WEP — WEb Programming

4 — PHP: dati e form

Lucidi per il corso di Basi di Dati tenuto da Paolo Baldan presso l'Università di Padova, anno accademico 2008/09

PHP e MySQL

Perché?

- Le applicazioni hanno bisogno di operare su dati persistenti
- ... e per gestire moli significative di dati (condivisi) la soluzione ovvia è un DB
- È opportuno separare logicamente (e fisicamente) la logica dell'applicazione, dalla gestione dei dati
- (cfr. Architetture software, es: Architettura three-tier, MVC (model-view-control, ...)

Come?

- L'accesso a un server MySQL da PHP tramite un API
- La sequenza dei passi da effettuare è
  - Effettuare una **connessione** al server
  - **Selezionare il DB** o crearlo se non esiste
  - **Eseguire** la/le query (creare tabelle, inserire o selezionare dati…)
  - Nel caso di una select, **elaborazione dei dati recuperati**
    - con logica a **cursore**
    - memorizzandoli in un array

Connessione

- Attiva una connessione e restituisce un **identificatore** per questa o una segnalazione di **errore** (FALSE)
- l'identificatore della connessione sarà usato in tutti gli accessi successivi

```
/* parametri per la connessione */
$host="localhost";          /* server MySQL */
$user="fabio";              /* utente       */
$pwd= "xxxxx";              /* password     */

/* connessione al server */
$conn=mysql_connect($host, $user, $pwd)
  or die($_SERVER['PHP_SELF'] . "Connessione fallita!");
```

Messaggi di errore

- Può essere comodo definire una funzione che dia informazioni dettagliate e personalizzate sull'errore

```
function fail($msg) {
  die($_SERVER['PHP_SELF'] . ": $msg<BR />");
 }
```

- da includere nei vari script

```
require('Errors.php');
...
...
/* connessione al server */
$conn=mysql_connect($host, $user, $pwd)
  or fail("Connessione fallita!");
```

Esecuzione di una query


```
/* seleziona il database da usare */

$dbname="Univ";
mysql_select_db($dbname);



/* prepara lo statement: media dei voti degli studenti di una
   data provincia */

$prov="SP";

$query="SELECT s.Matricola, s.Nome, s.Cognome,
               ROUND(AVG(e.Voto)) AS Media
        FROM Studenti s JOIN ESAMI e ON (s.Matricola=e.Candidato)
        WHERE s.Provincia=\"$prov\"
        GROUP BY s.Matricola, s.Nome, s.Cognome";
```

Esecuzione di una query


```
/* Stampa la query a video ... utile per debug */

echo "<B>Query</B>: $query <BR />";



/* ... e la esegue, ottenendo un handler per i risultati */

$studenti = mysql_query($query,$conn)
  or die("Query fallita" . mysql_error($conn));
```

Qualche risultato?

- Si può controllare la dimensione della tabella restituita dalla query eseguita con

```
mysql_num_rows($risultato)
```

- Continuando l'esempio ...

```
/* numero di righe nel risultato */
$num_righe=mysql_num_rows($studenti);

if (! $num_righe)
  echo "<P>Nessuno studente della provincia di $prov
            ha fatto esami.</P>";
else {
  /* gestione del risultato non vuoto */
}
```

Recuperare i risultati

- Il risultato di una query **SELECT** si elabora con una logica a cursore
- Per leggere la riga corrente in un array e posizionarsi sulla prossima
  - `$row = mysql_fetch_row($result)`

    ritorna la riga corrente come un array enumerativo ($row[0] = primo campo, $row[1] secondo campo, etc.)
  - `$row = mysql_fetch_assoc($result)`

    ritorna la riga corrente come un array associativo, indicizzato dai nomi dei campi
  - `$row = mysql_fetch_array($result)`

    ritorna un array enumerato e associativo

Recuperare i risultati: enum


```
if (! $num_righe)
  ...
else {
  echo "<P>Trovati $num_righe studenti di $prov
           che hanno fatto esami.<BR />";

  echo "Ecco le loro medie:<BR />";

  while ($row = mysql_fetch_row($studenti)) {

    $matricola=$row[0];     /* primo campo   */
    $nome=$row[1];          /* secondo campo */
    $cognome=$row[2];       /* ...           */
    $media=$row[3];         /* ...           */

    echo "$matricola - $nome $cognome - $media<BR />";
  };

  echo "</P>";
```

