Indice

PHP (parte prima)

testo: Francesco Mannarino
mailto:fm@docenti.org

Introduzione
Hypertext Preprocessor (preprocessore di ipertesti) [http://www.PHP.net/] è un software OpenSource, concorrente diretto di ASP, Perl, JSP, CGI e qualunque altra soluzione scripting lato server, è un linguaggio di script immerso nel HTML. Molta della sua sintassi è presa in prestito dai linguaggi C, Java e Perl, a cui sono state aggiunte alcune specifiche caratteristiche del PHP.
Effettivamente dopo la comprensibile ebbrezza che si prova nel vedere il proprio testo HTML visulizzato su qualsiasi computer collegato ad Internet, sorge successivamente l'esigenza di rendere il proprio sito interattivo ed anche il cruccio sulla portabilità del proprio software, combinata alla facilità d'uso, alla scalabilità, alla robustezza e alla portabilità.
Di primo acchito l'orientamento potrebbe essere verso il C o C++, ma ambedue obbligano, oltre alla consapevolezza di trovarsi di fronte a linguaggi di basso livello, anche una approfondita conoscenza del SO ospite, anzi, se guardiamo alla portabilità, di più SO ospiti !
Chi non dovesse mai trovarsi in simili condizioni può dormire sonni tranquilli, ma con l'evoluzione del software oggi, sempre più orientato alla facilità *end-user*, non si può certo pretendere che un programmatore, posto dinanzi a nuove scelte o soluzioni, non possa avere quel minimo di versatilità nell'affrontare con agilità le nuove problematiche.
Scegliere il PHP in questi casi è un'ottima soluzione, anche perché essendo OpenSource c'è parecchio da imparare dal codice sorgente e, se si conosce il linguaggio C/C++, è veramente semplice da apprendere.

I pacchetti e l'installazione
Gli ambienti di sviluppo (UNIX, Unix/Linux, Unix/HP-UX, Unix/Solaris, Unix/OpenBSD, Unix/Mac OS X, Windows 9X/NT/2000/XP; Apache, CGI/Commandline, fhttpd , Caudium, IIS/PWS, Netscape and iPlanet, OmniHTTPd Server, Oreilly Website, Pro, Xitami), i database (Oracle, Informix, Interbase, MS SQL, mySQL, PostgreSQL, ...) e le fonti (ODBC, attraverso moduli software supportati da terzi o via ADO, ...) supportati sono veramente impressionanti.
Il nostro ambiente operativo è WinNT (nessuna differenza con Win2000) con IIS 4.0 e, dopo aver scaricato i pacchetti [http://www.PHP.net/downloads.PHP] e il manuale in italiano (ma è preferibile quello in inglese) [http://www.PHP.net/docs.PHP], iniziamo l'installazione di **PHP410-installer.exe** nella versione 4.1.0, che crea una directory **c:\PHP**, che contiene due subdirectory (*sessiondata* e *uploaditem*) e quattro file (**install.txt**, **license**, **PHP.exe**, **PHP4ts.dll**).
Letta ed accettata la licensa d'uso, seguendo le istruzioni nel file *install.txt*, passiamo all'installazione veria e propria del nostro super CGI.
Copiamo **PHP4ts.dll** in **c:\winnt\system32**,avviamo **Internet Service Manager**, facciamo click col tasto destro su **Default Web Server** e scegliamo **Properties**, passiamo alla cartella **Home Directory** e facciamo click sul tasto **Configuration** ed in **App Mapping** facciamo click su **Add**: in **Executable** indichiamo la directory dove si trova *PHP.exe* (generalmente *c:\PHP\PHP.exe*) e in **Extension** l'estensione **PHP**, premiamo **OK** fin quando non chiudiamo la tab sheet di Properties, stoppiamo e riavviamo il servizio http.
A questo punto dopo aver scritto e salvato (*prova.PHP*) il nostro primo script nella *WWWroot*

script PHP (lato server)

```
<HTML>
<HEAD>
<TITLE>prova</TITLE>
</HEAD>
<BODY>
<?PHP
echo "mod PHP bnbmbmbmbmnbnm";
echo "\n<br>";

$num = 20; # intero
$prezzo = 345.45; # in virgola mobile
$grand = 'large'; # stringa
$myarray[0] = "rosso"; # array scalare, primo elemento (0)
$myarray["qwe"] = 6; # array associativo, elemento "qwe"

$var = "rosso $prezzo\n <br>";
echo $var;

$var = 'verde $prezzo <br>'; # qui non si puo usare \n
echo $var;

echo "$HTTP_USER_AGENT\n <br>";
?>
</BODY>
</HTML>
```

output codice HTML (lato client)

```
<HTML>
<HEAD>
<TITLE>prova</TITLE>
</HEAD>
<BODY>
mod PHP bnbmbmbmbmnbnm
<br>rosso 345.45
<br>verde $prezzo <br>Mozilla/4.0 (compatible; MSIE 6.0; Windows 98; Virgilio3pc; Q312461)
<br></BODY>
</HTML>
```

output browser (lato client)

```
mod PHP bnbmbmbmbmnbnm
rosso 345.45
verde $prezzo
Mozilla/4.0 (compatible; MSIE 6.0; Windows 98; Virgilio3pc; Q312461)
```

eseguiamolo dal client e vedremo comparire correttamente le informazioni richieste in Figura 1_1.

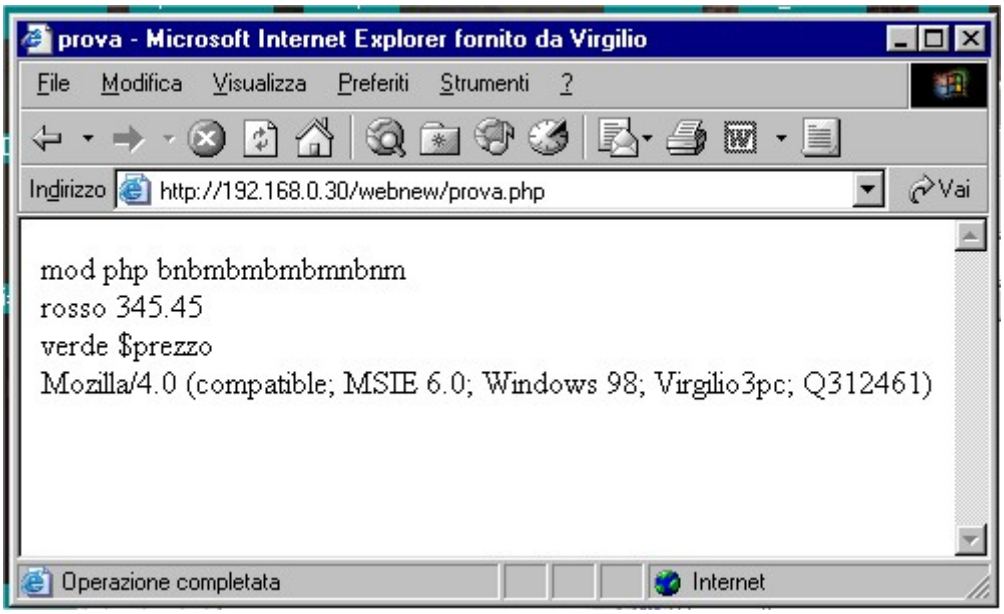

Figura 1_1

Sintassi di base e tipi
Abbiamo salvato il file con l'estensione .PHP, ma avremmo potuto anche salvarlo come .PHP4, .HTML in relazione all'impostazione di PHP nel file di configurazione del server Web, nel nostro caso IIS.
Il codice è racchiuso in uno dei seguenti tag:

```
<?PHP
echo "la mia casa";
?>
```

oppure

```
<? echo "la mia casa"; ?>
```

oppure

```
<script language = "PHP" > echo "la mia casa"; </script>
```

Il comando echo invia l'output sul broser client e dopo ogni istruzione PHP c'è necessariamente il punto e virgola (;) come carattere di fine istruzione, mentre è poco importante che le istruzioni si trovino tutte nella stessa riga di codice, identate o su diverse righe.
PHP può operare numeri interi, in virgola mobile, stringhe, array e oggetti tutti preceduti dal segno di dollaro ($) e sono tutti case sensitive. I valori delle stringhe possono essere racchiusi sia dai doppi apici ("), che dai apici singoli ('), e, come si può notare dall'esempio su riportato, nel primo caso la varibili racchiuse in esse sono valutate, quelle racchiuse dagli apici singoli non lo sono.
Le variabili presenti in ogni script PHP sono locali, mentre quelle incorporate in uno script con l'istrzione INCLUDE sono globali.

Ma si possono usare anche gli oggetti:
prima dichiarati

```
class myobj {
  function do_myobj ($par) {
     echo "la classe myobj con parametro $par. ";
  }
}
```

poi definiti dall'istruzione new

```
$var = new myobj;
$var -> do_myobj(8);
```

che avrà il seguente output

la classe myobj con parametro 8.

I tipi booleani posso essere usati sapendo che PHP considera una variabilie che contiene qualcosa TRUE, mentre una che indefinita, nulla o zero FALSE.

```
if $myVar {
    ... istruzioni ...;
}   # FALSE
```

invece

```
$myVar = "la mia casa";
if $myVar {
    ... istruzioni ...;
}   # TRUE
```

Per sapere le variabili predefinite dal proprio sistema bisogna usare la funzione

```
PHPinfo();
```

nell'esempio riportato su $HTTP_USER_AGENT è solo una delle tante.

