Laboratorio di Basi di Dati

PHP: Hypertext Preprocessor

Anno accademico 2014-15

Marco Mesiti

Parte di questi lucidi è tratta da una versione precedente di Marco Mesiti, Stefano Valtolina e Sergio Mascetti

La volta scorsa abbiamo visto

- Differenza tra
  - Pagine statiche
  - Pagine dinamiche:
    - script eseguito lato server
    - viene generato del codice HTML che viene inviato al client
- Introduzione ad alcune tecnologie lato client: HTML, CSS, Javascript

Programma

- Vediamo PHP come linguaggio di scripting lato server
  - costrutti di base del linguaggio
  - gestione dei cookie e delle sessioni
  - In seguito: interazione con una base di dati

Prima parte:
introduzione a XAMPP e PHP

XAMPP

- XAMPP *(pronunciato ZAMP)* è una distribuzione di Apache ridotta e semplice che contiene in un solo pacchetto i più comuni strumenti di sviluppo per applicativi Web
- E' l'acronimo per :
  - X- letto come "cross" e con il significato di cross-platform
  - A-Apache HTTP Server
  - M-MySQL
  - P-PHP
  - P-Perl

XAMPP

- XAMPP è una piattaforma di sviluppo Web
- Permette di far funzionare localmente degli script PHP **senza connettersi ad un server esterno**
- E' un ambiente di sviluppo che comprende:
  - Un web server (**Apache**)
  - Un database server (**MySQL**)
  - Un interprete di script **PHP**
  - Un amministratore grafico di MySQL (**phpMyAdmin**)

XAMPP: caratteristiche principali

- E’ freeware (http://www.apachefriends.org/en/xampp.html) con licenza GPL
- Sulle macchine del laboratorio è installata la versione **1.8.3** che contiene in un unico pacchetto:
  - Apache 2.4.10
  - MySQL 5.6.20
  - PHP 5.5.15
  - phpMyAdmin 4.2.7.1

XAMPP: caratteristiche principali

- Il software ha l'enorme vantaggio di installare tutti i software necessari per la progettazione e il funzionamento di un sito web in locale. **Il pc diventa client e server**
- Il server Apache crea automaticamente di default un dominio virtuale (in locale) all'indirizzo **localhost** (http://127.0.0.1)
- E' possibile vedere le pagine progettate semplicemente digitando l'indirizzo http://localhost nella barra degli indirizzi del proprio browser
- Le pagine devono essere inserite nella directory:
- *C:\xampp\htdocs*