MySQL1.php

Recuperare i risultati: assoc


```
if (! $num_righe)
  ...
else {
  echo "<P>Trovati $num_righe studenti di $prov
           che hanno fatto esami.<BR />\n";

  echo "Ecco le loro medie:<BR />";

  while ($row = mysql_fetch_assoc($studenti)) {

    $matricola=$row['Matricola'];
    $nome=$row['Nome'];
    $cognome=$row['Cognome'];
    $media=$row['Media'];

    echo "$matricola - $nome $cognome - $media<BR />";
  };

  echo "</P>";
```

Recuperare i risultati: array


```
if (! $num_righe)
  ...
else {
  echo "<P>Trovati $num_righe studenti di $prov
           che hanno fatto esami.<BR />\n";

  echo "Ecco le loro medie:<BR />";

  while ($row = mysql_fetch_array($studenti)) {

    $matricola=$row['Matricola'];
    $nome=$row[1];
    $cognome=$row['Cognome'];
    $media=$row[3];

    echo "$matricola - $nome $cognome - $media<BR />";
  };

  echo "</P>";
```

Visualizzare i risultati in una tabella HTML


```
/* funzione per stampare un array, come riga di tabella html */
function echo_row($row) {
  echo "<TR>";
  foreach ($row as $field)
    echo "<TD>$field</TD>";
  echo "</TR>";
};
...
```

```
/* Intestazione della tabella */
echo "
<TABLE border=\"1\">
<TR>
  <TH>Matricola</TH>
  <TH>Nome</TH>
  <TH>Cognome</TH>
  <TH>Media</TH>
</TR>";
```

Visualizzare i risultati in una tabella HTML


```
/* funzione per stampare un array, come riga di tabella html */
function echo_row($row) {
  echo "<TR>";
  foreach ($row as $field)
    echo "<TD>$field</TD>";
  echo "</TR>";
};
...
```

```
/* Intestazione della tabella con "here document" */
echo <<<END
<TABLE border="1">
<TR>
  <TH>Matricola</TH>
  <TH>Nome</TH>
  <TH>Cognome</TH>
  <TH>Media</TH>
</TR>
END;
```

Visualizzare i risultati in una tabella HTML


```
/* stampa le righe della tabella */

while ($row = mysql_fetch_row($studenti))
    echo_row($row);

echo "</TABLE>";
```

MySQL2.php

- **Nota**: L'uso di `mysql_fetch_row` è importante per il funzionamento di `echo_row`

LIMIT

- Talvolta interessa un numero limitato di record come risposta da una select
- Questo può essere ottenuto con il costrutto SQL **`LIMIT start, max`**

```
SELECT *
FROM   Studenti
LIMIT  0,9;
```

- Il primo argomento della LIMIT è l'offset (la riga di partenza) l'altro è il massimo numero di righe da ritornare
- Restituisce solo i record dal primo (indice 0) al decimo

Altre interrogazioni

- Con lo stesso meccanismo si possono eseguire altri statement SQL

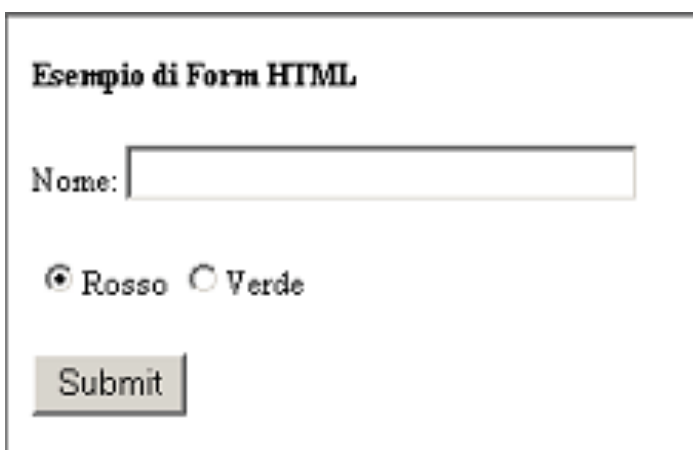
```
/* inserimento ... */

$query="INSERT INTO Docenti VALUES
        (\"MM1\", \"Mino\",\"Monti\")";

$ins=mysql_query($query,$conn)
  or die("Inserimento fallito" . mysql_error($conn));
```

```
/* ... e la esegue, ottenendo un handler per i risultati */

$query="DELETE FROM Docenti
        WHERE CodDoc=\"MM1\"";

$del=mysql_query($query,$conn)
  or die("Cancellazione fallita" . mysql_error($conn));
```