Una questione di stile
Ogni buon programmatore sa che lo stile imposto dal linguaggio è importante, ma soprattutto quello che mira alla chiarezza, alla semplicità e alla snellezza del proprio codice.
Identare è sempre importante, ma le variabili devono essere sempre chiare, soprattutto nel caso particolare di questo linguaggio che decide al momento dell'assegnazione il tipo di variabile utilizzata.
Si potrebbero usare prefissi importati dal altri linguaggi come il C o C++: int per gli interi, fl per la virgola mobile, str per le stringhe, ar per gli array, obj per gli oggetti e bo per le booleane; le variabili usate prima dovrebbero essere corrette in questo modo:

$num	$prezzo	$grand	$myarray[0]	$var	$myVar
$intnum	$flprezzo	$strgrand	$armyarray[0]	$objvar	$bomyVar

Attenzione: gli esempi qui mostrati funzionano con copia ed incolla a patto che vengano tolti tutti i caratteri superflui (colore, spaziatura, ...) utili per una migliore visualizzazione sui browser, ma fastidiosi per il parser PHP !

PHP (parte seconda)

testo: Francesco Mannarino
mailto:fm@docenti.org

Ancora sul linguaggio

Contrariamente allo C non è necessario dichiarare una variabile prima di usarla, mentre la conversione di una variabile in una altro tipo di dati è analoga, mediante il typecasting (o con la funzione settype)

```
<HTML>
<HEAD>
<TITLE>prova</TITLE>
</HEAD>
<BODY>
<?PHP
	// da floating point a stringa
	$intvar = (int)23.3222;
	echo "intvar=$intvar\n<br>";

	// da intero a stringa
	$strvar = (string)234 . 'qwe';
	echo "strvar=$strvar\n<br>";

	// da floating point a stringa
	$flvar = (float)123/56;
	echo "flvar =$flvar \n<br>";
?>
</BODY>
</HTML>
```

I commenti

```
<?PHP
	echo "primo commento"; // Questo è un commento su una linea nella stile c++

	/* Secondo commento su più linee ancora un'altra linea di commento
		evitare di annidare i commenti su più linee*/

	echo "terzo commento"; # Questo è un commento stile shell Unix su una linea
?>
```

Operatori ed espressioni
Molto simile al linguaggio C è il modo in cui sono trattate le espressioni, cioè ciò che viene valutato, tra queste quelle di assegnamento semplice che abbiamo già osservato: $flmiaVar = 123.322, è un'assegnazione floating point.

Operatori artimetici: basta osservarli per rendersene conto

```
<? PHP
$a + $b;          // somma di $a e $b
$a - $b;          // sottrazione fra $a e $b
$a * $b;          // prodotto di $a e $b
$a / $b;          // rapporto di $a e $b
$a % $b;          // resto della divisione di $a e $b
?>
```

Operatori di assegnazione:

```
<? PHP
$a = 0;                    // azzeriamo la variabile
$a = 2 + 4;                // $a = 6
$a *= 2;                   // $a = 12
$b = ++$a;                 // $b = 13, $a = 13
$c = $a-;                  // $c = 13, $a = 12
$c = $a = 5;               // $c = 5, $a = 5
?>
```

Operatori di confronto:

$myvar = = $myvar2	vero se entrambi i valori delle varibili sono veri
$myvar = = = $myvar2	vero se entrambi i valori delle variabili sono veri e sono dello steso tipo
$myvar < $myvar2	vero se la prima variabile ha un valore inferiore della seconda
$myvar < = $myvar2	vero se la prima variabile ha un valore inferiore o uguale alla seconda
$myvar > $myvar2	vero se la prima variabile ha un valore maggiore della seconda
$myvar > = $myvar2	vero se la prima variabile ha un valore maggiore o uguale alla seconda
$myvar ! = $myvar2	vero se le varibili hanno valori differenti
$myvar ! = = $myvar2	vero se le variabili hanno valori e tipi differenti

Operatori sui bit:

$var1 & $var2	And
$var1 | $var2	Or
$var1 ^ $var2	Xor
~ $var1	Not (complemento)
$var1 << $var2	Shift (scorrimento) a sinistra
$var1 >> $var2	Shift (scorrimento) a destra

Operatori logici:

$var1 and $var2	And
$var1 or $var2	Or
$var1 xor $var2	Xor
! $var1	Not (negazione)
$var1 && $var2	And con diversa precedenza
$var1 || $var2	Or con diversa precedenza

Controllo del flusso
if, che viene utilizzato per prendere decisioni e switch, usato come selettore:

```
<?PHP
    $a=8;
    $b=8;
    if ( $a == $b) {
        print " $a e $b sono uguali ";
    } else {
        print " $a e $b sono diversi ";
    }

// oppure

    $a=8;
    $b=8;
    If ($a<$b) { print “$a è maggiore di $b”;
       }
      elseif ($a>$b) { print “$a è minore di $b”;
          }
         elseif ($a == $b) { print “$a è uguale di $b”;
            }
// switch

$intvar = 3;
switch ($intvar) {
   case 0:
      echo "intvar contiene 0";
      break;
   case 1:
      echo "intvar contiene 1";
      break;
   case 2:
      echo "intvar contiene 2";
      break;
   case 3:
      echo "intvar contiene 3";
      break;
   default:
      echo "intvar contiene $intvar";
}

?>
```

While, ciclo (loop) di istruzioni ripetute finché non si verifica una condizione, for, che valuta tre espressioni e foreach, che permette di scorrere i valori di una array:

```
<?PHP
echo "\n<br> primo loop\n<br>";
```

```
// valutazione iniziale, stampa numeri da 1 a 99
$intvar = 0;
while ($intvar < 100) {
    echo $intvar++;
    echo "\n ";
}

echo "\n<br> secondo loop\n<br>";

// valutazione finale, stampa numeri da 1 a 99
$intvar = 0;
do
{
   echo $intvar++;
    echo "\n ";
}
while ( $intvar < 100);

echo "\n<br> terzo loop\n<br>";

// for, stampa numeri da 1 a 99
for ($intvar = 0; $intvar < 100; $intvar++) {
echo "$intvar\n ";
}

echo "\n<br> quarto loop\n<br>";

// foreach

$arMyArray = array (1, 2, 3, 17);
foreach ($arMyArray as $valore) {
print "Valore corrente di \$arMyArray: $valore.\n<br>";
}

?>
```

PHP (parte terza)

testo: Francesco Mannarino
mailto:fm@docenti.org

Installazione in ambiente PWS (W9X, WNTwks, ME, XP, W2000pro)

Iniziamo l'installazione di **PHP410-installer.exe** nella versione 4.1.0, che crea una directory **c:\PHP**, che contiene due subdirectory (*sessiondata* e *uploaditem*) e quattro file (**install.txt**, **license**, **PHP.exe**, **PHP4ts.dll**).
Letta e accettata la licenza d'uso, seguendo le istruzioni nel file *install.txt*, passiamo all'installazione vera e propria del nostro super CGI.
L'ambiente non è un server e, non potendo gestire servizi, deve essere configurato in modo che

possa interpretare correttamente la presenza dei file con estensione .PHP. Eseguiamo perciò il programma di gestione del file di registro, **regedit.exe**, ci portiamo su *HKEY_LOCAL_MACHINE/System/CurrentControlSet/Services/W3Svc/Parameters/ScriptMap*, facciamo click col tasto dx su ScriptMap e scegliamo *nuova \ stringa*, assegniamo al *nome* l'estensione .PHP e ai *dati* il percorso del nostro CGI, cioè c:\PHP\PHP.exe. Poi ci portiamo su *HKEY_CLASSES_ROOT*, facciamo click col tasto dx e scegliamo *nuova \ chiave* scrivendo la nuova estensione *.PHP*; evidenziamola e passiamo al pannello destro, doppio clic sul *nome (predefinito)* e digitiamo *PHPfile* (se desideriamo avere anche altre estensioni dobbiamo ripetere la stessa operazione). Ora bisogna creare una nuova chiave, quindi tasto dx e scegliamo *nuova \ chiave* PHPfile; evidenziamo la chiave, doppio clic sul *nome (predefinito)* e digitiamo PHP script, creiamo una nuova chiave in PHPfile (tasto dx e scegliamo *nuova \ chiave*) shell, poi una sotto chiave open, ancora una sottochiave command, doppio clic sul *nome (predefinito)* e digitiamo esattamente c:\PHP\.exe -q %1.
Chiudiamo Regedit e riavviamo il sistema per rendere effettive le modifiche. Tra l'altro, se non sapevate come inserire correttamente un'estensione, questo è ciò che bisogna apprendere; per comodità ho salvato le informazioni in un file allegato al testo, che si potrà comodamente editare se il percorso di PHP.exe è diverso da quello impostato, ricordandoci che lo slash è doppio per i path.
Infine settare l'accesso in Esecuzione alle cartelle che contengono i file .PHP.
Per essere sicuri che tutto sia andato bene, creiamo un file PHP che contenga le informazioni sull'installazione, usando la funzione PHPinfo():

```
<?PHP
PHPinfo()
?>
```

eseguiamolo dal client e vedremo comparire correttamente le informazioni richieste in Figura 3_1.