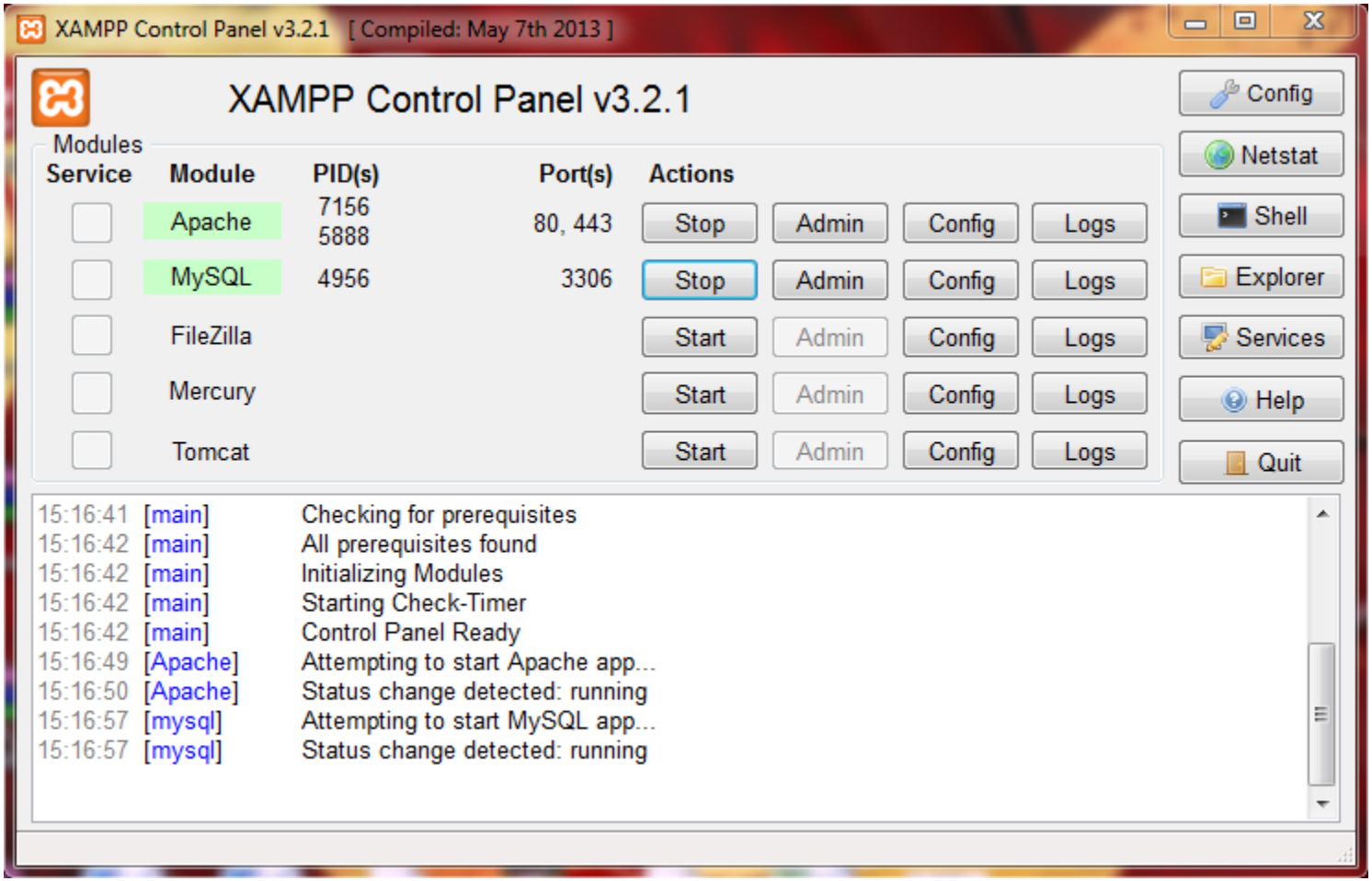
XAMPP: installazione

- Scaricare XAMPP dal sito

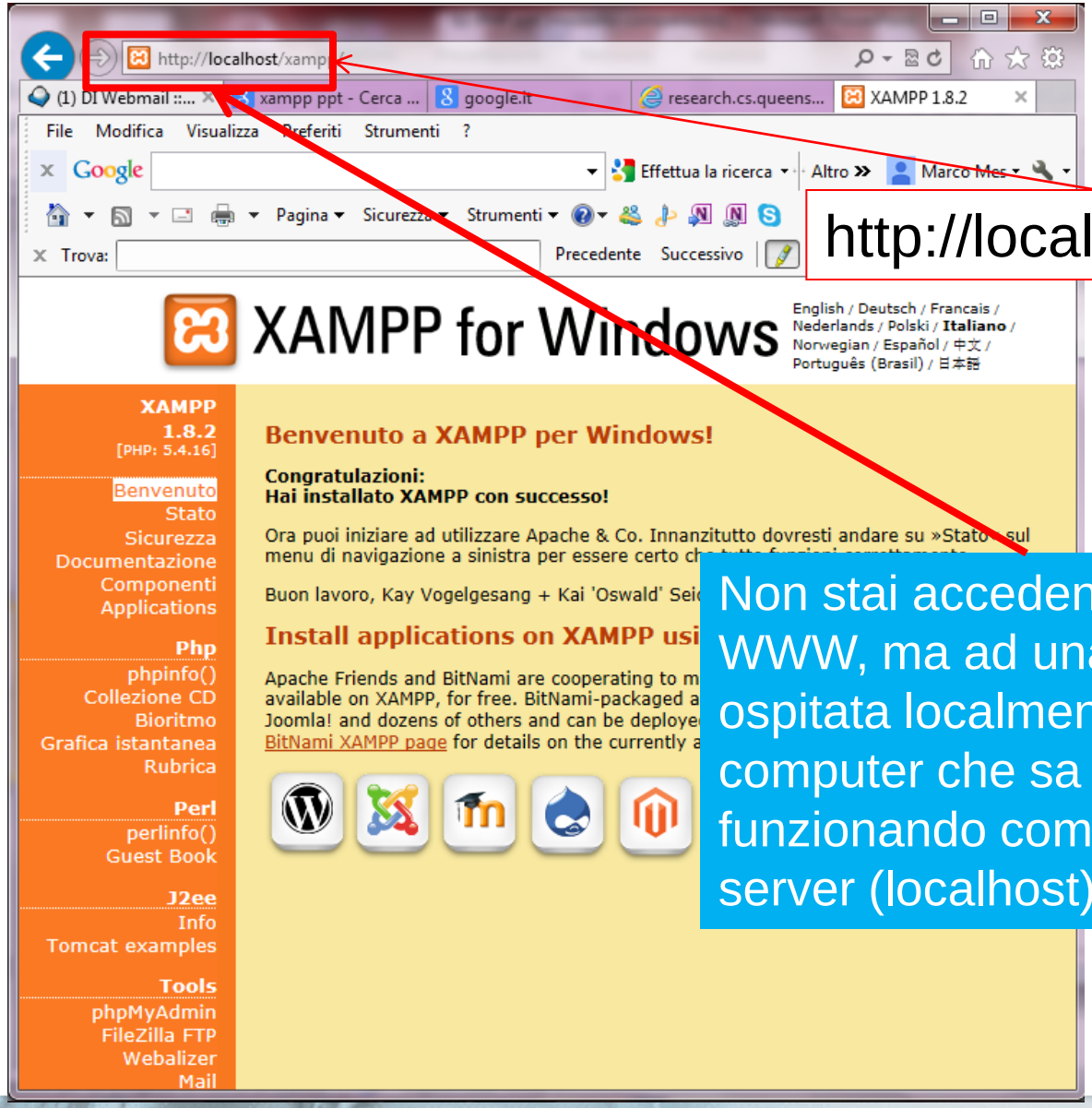
http://www.apachefriends.org/en/xampp.html

- Versioni disponibili per Linux, Windows, Solaris, Mac OS X
- Utilizzare l'installer (sotto windows) per effettuare l'installazione, scegliendo le opzioni di default
- Dopo l'installazione troverete XAMPP sotto Start/Programs/ XAMPP
- Potete usare il pannello di controllo per attivare o interrompere i servizi
- Far partire Apache

XAMPP: il pannello di controllo



XAMPP: Localhost



http://localhost/xamp

Non stai accedendo al WWW, ma ad una pagina ospitata localmente sul tuo computer che sa funzionando come Web server (localhost)

Il tuo primo sito

- Prendi dei file HTML che hai creato la lezione scorsa e assegnagli il nome ciao.html
  - se non ce l'hai creane uno nuovo
  - copia il file come ciao.html nella cartella *"C:"\xampp\htdocs*
- Nel browser apri la pagina http://localhost/ciao.html
- Vedi la tua pagina?

