Video Esercitazione

Titolo:

Le pagine web dinamiche – le applicazioni web - l'interazione utente con le form

Esercitazione suddivisa in tre parti:

- Analisi della pagina web HTML contenente un form
- Inoltro dei dati compilati al server
- Analisi della elaborazione dati ricevuti da parte del server e l'inoltro al client della risposta

Allegati: index.html – elabora.php

Contatti: prof.corti.daniele@gmail.com

PARTE 1

Problema

Si richiede la progettazione di una applicazione che consenta ad un utente l'acquisto di biglietti online.

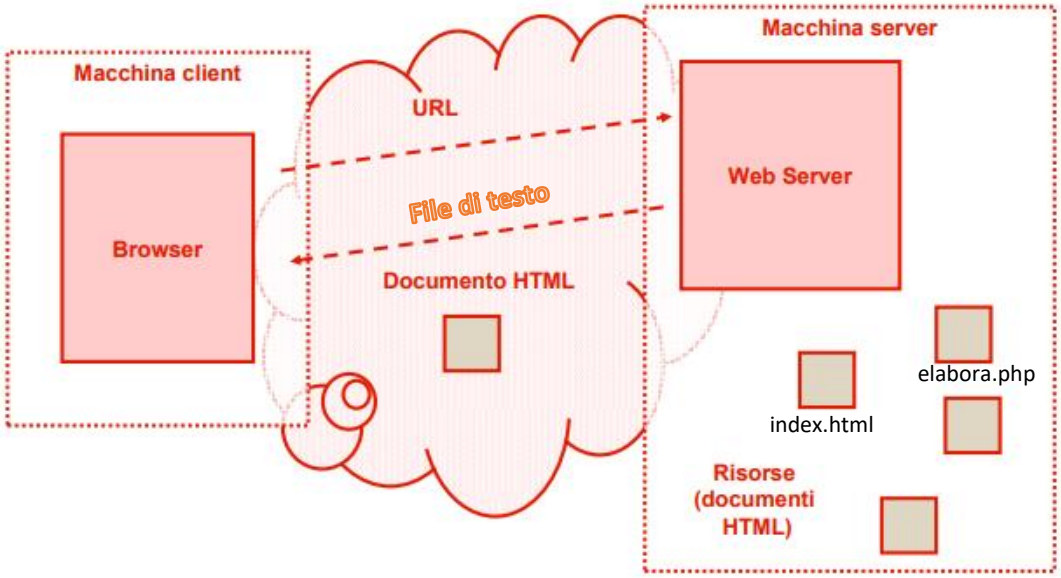
L'applicazione è costituita da due file, caricati sul web server:

- index.html: pagina HTML contenente il form compilabile dall'utente
- elabora.php: pagina PHP contenente lo script avente il compito di recuperare i dati inviati dall'utente, elaborarli e costruire una risposta HTML da inoltra all'utente stesso.

In questa video esercitazione verranno affrontati i seguenti problemi:

- Utilizzo di tecnologie lato server per lo sviluppo di pagine web dinamiche.
- Invio al server dei dati inseriti in un form dall'utente (client).
- Recupero, raccolta ed elaborazione dei dati ricevuti dal server.
- Generazione dei contenuti (in real time) da restituire al client contente i risultati della elaborazione (dipendenti dalle scelte fatte dall'utente).

Una delle principali possibilità offerte dai linguaggi di scripting lato server è quella di generare contenuti (dinamicamente) sulla base delle richieste degli utenti.



Pagina web statica

<u>Il contenuto della pagina viene stabilito nel momento in cui si crea e si memorizza la pagina sul Server Web e quindi ogni volta che un utente accede a tale pagina statica, questa gli presenta le stesse informazioni, a meno che non venga aggiornata, cioè riscritta</u>.

Pagina web dinamica

<u>Il contenuto di una pagina dinamica viene stabilito nel momento in cui si richiede quella pagina e quindi all'utente i contenuti gli vengono presentati in modi diversi a seconda di vari fattori (tipo utente, tempo, etc.).</u>

Analizziamo il processo di comunicazione client-server:

1. L'utente, attraverso il proprio computer (client), digita nella barra degli indirizzi del browser l'URL della pagina web richiesta: https://www.ingdanielecorti.it/videotutorial/PHP/post_ticket/index.html
2. L'utente preme INVIO e il browser invia la richiesta al web.
3. Il server riceve la richiesta e la prima cosa che fa è individuare se la pagina richiesta sia statica o dinamica.
4. Il server dopo aver individuato che si tratta di una pagina web statica, che altro non è che un semplice file di testo, la preleva dal proprio disco e la spedisce al browser che ne ha fatto richiesta.
5. Il browser quando riceve tale file, contenente semplicemente delle descrizioni su come il browser stesso deve mostrare a video gli elementi del form, lo interpreta (ovvero lo decodifica) mostrandone a video i contenuti (gli elementi del form).
6. L'utente compila il form (fase di interazione dell'utente con la pagina web – INPUT) e al termine clicca sul relativo pulsante "Invia" per inviare i dati al server.