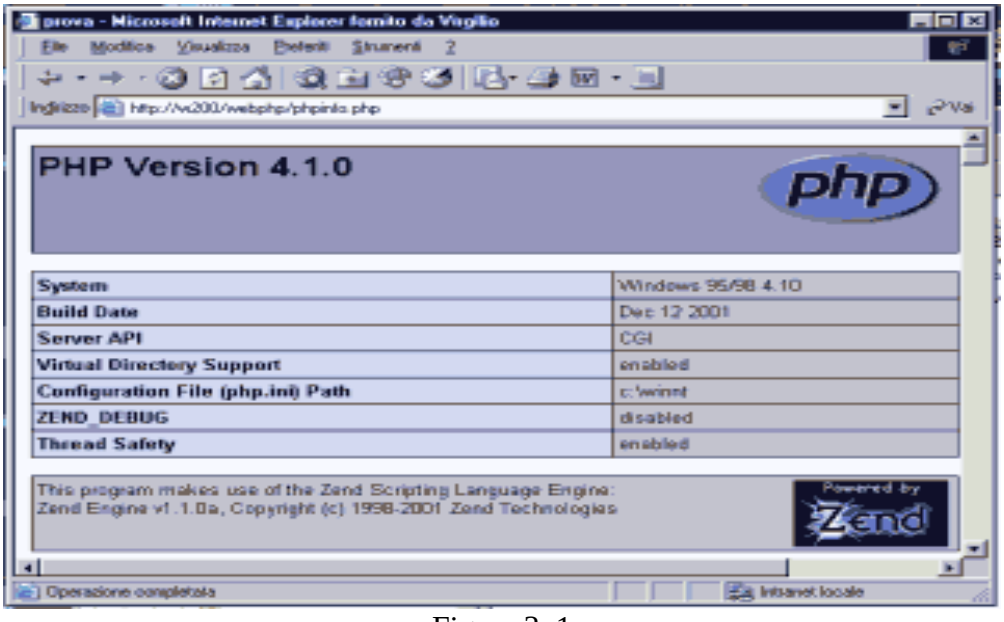

Figura 3_1

Le costanti
Si definiscono con la funzione define(); naturalmente non possono essere ridefinite e sono globali.

```
<?PHP
define("MIACOSTANTE", "Ciao mondo.");
echo MIACOSTANTE; // stampa "ciao mondo."
?>
```

Non va anteposto il simbolo del dollaro ($) come per le variabili (vale solo per i tipi boolean, integer, double, string).

Le funzioni
Come nel linguaggio C, le subroutine sono solo funzioni con o senza argomenti e con identica struttura

```
<?PHP
function moltiplica($intarg1, $intarg2)
{
  return $intarg1 * $intarg2;
}
$IntRisultato = moltiplica(2,3);
echo $IntRisultato;
?>
```

Nell'esempio testé mostrato le variabili sono passate per valore, ma posso essere passate anche per riferimento oppure argomenti di default, come in C++

```
<?PHP
function aggiungi(&$string)
{
    $string .= "questa è l'aggiunta.";
}
$str = "La stringa, ";
aggiungi($str);
echo $str;    // stampa 'La stringa, questa è l'aggiunta.'
?>

oppure

<?PHP
function CostruisciCasa ($Modello = "villa")
{
    return "Costruisci una $Modello.\n";
}
echo CostruisciCasa ();                  // stampa costruisci una villa.'
echo "<br>";
echo CostruisciCasa ("catapecchia"); // stampa costruisci una catapecchia.'
?>
```

Mantenimento dello stato: Le sessioni
La funzione session_start() inizializza una variabile per poter essere usata nelle stesse pagine; session_register($strarg1, $strarg2, ..) registra la variabile nel server e session_is_registered($strarg1) verifica che la variabile sia effettivamente registrata.

```
<?PHP
session_start();
if (session_register("mionome"))
{
  echo "sessione attiva";
  $mionome = "la mia sessione";
}
 else
  echo "sessione non attiva";

echo "<BR><BR>         adesso verifico la sessione <BR><BR>";

if (session_is_registered("mionome"))
  echo $mionome;
else
  echo "sessione fallita";
?>
```

Ci sono altre quindici funzioni per manipolare le sessioni ma, come in ASP (Active Server Page), JSP (Java Server page), anche in PHP valgono le solite raccomandazioni.
Le variabili di sessione risiedono sul server, perciò lo sovraccaricano; i cookie risiedono sul client, ma non sappiamo se questi saranno accettati dal client, però possiamo anche passare le variabili da una pagina all'altra semplicemente tramite l'URL, come in ASP e JSP:
http://IndirizzoServer/MiaPagina.PHP?variabile=casa
dopo la pagina richiesta facciamo seguire un punto interrogativo (?), nome della variabile(variabile), segno uguale(=) e infine la variabile(casa); la pagina PHP (MiaPagina.PHP) dovrà solo avere la variabile $variabile, che potrà essere immediatamente utilizzata. Bisogna solo ricordarsi che *dopo il punto interrogativo non ci deve essere alcuno spazio*!

PHP (parte quarta)

testo: Francesco Mannarino
mailto:fm@docenti.org

Mantenimento dello stato: I cookie
La funzione setcookie() definisce un cookie da inviare insieme alle altre informazioni di header. Essi permettono di memorizzare dei dati o informazioni addizionali tramite il browser dell'utente, poihcé fanno parte degli header del protocollo HTTP, tale funzione deve essere la prima in assoluto ad essere inviata, prima di qualsiasi contenuto della pagina HTML.
La funzione ha questa sintassi
int setcookie (string name [, string value [, int expire [, string path [, string domain [, int secure]]]]])
come si può notare tutti gli argomenti sono opzionali tranne name e con i seguenti significati:
name, nome del cookie da creare;
value, il valore da attribuirgli;
expire, data di scadenza, che il browser provvederà a cancellare;
path, percorso della pagina che deve restituirlo;
domain, dominio per cui esso è valido;
secure, indica se esso deve essere trasmesso solo in una connesisone sicura di tipo HTTPS
É conseguente che i cookie diventino disponibili solo dopo la pagina successiva a quella che li ha

generati, al refreshing della medesima e possono essere cancellati solo con gli stessi parametri che gli ha generati.

```
<?PHP
 // è stata utilizzata la funzione time() + 3600 secondi = 1 ora
 if ($scelta==impostacookie) {
   // imposta il cookie
   SetCookie( 'fms', $valore, time()+3600, '/');
   echo '<a href="cookie.PHP"> per attivare il cookie ricaricare la pagina, clicca qui </a>';
   exit;
 } else
 if ($scelta==rimuovicookie) {
   // cancella il cookie
   setcookie('fms', '', time()-1, '/' );
   echo '<a href="cookie.PHP"> per disattivare il cookie ricaricare la pagina, clicca qui </a>';
   exit;
 }
?>
<HTML>
<head><title>cookie</title></head>
<body>
<?
 // Controlla se il cookie e' stato trasmesso dal browser
 //  e precisamente il nome utilizzato, fms
 if ( ! isset($fms) ) {
   echo "Cookie non impostato";
 } else {
 // bisogna chiedere il nome del cookie, in questo caso fms
 // per conoscere il valore immesso
   echo "nome  valore<br>";
   echo "-----------<br>";
   echo "fms = $fms";
 }
?>

<? if ( !isset($fms) ) {
 // possiamo scegliere se visualizzare l'impostazione oppure no
?>
  <br><br>
   <form action="cookie.PHP" method=POST>
     <input type=hidden name="scelta" value="impostacookie">
     Inserisci il valore<br>
     <input type=text name="valore" value="">
     <br>
     <br>
     <input type=submit value="Imposta il cookie">
   </form>
<? } else { ?>

   <form action="cookie.PHP" method=POST>
```

```
    <input type=hidden name="scelta" value="rimuovicookie">
    <input type=submit value="Rimuovi il cookie">
   </form>
<? } ?>
</body>
</HTML>
```

da notare la funzione int isset (mixed var), che verifica se una variabile è definita.

Comunicazione con l'utente: i form
Proprio nel precedente paragrafo sono stati usati i form o moduli per inviare dati. Non c'è nulla di differente rispetto a tutti gli altri metodi di gestione dei form e nemmeno sull'uso delle variabili nei confronti di ASP e JSP: *quando si inviano variabili tramite form queste sono considerate dalla pagina ricevente delle varibili globali*.
Il metodo GET usato in un form equivale al passaggio dei parametri via URL, cioè la pagina richiamata con tutte le variabili e i parametri sono visualizzati nella barra degli indirizzi, salvati nella cache e o potrebbero esseri salvati nella cartella cronologia, cosa che non è per nulla consigliato: pensate ad esempio che l'utente immetta una password o un altro tipo di dato personale. In sostanza è sempre consigliato usare il metodo POST nei form.

