Passaggio dei dati tra HTML e PHP : array superglobali $_GET e $_POST

I valori inviati da un modulo (con metodo *post* o *get*), sono automaticamente memorizzati da PHP sia in **variabili di ambiente** associate con lo stesso nome[1] (retrocompatibilità con versioni del linguaggio precedenti alla 4.1.0) sia come **elementi** di **array superglobali** individuati da chiavi con lo stesso nome delle caselle del modulo stesso.

Esempio di interazione client-server: creazione dinamica di pagina con saluto personalizzato

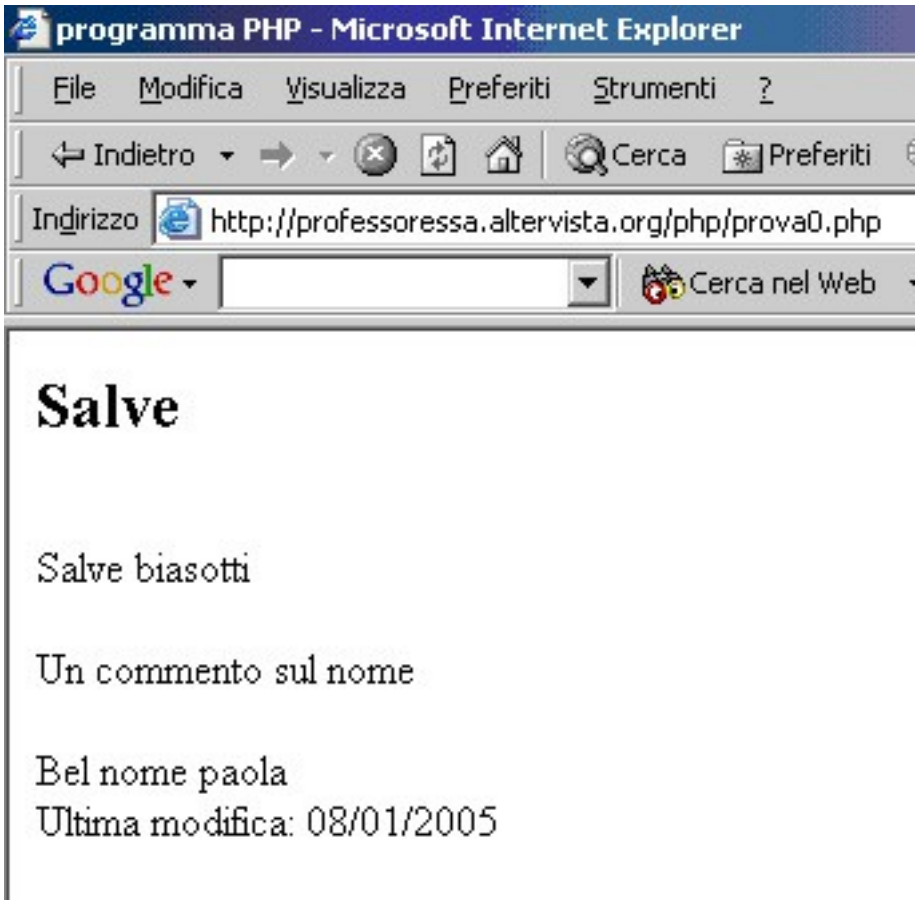
1. Costruiamo una pagina HTML **lato client** che contenga un modulo con la richiesta del nome e cognome dell'utente per un invio di tali dati al server con metodo "post":

Codice sorgente (file "prova0.htm"):

```
<html>
<head><title>Scheda alla prof. Paola Biasotti</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
</head>

<body Style ="background-color:#FFFFFF; color:#000000">

<h2>inserimento dati</h2>
<form method="post" action ="http://professoressa.altervista.org/php/prova0.php">
<p>inserisci il tuo cognome: <input type="text" name ="cognome"><p>
<p>inserisci il tuo nome: <input type="text" name ="nome"><p>
<p><input type = "submit" value = "invio"><p>
<p><input type = "reset"><p>

</body>
</html>
```