PHP

- E’ un software open source (OSS)
- I file PHP
  - contengono testo, tags HTML e scripts
  - una volta eseguiti gli script lato server, ritornano al client HTML puro
  - hanno estensione ".php", ".php3", o ".phtml“
- PHP supporta l’interazione con diversi database (MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, etc.)

Sintassi Base

- Un file PHP consiste generalmente di
  - tag HTML e
  - frammenti di codice di scripting PHP
- Un frammento di scripting PHP
  - inizia sempre con il tag “<?php”
  - termina con il tag “?>”
  - può essere posizionato in qualsiasi punto del documento

Sintassi di base

- Ogni linea di codice PHP deve terminare con un “;”
- Start ed end tags alternativi:
  - <script language="php"> ... </script>
  - <? ... ?> (abilitare il parametro *short_open_tag* di *php.ini*)
  - <% ... %> (abilitare il parametro *short_open_tag* di *php.ini*)
- PHP reindirizza l’output attraverso le funzioni echo e print.

Sintassi Base: esempio `hello.php`

```
<html>
   <body>

        <?php echo "Hello World"; ?>

   </body>
</html>
```

Le variabili

- Le variabili cominciano col simbolo $
  - I nomi delle variabili sono case sensitive
- Assegnazioni di informazioni
  - $variabile=…
- PHP è un linguaggio loose typed
  - le variabili non vengono definite dal programmatore, ma il loro tipo è deciso a runtime dall'interprete in base al contesto
  - Es. $number = 25;
  - Es. $number = "questa è una stringa";

Varibili e costanti

- Per concatenare due o più variabili si usa l'operatore punto ".". Esempio:
  - $nome_completo=$nome.$cognome;
- Definizione costanti:
  - define ("<nome costante>", <valore costante>);
- Per recuperare il valore di una costante si usa il nome della costante. Esempio:

```
define ("FILM", "CODICE D'ONORE");

echo "Il film ". FILM;
```

Commenti

- Simbolo “//”
  - singola linea di commento
- Simboli “/*” e “*/”
  - commento costituito da più linee.

Variabili, costanti e commenti: esempio

```
<html> <body>
    <?php
    define (“world", "Hello World");
    $txt2="1234";
    echo world . " " . $txt2 ;
    //This is a comment
    /* This is a
    comment block */
    ?>
</body> </html>
```

Tipi di Dato

- Tipo Boolean
  - Può assumere solo due valori: TRUE o FALSE
- Tipo Integer
  - Può assumere un valore qualsiasi appartenente all'insieme numerico Z={...-1,0,1...}
- Tipo Float
  - Permette di esprimere i numeri in virgola mobile; "double" è un sinonimo
- Tipo String
  - Una stringa è un insieme di caratteri a 8 bit

Tipi di Dato

- Tipo Resource
  - Rappresenta un puntatore ad una risorsa esterna
- Tipo Null
  - Tipo speciale usato per denotare che una variabile contiene un puntatore di valore nullo, ovvero non punta ad alcun oggetto
- Tipi complessi: Object ed Array

Tipo di Dato String

- Una stringa è una sequenza arbitraria di caratteri ASCII. Può essere specificata in diversi modi:
  - Singoli apici. Es: $string = ‘esempio di stringa’
  - Doppi apici. Es: $string = “altro esempio di stringa”;
  - Notazione *here document* (vedi prossime slide)
- Le notazioni a singoli o doppi apici differiscono in alcuni aspetti di sintassi. Esempio:

```
$nome = 'Anna';
print "$nome è simpatica".'</br>'; /*stampa: Anna è simpatica*/
print '$nome è simpatica'.'</br>'; /*stampa: $nome è simpatica*/
print $nome.' è simpatica'.'</br>'; /*stampa: Anna è simpatica*/
```

Notazione *here document*

- ESEMPIO 1:

```
$prova = <<<EOT
Il mio nome è Sergio
EOT;
print $prova;  /*stampa: Il mio nome è Sergio*/
```

- ESEMPIO 2:

```
$prova = <<<FINE
Il mio nome è
Sergio
FINE;
    print $prova;  /*stampa: Il mio nome è Sergio*/
```

Tipo Array

- In PHP un array è una mappa ordinata di chiavi e valori.
  - Per creare un array si usa la funzione array()

```
$giorni=array(‘primo’=>‘Lun’, ‘secondo’=>‘Mart’, ‘terzo’=>‘Merc’);
```

  - Per accedere ad un elemento si usa l’operatore []

```
echo $giorni[‘primo’];
```

- Se non si indica la chiave, viene associato automaticamente un indice per riferirsi agli elementi dell’array

```
$giorni=array(“Lu”,“Ma”,“Me”,“Gio”,“Ve”);
echo $giorni[2];
```

Aggiunta/modifica dati ad un array

- L'istruzione: `$mioArray['a'] = 'b';`
  - Aggiunge un elemento all'array con chiave 'a' e valore 'b' se non esiste nessun elemento con chiave 'a'
  - Altrimenti imposta a 'b' il valore dell'elemento con chiave 'a'.
  - NOTA: se l'array non esiste, viene creato.