Di seguito c'è un form d'invio dati (form.PHP)

```
<HTML><HEAD><TITLE>primo form</TITLE></HEAD>
<BODY>
 <form method="post" action="form2.PHP">

 <!-- **** quello che segue è una tetxtbox (casella di testo) **** -->
  Cognome: <input name="cognome" type="text"><br>
  Nome: <input name="nome" type="text"><br>

 <!-- **** qui invece una combobox (casella di riepilogo combinata) **** -->
  Tipo di scuola
  <select name="tiposcu">
    <option value="Materna">Materna
    <option value="Elementare">Elementare
    <option value="Media">Media
    <option value="Superiore">Superiore
  </select><br>

 <!-- **** qui una checkbox (casella di controllo) **** -->
  <input type="checkbox" name="materia"> Materia Insegnata<br>

 <!-- **** qui invece una listbox (casella di riepilogo) a cui viene passato un array **** -->
  <select multiple name="materia[]">
   <option value="Italiano">Italiano
   <option value="Matematica">Matematica
   <option value="Altro">Altro
  </select><br>

 <!-- **** tipo pulsante di invio dati (submit) con testo "invia" **** -->
```

```
 <input type="submit" value="invia">

 </form>
</BODY>
```

poi uno di ricezione ed elaborazione (form2.PHP)

```
<HTML><HEAD><TITLE>prelievo dati primo form</TITLE></HEAD>
<BODY>
<?PHP
 echo "lei è il sig <B>$cognome <i>$nome</i></b>";
 echo "\n<br>";
 echo "la sua scuola d'appartenenza è $tiposcu";
 echo "\n<br>";
 // adesso controlliamo se esiste la variabile 'materia', la checkbox

// viene valutata l'esistenza dell'array
 if (isset($materia))
 {
   echo "materia insegnata: ";
   echo "\n<br>";
   // viene scandito l'array per estrarre il valore scelto con foreach
   foreach($materia as $val)
     echo "$val ";
 }
?>
</BODY>
</HTML>
```

Attenzione: per le versioni dalla 4.1 in poi e anche la 5.x occorre usare uno stile diverso per prelevare le informazioni tra le pagine web. Gli esempi che seguono si commentano da sè.

Uso del metodo GET
pagina chiamante

```
<HTML>
<HEAD>
<TITLE>metodo GET</TITLE>
</HEAD>
<BODY>
Prova d'invio informazioni<BR>
<A HREF= "metodoget2.PHP?strCognome=Crispino&strNome=Santo"> Elabora dati </A>
</BODY>
</HTML>
```

pagina ricevente

```
<HTML>
<HEAD>
<TITLE>metodo GET</TITLE>
</HEAD>
<BODY>
<?PHP
echo "Prova ricezione informazioni<BR>";
echo "Sono stati inviati i seguenti dati<BR>";
echo "Primo dato: " .$_GET['strCognome']. "<BR>";
echo "Secondo dato: " .$_GET['strNome']. "<BR>";
?>
</BODY>
</HTML>
```

Uso del metodo POST e i form
Modulo casella di testo con pagina chiamante e ricevente

```
<HTML>
<HEAD>
<TITLE> modulo</TITLE>
</HEAD>
<BODY>
Immetti il tuo nome<BR>
<FORM METHOD="post" ACTION="elaboradati1.PHP">
<INPUT TYPE="text" NAME="strnome">
<BR>
<INPUT TYPE="submit" VALUE="Invia dati">
<INPUT TYPE="reset" VALUE="cancella i dati immessi">
</FORM>
</BODY>
</HTML>

---------- file elaboradati1.PHP ----------
<HTML>
<HEAD>
<TITLE> elaborazione testo</TITLE>
</HEAD>
<BODY>
<?PHP
$strrisultato = $_POST['strnome'];
echo "hai inserito il testo è <b> $strrisultato </b>";
?>
</BODY>
</HTML>
```

Modulo pulsanti di opzione con pagina chiamante e ricevente

```
<HTML>
<HEAD>
```

```
<TITLE>modulopulsantiopzione</TITLE>
</HEAD>
<BODY>
Quale nazione preferisci per le tue vacanze ?<BR>
<FORM METHOD="post" ACTION="elaboradati2.PHP">
<INPUT TYPE="radio" NAME="opVacanze" VALUE="Francia">
Francia
<BR>
<INPUT TYPE="radio" NAME="opVacanze" VALUE="Spagna" CHECKED>
Spagna
<BR>
<INPUT TYPE="radio" NAME="opVacanze" VALUE="Spagna">
Inghilterra
<BR>
<BR>
<INPUT TYPE="submit" NAME="Submit" VALUE="Invia">
</FORM>
</BODY>
</HTML>


---------- file elaboradati2.PHP ----------
<HTML>
<HEAD>
<TITLE> elaborazione testo</TITLE>
</HEAD>
<BODY>
<?PHP
$strrisultato = $_POST['opVacanze'];
echo "tu preferisci andare in vacanza in: <b> $strrisultato </b>";
?>
</BODY>
</HTML>
```

Modulo casella di controllo con pagina chiamante e ricevente

```
<HTML>
<HEAD>
<TITLE>modulocaselladicontrollo</TITLE>
</HEAD>
<BODY>
<BR>
Qual'&egrave; il tuo hobby ? <BR>
<FORM METHOD="post" ACTION="elaboradati3.PHP">
<INPUT TYPE="checkbox" NAME="chkHobby[]" VALUE="sport">
Lo sport
<BR>
<INPUT TYPE="checkbox" NAME="chkHobby[]" VALUE="francobolli" CHECKED>
I francobolli
<BR>
```

```
<INPUT TYPE="checkbox" NAME="chkHobby[]" VALUE="computer">
Il computer
<BR>
<INPUT TYPE="submit" NAME="Submit" VALUE="Invia">
</FORM>
<BR>
</BODY>
</HTML>



---------- file elaboradati3.PHP ----------
<HTML>
<HEAD>
<TITLE> elaborazione testo</TITLE>
</HEAD>
<BODY>

<?PHP
if (is_array ($_POST['chkHobby'])) {
echo "I tuoi hobby sono: <br>";
foreach ($_POST['chkHobby'] as $value) {
echo "<b>$value</b> <br>";
}
}
var_dump( $_POST['chkHobby']);
?>
```

Modulo casella di riepilogo con pagina chiamante e ricevente

```
<HTML>
<HEAD>
<TITLE>modulocaselladiriepilogo</TITLE>
</HEAD>
<BODY>
Scegli il tuo libro preferito e la poesia preferita:<BR>
<FORM METHOD="post" ACTION="elaboradati4.PHP">
<SELECT NAME="cbLibro[]">
<OPTION>I Promessi Sposi</OPTION>
<OPTION>Orlando Furioso</OPTION>
<OPTION>La Divina Commedia</OPTION>
</SELECT>
<BR>
<BR>
<SELECT NAME="lstPoesia[]" SIZE="3" MULTIPLE>
<OPTION>La cavallina storna</OPTION>
<OPTION>La pioggia nel pineto</OPTION>
<OPTION>Il porto sepolto</OPTION>
<OPTION>L&acute; isola</OPTION>
<OPTION>La capra</OPTION>
</SELECT>
```

```
<BR>
<BR>
<INPUT TYPE="submit" NAME="Submit" VALUE="Invia">
</FORM>
</BODY>
</HTML>



---------- file elaboradati4.PHP ----------
<HTML>
<HEAD>
<TITLE> elaborazione testo</TITLE>
</HEAD>
<BODY>

<?PHP
if (is_array ($_POST['cbLibro'])) {
echo "hai scelto il libro <br>";
foreach ($_POST['cbLibro'] as $value) {
echo "<b>$value</b> <br>";
}
}

if (is_array ($_POST['lstPoesia'])) {
echo "hai scelto la poesia <br>";
foreach ($_POST['lstPoesia'] as $value) {
echo "<b>$value </b> <br>";
}
}
var_dump( $_POST['cbLibro']);
echo "<BR>";
var_dump( $_POST['lstPoesia']);
?>

</BODY>
</HTML>
```

PHP (parte quinta)

testo: Francesco Mannarino
mailto:fm@docenti.org

Installazione in ambiente Linux

Tutte le attuali distribuzioni includono il supporto PHP; quella che adopero attualmente, RedHat 7.2, è stabile ed efficiente. Ma vi capiterà sicuramente di dover aggiornare il vostro server Linux, di conseguenza è necessario configurarlo manualmente, dopo aver scaricato i file opportuni dal sito della GTK [http://gtk.PHP.net/], da quello Zend [http://www.zend.com/zend/optimizer.PHP] oppure da quello ufficiale PHP [http://www.PHP.net/].

Si possono scegliere le confezioni precompilate RPM, che si installano facilmente col comando
rpm -ivh nomefile.rpm.
Altrimenti bisogna compilare direttamente il pacchetto e comunemente si usa come web server Apache, occorrerà decomprimere i sorgenti col comando
tar xvfz nomefile.tar.gz
poi bisognerà eseguire il comando di configurazione dopo esser entrati nella directory appena creata
./configure
se si desiderano altre opzioni si potrà acceder all'help con ./configure --help come ad esempio la configurazione sotto Apache con MySQL ./configure --with-mysql --with-apxs
successivamente occorrerà configurarlo col comando
make
make install
Poi occorre configurare e copiare PHP.ini (in genere in /usr/local/lib ma anche in /etc/) anche se è meglio lasciarlo con le impostazioni predefinite, almeno inizialmente per chi è alle prime armi, poi occorrerà aggiungere file di configurazione del server apache (http.conf in /etc/httpd/conf/) aggiungendo una riga che permetterà al demone di interpretare correttamente i file con estensione PHP
AddType application/x-httpd-PHP .PHP
a questo punto bisogna riavviare il server web con la nuova configurazione
/usr/sbin/httpd -f /etc/httpd/conf/httpd.conf
La directory in cui potremo inserire in nostri file PHP è la stessa che viene usata, naturalmente, da linux
/var/www/HTML
Infine l'esecuzione in un browser, Konqueror per esempio, di un file che contenga lo script <?PHP PHPinfo() ?> ci dirà se la configurazione e l'installazione ha avuto successo (vedi Figura 5_1).