Interazione con l'utente

Form

- L'interazione con l'utente avviene principalmente mediante l'uso di form HTML
  - HTML fornisce vari tag per visualizzare e formattare opportunamente le FORM
  - Quando l'utente "conferma" i dati nella form, le informazioni vengono codificate e inviate al server tramite HTTP
  - Il server elabora i dati e li gestisce (nel nostro caso tramite PHP)




```
<FORM  action="pagina.php"
       method="GET"|"POST">
<INPUT type="text" name="username" />
<INPUT type="radio" name="color"
       value="Rosso">
…
<INPUT type="submit" />
```

Form

- Una form ha in generale la seguente struttura

```
<FORM action="manage_form.php"  method="GET|POST">

input_1: <INPUT type="..." name="campo_1">

   ...

input_n: <INPUT type="..." name="campo_n">

<INPUT type="submit" value="Procedi">
<INPUT type="reset"  value="Cancella">

</FORM>
```

Input

- L'input può essere di vari tipi
  - Text
  - Password
  - Textarea
  - Radio
  - Checkbox
  - Selection/Option (menù)
  - ...

Text, password, textarea


```
<FORM method="GET" action="manage.php">

Nome <INPUT type="text" name="username">
<BR /><BR/ >

Password <INPUT type="password" name="password"
                maxlength="8" size="8">
<BR /><BR/ >

Commenti<BR />
<TEXTAREA name="comments" rows="7" cols="40">
</TEXTAREA>

<BR />
<INPUT type="submit" value="Invia">
<INPUT type="reset" value="cancella">

</FORM>
```

FormText.html

Radio, Checkbox, Option


```
<FORM method="GET"  action="manage.php">

  Iniziali del nome:
  <SELECT name="nome">
  <OPTION>---</OPTION>
  <OPTION>A-H</OPTION>
  <OPTION>I-Z</OPTION>
  </SELECT>

  Provincia:
  <INPUT type="radio" name="prov" value="PI" /> PI
  <INPUT type="radio" name="prov" value="SP" /> SP

  <INPUT type="checkbox" name="tutor[]" value="HAS"/> Ha tutor
  <INPUT type="checkbox" name="tutor[]" value="IS" /> E' tutor
  ...

</FORM>
```

FormChoice.html

Passaggi parametri: GET e POST

- **GET**
  - i parametri sono passati accodandoli alla URL del gestore della form

```
http://localhost/manage.php?dato1=pippo&dato2=pluto
```

  - la stringa dei parametri (**visibile** nel browser) viene detta **query string**
  - può essere costruita artificialmente o inserita in un bookmark
  - lunghezza massima querystring 256 caratteri
- **POST**
  - i parametri vengono passati direttamente tramite protocollo HTTP
  - non sono visibili

Gestione dei parametri

- Uno script PHP può accedere ai parametri in tre modi
  - `$_POST["nomepar"]` per il metodo POST
  - `$_GET["nomepar"]` per il metodo GET
- Oppure accedendo all'array globale delle richieste
  - `$_REQUEST["nomepar"]`
  - vale per entrambi i metodi
  - lo script PHP è indipendente dal metodo usato dalla FORM
- Tutti i parametri di passaggio nelle form sono nell'ambiente predefinito di php e sono quindi visualizzabili con la funzione `phpinfo()`

Pagine per form

- La gestione delle form prevede due passi
  - la visualizzazione della FORM in HTML
  - gestione dei parametri in PHP
- Soluzione tipica: Due pagine distinte
  - una in HTML (estensione .html)
  - l'altra come pagina PHP (estensione .php)