- L'istruzione: $mioArray[] = 'b'
  - aggiunge un nuovo elemento con valore 'b'
  - la chiave del valore aggiunto è il valore massimo delle chiavi numeriche aumentato di 1. Esempio:
  - `$ar = array(1 => 'Lun', 'secondo' => 'Mart');`

    `$ar[] = 'Merc';` Equivale a `$ar[2] = 'Merc';`

Altre funzioni sugli array

- Stampa:
  - print_r(*$array*);
  - Attenzione: print($array) restituisce sempre la stringa "Array"
- Rimozione di elementi di un array: funzione unset(nomevar)
  - unset($ar[2]) rimuove dall'array $ar l'elemento con chiave 2
  - unset($ar) cancella l'intero array
- Come esercizio per casa vi vengono mostrate altre funzioni per gestire gli Array

Uso degli array, un esempio

```
$ar=array('nome'=>'Sergio',
   'cognome'=>’Mascetti');

$ar["nome"]="Andrea";

echo "ar['nome']
   =".$ar['nome']."<br>ar['cognome'] =
   ".$ar['cognome']."<br><br>”;
```

- Cosa viene stampato dalla funzione echo?

Operatori Aritmetici

Operator	Description	Example	Result
+	Addition	x=2 x+2	4
-	Subtraction	x=2 5-x	3
*	Multiplication	x=4 x*5	20
/	Division	15/5 5/2	3 2.5
%	Modulus (division remainder)	5%2 10%8 10%2	1 2 0
++	Increment (pre or post)	x=5 ++x o x++	x=6
--	Decrement (pre or post)	x=5 --x or x--	x=4

Operatori di Assegnamento

Operator	Example	Is The Same As
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

Operatori di Confronto

Operator	Description	Example
==	is equal to	5==8 returns false
===	is identical to (equal and of the same type)	0==‘0’ returns true 0===‘0’ returns false
!=	is not equal	5!=8 returns true
>	is greater than	5>8 returns false
<	is less than	5<8 returns true
>=	is greater than or equal to	5>=8 returns false
<=	is less than or equal to	5<=8 returns true

Operatori Logici

Operator	Description	Example
&&	and	x=6 y=3 (x < 10 && y > 1) returns true
||	or	x=6 y=3 (x==5 || y==5) returns false
!	not	x=6 y=3 !(x==y) returns true

Variabili Predefinite

- PHP mette a disposizione un certo numero di variabili predefinite
- Sono usate per memorizzare dati diversi
  - Environment
  - Input utente
  - Dati di sessione
  - …
- Tali variabili sono definite come *superglobals*: sono quindi disponibili automaticamente, senza doverle dichiarare, nello scope desiderato

Variabili Predefinite

- $GLOBALS

  Contiene i riferimenti alle variabili globali dello script
- $_SERVER

  Variabili settate dal web server o relative al contesto di esecuzione dello script
- $_REQUEST

  Variabili trasmesse allo script mediante i meccanismi GET, POST e COOKIE
- $_GET

  Variabili trasmesse allo script tramite il metodo GET
- $_POST

  Variabili trasmesse allo script tramite il metodo POST

Variabili Predefinite

- $_FILES

  Variabili trasmesse allo script mediante upload di file attraverso il metodo HTTP POST

