

La gestione dei form in PHP avviene in due fasi:

1. Preparazione del form HTML
2. Preparazione dello script PHP

Un form HTML si crea mediante il tag `form` con molti possibili tipi di input. Il tag `form` è di questo tipo:

```
<form action="script.php" method="post">  
</form>
```

L'attributo più importante è `action`, che indica il nome dello script che gestirà i dati inseriti col form. L'attributo `method` è quasi sempre settato a POST (v. seguito).

Fra i tag di apertura e chiusura del form si possono inserire *text box*, *radio button*, *menu a tendina*, *check box*, ecc. L'attributo `name` nei tag di input indica il nome con cui lo script PHP farà riferimento al dato inserito dall'elemento di input.

Scelta del metodo (<http://www.w3.org/2001/tag/doc/whenToUseGet.html>)

Il metodo GET invia i dati allo script in forma di una serie di coppie nome-valore aggiunte alla URL.

Esempio:

`http://www.dmcinsights.com/script.php?name=Homer&gender=M`

Vantaggi di GET:

- La pagina può essere inserita come bookmark nel browser (è una URL).
- E' possibile utilizzare il pulsante Back del browser per tornare alla pagina che genera l'input come GET, o ricaricarla senza problemi (questo non accade con POST).

Svantaggi di GET:

- I dati sono visibili e limitano quindi la riservatezza dei dati stessi; i dati trasmessi con POST sono invisibili.
- Il numero di dati che può essere trasmesso è limitato (nessun problema con POST).

Gestione dei dati tramite script PHP

Una volta preparato il form HTML, è necessario predisporre lo script PHP per gestirlo.

Lo script PHP:

- Riceve i dati dal form
- Li elabora
- Restituisce al browser un flusso di codice HTML attraverso opportune istruzioni di output.

Il metodo tradizionale di gestione di form (CGI) richiedeva che lo script facesse il parsing dei dati: PHP inserisce le informazioni in variabili speciali.

PHP: Funzioni di output

L'istruzione `echo` è il modo più semplice per generare output.

E' generalmente seguita dalla stringa che deve essere stampata:

- se la stringa è fra apici viene riprodotta in output in modo esatto

```
echo 'ho speso $100'; produce in output
```

```
ho speso $100
```

- se è fra doppi apici le eventuali variabili (denotate da stringhe aventi `$` come primo carattere) contenute al suo interno sono sostituite dal valore corrispondente

```
$a = '100 Euro';
```

```
echo "ho speso $a"; produce in output
```

```
ho speso 100 Euro
```

Un punto è l'operatore di concatenazione fra stringhe

```
$a='Carlo'; $b='Luca';
```

```
echo "Ho visto ". "$a" . " e ". "$b"; produce
```

```
Ho visto Carlo e Luca
```

Dato un form contenente

```
<input type="text" name="weight" size="20" />
```

Lo script PHP inserisce l'input dell'utente in una variabile speciale chiamata `$_REQUEST['weight']` il cui argomento è uguale al corrispondente attributo `name` del form.

NB

Le variabili in PHP sono case-sensitive.



Ora `$_REQUEST['weight']` è utilizzabile come una variabile nello script.

In effetti, `$_REQUEST` per PHP è un array e

`$_REQUEST['weight']` è l'elemento di `$_REQUEST` identificato dalla chiave `'weight'`.

In PHP, infatti (v. seguito), ogni elemento di un array può essere visto come una coppia indice/valore. L'indice non è necessariamente numerico, ma può anche essere una stringa.

PHP: Registrazione di variabili globali

Una variabile è definita attraverso una stringa avente `$` come iniziale. Nelle versioni PHP precedenti alla 4.2 i dati passati dai form erano direttamente utilizzabili come variabili globali (`register_globals=On` nel file di configurazione `php.ini`). Dalla versione 4.2 il default è `Off` per aumentare la sicurezza degli script.

Per ovviare al 'problema' ci sono vari modi:

- Mettere `on` `register_globals` in `php.ini`. (richiede diritti di amministratore)
- Attivare localmente le variabili globali attraverso il comando `import_request_variables('gpc')` in apertura di script (`gpc` fa riferimento alle variabili speciali `$_GET`, `$_POST`, `$_COOKIE`)
- Usare esplicitamente le variabili speciali `$_REQUEST`, `$_GET`, `$_POST` come visto in precedenza

NB `$_REQUEST` generalizza `$_GET`, `$_POST` e `$_COOKIE` e può essere usata per gestire i dati trasmessi con tutti e tre i metodi

PHP: Controllo dell'input

Le principali strutture di controllo di PHP (if, switch, while, for) hanno la stessa sintassi che hanno in C.

Analogamente i blocchi di codice sono racchiusi in parentesi graffe e ogni istruzione è chiusa da punto e virgola.

I commenti sono introdotti da // (vale fino a fine riga)

Funzione `isset()`

Verifica se una variabile è stata istanziata o se ha un valore NULL (con funzione analoga al valore usato per le basi di dati)

```
if (isset($var)) {  
    // $var ha un valore.  
} else {  
    // $var non ha un valore.  
}  
// comprese stringhe vuote e FALSE
```

NB Per le variabili logiche la convenzione è la stessa del C

(FALSE $\leftrightarrow$ 0 TRUE $\leftrightarrow$ $x \neq 0$): `if($a)` è vera se \$a è diversa da 0

Funzione `strlen()`

Un altro modo per vedere se una variabile è istanziata è usare la funzione `strlen()` che restituisce la lunghezza, in caratteri, della variabile. Se è pari a zero, la variabile non è istanziata.

```
if (strlen($var) > 0) {  
    // $var ha un valore.  
} else {  
    // $var non ha un valore.  
}
```

Funzione `is_numeric()`

Verifica se la variabile usata come argomento è di tipo numerico.