Esempio di Form


```
<FORM action="FormStudenti.php"  method="GET">
  Cognome:   <INPUT type="text" name="cognome"><BR />
  Matricola: <INPUT type="text" name="matricola" maxlength="6">
  <BR /><BR />

  Iniziale del nome:
  <SELECT name="nome">
    <OPTION>---</OPTION>
    <OPTION>A-H</OPTION>
    <OPTION>I-Z</OPTION>
  </SELECT>
  <BR /><BR />

  Provincia:
  <INPUT type="radio" name="prov" value="PI" /> PI
  <INPUT type="radio" name="prov" value="SP" /> SP
  <BR /><BR />
```

FormStudenti.html

Esempio di Form


```
  <INPUT type="checkbox" name="tutor[]" value="HAS" />
  Ha un tutor

  <INPUT type="checkbox" name="tutor[]" value="IS" />
  E' tutor <BR /><BR />

  <TEXTAREA name="commento" rows="5">
  Mah ...
  </TEXTAREA> <BR /><BR />

  <INPUT type="submit" value="Invia">
</FORM>
```

FormStudenti.html

Gestore della Form: Recupero dei parametri

- I parametri della form possono essere recuperati dall'array **$_GET**

```
<?PHP
$cognome = $_GET["cognome"];
$matricola = $_GET["matricola"];
$nome = $_GET["nome"];
$provincia = $_GET["prov"];
```

- Nel caso di scelte multiple (es. checkbox e select), il parametro è un array

```
if ($_GET["tutor"]) {
   $hatutor = in_array('HAS', $_GET["tutor"]);
   $etutor =  in_array('IS',  $_GET["tutor"]);
};
$commento = $_GET["commento"];
```

Gestore della Form: Recupero dei parametri

- In fase di debug è una buona idea stampare i parametri della form
- Qui lo facciamo con HERE document

```
echo<<<END
Ecco i parametri:<BR />
<UL>
  <LI>Cognome:  $cognome</LI>
  <LI>Iniziali del nome: $nome</LI>
  <LI>Matricola: $matricola</LI>
  <LI>Provincia: $prov</LI>
  <LI>Ha tutor?: $hatutor</LI>
  <LI>&Egrave; un tutor?: $etutor</LI>
  <LI>Commento: $commento
  </LI>
</UL>
END;
```

Gestore della Form: Costruzione della query

- Costruzione della query secondo le esigenze dell'utente

```
$query="SELECT DISTINCT s.* FROM Studenti s";


/* da aggiungere in JOIN se si vuole che lo studente sia
   un Tutor */
if ($etutor)
  $join=" JOIN Studenti s1 ON (s.Matricola = s1.Tutor)";


/* clausola WHERE */
$where=" WHERE TRUE";

/* verifica se c'e` un vincolo sul cognome ed eventualmente
   lo aggiunge al WHERE */
 if ($cognome)
   $where .= " AND  s.cognome =\"".  $cognome . "\"";
```

Gestore della Form: Costruzione della query


```
/* verifica se c'e` un vincolo su matricola ed
   eventualmente lo aggiunge al WHERE */
if ($matricola)
  $where .= " AND  s.matricola =\"".  $matricola . "\"";


/* verifica se c'e` un vincolo su provincia ed
   eventualmente lo aggiunge al WHERE */
if ($prov)
  $where .= "AND  s.Provincia =\".  $prov . "\"";
```

Gestore della Form: Costruzione della query


```
/* da aggiungere nel WHERE se vogliamo che lo studente
   abbia un nome con iniziali prefissate */
switch ($nome) {
case 'A-H':
  $where .= " AND (s.Nome REGEXP \"[A-H].*\")";
  break;

case 'I-Z':
  $where .= " AND (s.Nome REGEXP \"[I-Z].*\")";
  break;

/* se ($nome='---') non inserisce niente nel where! */
}
```

Gestore della Form: Costruzione della query

- Costruzione della query secondo le esigenze dell'utente

```
/* da aggiungere nel WHERE se vogliamo che lo studente
   abbia un tutor */
if ($hatutor)
  $where .= " AND (s.Tutor IS NOT NULL)";

/* completa la query */
$query = $query . $join . $where;

/* come al solito conviene stamparla ... */
echo "<B>Query</B>: $query";

/* e la esegue */
$studenti = mysql_query($query,$conn)
   or die("Query fallita" . mysql_error($conn));
```