- $_ENV

  Variabili di environment trasmesse allo script

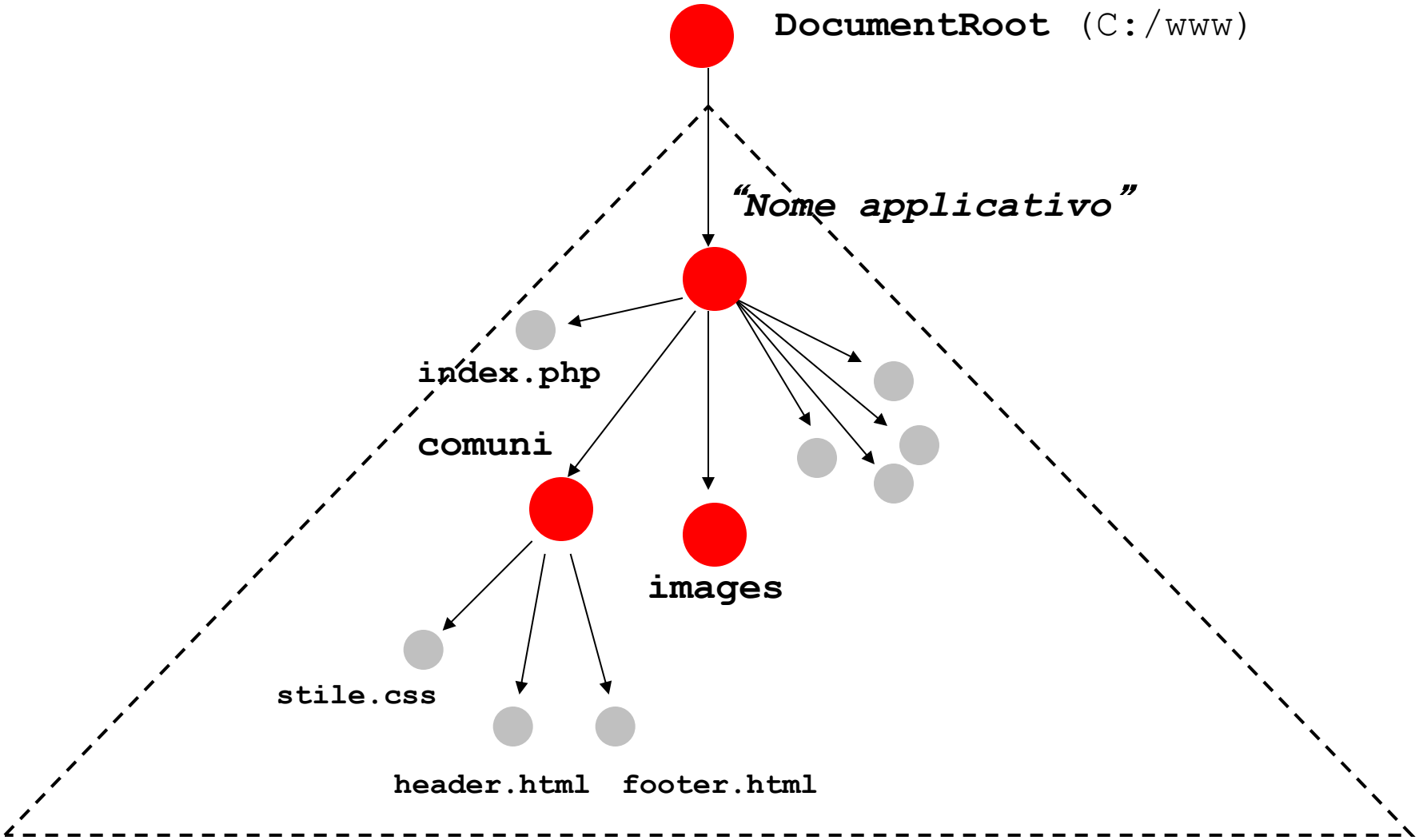
- $_COOKIE

  Variabili trasmesse allo script dal browser attraverso i cookie

- $_SESSION

  Variabili mantenute in sessione

Struttura di un applicativo PHP



Seconda parte:
ora provate voi

Esecizio 1

- Crea un file .PHP
- Assegna ad una variabile il valore 5, ad un’altra variabile il valore 7, ad una terza variabile il prodotto delle prime due.
- Fai stampare a monitor la terza variabile.

Strutture di Controllo

- **IF**

if (*condition*) {

code to be executed if condition is true; }

else {

code to be executed if condition is false;}

- **IF**

if (*condition*) {

code to be executed if condition is true; }

elseif (2nd *condition*) {

code to be executed if 2nd condition is true; }

else {

code to be executed if condition is false;}

Esercizio 2

- Modifica il file .PHP creato in precedenza per determinare quale sia il valore più grande tra quelli contenuti nelle tre variabili

Strutture di Controllo: Switch

```
switch (expr) {
case label1:
    code to be executed if expr = label1;
    break;
case label2:
    code to be executed if expr = label2;
    break;
default:
    code to be executed if expr is different from both label1 and
    label2;
}
```

Strutture di Iterazione

- **While**

while (*condition*)

code to be executed;

- **Do…While**

do {

code to be executed;

} while (*condition*);

- **FOR**

for (*initialization*; *condition*; *increment*) {

code to be executed; }

Strutture di Iterazione

- FOREACH

```
foreach ($array as $value) {
        codice;
}
```

- oppure

```
foreach ($array as $key=>$value) {
        code to be executed;

}
```

- NOTA: la seconda forma permette di conoscere il valore delle chiavi di un array

Esempio di foreach

```
$piloti_f1= array (1=>"Michael Schumacher","Mika
    Hakkinen","Rubens Barrichello","Jacques
    Villeneuve","David Coulthard","Harald Frentzen","Ralf
    Schumacher");

foreach ($piloti_f1 as $chiave => $valore) {

echo “<B> $chiave</B>$valore<BR>";

}
```

- Cosa viene stampato dalla funzione echo?

Strutture di Iterazione

- Per uscire da un qualunque ciclo del tipo
  - For, foreach, while, do...while, switch

  è possibile usare la funzione di uscita non condizionale "BREAK"
- Il costrutto "CONTINUE" viene usato all'interno di strutture di loop per saltare la parte rimanente dell'iterazione e cominciare dall'inizio quella successiva

```
for ($i = 0; $i < 10 ; $i++) {
  if ($i == 5) {
       continue;
  }
  echo $i;
}
```

Il numero 5 non viene stampato

Funzioni: Definizione

- Sintassi:

```
function nomefunzione($arg1, …., $argn){
    istruzioni;
    return($ret);
}
```

- Nella definizione dopo la parola chiave *function* viene specificato il nome della funzione seguito dai parametri (senza specifiche di tipo)
- L'istruzione *return* fa terminare l'esecuzione della funzione
- Il corpo di una funzione può essere un qualunque frammento di codice PHP valido
- Una funzione può ritornare un qualsiasi tipo di dato

Funzioni: Definizione

- Es.

```
<?php
function getPow($base, $esp){
    return pow($base,$esp);
}
print getPow(5,3)."</br>";
 ?>
```

