Form PHP

Slide tratte da:

W3CSCHOOLS.COM

(https://www.w3schools.com/php/php_forms.asp)

Gestione dei Form in PHP

- Impariamo a creare e gestire form in PHP per l'elaborazione dei dati.

HTML

```
<html>
<body>
<form action="welcome.php" method="post">
  Nome: <input type="text" name="name"><br>
  E-mail: <input type="text" name="email"><br>
  <input type="submit">
</form>
</body>
</html>
```


Variabili Superglobali in PHP

- In PHP, le variabili superglobali `$_GET` e `$_POST` vengono utilizzate per raccogliere i dati dei form.
- Queste variabili sono accessibili da qualsiasi parte del codice PHP.
- Ecco una lista:

- `$GLOBALS`
- `$ SERVER`
- `$_REQUEST`
- `$_POST`
- `$_GET`
- `$_FILES`
- `$_ENV`
- `$_COOKIE`
- `$_SESSION`

Variabili Superglobali in PHP (caso POST)

PHP

```
<?php
// Utilizzo delle variabili superglobali $_GET e
$_POST

if ($_SERVER["REQUEST_METHOD"] == "POST") {
  $name = $_POST["name"];
  $email = $_POST["email"];
  // Elaborazione dei dati del form...
}

?>
```


Variabili Superglobali in PHP (caso GET)

- Esaminiamo l'attributo method inizializzato a:"get"

HTML

```
<html>
<body>
<form action="welcome.php" method="get">
  Nome: <input type="text" name="name"><br>
  E-mail: <input type="text" name="email"><br>
  <input type="submit">
</form>
</body>
</html>
```


Variabili Superglobali in PHP (caso GET)

PHP

```
<?php
// Utilizzo delle variabili superglobali $_GET e
$_POST

if ($_SERVER["REQUEST_METHOD"] == "GET") {
  $name = $_GET["name"];
  $email = $_GET["email"];
  // Elaborazione dei dati del form...
}

?>
```


Differenze tra GET e POST (1)

	GET	POST
BACK button/Reload	Harmless	Data will be re-submitted (the browser should alert the user that the data are about to be re-submitted)
Bookmarked	Can be bookmarked	Cannot be bookmarked
Cached	Can be cached	Not cached
Encoding type	application/x-www-form-urlencoded	application/x-www-form-urlencoded or multipart/form-data. Use multipart encoding for binary data
History	Parameters remain in browser history	Parameters are not saved in browser history

Differenze tra GET e POST (2)

- Abbiamo visto che entrambi i metodi GET e POST creano un array di dati.
- `$_GET` contiene i dati passati tramite i parametri URL.
- `$_POST` contiene i dati inviati tramite il metodo POST.
- Cerchiamo di capire quando utilizzare uno o l'altro.

Quando usare GET

- I dati inviati con GET sono visibili nell'URL.
- GET è appropriato per dati non sensibili.
- Non utilizzare GET per password o informazioni sensibili.

Quando usare POST

- POST invia dati invisibili nell'URL.
- Non ha limiti sulla quantità di dati.
- Utilizzato per dati sensibili.
- Utilizzato per il caricamento di file.

$_SERVER[“PHP_SELF”] (Tecnica del POSTBACK)

HTML - PHP

```
<form method="post"
action="<?php echo $_SERVER["PHP_SELF"]);?>">
...
...
</form>
```

- `$_SERVER["PHP_SELF"]` è una variabile super globale che restituisce il nome file dello script attualmente in esecuzione.
- Il form invia i dati alla pagina stessa, invece di passare a una pagina diversa. (p.e. l'utente riceverà messaggi di errore sulla stessa pagina del form)

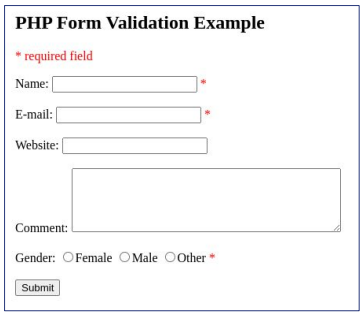
Sicurezza nei form PHP

- Attraverso i form inviamo dati dal client al server attraverso la rete.
- Quindi è importante gestire i form in modo sicuro.
- Sino ad ora abbiamo visto solo l'invio e il recupero dei dati, senza validazione.
- È essenziale validare i dati dei form per proteggere lo script da code injection.

Validazione dei campi di un form

Il seguente è il form di esempio che utilizzeremo:

Field	Validation Rules
Name	Required. + Must only contain letters and whitespace
E-mail	Required. + Must contain a valid email address (with @ and .)
Website	Optional. If present, it must contain a valid URL
Comment	Optional. Multi-line input field (textarea)
Gender	Required. Must select one



Validazione dei campi di un form

- Passiamo tutte le variabili attraverso la funzione `htmlspecialchars()` di PHP.
- Eliminiamo i caratteri non necessari (spazio extra, tabulazione, nuova riga) dai dati di input dell'utente mediante la funzione PHP `trim()`
- Rimuoviamo le barre rovesciate (\) dai dati di input dell'utente utilizzando la funzione `tripslashes()`

Validazione dei campi di un form

- Definiamo una funzione che esegua la “pulizia” del codice.

PHP

```
<?php
function test_input($data) {
  $data = trim($data);
  $data = stripslashes($data);
  $data =
htmlspecialchars($data);
  return $data;
}
// define variables and set to
empty values
$name = $email = $gender =
$comment = $website = "";
```

```
if ($_SERVER["REQUEST_METHOD"] == "POST")
{
  $name = test_input($_POST["name"]);
  $email = test_input($_POST["email"]);
  $website =
test_input($_POST["website"]);
  $comment =
test_input($_POST["comment"]);
  $gender = test_input($_POST["gender"]);
}
?>
```

Validazione del campo “name”

- Utilizziamo una *espressione regolare* con `preg_match()`.

PHP

```
$name = $_POST["name"];
if (!preg_match("/^[a-zA-Z-' ]*$/", $name)) {
  $nameErr = "Sono consentite solo lettere e spazi";
}
```


Validazione del campo “email”

- Per verificare il formato utilizziamo `filter_var()`.
- I tipi di filtri li trovate qui (php.net)

PHP

```
$email = $_POST["email"];
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
  $emailErr = "Formato e-mail non valido";
}
```


Validazione del campo “website”

- Utilizziamo una *espressione regolare* con `preg_match()`. per verificare la sintassi dell’URL

PHP

```
$website = $_POST["website"];
if (!preg_match(

"/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-9+&@#\/%?=~_|!:,.;]*[
-a-z0-9+&@#\/%=~_|]/i", $website)) {

  $websiteErr = "URL non valido";
}
```


Memorizzazione valore dei campi

- Vediamo come mantenere i valori dei campi del form dopo l'invio utilizzando script PHP.

PHP

```
Name: <input type="text" name="name" value="<?php echo
$name;?>">

E-mail: <input type="text" name="email" value="<?php echo
$email;?>">
```

Memorizzazione valore dei campi

- Vediamo come mantenere i valori dei campi del form dopo l'invio utilizzando script PHP.

PHP

```
Name: <input type="text" name="name" value="<?php echo
$name;?>">

E-mail: <input type="text" name="email" value="<?php echo
$email;?>">

<input type="radio" name="gender"
<?php if (isset($gender) && $gender == "female") echo
"checked";?> value="female">Female
```


Abbiamo finito!

Ora è importante fare parecchi esercizi!