Videocorso PHP

PHP, HTML, GET e POST

Integrazione di PHP nelle pagine HTML

Alessandro Flora

L’integrazione di PHP nell’HTML

Come sarà impostata questa lezione

- All’interno delle slide ci saranno alcuni riferimenti utili per il successivo ripasso
- In particolare nelle slide sono riportati i concetti di base e la sintassi
- La seconda parte sarà un esercizio svolto; è vivamente consigliato provare a ragionare insieme creando uno script per provare in prima persona

L’integrazione di PHP nell’HTML

Cosa dobbiamo fare?

- Non esiste un solo approccio per gestire l’integrazione tra il codice PHP e le pagine HTML
- In questo corso vedremo l’approccio di integrazione ovvero inserire del codice PHP direttamente nella pagina web con l’idea di modificarla grazie allo stesso
- Un altro approccio, più interessante del precedente, vede quello di utilizzare nelle pagine HTML del codice JavaScript per effettuare delle richieste Ajax a pagine PHP senza grafica e riutilizzarne il risultato per ricomporre la pagina. Tale approccio è più complesso ma affatica di meno il server e risulta più *fluido* per l’utente.

L’integrazione di PHP nell’HTML

Integrare il codice PHP

- Per farlo basta inserire i delimitatori che abbiamo usato per racchiudere il codice PHP
- Tutto ciò che è fuori dai delimitatori è HTML

index.php

```php
<!DOCTYPE html>
<html lang="it-IT">
    <head>
        <title>Prova</title>
        <meta charset="UTF-8">
    </head>

    <body>
        <ul>

        <?php

            $vettore = ['uno','due','tre'];

            foreach($vettore as $elemento) {
                echo "<li>$elemento</li>";
            }

        ?>

        </ul>
    </body>
</html>
```

In questo esempio viene generato un elenco puntato iterando su un vettore

L’integrazione di PHP nell’HTML

Il delimitatore <?=?>

- Il delimitatore **<?php?>** che abbiamo usato finora può essere usato nella pagina HTML sia per svolgere operazioni interne allo script sia per stampare mediante la sintassi

```
<?php echo "Hello, World!"; ?>
```

- Il delimitatore **<?=?>** è invece una *scorciatoia* per poter stampare ciò che gli viene posto all’interno; pertanto la linea di codice precedente può essere scritta come

```
<?= "Hello, World!" ?>
```

L’integrazione di PHP nell’HTML

Stampare le strutture HTML

- Per stamparle ci avvaliamo dell’echo o del delimitatore **<?=?>**
- Se stiamo usando una stringa con gli apici dobbiamo concatenare eventuali variabili
- Se stiamo usando una stringa con le virgolette possiamo includere il nome della variabile direttamente nella stringa o tra parentesi graffe. In questo caso però eventuali parametri dei tag dovranno usare il codice di escape **\"** anziché **"**

```
echo "<p class=\"testoNormale\">$variabile1 {$variabile2}</p>";

echo '<p class="altroTesto">'.$variabile1.' '.$variabile2.'</p>';
```

L’integrazione di PHP nell’HTML

Cosa è lecito stampare nella pagina?

- Sicuramente i testi o le quantità in numero variabile di una struttura (righe di una tabella, punti di un elenco puntato, paragrafi di un testo, ...)
- Raramente degli script JavaScript (è meglio includerli nella pagina e al massimo stampare le chiamate tramite PHP ricordandosi che il codice PHP viene eseguito prima)
- Mai il CSS: non ha senso non includerlo della pagina, sovraccarichiamo inutilmente il server

In generale l’approccio è usare PHP il meno possibile per non sovraccaricare il server

Le richieste GET e POST

Le richieste HTTP

- Quando ci connettiamo a una pagina usando il protocollo HTTP stiamo compiendo una richiesta che segue un certo metodo trasportando dei dati
- Le richieste che vediamo in questo corso sono le GET e le POST
- Quando ci connettiamo a una pagina usando un browser e inserendo il percorso stiamo compiendo una richiesta GET
- Per compiere una richiesta POST dobbiamo ricorrere a un form HTML oppure a una richiesta Ajax tramite JavaScript
- Entrambe le richieste possono trasportare dei dati sotto forma di variabili

Le richieste GET e POST

Le richieste GET

- Sono le più semplici ma anche le meno sicure
- Trasportano i dati sotto forma di stringa esplicita nell'URL

```
http://indirizzo.dominio/pagina.php?parametro1=valore1&parametro2=valore2
```

Collegandoci a questo indirizzo stiamo effettuando una richiesta GET passando come dati alla pagina **`parametro1`** e **`parametro2`** posti rispettivamente a **`valore1`** e **`valore2`**

Le richieste GET e POST

Le richieste POST

- Sono più complesse da impostare perché richiedono del codice JavaScript
- Sono più sicure perché i dati non sono lasciati *in chiaro* ma sono invisibili all'utente e l'utente non può inoltrarle in maniera inconsapevole
- Anche in questo caso possiamo passare delle informazioni sotto forma di variabili

Le richieste GET e POST

Accedere ai dati da PHP

- Possiamo accedere ai dati mediante i vettori associativi **$_GET** e **$_POST**
- Nell'esempio precedente potremo accedere a **valore1** tramite **$_GET['parametro1']**
- Per verificare che la richiesta alla pagina sia GET o POST possiamo controllare il valore di **$_SERVER['REQUEST_METHOD']** come nell'esempio successivo

```php
if($_SERVER['REQUEST_METHOD']=="POST") {
    // fai qualcosa
} else
    echo "Richiesta non valida!";
```

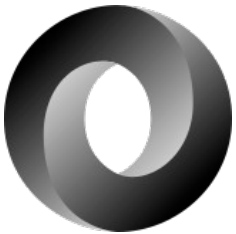
Le richieste GET e POST

Il formato JSON JavaScript Object Notation

È un formato con cui possiamo *trasformare* i vettori (specialmente gli associativi) in una stringa e viceversa; è molto utile perché facilita di gran lunga il passaggio di dati specialmente tra JavaScript e PHP

Array ([c1] => uno [c2] => due [c3] => tre [c4] => quattro [c5] => cinque)

{"c1":"uno","c2":"due","c3":"tre","c4":"quattro","c5":"cinque"}



Le richieste GET e POST

Il formato JSON JavaScript Object Notation

Possiamo codificare tramite JSON le stringhe, gli interi, i float, i booleani, i vettori e NULL.

In PHP possiamo usare le seguenti funzioni per trasformare un array associativo in una stringa JSON e viceversa

```php
// codifichiamo un ipotetico vettore in una stringa JSON
$stringa_json = json_encode($vettore);

// torniamo al vettore di partenza
$vettore_json = json_decode($stringa_json,true);
```

Il secondo parametro **`true`** indica che vogliamo decodificare la stringa in un vettore associativo e non in un oggetto