Funzioni: Scope delle Variabili

- Scope delle variabili
  - Le variabili definite all'interno di una funzione sono chiamate *variabili locali*
  - Le variabili definite al di fuori di una funzione sono chiamate *variabili globali* e sono accessibili attraverso la variabile predefinita *$GLOBALS (che è un array)*

Funzioni: Scope delle Variabili

- Es.

```
<?php
$a=1;
$b=2;
print $a+$b."</br>";
function getSum(){
    print $a+$b."</br>";
    $a=3;
    $b=4;
    print $a+$b."</br>";
    $c=5;}
getSum();
print $a+$b."</br>";
foreach($GLOBALS as $key=>$val){
    print $key."=>".$val."</br>";}
?>
```

Funzioni: Passaggio di Parametri

- Passaggio per valore:

  function *nomefunzione*(*$arg1*, …., *$argn*){

  …

  }

- Passaggio per riferimento

  function *nomefunzione*(&*$arg1*, …., &*$argn*){

  …

  }

Passaggio parametri per valore e riferimento

- Passaggio per valore:
  - se la variabile passata viene modificata dalla funzione, l'effetto della modifica non è visibile al di fuori della funzione
- Passaggio per riferimento:
  - se la variabile passata viene modificata dalla funzione, l'effetto della modifica è visibile al di fuori della funzione

Prova tu

```
<?php
function change(&$num1, &$num2){
  $tmp=$num1;
  $num1=$num2;
  $num2=$tmp;
}
$a=10;
$b=25;
print "Prima dello scambio a vale $a e b vale $b</br>";
change($a,$b);
print "Dopo lo scambio a vale $a e b vale $b</br>";
?>
```

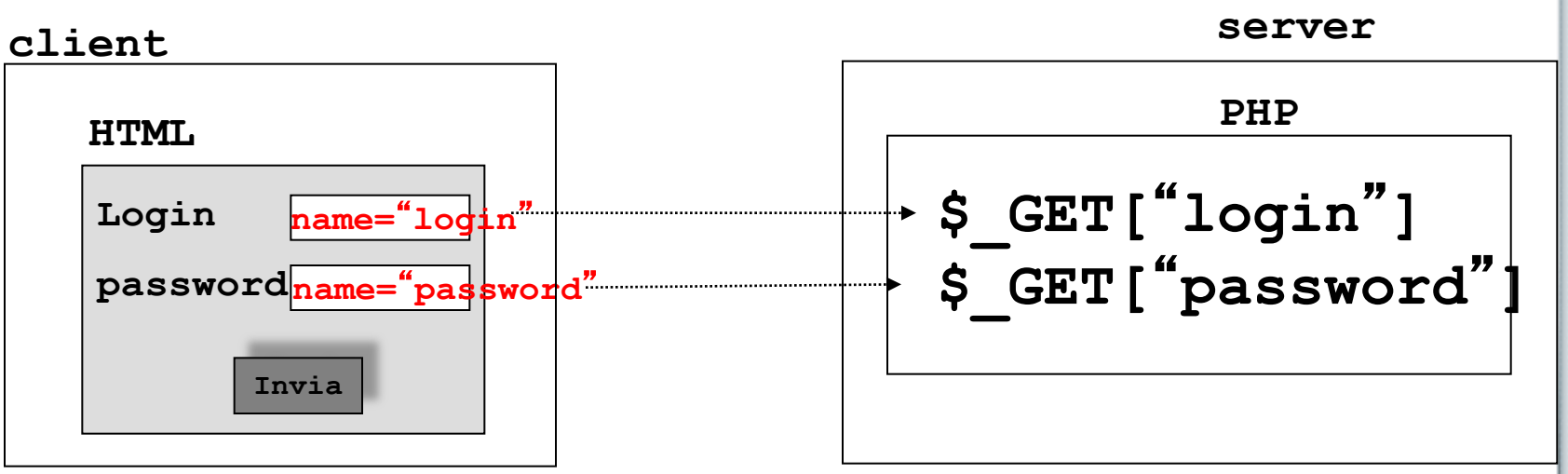
- Esegui questo codice. Poi prova a togliere gli “&”

L’interazione con l’utente

- Fino ad ora abbiamo visto come si esegue del codice lato server
  - ora vediamo come si fa a interagire con una pagina HTML
- La volta scorsa abbiamo visto come si creano le form in HTML
  - le form sono lo strumento per trasmettere dei dati da una pagina HTML ad un web server
  - i dati possono essere usati in uno script PHP

Passaggio di Parametri

- Esistono due metodi principali per passare parametri ad uno script PHP
  - GET
  - POST
- I parametri passati al PHP attraverso il metodo GET sono accessibili attraverso l'array $_GET
- I parametri passati al PHP attraverso il metodo POST sono accessibili attraverso l'array $_POST



Differenze tra GET e POST

- Con il metodo GET i parametri vengono esplicitati all'interno della URL.
  - Es: www.google.it/search?q=ciao
- Con GET si possono passare solo parametri testuali
- GET non è adatto quando si hanno molti parametri
  - o pochi parametri ma molto lunghi
- GET può essere utilizzato **anche** in un link
  - ES: <A HREF=www.google.it/search?q=ciao>link</A>
- POST può essere usato solo come risultato di una richiesta inoltrata da una form

Esempio: la form `example01.html`

```
<HTML> <BODY>

<FORM METHOD="POST"  ACTION="example01.php">

<p>

Inserisci un numero

<INPUT TYPE="TEXT" NAME="numero">

</p>

<INPUT TYPE="SUBMIT" VALUE="Invia!">

<INPUT TYPE="RESET" VALUE="Cancella!">

</FORM></BODY></HTML>
```

Esempio: il file .php example01.php

```
<HTML><BODY>
   <?php
        $num = $_POST['numero'];
        echo "Hai inserito il numero  $num <BR>";
        $a = $num*2;
        echo "$num per due fa $a";
   ?>
<BR/><A HREF="example01.html" >Indietro</A>
</BODY></HTML>
```

Prova tu!