[1] Usando **EasyPHP** è necessario usare array ***superglobali*** $_GET e $_POST. Per motivi di sicurezza, infatti, il server imposta *register_globals* su *OFF*, e non si possono usare direttamente le variabili (con il loro nome) se non dopo averle ricavate dall'array superglobale. Per visualizzare le ***impostazioni*** di ***configurazione*** si può creare un file (*.php*) con il seguente codice <html><body> **<? phpinfo(); ?>**</body></html>

2. Costruiamo una pagina PHP **lato server** che crei *al volo* una pagina HTML visualizzata sulla finestra del client con l'effetto mostrato in figura:



Codice sorgente
(file "prova0.php")

```
<html>
<head><title>programma PHP</title></head>
<body>
<h2>Salve</h2><br>
<?php
        /* uso dell'istruzione echo e di variabili associate ai valori che il metodo POST
           memorizza in variabili di ambiente : retrocompatibilità con old version   */

    echo ("Salve $cognome");
?>
<p>Un commento sul nome</p>
<?php
    echo ("Bel nome"." ".$nome. "<br>") ;

    # per concatenare si usa il carattere punto ad esempio echo("Ciao"." "."Alberto")
    // altro commento

   echo("Ultima modifica: ".date("d/m/Y",filemtime($PATH_TRANSLATED)));

?>
</body>
</html>
```

Dalla versione 4.1.0, i valori contenuti in una query string ed inviati da un modulo col metodo *get*, sono automaticamente memorizzati da PHP nell'**array *superglobale*** (in quanto è disponibile anche all'interno delle funzioni) **$_GET**.

Analogamente, i valori inviati da un modulo col metodo *post*, sono automaticamente memorizzati da PHP nell'**array *superglobale*** (in quanto è disponibile anche all'interno delle funzioni) **$_POST**.

Lato server, si potranno recuperare tali dati, dunque, come elementi di un array[2].

Codice sorgente (file "prova_new.htm"):

```
<html>
<head><title>Scheda alla prof. Paola Biasotti</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
</head>

<body Style ="background-color:#FFFFFF; color:#000000">

<h2>inserimento dati</h2>
<form method="post" action ="http://professoressa.altervista.org/php/prova_new.php">
<p>inserisci il tuo cognome: <input type="text" name ="cognome"><p>
<p>inserisci il tuo nome: <input type="text" name ="nome"><p>
<p><input type = "submit" value = "invio"><p>
<p><input type = "reset"><p>
</body>
</html>
```

Codice sorgente (file "prova_new.php")

```
<html>
<head><title>programma PHP</title></head>
<body>
<h2>Salve</h2><br>
<?php
        /* uso dell'istruzione echo e di array superglobale $_POST  */

    echo ("Salve $_POST[cognome]");
?>
<p>Un commento sul nome</p>
<?php
    echo ("Bel nome"." ". $_POST['nome'] . "<br>") ;

    # per concatenare si usa il carattere punto ad esempio echo("Ciao"." "."Alberto")
    // altro commento

   echo("Ultima modifica: ".date("d/m/Y",filemtime($PATH_TRANSLATED)));
?>
</body>
</html>
```

[2] In PHP un **array** è un modo di raggruppare un insieme di valori ***diversi*** cioè non omogenei e di indicizzarli con un numero (***chiave numerica***) o con una stringa (***chiave associativa***) o in modo misto con chiavi dei due tipi.

Cos'è PHP?

A metà degli anni novanta il Web era ancora formato in gran parte da pagine statiche, cioè da documenti HTML il cui contenuto non poteva cambiare fino a quando qualcuno non interveniva manualmente a modificarlo. Con l'evoluzione di Internet, però, si cominciò a sentire l'esigenza di rendere dinamici i contenuti, cioè di far sì che la stessa pagina fosse in grado di proporre contenuti diversi, personalizzati in base alle preferenze degli utenti, oppure estratti da una base di dati (database) in continua evoluzione.

PHP nasce nel 1994, ad opera di Rasmus Lerdorf, come una serie di macro[3] la cui funzione era quella di facilitare ai programmatori l'amministrazione delle home page personali: da qui trae origine il suo nome, che allora significava appunto **Personal Home Page**. In seguito, queste macro furono riscritte ed ampliate fino a comprendere un pacchetto chiamato Form Interpreter[4] (PHP/FI).