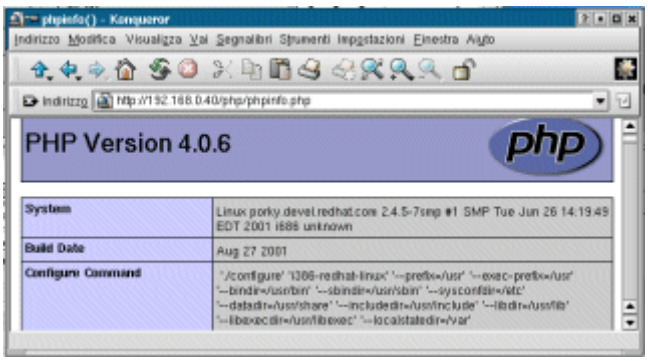

Figura 5_1

Le funzioni Include() e Require()
Le due funzioni permettono di includere un secondo file in quello in cui sono presenti, mentre il primo può essere utilizzato in espressioni condizionali il secondo ne è refrattario, ciò spiega benissimo il maggior uso del primo a causa della sua notevole flessibilità.
Ora ammesso che voi dobbiate ripetere la stessa pagina numerose volte e dover cambiare solo il corpo centrale, potreste benissimo includere, a seconda delle esigenze ora l'una ora l'altro, se poi aggiungete il fatto che include(), come detto poc'anzi risente delle istruzioni condizionali, comprenderete certamente la sua enorme utilità. *Prima di mostrarvi un esempio, ragionate sul fatto che in ASP il comando corrispondente non è una funzione, ma un comando vero e proprio che il IIS legge e processa prima della pagina HTML, poi la esegue con quest'ultima, obbligando il programmatore a far dei ragionamenti estremamente astrusi proprio sull'uso delle espressioni*

condizionali pari a quelle dell'uovo e della gallina. In PHP invece è una normale funzione presente nel normale flusso di istruzioni del programma e con tutti i vantaggi che ne derivano!

il file da includere myinclude

```
<b><i> Buon giorno papi </i></b>
```

il file richiamante include.PHP

```
<HTML><head><title>include file</title></head>
<body>
<?PHP
    include("myinclude");
 ?>
</body>
</HTML>
```

La funzione mail()
Essa permette di inviare e-mail ed ha questa sintassi
bool mail (string a, string soggetto, string messaggio [, string header_addizionali, string [parametri_addizionali]])
come si può intuire facilmente essa restisce TRUE se è andato tutto bene, FALSE al contrario, ecco un esempio completo tratto direttamente dal manuale ufficiale

```
<?PHP
/* destinatari */
$destinatari .= "Mary <mary@u.college.edu>" . ", " ; // da notare il comma
$destinatari .= "Kelly <kelly@u.college.edu>" . ", ";
$destinatari .= "ronabop@PHP.net";

/* soggetto */
$soggetto = "Promemoria compleanno per Agosto";

/* messaggio */
$messaggio .= "La seguente mail include una tabella ASCII formattata\n";
$messaggio .= "Giorno \t\tMese \t\tAnno\n";
$messaggio .= "3-terzo \t\tAgo \t\t1970\n";
$messaggio .= "17-esimo\t\tAgo \t\t1973\n";

/* è possibile inserire una blocco con la firma */
$messaggio .= "--\r\n"; //Delimitatore di firma
$messaggio .= "Promemoria di compleanno con senza proprietà d'autore ma di pubblico dominio";

/* intestazioni addizionali per errori, campo CC, BCC, etc */

$intestazioni .= "From: Birthday Reminder <birthday@PHP.net>\n";
$intestazioni .= "X-Sender: <birthday@PHP.net>\n";
$intestazioni .= "X-Mailer: PHP\n"; // mailer
$intestazioni .= "X-Priority: 1\n"; // Messaggio urgente!
$intestazioni .= "Return-Path: <birthday@PHP.net>\n";  // Indirizzo di ritorno per errori
```

```
/* Se si vuole inviare una mail in formato HTML, togliere il commento alla seguente linea */
// $intestazioni .= "Content-Type: text/HTML; charset=iso-8859-1\n"; // Tipo Mime

$intestazioni .= "cc:birthdayarchive@PHP.net\n"; // CC in copia a
$intestazioni .= "bcc:birthdaycheck@PHP.net, birthdaygifts@PHP.net\n"; // BCC in copia cieca a

/* ed infine l'invio */
mail($destinatari, $soggetto, $messaggio, $intestazioni);

?>
```

Upload dei file

É una delle più interessanti caratteristiche del PHP infatti con due soli file, uno è un form e l'altro che processa i dati ricevuti, è possibile inviare, con tutta sicurezza, dati di qualsiasi tipo via internet in una di dedicata appositamente all'upload.
Le variabili necessarie da impostare sono tre:
$userfile_name il nome del file originario inviato dal client
$userfile_size lunghezza del file in byte
$userfile il nome temporaneo del file che dovrà essere memorizzato nel server
$userfile_type tipo di file inviato

a questo punto bisogna costruire il form d'invio opportuno formload.PHP

```
<HTML><head><title>formload.PHP</title></head><body>

<form enctype="multipart/form-data" action="filesend.PHP" method="POST">
<!-- **** imponiamo un valore di 100000 byte (100 kbyte)**** -->
<input type="hidden" name="MAX_FILE_SIZE" value="100000">
<!-- **** nome del file 'ilfile'
  accanto apparirà il pulsante [sfoglia] **** -->
File da inviare: <input type="file" name="ilfile"><br>
<input type=submit value="Invia"></td>
</form>

</body></HTML>
```

e il file di ricezione filesend.PHP

```
<HTML><head><title>filesend.PHP</title></head><body>

<b>variabili inviate</b> <br>
$ilfile          <b><? echo $ilfile ?> </b><br>
$ilfile_name <b><? echo $ilfile_name ?> </b><br>
$ilfile_size    <b><? echo $ilfile_size ?> </b><br>
$ilfile_type   <b><? echo $ilfile_type ?> </b><br>
 <br>
<?
 // non è stato inviato nulla ?
 if ($ilfile_size == 0 ) {
   echo "Non è stato inviato alcun file <br>";
   exit;
```

```
 }
 // grandezza superiore a quello richiesto ?
 if ($ilfile_size > 100000 ) {
   echo "Il file eccede la grandezza di 100000 byte <br>";
   exit;
 }
 // copia il file con funzione copy() dalla directory temporanea a quella di destinazione
 $filetosend = './'. strtolower($ilfile_name);
 if (! copy($ilfile, $filetosend) ) {
   echo "Errore nel copiare file: la directory non ha i permessi di scrittura";
   exit;
 }

?>
</body></HTML>
```

PHP (parte sesta)

testo: Francesco Mannarino
mailto:fm@docenti.org

L'uso dei database

Il database più usato è sicuramente MySQL, un fior di database enterprise che consente di instaurare fino a 101 connessioni simultanee, ciò sta ad indicare che il numero degli utenti è virtualmente illimitato, solo Microsoft SQL Server riesce a far di più, ma non è free.
L'installazione sotto Linux è pressoché identica a quella operata con PHP. Ancora consiglio la versione RPM precompilata per la versione di Linux installata sul proprio computer. Invece quella sul Windows NT/ 2000 è un semplice file eseguibile.
Dopo bisogna attivarne il servizio dopo essersi portati nella dir: sotto Linux è mysql.server start (stop) dalla Konsole in /var/sbin; in windows NT/2000 NET START (STOP) mysqld (oppure mysqld-nt) dal Command Prompt in c:\mysql\bin.
Per ambedue bisogna verificare che il daemon sia in funzione digitando il comando mysqladmin -p ping, vi verrà chiesta la password, cha al momento non è stata settata, quindi premere invio e se l'esecuzione ha avuto successo avrete come risposta msqld is alive. Una ulteriore conferma la si potrà avere digitando il comando mysqlshow, che permetterà di osservare i database attualmente presenti (Vedi Figura 6_1)

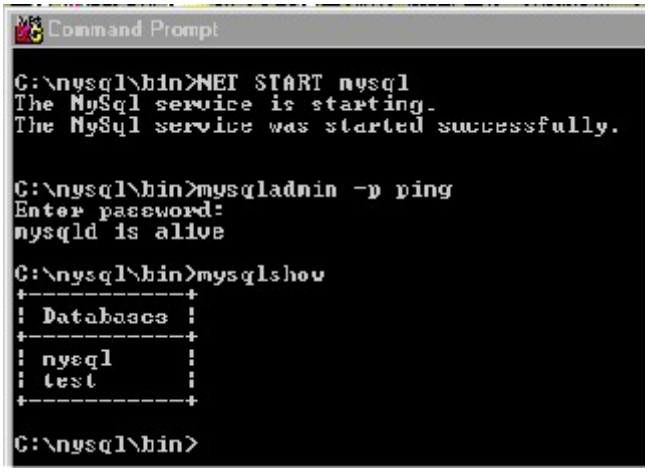

Figura 6_1

La versione per Window include un GUI, MySqlManager, sufficientemente friendly che aiuta l'utente nella gestione dei database, mentre quella per Linux, MysqlGUI, è la negazione totale del concetto di interfaccia grafica user-friendly, poiché i comandi bisogna darli comunque a mano, infatti il programma, probabilmente scritto in TK/TCL , integra la Konsole, e ci sono solo finestre di utilità per la gestione delle tabelle standard. In ambedue i sistemi è meglio agire via console.