E' possibile attivare l'uso di metodi per la gestione corretta di caratteri speciali derivanti dall'input che in HTML devono essere preceduti da \

Questi metodi si chiamano Magic Quotes e sono di tre tipi:

- `magic_quotes_gpc`, usato per filtrare dati da form, URL, and cookie (gpc sono le iniziali di GET, POST, COOKIE);
- `magic_quotes_runtime`, che filtra dati estratti da file esterni e database. Se Magic Quotes è attivo sul server, è possibile neutralizzarne l'effetto usando la funzione `stripslashes()` che elimina le \ dalla stringa inserita come argomento.

```
$var = stripslashes($var);
```

- `magic_quotes_sybase` — usata raramente.

- Un `\\` inserito da `magic_quotes_sybase` diventa un singolo backslash usando la funzione `stripslashes()`.
- E' comodo usare Magic Quotes quando si consulta una base di dati per evitare che caratteri speciali interrompano la trasmissione dei risultati di una query
- E' possibile emulare Magic Quotes se è disabilitato usando la funzione duale a `stripslashes()`: `addslashes()`

NB Da Php 5.4 MagicQuotes non è più supportato.

- Quando si usano stringhe generate da form, è consigliabile usare la funzione `trim()` che elimina gli spazi eventualmente presenti all'inizio e alla fine della stringa.
`$name = trim($name);`

PHP: Array

Gli array sono strutturati come coppie *chiave-valore* in cui ogni coppia è un elemento dell'array. Il *valore* (contenuto informativo) di ogni elemento può essere costituito da un numero, una stringa o anche un altro array.

Ogni elemento è associato a una *chiave* (indice) per cui la struttura di un array è simile a quella di una tabella di dati.

Array	
Key 1	Value 1
Key 2	Value 2
Key 3	Value 3



Array

Esistono due tipi di array:

- *Array indicizzati*

Usano come chiave un numero (sono di fatto gli array utilizzati in tutti i linguaggi di programmazione)

- *Array associativi*

Usano come chiave una stringa il cui contenuto consente di identificare univocamente il corrispondente valore dell'array.

`$sigle`

Chiave	Valore
MI	Milano
TO	Torino

`$sigle['MI']`

`$sigle['TO']`

NB Le chiavi di un array devono essere tutte diverse, i valori possono ripetersi (es. un array inizializzato tutto a zero).

Array

Ogni elemento è individuato dal nome dell'array e dalla chiave fra parentesi quadre. Se la chiave è una stringa andrà indicata fra singoli apici. Se è numerica gli apici non sono necessari.

```
$sigle[ 'MI' ]  
$temperatura[2]
```

Se si vuole stampare l'elemento di un array associativo è necessario racchiuderlo fra graffe per evitare errori di parsing. Non è necessario fare lo stesso per un array indicizzato.

```
echo "Il panettone è fatto a {$sigle[ 'MI' ]}"  
echo "L'elemento con indice 2 dell'array  
temperatura è $temperatura[2]"
```

Un array associativo può avere sia numeri che stringhe come chiavi, purché le chiavi siano uniche.

Creazione di array

Un array può anche essere definito all'interno di uno script:

- Assegnando il valore di un elemento alla volta

```
$single['MI'] = 'Milano';
```

- Costruendo un intero array in un singolo passo con la funzione `array()`

```
$stati = array ('IA' => 'Iowa',  
               'MD' => 'Maryland');
```

- Senza specificare la chiave

```
$artisti = array ('Vasco Rossi', 'Beatles', 'Mina');
```

- Inserendo solo l'indice numerico del primo elemento

```
$days = array(1=> 'Sunday', 'Monday', 'Tuesday');
```

- Creando un array di numeri in sequenza

```
$ten = range (1, 10);
```

Accesso ad array

Un array non può essere stampato con il comando
`echo $vettore`

Il ciclo `foreach` può scandire gli elementi di un array

```
// Sintassi del ciclo foreach
foreach ($array as $value) { 
    Fai qualcosa con $value.
}
```

// Esempio di ciclo `foreach` che accede a chiavi e valori.

```
foreach ($array as $key => $value) {
    echo "Il valore dell'elemento $key è $value.";
}
```

La funzione `count($array)` è equivalente a `sizeof($array)` e restituisce il numero di elementi contenuti in `$array`

PHP: Array multidimensionali

```
$states=array('TX'=>'Texas','UT'=>'Utah',...);  
$provinces=array('QC'=>'Quebec','AB'=>'Alberta',...);
```

possono essere fusi in un singolo array bidimensionale

```
$abbr=array('US'=>$states,'Canada'=>$provinces);
```

In questo caso ogni elemento di `$abbr` potrà essere individuato attraverso due chiavi, ognuna delle quali permetterà di accedere ad uno dei due array.

```
echo "Lo stato con sigla TX è: $abbr['US']['TX']"
```

NB Un array bidimensionale in PHP non è una matrice; ogni 'riga' può avere un diverso numero di elementi. Concettualmente è più simile ad una lista di liste.