Essendo un progetto di tipo *open source* (cioè "codice aperto", quindi disponibile e modificabile da tutti), ben presto si formò una ricca comunità di sviluppatori che portò alla creazione di PHP3: la versione del linguaggio che diede il via alla crescita esponenziale della sua popolarità. Tale popolarità era dovuta anche alla forte integrazione di PHP con il Web server Apache (il più diffuso in rete), e con il database MySql. Tale combinazione di prodotti, integralmente ispirata alla filosofia del free software, diventò ben presto vincente in un mondo in continua evoluzione come quello di Internet.

Alla fine del 1998 erano circa 250.000 i server Web che supportavano PHP[5]: un anno dopo superavano il milione. I 2 milioni furono toccati in aprile del 2000, e alla fine dello stesso anno erano addirittura 4.800.000. Il 2000 è stato sicuramente l'anno di maggiore crescita del PHP[6], coincisa anche con il rilascio della versione 4.0.0, con un nuovo motore (Zend) molto più veloce del precedente (realizzando la fase di traduzione in codice eseguibile prima dell'esecuzione) ed una lunga serie di nuove funzioni, fra cui quelle importantissime per la gestione delle sessioni.

La crescita di PHP, nonostante sia rimasta bloccata fra luglio e ottobre del 2001, è poi proseguita toccando quota 7.300.000 server alla fine del 2001, per superare i 10 milioni alla fine del 2002, quando è stata rilasciata la versione 4.3.0.

Oggi PHP è conosciuto come **PHP: Hypertext Preprocessor**, ed è un linguaggio completo di scripting, sofisticato e flessibile, che può girare praticamente su qualsiasi server Web, su qualsiasi sistema operativo (Windows o Unix/Linux, ma anche Mac, AS/400, Novell, OS/2 e altri), e consente di interagire praticamente con qualsiasi tipo di database (MySql, PostgreSql, Sql Server, Oracle, SyBase, Access e altri). Si può utilizzare per i più svariati tipi di progetti, dalla semplice home page dinamica fino al grande portale o al sito di e-commerce.

---

[3] Nel giugno 1995 Rasmus Lerdof annuncia su comp.infosystem.www.authoring.cgi il rilascio di un piccolo insieme di file binari scritti in C con il fine di registrare le visite alla pagina web contenente il suo curriculum...

[4] **FI** (Form Interpreter) un parser di pagine HTML con la possibilità di interagire con mSQL, rilasciato al settembre '95 che a fine '95 è già utilizzato da diversi beta-tester e sviluppatori. Nel novembre '97 esce la **PHP/FI 2.0** che supporta mSQL, Postgres95 e MySQL, pare che fosse già usato da 50mila domini.

[5] Il salto di qualità avviene nel giugno 1998 con **PHP 3** grazie a due israeliani Zeev Suraski e Andi Gutmans che hanno creato un parser chiamato Zend Engine (la versione 2 è inclusa in **PHP 5**, rilasciato nel 2004, che introduce un nuovo modello di programmazione ad oggetti, interazione con file XML e il supporto di Web Services). I punti di successo di **PHP 3** furono il nuovo parser, il supporto di altri database, la compatibilità con Windows e soprattutto il numero crescente di sviluppatori che garantivano una continuità e una larga base di testing. PHP3 pare fosse installato sul 10% dei domini dell'epoca.

[6] Nel maggio 2000 esce **PHP 4**, non più con licenza GPL ma PHP license (più restrittiva ma sempre open source) che supporta nativamente le sessioni (prima in libreria esterna) e offre una modularità avanzata.

EasyPHP

Lanciare EasyPHP

Non si può propriamente parlare di "lanciare EasyPHP", il fatto è che vengono avviati **Apache** e **MySQL server**.

Dopo l'installazione, sarà stato creato un collegamento nella cartelle "Start/Programmi/EasyPHP".

La prima volta che si avvia EasyPHP verrà aggiunta un'icona nella barra di sistema, vicino all'orologio.