Tralasciamo tutte le operazioni necessarie che bisognerebbe effettuare sul database e sulla creazione delle tabelle (password, tabelle assegnandole a gruppi, utenti, ...) ed avviamoci alla creazione della nostra tabella utenti con questi comandi, dopo ave fatto login all'interno di mysql, proprio col comando mysql, ricordandosi come in C/C++ che ad ogni istruzione bisogna far seguire il segno d'interpunzione punto e virgola (;) per far sì che esso venga eseguito, altrimenti vi si presenterà una freccia che indica l'attesa di nuove istruzione per eseguire il comando:

```
c:\mysql\bin>mysql
Welcome to the My monitor. Commands end with , or \g.
Your MySql connectione id is 3 to server version: 3.23.28-gamma

Type 'help;' or '\h' for help. Type '\c' to clear the buffer

mysql>create database utenti_db;
Query OK, 1 row affected (0.09 sec)

mysql>USE utenti_db;
Databse changed

mysql>CREATE TABLE utenti ( id INT (3) AUTO_INCREMENT, nome CHAR (20), cognome
CHAR (20), email CHAR(20), PRIMARY KEY(id) );
Query OK, 0 rows affected (0,21 sec)

mysql>INSERT INTO utenti VALUES ('', 'Francesco', 'Mannarino', 'fm@docenti.org');
Query OK, 1 row affected (0,17 sec)

mysql>SELECT * FROM utenti;
+---+-----------+------------+-------------------+
```

```
| id  | nome      | cognome    | email              |
+---+-----------+------------+--------------------+
|  1 | Francesco | Mannarino | fm@docenti.org |
+---+-----------+------------+--------------------+
1 row in set (0.08 sec)

mysql>quit
Bye
c:\mysql\bin>
```

L'esempio si commenta da sè. Con mysql siamo entrati nel prompt client di mysql; CREATE nomedatabase premette di create un database, USE nomedatabase di connettersi ad esso, CREATE TABLE nometabella (campo1, campo1, ...) di creare una tabella con i campi id intero (INT) contatore tre caratteri che sarà dichiarato chiave primaria (KEY); nome, cognome, email CHAR di 20 cartteri ciascuno. INSERT INTO nometabella VALUES (campo1, campo1, ...) con i valori inclusi negli apici (') serve per inserire i dati nella tabella, mentre SELECT * FROM utenti ci mostra la tabella.

PHP (parte settima)

testo: Francesco Mannarino
mailto:fm@docenti.org

La connesione al database e la visualizzazione dei dati
La connessione è estremamente semplice e consta di poche istruzioni, tutte definite da un sistema gestione delle eccezioni integrato, istruzione (argomenti) ord die (argomenti):
resource mysql_connect (string [indirizzo server], string [nome utente], string [password]) connessione
bool mysql_select_db (string nome database, resource [connessione]) seleziona un database MySQL
resource mysql_query (string query [, connessione]) esecuzione di una query
array mysql_fetch_array (resource result, int [result_type]) preleva i valori dei campi restituiti dalla query in un array
bool mysql_close (resource [connessione]) chiude la connessione

```
<HTML>
<HEAD>
<TITLE>prova database</TITLE>
</HEAD>
<BODY>
<?PHP

 // variabili di connessione
 // nome server, nome utente,
 // Password, nomedatabase, tabella
 // nel precedente articolo non abbiamo
 // dato nessuna UID e PWD
 $ServerName = "localhost";
 $UserName = "";
 $Password = "";
```

```
 $DbName = "utenti_db";
 $TableName = "utenti";

 // la connessione al database e alla tabella
 $MyConn = mysql_connect($ServerName, $UserName, $Password )
 or die ("Connessione fallita sul server $ServerName<br>");
 $MyDb = mysql_select_db ($DbName, $MyConn)
 or die ("Selezione del database fallita su $DbName<br>");

 // istruzione SQL di selezione dei dati
 $MyVarSQL = "SELECT * FROM $TableName";
 $MyQuery = mysql_query ($MyVarSQL, $MyConn)
 or die ("Query di selezione fallita $MyVarSQL<br>");

 // estrazione dei dati
 while($MyValues = mysql_fetch_array ($MyQuery))
 {
   $id = $MyValues["id"];
   $nome = $MyValues["nome"];
   $cognome = $MyValues["cognome"];
   $email = $MyValues["email"];
   echo "$id   ";
   echo "$nome   ";
   echo "$cognome   ";
   echo "$email   ";
   echo "<br>";

 // chiusura della connessione
 mysql_close($MyConn);
}
?>
</BODY>
</HTML>
```

ecco il risultato Figura 7_1

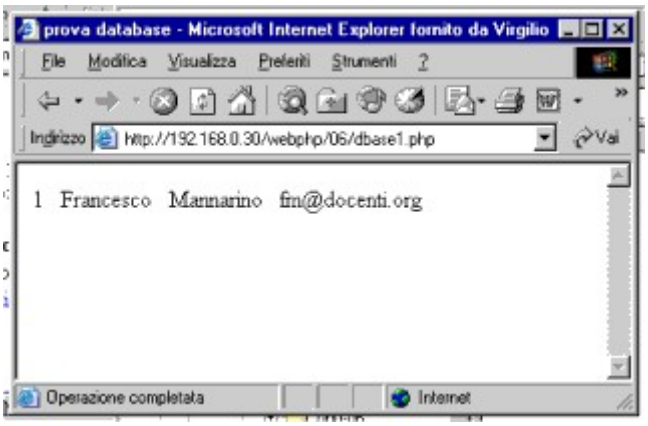

Figura 7_1

Tutte le altre operazioni sono semplicissime, perché basta passare nella stringa SQL l'istruzione che intendiamo eseguire (Figura 7_2)

```
 // istruzione SQL di inserimento dati
 $MyVarSQL = "INSERT INTO $TableName VALUES ('', 'Annarella', 'Perra',
'AnnarellaP@docenti.org')";
 $MyQuery = mysql_query ($MyVarSQL, $MyConn)
 or die ("Query di selezione fallita $MyVarSQL<br>");

// ricerca d'un dato
 $MyDatoCercato = "casa";
 $MyVarSQL = "SELECT * FROM $TableName WHERE nome LIKE '%$MyDatoCercato%'";
 $MyQuery = mysql_query ($MyVarSQL, $MyConn)
 or die ("Query di selezione fallita $MyVarSQL<br>");
```

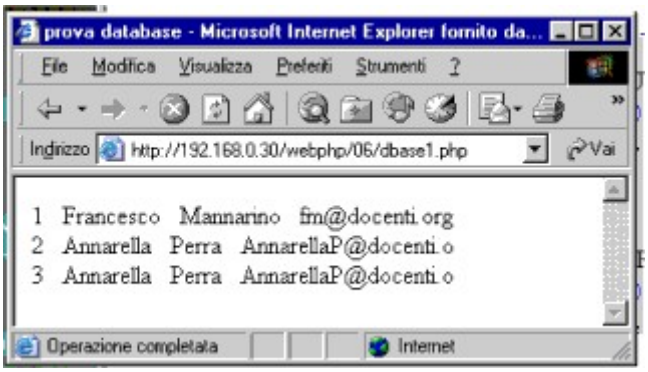

Figura 7_2

PHP (parte ottava)

---

testo: Francesco Mannarino
mailto:fm@docenti.org

Autenticazione HTTP

Possiamo controllare l'accesso a una pagina in diversi modi: tramite i cookie, in Windows con le SSL, in Linux tramite i permessi sulle cartelle e i file; ma se usate come web server Apache, allora PHP è un modulo che permette anche connessioni controllate con l'istruzione

int header (string string [, bool replace])

Gli argomenti sono parecchi e permettono di controllare la data, le ultime modifiche, la cancellazione della cache di autenticazione, ecc.

Il funzionamento è molto semplice: PHP controlla se alcune variabili sono state impostate ($PHP_AUTH_USER e $PHP_AUTH_PW); se non sono presenti si può richiedere che vengano impostate, poi si può stabilire solo il tipo d'autenticazione Basic (PHP_AUTH_TYPE).

```
<?PHP

 // intenzionalmente inseriamo una data
 // molto vecchia per impedire che la
 // pagina venga salvata nella cache
 // del proprio computer
```

```
 Header ("Expires: Mon, 11 Dec 1999 08:08:08 GMT");
 Header ("Last-Modified: " .gmdate("D, d M Y H:i:s"). " GMT");
 Header ("Pragma: no-cache");

// poi evitiamo che l'header inviato in
// precedenza non permetta un nuovo login
 Header ('WWW-Authenticate: Negoziate');
 Header ('WWW-Authenticate: NTLM', false);

if (!isset($PHP_AUTH_USER))
{
  // password errata
 echo "Accesso ad una area protetta ...<br>";
 // PHP_AUTH_TYPE è disponibile solo quella BASIC
 Header ("WWW-Authenticate: BASIC realm=\"Restricted Area\"", replace);

 Header ("HTTP/1.0 401 Unauthorized");
 echo "Inserire la password per accedere";
  exit;
}

else

{
  // viene controllata la password
  // che abbiamo defino: casa
  if ($PHP_AUTH_PW !="casa")
    exit;

  echo "$PHP_AUTH_USER: la password [$PHP_AUTH_PW] è stata accettata";
 }
?>
```

L'esempio è già commentato, occorre solo osservare che la funzione isset serve a controllare la presenza della variabile nella cache.