- Fai funzionare l'esempio delle due slide precedenti sulla tua macchina.
- Modifica l'esempio in modo che funzioni con GET
- Modifica l'esempio per fare in modo che venga effettuata la somma tra due numeri forniti dall'utente.
- Modifica ancora l'esempio per fare in modo che l'operazione da svolgere tra i due numeri sia scelta dall'utente
  - ti ricordi come si crea una casella di scelta multipla?

Esempio

- Considera la pagina web che richiede login e password
- Scrivi uno script PHP che verifica che
  - Se la password è corretta (=“abc”) scrive “sei connesso”
  - Se la password è sbaglia viene richiesta una nuova immissione

Esempio: uso di PHP_SELF

```
<?php
$correct_passwd="abc";
if(!isset($_POST['password'])) {
    echo "<form method=post action=". $GLOBALS['PHP_SELF'] .">";
    echo "<input type=text name=password>";
    echo "<input type=submit>";
}
else{
    if($_POST['password'] == $correct_passwd) {
        echo "Sei connesso!";
    }
    else {
        echo 'Mi spiace, password ERRATA, <a href="'.
        $GLOBALS['PHP_SELF'] .'">Rifare la     Login</a>.';
    }
}
?>
```

Altro esercizio

- Considera la seconda form che abbiamo sviluppato nella scorsa lezione
- Memorizza i valori inseriti dall'utente ed effettua una visualizzazione in una pagina web
- Dopodiché metti insieme le due form secondo il seguente algoritmo:
  - Richiedi login e password
  - Se la password è corretta attiva la seconda form (in caso negativo richiedi di inserire nuovamente login e password)
  - Una volta inseriti i dati richiesti nella seconda form, presenta una pagina riassuntiva contenente login, password e tali dati

Attenzione!

- I dati in arrivo dal client devono essere validati!
- PHP ha varie funzioni che permettono di testare lo stato delle variabili

  Il test dipende dalle abilità del programmatore!

Esempio di controllo dell'input

```
<?php
$msg="";
if (empty($_POST["nome"]))
    $msg = $msg . "<li>manca il nome</li>\n";
if (empty($_POST["cognome"]))
    $msg = $msg . "<li>manca il cognome</li>\n";
if (empty($_POST["indirizzo"]))
    $msg = $msg . "<li>manca l'indirizzo</li>\n";

if ($msg!=""){
    $msg = "Attenzione torna indietro … " . $msg;
    echo $msg;
}
else
{ /* l'input inserito e' corretto. Si procede
     nella gestione dei dati */ }
?>
```

Esercizi per casa

Le funzioni per gestire le variabili

- La sintassi: nome_funzione();
  - empty(valore)
  - isset(valore)
  - is_null(valore)
  - is_int(valore), is_integer(valore), is_long(valore)
  - is_float(valore), is_double(valore), is_real(valore)
  - is_string(valore)
  - is_array(valore)
  - is_numeric(valore)
  - gettype(valore)
  - print_r(valore)
  - unset(valore)

Esercizio

- Data una variabile $a non ancora definita assegnarla alla variabile $b e verificare se $b è vuota
- Data $a = 5 assegnarla a $b e controllare se $b è settata
- Controllare se $b è un float
- Controllare se $b è una stringa
- Data $a = ‘5’ assegnarla a $b e controllare se $b è un intero
- Controllare se $b è una stringa
- Controllare se $b è numerico
- Definire una variabile $c a cui assegnare il “tipo” di $b
- Eliminare la variabile $a

Le funzioni per gestire le stringhe

- strlen(stringa)
- trim(stringa)
- ltrim(stringa)
- rtrim(stringa)
- substr(stringa, intero [, intero])
- str_replace(stringa, stringa, stringa)
- strpos(stringa, stringa)
- strstr(stringa, stringa)
- strtolower(stringa)
- strtoupper(stringa)
- ucfirst(stringa)
- ucwords(stringa)
- explode(stringa, stringa [, intero])

Esercizio

- Contare il caratteri della stringa ‘abcd’
- Data la stringa ' Buongiorno a tutti ‘ eliminare gli spazi all’inizio e alla fine della stringa
- Della stringa ‘Buongiorno a tutti’ creare una sottostringa a partire dal 5° carattere lunga 6 caratteri
- Della stringa ‘Buongiorno a tutti’ creare una sottostringa con gli ultimi 4 caratteri
- Rimpiazzare la parola ‘buongiorno’ con la parola ‘ciao’ della stringa ‘Buongiorno a tutti’
- Rimpiazzare l’occorrenza ‘dom’(con d minuscola) con il carattere ‘x’ della stringa ‘Domani è domenica
- Individuare la posizione della prima occorrenza del carattere ‘m’ nella stringa ‘Domani è domenica’
- Scrivere ‘Buongiorno a tutti’ in maiuscolo
- Scrivere ‘buongiorno a tutti’ con solo la prima lettera in maiuscolo
- Scrivere ‘buongiorno a tutti’ con in maiuscolo la prima lettera di tutte le parole