7. I dati vengono inviati al server … ma in quale modo vengono intercettati i dati ed elaborati? Questo sarà l'aspetto più importante della esercitazione.

Guardiamo però prima il codice HTML contenuto nel file index.html, contenente il form, possiamo esaminare la riga 13 per comprendere in quale modo i dati compilati dall'utente vengono inviati al server, e in quale modo gli stessi vengano elaborati.

La riga 13 contiene infatti il tag form con i seguenti due importanti attributi:

- Action
- Method

All'attributo "action" è necessario assegnare come valore il nome della pagina web presente sul web server incaricata di elaborare i dati compilati dall'utente: nel nostro caso "elabora.php".

All'attributo "method" invece occorre assegnare come valore il metodo utilizzato per l'invio dei dati stessi: nel nostro caso è stato scelto il metodo "post" (l'alternativa è il metodo "get") quello più performante per l'invio dei dati.

```
<form action="elabora.php" method="post">
```

Si comprende a questo punto in che modo il client riesca a contattare il server:

1. l'utente cliccando sul pulsante "Invia" permette al browser l'invio dei dati (col metodo "post") alla pagina web "elabora.php", indicata dall'attributo "action", pagina presente sul server.
2. Il server, riconoscendo che si tratta di una pagina dinamica (contenente uno script in PHP) non fa altro che eseguirla. Lo script ha come compito di raccogliere ed elaborare i dati inviatigli dal client.
3. Il server genera la risposta, che è contenuta in semplice file di testo (pagina HTML) e la inoltra al browser dell'utente.
4. Il browser la decodificherà per mostrarne a video i risultati.

Si faccia attenzione che la pagina restituita al browser non è la pagina "elabora.php" presente fisicamente sul server, ma è la pagina HTML generata dinamicamente dallo script contenuto nel file PHP.

Provando ad aprire il codice sorgente di questa pagina restituita dal server l'utente non sarà in grado di individuare nessun codice PHP che ha generato i contenuti mostrati dal browser. Utilizzando Chrome, per esempio, cliccando con il tasto destro del mouse in un qualsiasi punto della pagina "elabora.php" e scegliendo la voce "Visualizza sorgente pagina", si potrà notare solo codice HTML generato dallo script stesso.

Analizziamo adesso in dettaglio le parti più importanti del file "index.html".

Il form è costituito da differenti elementi (controlli) classificabili in:

- Caselle di testo, come quella presente alla riga 15, utilizzabili dall'utente per inserire delle stringhe di testo.
- Caselle di opzione, come quella presente dalla riga 17 alla riga 21, che consento all'utente di scegliere una opzione fra differenti alternative.
- I radio button, come quelli presenti dalla riga 45 alla riga 47, che consentono all'utente (come le caselle di opzione) di scegliere una fra differenti alternative.

- Caselle di spunta (checkbox), come quelle presenti dalla riga 39 alla riga 42, che consentono all'utente di fare scelte multiple.
- L'area di testo (textarea), come quella presente alla riga 50, che consente all'utente l'inserimento di stringhe di testo su più righe.
- Le caselle number, come quella presente alla riga 36, che limitano l'utente nella scelta di un valore numerico compreso in un determinato range e secondo determinati step incrementali.
- Il pulsante submit, presente alla riga 53, utilizzato dall'utente per l'invio dei dati al server.
- Il pulsante reset, presenta alla riga 54, per cancellare la compilazione del form.

Un concetto fondamentale dei form, lato server, è il riconoscimento degli elementi, al fine di distinguerli fra di loro. Ogni elemento del form deve quindi avere l'attributo "name" impostato ad un determinato valore. Per esempio, l'elemento presente nella riga 15, ha l'attributo "name" impostato a "nome", quello nella riga 16 invece è impostato a "cognome". Questo è quindi l'unico modo utilizzabile dal server per distinguere i vari elementi del form, per distinguere a quale elemento del form corrisponde un determinato valore impostato dall'utente.