Compatibilità dell'autenticazione HTTP? La lettura e scrittura dei dati utente.
Avrete sicuramente compreso che questo codice funziona solo sotto Apache e quando PHP è un suo modulo, cioè nei sistemi UNIX like; ma un passaggio è stato volutamente sottratto: dove sono memorizzate tutte le UID e PWD degli utenti?
PHP non prevede questo o, meglio, bisogna gestirle creando un database, esattamente come si farebbe in ASP o JSP.
MySQL per esempio, oppure un semplice file ASCII in lettura/scrittura utilizzando le funzioni del tutto C like che permettono di scrivervi

dio_open -- Open a new filename with specified permissions of flags and creation permissions of mode
dio_read -- Read n bytes from fd and return them, if n is not specified, read 1k
dio_write -- Write data to fd with optional truncation at length
dio_truncate -- Truncate file descriptor fd TO offset bytes
dio_stat -- Get stat information about the file descriptor fd

dio_seek -- Seek TO pos on fd from whence
dio_fcntl -- Perform a c library fcntl on fd
dio_close -- Close the file descriptor given by fd

Queste sono funzioni da usare con cautela, non tanto per gli errori che si potrebbero commettere in fase di lettura/scrittura, ma per l'uso che ne potrebbe fare un hacker sapendo che sono state implementate, e per il fatto che dovremmo usare una cartella isolata dalla WWWRoot con permesso di lettura/scrittura/modifica per tutti gli utenti (EveryOne). Allora non rimane che usare un database come MySQL o PostgreSQL. Ma una cosa qui non funziona proprio: la inutile ridondanza del codice.
Il pregio del PHP sta tutto nella sua portabilità; se ci priviamo di questa, è meglio utilizzare il il linguaggio di cui nativamanete il server dispone: ASP è di gran di lunga più accettabile su IIS se non ci si pone il problema della portabilità!
Infatti è possibile:

- Usare le variabili di sessione per stabilire un contatto col client e per mantenerne lo stato
- usare un database come MySQL per inserire/modificare i dati utenti
- usare il metodo POST per legare le pagine

Più semplice di così!

Precisazioni sull'installazione di MySQL sotto Windows 2000 Server
Se l'installazione del servizio dovesse fare le bizze in questo ambiente, a causa della vostra versione non proprio compatibile con esso, potete sempre avviare l'utility winmysqladmin.exe nella directory c:\mysql\bin e verificare che il servizio sia veramente presente nei servizi standard sia digitando dal Command Prompt NET START sia usando Administrative Tools\Services.

PHP (parte nona)

---

testo: Francesco Mannarino
mailto:fm@docenti.org

I Socket

Nella parte quinta abbiamo visto come inviare una Mail attraverso una funzione, che a sua volta utilizzava proprio i socket.
Brevemente diremo che i socket sono una interfaccia di comunicazione UNIX per consentire la comunicazione tra processi ed applicazioni attraverso la rete, cosa che permette di colloquiare con un processo funzionante su un'altra macchina servendosi di un protocollo di rete come il TCP/IP. L'applicazione apre un "socket" (innesto) e dichiara l'indirzzo di destinazione, di seguito la rete si occupa di creare la connessione.
Infatti si può benissimo inviare una mail utilizzando direttamente i socket, infatti che cosa è una mail se non una serie di dati da inviare alla porta 25 (Simple Mail Transfer) ?
Per il resto bisogna scriverci dentro (fputs) tutto, incluso le intestazioni, rispettando lo standard per le SMTP (rfc1090, rfc974, rfc822, rfc821 scaricati dal sito AlterNIC e in allegato) ed attendere la risposta (fputs).
Ci sono una quarantina di funzioni, ma la princiaple, quella che permette di aprire la connessione, è
int fsockopen (string [udp://]hostname, int port [, int errno [, string errstr [, float timeout]]])
Gli argomenti sono semplici da comprendere:
L'hostname (mbox.docenti.org p.e.), la porta (25 per l'SMTP), l'errore restituito sottoforma

numerica e descrittiva se la connessione non ha avuto successo, il tempo d'attesa per la chiamata al sistema. La funzione restituisce un *int* di valore booleano (TRUE/FALSE, nr/0), che permette di sapere se la connessione ha avuto successo.
Poi inviare le richieste al server web con

```
int fputs (int fp, string str, int [length])
```

e attendere la risposta con

```
string fgets (int fp, int length)
```

fp è il numero della connessione (file puntato), str la stringa che viene inviata e length la grandezza del buffer
infine si deve chiudere la connessione con

```
bool fclose (int fp)
```

il programmino sockmail.PHP:

```
<?PHP
 // impostazione variabili
 // il mail server e il tempo d'attesa
 // a 30 secondi
 $strMailServer = "mbox.docenti.org";
 $flTempodAttesa = 30;
 $strMyTemp = "";

 // connessione
 // nome host, porta, errore numero, errore descrizione, timeout
 $intMyConn = fsockopen($strMailServer, 25, $intErr, $strErr, $flTempodAttesa);

 // verifica connessione TRUE/FALSE
 if ($intMyConn)
 {
  // connessione avvenuta
  fputs ($intMyConn, "HELO host\n");
  fgets ($intMyConn, 128); // colloca risposta nel buffer

  fputs ($intMyConn, "MAIL FROM: <info@docenti.org>\n");
  fgets ($intMyConn, 128); // colloca risposta nel buffer

  fputs ($intMyConn, "RCPT TO: <fm@docenti.org>\n");
  fgets ($intMyConn, 128); // colloca risposta nel buffer

  fputs ($intMyConn, "DATA\n");
  $strMyTemp = fgets ($intMyConn, 128); // colloca risposta nel buffer

  fputs ($intMyConn, "Subject: socket mail!\n");
  fputs ($intMyConn, "inviata dal web.\n");
  fputs ($intMyConn, "\n");
  fputs ($intMyConn, "Bye\n");
  fputs ($intMyConn, ".\n");

  fputs ($intMyConn, "QUIT\n"); // termine connesisone

  fclose ($intMyConn);
```

```
 }
 else
 {
   // errore
   echo "Connessione al server $strMailServer fallita <br>\n";
   echo "Errore nr $intErr - descr. $strErr <br>\n";
 }
?>
```

Rispetto ad ASP è una operazione visibilente più complessa, dato che PHP utilizza i socket e per forza di cose bisogna scrivere, dopo aver aperto la connessione, tutti i valori da inviare. In ASP è un oggetto CDONTS, preconfezionato ed estremamente semplice.
Chi naturamente volesse utilizzare oggeti preconfezionati in PHP può far uso di PHPNuke (http://www.PHPnuke.org/), che offre svariate soluzioni per il WEB programmabili con pochissime istruzioni: forum, chat, mail, impaginatori, carreli spese, ecc
Indubbiamente PHP offre soluzioni molto potenti, ma scarseggia fortemente, di oggetti che devono essere costruiti a mano.
Altresì si possono richiedere anche infomazioni sul proprio server, utilizzando tutte le porte disponibili (svrinfo.PHP):

```
<?PHP
 $strHostName = "localhost";
 function GetMyWebServices($strHostName, $strService, $intPort)
 {
  $intMyConn = fsockopen($strHostName, $intPort);
  if ($intMyConn)
   {echo "Servizio: <i>$strService su porta $intPort</i> <b>attivo</b><br>\n";
    fclose ($intMyConn);
   }
  else
   {echo "Servizio: <i>$strService su porta $intPort</i> non attivo<br>\n"; }

 }

 echo "Host Name: <i>$strHostName</i><br>\n";
  $strMyConn = GetMyWebServices($strHostName, "http", 80);
  $strMyConn = GetMyWebServices($strHostName, "ftp", 21);
  $strMyConn = GetMyWebServices($strHostName, "smtp", 25);
  $strMyConn = GetMyWebServices($strHostName, "pop3", 110);
  $strMyConn = GetMyWebServices($strHostName, "ssh", 22);
  $strMyConn = GetMyWebServices($strHostName, "mysql", 3306);
  $strMyConn = GetMyWebServices($strHostName, "news", 119);
?>
```

L'esempio si commenta da sè, come anche il risultato del mio server (Vedi Figura 9_1)

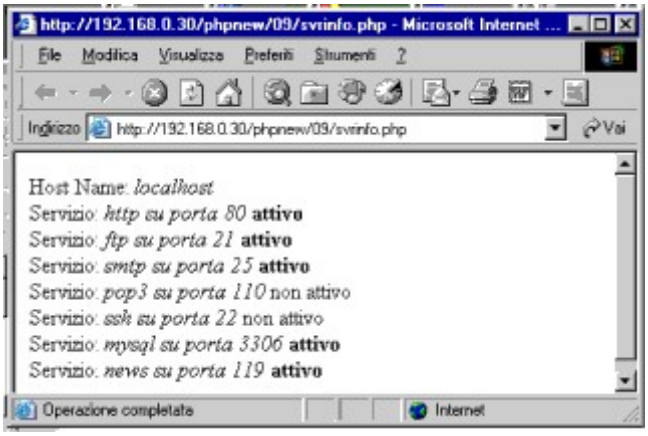

Figura 9_1

PHP (parte decima)

testo: Francesco Mannarino
mailto:fm@docenti.org

Il debugger

Il PHP si può dire che abbia già un debugger integrato nella configurazione standard, dato che segnala con precisione la riga di codice che il parser non è riuscito a mandar giù assieme ad una microscopica descrizione.
Ma per gli errori più complessi, quelli che si evidenziano solo in particolari condizioni, bisogna abilitarlo agendo sul file di configurazione PHP.ini e su quello di Apache se vi gira su. Così è possibile prelevare le informazioni tramite il listener socket, evitando di far apparire sulla pagina web l'orrendo output degli errori tracciati.
Nel caso in cui non si fosse soddisfatti di questa soluzione, si può utilizzare un debugger di terze parti DBG, cheoffre un ambiente di debugging sufficiemente versatile ed affidabile, ma non è molto preciso sulle indicazioni di configurazione, infatti la libreria (PHP_dbg.dll) che secondo la faq dovrebbe esser collocato in una sotto directory di PHP, invece per funzionare a dovere deve essere copiato in c:\winnt\system32. Sotto Linux non v'è stato alcun problema nell'installazione, segno che questo tool ancor prima d'essere stato creato per la piattaforma Microsoft, lo è stato per Linux.