Gestore della Form: Output dei risultati

- Funzioni di supporto per la gestione delle tabelle HTML

```
/* Inizia una tabella html. In input l'array degli
   header delle colonne */
function table_start($row) {
  echo "<TABLE border=\"1\">";
  echo "<TR>";
  foreach ($row as $field)
    echo "<TH>$field</TH>";
  echo "</TR>";
};
```

table_fun.php

Gestore della Form: Output dei risultati

- Funzioni di supporto per la gestione delle tabelle HTML

```
/* Stampa un array, come riga di tabella html */
function table_row($row) {
  echo "<TR>";
  foreach ($row as $field)
    if ($field)                /* gestione valori nulli! */
        echo "<TD>$field</TD>";
    else
        echo "<TD>---</TD>";
  echo "</TR>";
};

/* funzione per terminare una tabella html */
function table_end() {
  echo "</TABLE>";
};
```

table_fun.php

Gestore della Form: Output dei risultati

- Occorre includere il file delle funzioni con

```
require("table_fun.php");
```

- Quindi l'ultima parte del codice per l'ouput è

```
/* fornisce in output i risultati in forma di tabella */

table_start(array("Nome", "Cognome",
                  "Matricola", "Nascita",
                  "Provincia", "Tutor"));

while ($row = mysql_fetch_row($studenti))
  table_row($row);

table_end();
```

Gestore della Form: Controllo dei parametri

- Spesso la prima cosa da fare è controllare che i parametri immessi nella form soddisfino i requisiti ...
  - valori nulli
  - campi numerici
  - lunghezza dei campi stringa
  - spazi in eccesso all'inizio o alla fine ... (trim)

Gestore della Form: Controllo dei parametri


```
/* Verifica dei dati immessi */

/* Elimina spazi superflui */
$cognome   = trim($cognome);
$commento  = trim($commento);
$matricola = trim($matricola);

/* la variabile "errore" indica se tra i dati abbiamo
   trovato un errore */
$errore=FALSE;

/* verifica che almeno uno tra matricola e cognome siano
   non vuoti */
if (!$cognome && !$matricola) {
  echo "<B>Errore! Almeno uno tra nome e matricola devono
            essere specificati!</B><BR />";
  $errore=TRUE;
};
```

Gestore della Form: Controllo dei parametri


```
/* verifica che la matricola sia numerica */
if ($matricola && ! ctype_digit($matricola)) {
  echo "<B>Errore! Matricola deve essere numerica!</B><BR />";
  $errore=TRUE;
};

/* ... e il cognome alfabetico */
if (preg_match('/^[a-zA-Z]*$/', $cognome)) {
  echo "<B>Errore! Cognome deve essere alfabetico!</B><BR />";
  $errore=TRUE;
};


if (!$errore) {
  /* Inizia la costruzione della query */
  ...
};
```

Form e gestore in un unico script

- Soluzione diversa: form e gestore in un unico script
- Necessità di distinguere il caso in cui si deve mostrare la form e quello in cui si deve elaborarla
  - **Valore** associato **all'input submit**
  - Uso di **self**

```
if (isset($_REQUEST['submit']))
  # trattamento dei parametri
else {
  # visualizza la form
  echo<<<END
  <FORM method="POST"
      action="$_SERVER['PHP_SELF']">
  ...
  ...
  <INPUT type="submit" name="submit">
END;
};
```

form.php

Form e gestore in un unico script

- Es. mantenere il valore dei campi in caso di errore

```
/* verifica se si esegue lo script per la prima volta:
   in questo caso bisogna presentare la form  */
$first= ! isset($_REQUEST["submit"]);

if (! $first) {
  /* recupera i dati della form */
  $nome=$_REQUEST['nome'];
  $cognome=$_REQUEST['cognome'];

  /* e verifica se il nome soddisfa i criteri */
  $errore_nome = ! preg_match("/^[a-zA-Z]*$/",$nome);
};

...
```

SingleFile.php

Form e gestore in un unico script


```
if ($first || $errore_nome) {
/* se lo script si esegue per la prima volta oppure i dati
   sono errati mostra la form ed eventualmente segnala
   l'errore */

$self = $_SERVER['PHP_SELF']; /* nome dello script corrente */
echo<<<END
<FORM method="POST" ACTION="$self">
  Nome: <INPUT type="text" name="nome" value="$nome">
  Cognome: <INPUT type="text" name="cognome" value="$cognome">
  <INPUT type="submit" name="submit" value="Invia">
  <INPUT type="reset" value="Cancella">
</FORM>
END;

/* e se c'era un errore nei dati lo segnala */
...
```