Le funzioni per gestire gli array

- count(array)
- array_reverse(array [, booleano])
- sort(array)
- rsort(array)
- asort(array)
- arsort(array)
- in_array(valore, array)
- array_key_exists(valore, array)
- array_search(valore, array)
- array_merge(array, array [, array...])
- array_pop(array), array_push(array, valore [,valore...])
- array_shift(array), array_unshift(array, valore [,valore...])
- implode(stringa, array)

Esercizio

- Creare un array così composto ('Luca', 'Giovanni', 'Matteo', 'Paolo', 'Antonio', 'Marco', 'Giuseppe');
- Contare gli elementi dell'array
- Creare un array invertendo gli elementi del precedente array
- Ordinare gli elementi del primo array
- Cercare se 'Giovanni' è presente nell'array
- Estrarre e stampare l'ultimo elemento dell'array precedentemente ordinato
- Estrarre e stampare il primo elemento dell'array precedentemente ordinato
- Inserire in testa a quest'ultimo array i due nominativi precedentemente estratti
- Creare una stringa con tutti gli elementi di questo array
- Creare un array così composto ('padre' => 'Claudio', 'madre' => 'Paola', 'figlio' => 'Marco', 'figlia' => 'Elisa');
- Ordinare in modo decrescente l'array (comprese le chiavi)
- Cercare se esiste un elemento di chiave 'zio'
- Cercare se esiste l'elemento 'Claudio' e indicarne la chiave

Le funzioni per gestire le date

- time():
- date(formato [,timestamp]
  - Y - anno su 4 cifre, y - anno su 2 cifre
  - n - mese numerico (1-12), m - mese numerico su 2 cifre (01-12),
  - F - mese testuale ('January' - 'December')
  - M - mese testuale su 3 lettere ('Jan' - 'Dec')
  - d - giorno del mese su due cifre (01-31), j - giorno del mese (1-31)
  - w - giorno della settimana, numerico (0=dom, 6=sab)
  - l - giorno della settimana, testuale ('Sunday' - 'Saturday')
  - D - giorno della settimana su 3 lettere ('Sun' - 'Sat')
  - H - ora su due cifre (00-23)
  - G - ora (0-23), i - minuti su due cifre (00-59), s - secondi su due cifre (00-59)
- mktime(ore, minuti, secondi, mese, giorno, anno)
- checkdate(mese, giorno, anno)

Esercizio

- Calcolare il timestamp della data 24/4/2007 ore 15.56.20
- Impostare la data con giorno in due cifre, mese in tre lettere, anno in due cifre, ora in due cifre, minuti in due cifre
- Calcolare il timestamp di 60 giorni dopo la data 24/4/2007 ore 14
- Verificare se la data 1/5/2007 è una data valida
- Verificare se la data 7/19/2007 è una data valida

Strutture di Controllo

```
<html>
<body>
<?php
$i=0;
while($i<5) {
echo "The number is " . $i .
    "<br />";
$i++;
}
?>
</body>
</html>
```

```
<html>
<body>
<?php
$i=0;
do {
$i++;
echo "The number is " . $i .
    "<br />";
} while ($i<5);
?>
</body>
</html>
```

- Le due funzioni ritornano lo stesso output?

Esercizio 1: form +PHP

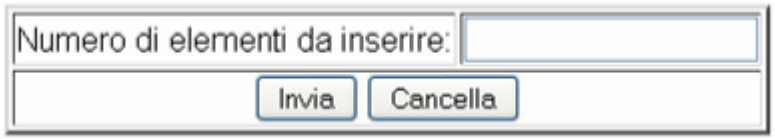
- Scrivere una form HTML che invoca script stampa.php.
  - La form deve contenere quattro campi di testo:
    - nome,
    - indirizzo email
    - due password (il secondo serve per verificare che l'utente inserisca la stessa password due volte)
  - Lo script stampa.php restituisce i dati ricevuti dal client in una tabella solo se le due password inviate coincidono; in caso negativo, lo script deve inviare un messaggio di errore che spieghi all'utente che cosa sia successo.

Esercizio 2: form +PHP

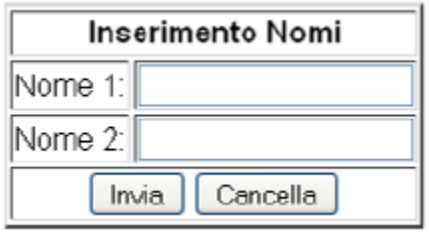
- Scrivere una form per ordinare un menu ad un ristorante on-line
- La form deve contenere una serie di pietanze che possono essere scelte dall'utente, il relativo prezzo e la quantità
- La form deve chiamare lo script `calcola.php` che si occupa di calcolare il prezzo (incluso il costo del servizio a domicilio) del pranzo ordinato
- Inoltre, lo script PHP restituisce in una tabella il menu ordinato dall'utente con il costo delle singole pietanze e il costo totale

Esercizio 3: form +PHP

- Crea la seguente form:



- Quando si preme “invia”, viene creata una nuova form con numero di campi uguale al numero inserito nel campo di testo (verificare che sia inserito un numero). E.g., se viene inserito 2, allora sarà creata la form:



- Inserire i nomi in un array e visualizzarli a video in una tabella ordinati in base alla lunghezza