utilizzando questa struttura occorre ricordarsi di due o tre cose: ogni caso (**case**) va terminato con il simbolo due punti (:), dopo ogni case va inserita il comando **break** per saltare subito a fine struttura (a meno che non si voglia per qualche ragione continuare con il case seguente) e infine ricordarsi di indicare l'ultimo caso con la parola chiave **default** (significa in difetto di, in mancanza di) per racchiudere tutti i casi non contemplati fino a questo punto.

Nel nostro esempio il caso di default è stato aggiunto anche se non serviva, poiché è impossibile che il giorno della settimana abbia un valore minore di zero o maggiore di sei.

In questo corso, visto il breve tempo a disposizione, vengono trattate solamente le strutture principali del linguaggio PHP, rimandando comunque ad una qualsiasi guida, ufficiale e non, del linguaggio stesso o, addirittura, ad un libro di testo vero e proprio con molte decine di esempi anche molto più complessi di questi.

Per quello che riguarda il linguaggio PHP di base il corso si conclude quindi a questo punto, proseguiremo con il prossimo capitolo dedicato alle istruzioni PHP per la connessione ad un database, riprendendo quindi il discorso lasciando in sospeso con le precedenti lezioni sul linguaggio MySQL.

Si consiglia la consultazione dei seguenti documenti, alcuni reperibili sul nostro sito, altri invece disponibili in Internet.

Manuale in italiano di PHP, scaricabile dal nostro sito, in formato CHM (Help di Windows), http://www.ipsiapacinotti.it/docs/php_manual_it.zip

Sito italiano, forse il migliore, dedicato al PHP <http://php.html.it> raggiungibile anche dalla home page <http://www.html.it>

Per poter collaudare in locale tutte le pagine scritte in HTML, PHP oppure testare le istruzioni MySQL, consiglio di installare un server WAMP sul proprio personal computer, in modo da verificare la correttezza del funzionamento del tutto.

WAMP Server è un acronimo che significa Windows Apache MySQL e PHP Server, in maniera analoga al ben più famoso **LAMP Server** che significa invece Linux Apache MySQL e PHP Server, ovvero il set completo di programmi per gestire un Web Server sotto ambiente Linux oppure Windows.

Apache è forse il miglior programma per gestione dei Server Web e comunque il più utilizzato al mondo, con questo set di programmi si mette in piedi un "serverino" completo e perfettamente funzionante in pochi minuti e senza eccessive difficoltà.

Rimando agli appunti per l'installazione e l'uso di questo programma.

Comunque il tutto è scaricabile dal sito <http://www.wampserver.com/en/download.php>

In rete comunque esistono svariate decine di manuali, per lo più in formato PDF Acrobat, la maggior parte dei quali in lingua inglese, sia di livello introduttivo che molto più approfonditi, con cui potrete proseguire lo studio di questo linguaggio e, comunque, ricordate che la maggior parte delle cose si impara dagli "esempi", ovvero guardando e cercando di capire cosa hanno fatto gli altri.