Titolo:

Le pagine web dinamiche – le applicazioni web - l'interazione utente con le form

PARTE 2

ARGOMENTI:

- Invio dei dati attraverso una form: GLI ATTRIBUTI ACTION E METHOD
- Gli array associativi
- La variabile array associativa $_POST utilizzata per raccogliere i dati del form

IL NOSTRO FORM

Abbiamo visto come i form permettano una interazione con gli utenti. Il form è costituito da differenti elementi anche detti **campi**.

```
<form action="elabora.php" method="post">
        <input type="text" name="nome" required> Nome<br />
        <input type="text" name="cognome" required> Cognome<br />
        <input type="text" name="tel"> Recapito telefonico<br />
        ….
</form>
```

Osserviamo come ogni elemento del form abbia un attributo **name** a cui è stato assegnato un nome univoco (**nome del campo**) al fine di riconoscere e distinguere un campo dagli atri.

```
<input type="text" name="nome_campo">
```

Vedremo fra poco come il nome del campo venga utilizzato da PHP per recuperare il valore che l'utente ha assegnato al campo stesso.

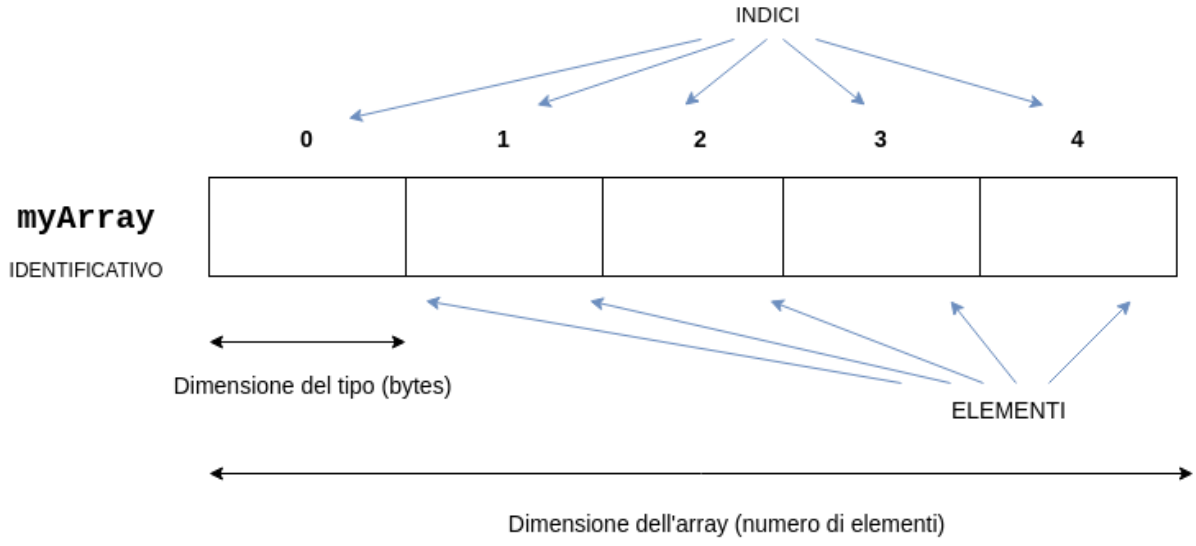
In altre parole: il valore del campo, presente nel nostro modulo HTML, avente come nome "campo1" può essere recuperato da PHP sfruttando, appunto, la variabile globale $_POST.

```
$campo1 = $_POST['campo1'];
// recupero il valore del parametro 'campo1' trasmesso mediante il metodo POST
```

Prima però ripassiamo il concetto di array

ARRAY

Un’array può essere considerato come una matrice matematica. In altri termini è una **collezione di elementi** ognuno identificato da un **indice**.



Nel linguaggio PHP gli elementi possono essere anche di differente tipologia (mentre in C++ NO), per esempio il primo può essere un intero, il secondo una stringa, il terzo un numero reale, etc.

Per poter individuale un elemento all’interno dell’array occorre associargli un indice (posizione), così la prima locazione dell’array è associata all’indice 0, la seconda all’1, etc. Se l’indice non viene esplicitato di default è di tipo numerico.

Un array in PHP può essere valorizzato in questo modo:

$nomi[0] = 'paolo'; //al primo elemento assegno il valore 'paolo'

$nomi[1] = 'luca';

$nomi[2] = 'claudio';

L’accesso ai singoli elementi avviene attraverso il proprio indice:

echo "Il mio nome è " . $nomi[2]; //visualizza claudio

Un altro modo molto di utilizzare gli array è quello di sfruttare gli indici come stringhe anziché come valore numerico (**ARRAY ASSOCIATIVI**):

$eta['paolo']=18;

$eta['luigi']=16;

$eta['carlo']=14;

Ogni elemento dell’array è caratterizzato da una coppia **chiave -> valore** in cui la chiave (o indice) è il nome di una persona e il valore è il suo anno di nascita.