Clicca su di essa per accedere ai menu :

- Log file: riporta ogni errore generato da Apache e MySQL
- Configuration : una semplice interfaccia per configurare EasyPHP
- Web local : apre l'URL "http://localhost/"
- Start/Stop : avvia/ferma i server Apache e MySQL
- Quit : uscita

Uso della cartella www

Per fare in modo che uno script possa essere eseguito, si posiziona il file della cartella "www".

Il server Apache è configurato automaticamente per aprire un file indice (**index**) quando viene digitato l'indirizzo "**http://localhost/**" che corrisponde al percorso assoluto della cartella"www".

Questa è per definizione la pagina di partenza e verifica che EasyPHP è in esecuzione.

E' consigliablie creare una cartella per ciascun progetto all'interno della cartella "www"; in questo modo diviene più semplice gestire diversi progetti.

Possibili estensioni : **php, php3, php4**

Il codice PHP può essere inserito all'interno del codice HTML :

- cominciando a racchiuderlo tra **<?** oppure <?php
- e concludendo con **?>**

Nota Bene:
AZIONI DA EVITARE : cercare di lanciare lo script facendo doppio click sul file all'interno della cartella del progetto: questo genera una pagina di errore.

AZIONI CORRETTE : lanciare EasyPHP, connettersi tramite il browser a "**http://localhost**", aprire la cartella del progetto, quindi cliccare su "*nome_file*.php".

A questo punto la pagina è correttamente interpretata dal server Apache

Lanciando da browser:
http://localhost/Esempi_PHP/prova_post.htm
il file html contiene un modulo dove il metodo scelto per inviare i dati è "**post**"
e come attributo action ="**http://localhost**/Esempi_PHP/prova_post.php"
oppure action =" prova_post.php" essendo nella stessa cartella

Codice prova_post.htm

```
<html><head><title>Scheda in locale</title></head>
<body Style ="background-color:#FFFFFF; color:#000000">
<h2>inserimento dati</h2>
<form  method="post" action ="http://localhost/Esempi_PHP/prova_post.php">
<p>inserisci il tuo cognome: <input type="text" name ="cognome"><p>
<p>inserisci il tuo nome: <input type="text" name ="nome"><p>
<p><input type = "submit" value = "invio"><p>
<p><input type = "reset"><p>
</body></html>
```

Codice prova_post.php

```
<html>
<head><title>programma PHP</title></head>
<body>
<h2>Salve</h2>
<br>
<?php
        /* uso dell'istruzione echo e di elementi dell'array superglobale individuati  da chiavi con lo stesso nome delle
           variabili che il metodo POST invia separatamente */

    echo ("Salve"." ". $_POST['cognome'] );
?>
<br>Un commento sul nome:

<?php
    echo (" bel nome"." ". $_POST['nome'] . "<br>") ;

    # per concatenare si usa il carattere punto ad esempio echo("Ciao"." "."Alberto")
   // altro commento
?>
Data Corrente : <? print (Date("l F d, Y")); ?>
</body>
</html>
```

Codice prova_get.htm

```
<html><head><title>Scheda in locale</title></head>
<body Style ="background-color:#FFFFFF; color:#000000">
<h2>inserimento dati</h2>
<form  method="get" action ="prova_get.php">
<p>inserisci il tuo cognome: <input type="text" name ="cognome"><p>
<p>inserisci il tuo nome: <input type="text" name ="nome"><p>
<p><input type = "submit" value = "invio"><p>
<p><input type = "reset"><p>
</body>
</html>
```

Codice prova_get.php

```
<html>
<head><title>programma PHP</title></head>
<body>
<h2>Salve</h2>
<br>
<?php
        /* uso dell'istruzione echo e di elementi dell'array superglobale individuati  da chiavi con lo stesso nome delle
           variabili che il metodo GET accoda all'URL */

    echo ("Salve"." ". $_GET['cognome'] );
?>
<br>Un commento sul nome:

<?php
    echo (" bel nome"." ". $_GET['nome'] . "<br>") ;

    # per concatenare si usa il carattere punto ad esempio echo("Ciao"." "."Alberto")
   // altro commento
?>
Data Corrente : <? print (Date("l F d, Y")); ?>
<br>
<? echo("Ultima modifica: ".date("d/m/Y") ); ?>    <!--  indefinita  variabile $PATH_TRANSLATED -->
</body>
</html>
```