SingleFile.php

Form e gestore in un unico script


```
/* e se c'era un errore nei dati lo segnala */
  if ($errore_nome)
    echo "<B>Errore nel nome!!! Deve essere alfabetico!</B>";
}

else { /* ! $first && ! $errore_nome */
  /* se invece i dati ci sono e sono corretti, li elabora */
  echo "elaboro i dati <BR />";
  echo "Nome: $nome<BR />";
  echo "Cognome: $cognome<BR />";
};
```

SingleFile.php

Invio di mail

- Da uno script php possiamo inviare una email con la funzione

```
mail (to, subject, body, headers)
```

- oppure

```
mail (to, subject, body, headers)
```

```
$to      = 'fabio@cli.di.unipi.it';
$subject = 'Ieri';
$message = 'Ciao, come va?';
$headers = 'From: webmaster@bd.com' . "\r\n" .
           'Reply-To: webmaster@bd.com';

mail($to, $subject, $message, $headers);
```

Upload di file: form

- Occorre utilizzare una form HTML adeguata ...

```
<FORM enctype="multipart/form-data"
      action="Upload.php" method="POST">

  Nome del file:
  <INPUT type="hidden" name="MAX_FILE_SIZE" value="30000" />
  <INPUT type="file" name="myfile" />
  <BR />

  <INPUT type="submit" name="submit" value="Invia" />
</FORM>
```

Upload.html

Upload di file

- I dati relativi ai file inviati sono accessibili mediante la variabile **`$_FILES`**
  - **`$_FILES['userfile']['name']`**

    Il nome del file sulla macchina di origine
  - **`$_FILES['userfile']['type']`**

    Il mime type del file (es. "image/gif"). Non sempre affidabile
  - **`$_FILES['userfile']['size']`**

    La dimensione del file
  - **`$_FILES['userfile']['tmp_name']`**

    Nome temporaneo del file sul server
  - **`$_FILES['userfile']['error']`**

    Codice di errore associato all'upload

Upload di file: recupero dei dati


```php
/* dimensione massima di upload */
define(DIM_MAX,30000);

/* directory locale per la memorizzazione */
$localdir="upload";

/* informazioni sul file */

$error= $_FILES["myfile"]["error"];    /* codice di errore */
$size = $_FILES["myfile"]["size"];     /* dimensione       */
$type = $_FILES["myfile"]["type"];     /* mime-type        */
$name = $_FILES["myfile"]["name"];     /* nome sul client  */
$tmp  = $_FILES["myfile"]["tmp_name"]; /* nome sul server  */
```

Upload.php

Upload di file: error check


```
/* verif. se il file uploaded ha caratteristiche appropriate */

/* si e` verificato un errore ? */
if ($error != UPLOAD_ERR_OK)  {
  echo "Errore di upload. Codice: $error<BR />";
}
/* tipo corretto ? */
elseif (($type != "image/gif")
       && ($type != "image/jpeg")) {
  echo "Errore: File di tipo $type. Errato! <BR />";
}
/* dimensione */
elseif ($size > DIM_MAX) {
  echo "Errore: File troppo grande ($size bytes)! <BR />";
}
else {
  /* Tutto ok! Mostra i dati */
```

Upload.php

Upload di file: recupera il file


```
  /* recupera il file */

  /* se non è già presente ... */
  if (file_exists($locadir . "/" . $name))
    {
      echo $name . " gia` presente. ";
    }
  else
    {
     /* lo porta a destinazione ... */
      move_uploaded_file($tmp, $localdir . "/" . $name);
      echo "Memorizzato in: " . $localdir . "/" . $name;
    }
};
```

Upload.php

Upload di file

- Quando si gestiscono operazioni invasive come un upload, può essere opportuno tenere un logfile
- Supporto in PHP per operare sul log di sistema (**/var/log/system.log**)
- **Esempio**
  - **openlog("Upload.php", LOG_PID, LOG_LOCAL0)**
  - **...**
  - **syslog(LOG_WARNING,**
    "Upload di $name da parte di " . $_SERVER["SERVER_ADDR"]);
  - **closelog()**