Per accedere ad un elemento dell'array si può scrivere:

echo $eta['paolo'];

Se si vuole per esempio stampare l'età di uno degli utenti, si può scrivere:

echo "L'età di paolo è " . $eta['paolo'];

Come per gli indici numerici è sufficiente indicare l'indice tra parentesi quadre per accedere al valore di un elemento.

A questo punto dobbiamo analizzare il codice della pagina "elabora.php" per comprendere in che modo il server riesce a intercettare i dati inviati dal client, ovvero i valori impostati dall'utente attraverso i differenti elementi del form.

Grazie ad una particolare variabile **$_POST**, del linguaggio PHP, è possibile raccogliere in un super contenitore gli input dell'utente. Questa variabile viene **valorizzata in automatico** da PHP e contiene, sotto forma di ARRAY, tutti i valori ricevuti tramite il metodo POST.

Nome del vettore: $_POST	paolo	rossi	340334455	…
Indice del vettore	'nome'	'cognome'	'tel'	…

$nome=$_POST['nome'];

echo $nome; //stampo il valore che l'utente ha inserito nel campo nome del form

NB

Si ricordi che una generica riga di codice del tipo:

nome_variabile = espressione ;

è un'operazione di assegnamento.

L'espressione, qualsiasi sia, scritta alla destra dell'uguale, viene valutata e il valore determinato viene assegnato alla variabile scritta alla sinistra dell'uguale.

Alla destra dell'uguale troviamo scritto:

$_POST['nome']

Dove:

- $_POST: nome della variabile VETTORE discussa precedentemente.
- ['nome']: indice attraverso la quale è possibile accedere alla posizione (associativa) 'nome' all'interno del vettore stesso al fine di recuperarne il relativo valore ('paolo').

Alla sinistra dell’uguale troviamo scritto:

$nome

Il nome della variabile utilizzata per salvare il valore estratto dalla posizione 'nome' del vettore.

In modo analogo tutto il codice contenuto dalla riga 4 fino alla riga 36 contiene un blocco di operazioni atte a recuperare tutti i dati compilati dall’utente nel form e a salvarle in variabili utilizzabili nella restante parte dello script.

Si osservi inoltre quanto segue:

- il codice dalla riga 12 alla riga 19 necessario per attribuire un valore alla variabile $prezzo_biglietto a seconda dell’opzione scelta dall’utente nel relativo elemento del form. Per esempio, se il biglietto scelto è “biglietto platea” alla variabile $prezzo_biglietto viene assegnato il valore 35.
- Il codice dalla riga 21 alla riga 35 consente di determinare la somma dei prezzi delle eventuali opzioni scelte dall’utente nel relativo elemento del form.

A partire dalla riga 38 il codice inserito è relativo alla elaborazione dei dati e alla generazione della risposta che il server invierà al browser dell’utente.

Titolo:

Le pagine web dinamiche – le applicazioni web - l'interazione utente con le form

PARTE 3

ARGOMENTI:

- La variabile array associativa $_POST utilizzata per raccogliere i dati del form
- Lo script PHP contenuto nel file elabora.php

Per comprendere ciò, si analizzi per esempio la riga 3 contenente il seguente codice:

$nome=$_POST['nome'];

il codice contiene un'operazione di assegnamento.

I dati inseriti dall'utente nel form vengono inviati al server e salvati nella variabile super globale array $_POST.

Per estrarre un dato da questo campo occorre scrivere un'operazione di assegnazione come quella della riga 3, in cui il valore inserito nel campo **nome** viene estratto dalla variabile array $_POST e salvato nella variabile locale di appoggio $nome. Quest'ultima variabile verrà utilizzata nel proseguo del codice per eventuali elaborazioni e per la costruzione della risposta da inviare all'utente.

In modo analogo tutto il codice contenuto dalla riga 4 fino alla riga 36 contiene un blocco di operazioni atte a recuperare tutti i dati compilati dall'utente nel form e a salvarle in variabili utilizzabili nella restante parte dello script.

Si osservi inoltre quanto segue:

- il codice dalla riga 12 alla riga 19 necessario per attribuire un valore alla variabile $prezzo_biglietto a seconda dell'opzione scelta dall'utente nel relativo elemento del form. Per esempio, se il biglietto scelto è "biglietto platea" alla variabile $prezzo_biglietto viene assegnato il valore 35.
- Il codice dalla riga 21 alla riga 35 consente di determinare la somma dei prezzi delle eventuali opzioni scelte dall'utente nel relativo elemento del form.

A partire dalla riga 38 il codice inserito è relativo alla elaborazione dei dati e alla generazione della risposta che il server invierà al browser dell'utente.