L'editor

Il *notepad* è forse il migliore assieme al *wordpad* che offre, nel caso in cui vi fosse la necessità, anche il salvataggio nel formato ASCII UNIX. Sembra promettere bene il Maguma Studio, ma non sono riuscito ad apprezzare pienamente la versione che ho utilizzato, la light, che per chi è abituato a scrivere tutto a mano, non offre nulla di più di quanto ci si aspetterebbe: codice e tool pronti all'uso.
Invece ho potuto pienemente apprezzare ConTEXT, che ha un grande pregio: l'highlight colorato di identificatori, stringhe, numeri, commenti, ... l'identazione automatica, senza aver pretese di far tutto da sè, supportando il C/C++, Fortran, HTML, Java, JavaScript, ObjectPascal, PHP, Perl, Python, SQL, Tcl/Tk, VB, X86 ASM e XML ed è, come i precedenti, anche freeware (Vedi Figura 10_1).

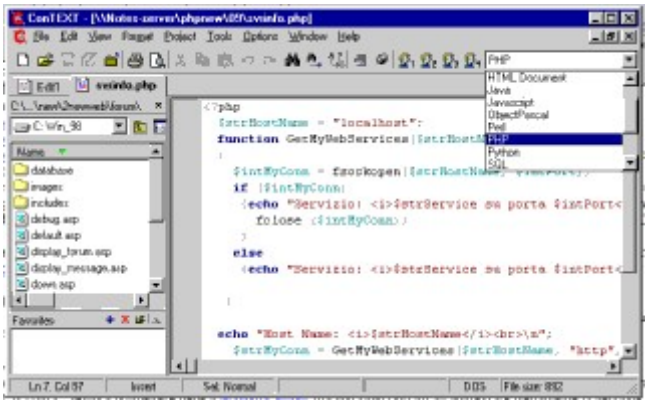

Figura 10_1

Veramente superbo è il *KEdit* sotto Linux (RH 7.2), un pluri editor per linguaggi di programmazione con Konsole inclusa: in un SO in cui operare a linea di comando è una necessità, questo editor è un vero aiuto!

Ma uno degli editor più interessanti sia per Windows che per Linux è SciTE, che supporta numerosissimi linguaggi pur essendo di modeste dimensioni ma velocissimo, prelevabile dal sito http://www.scintilla.org (Vedi Figura 10_2)

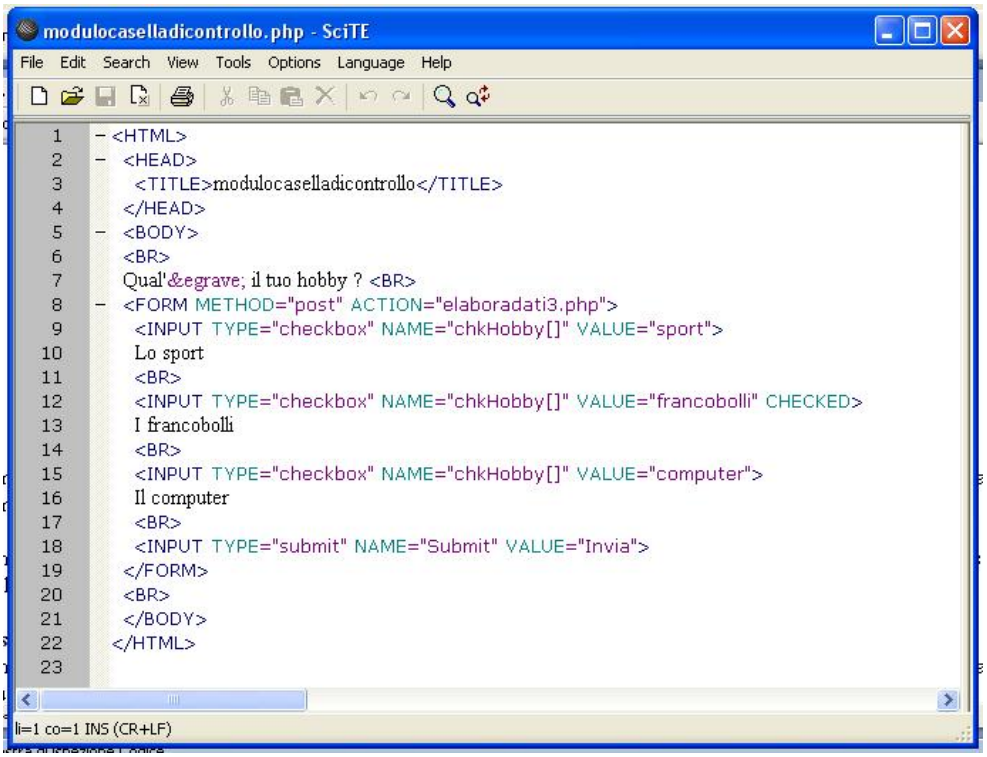

Figura 10_2

Le risorse

Per uno strumento così potente non potevano mancare bibliografia e siti di enorme interesse, tutti utili per affrontare con agilità tutte le problematiche che il linguaggio offre.
Il punto di partenza è sicuramente il sito produttore
http://www.PHP.net/ dove c'è il manuale anche in italiano, poi Zend Technologies, Apache, MySQL, PostgreSQL.
http://www.PHPbuilder.com, chiamarla una risorsa è poco, in quanto è un vero laboratorio d'idee e

di codice sorgente.
http://www.PHPcenter.it/, http://www.PHPnuke.org/, http://freePHP.HTML.it/, http://www.nukeitalia.com/ sono un valido complemento ai siti precedentemente citati
Di testi e di case editrici ve ne sono numerose in giro, la Wrox si eleva sopra tutte per la sua professionalità (se volete del codice sorgente gratis qui ve n'è parecchio), come anche la Sams, famosa per le sue guide iperveloci e di buon livello. Buone le traduzioni italiane della Apogeo con Maslakowski, MySQL - Guida completa, che offre nel brevissimo spazio di 450 pagine una guida esaustiva su MySQL e una praticissima anche parecchio breve proprio su PHP. Da studiare il testo di Tim Converse e Joyce Park - Guida a PHP 5 - Mc Graw Hill Italia.

Per una guida veloce e precisa rifersi a questo libro

Autore: Francesco Mannarino

Formato
cm 12 x 16
Pagine 160
ISBN 88-487-3526-6

solo € 5.99

http://www.finson.it
http://www.finson.it/prodotti/catalogo/intasca/intasca.asp

Il PHP è una tecnologia Web Server OpenSource che si sta diffondendo rapidamente grazie alla sua semplicità e alla sua portabilità. Con esso è possibile creare pagine Web dinamiche complesse e veri programmi che possono interagire con l'utente. Questo tascabile di pone l'obiettivo di introdurre il lettore nel mondo PHP 4/5 in modo semplice e chiaro anche per chi ha poca dimestichezza con i linguaggi di programmazione.

e di prossima uscita il volume di circa 600 pagine della medesima casa editrice
PHP Professionale collana the Big One
Autore: Francesco Mannarino

Via Cavalcanti, 5 - 20127 Milano
Telefono: 02.283.1121 (r.a.)
Email: finson@finson.it
U.R.L. http://www.finson.com

Questo testo si propone di guidare il lettore nel mondo di PHP 4 /5 in modo semplice e chiaro con frequenti suggerimenti sia sullo stile che sulle tecniche di programmazione. Gli argomenti sviluppati nel testo riguardano l'installazione, la sintassi, i tipi, il controllo del flusso, il mantenimento dello stato, la gestione dell'I/O, il database MySQL, la programmazione orientata agli oggetti ed le funzioni di PHP.
Il testo segue un percorso graduale, distinto in tre sezioni: i capitoli che vanno dal primo al terzo trattano delle basi del linguaggio PHP: l'ambiente PHP, il linguaggio, il controllo del flusso e le strutture decisionali; quelli che vanno dal capitolo quarto all'ottavo trattano argomenti avanzati: comunicazione con l'utente, il mantenimento dello stato, il database MySQL, PHP con MySQL,

programmazione Object Oriented; i capitoli restanti, dal decimo al dodicesimo, trattano delle funzioni necessarie e maggiormente usate: funzioni sulle stringhe, funzioni sugli array, funzioni matematiche, espressioni regolari, funzioni di sistema, di data/ora e di rete.

Infine se si ha bisogno d'aiuto è meglio rivolgersi ad un newsgroup, gentilmente ed in modo pertinente, dopo aver letto manifesto e faq, it.comp.www.PHP