I moduli “elettronici” – Le Form

Introduzione

Il software della Web è stato progettato in modo da funzionare su una architettura di tipo client/server.

Sul Web server vi è un programma che, ricevuta una richiesta, spedisce un documento (o un messaggio appropriato di errore) al Client richiedente.

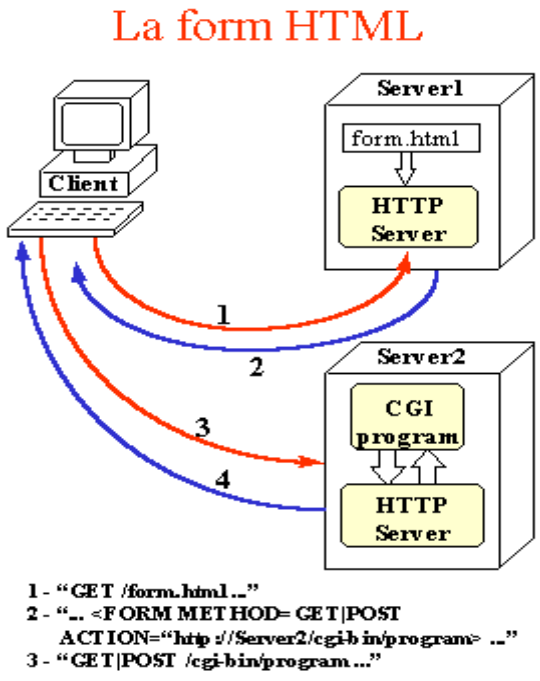
Un browser è un programma, che risiede su di un computer Client, che può inviare richieste ad un qualsiasi Web server.

L'uso di una architettura di questo tipo implica che *un programma Client* può girare su una macchina completamente distinta da quella su cui gira il server.

Il compito di immagazzinare i documenti è proprio del server mentre quello della presentazione è lasciato al cliente.

Ogni programma così può avanzare nei propri compiti indipendentemente dall'altro. I Server riescono ad ottimizzare il tempo di CPU dal momento che operano solo quando si richiedono documenti.

Ogni minuto nel WWW avvengono migliaia di transazioni tra server e client grazie a questa tecnica che utilizza il protocollo di comunicazione http.



L'equivalente informatico dei “moduli su carta”, in gergo informatico è chiamato **form**.

Un form può contenere elementi di input come campi di testo, caselle di password, caselle di controllo, radio-pulsanti, pulsanti, elenchi di selezione, ecc.

L'elemento più importante del form è l'elemento <**input**>, esso viene usato per selezionare le informazioni dell'utente.

Un elemento <input> può variare in molti modi, a seconda del tipo di attributo e quindi può essere di tipo di campo di testo, checkbox, password, pulsante, pulsante di invio, e altro ancora.

Con lo "strumento" form è possibile trasmettere dati dalla macchina sulla quale gira (Client) ad un'altra pagina che risiede su di un Server.

Metodo

Se verrà specificato l'attributo **action** all'interno del form di una pagina web, alla pressione del tasto, **sarà possibile inviare i dati del modulo compilato ad un server**; conseguentemente, la pagina di cui all'attributo action (sul server) elaborerà il flusso dati dell'input ricevuto.

Esistono due metodi di accesso ad una form: **POST** e **GET** che si differenziano nel modo di trasferire i dati: Il metodo POST non permette la visualizzazione dei dati nella fase di invio, con questo metodo i dati vengono inviati separatamente dall'url a differenza del metodo GET dove i dati vengono inviati alla pagina indicata come appendice dell'url permettendone anche la visualizzazione sulla barra indirizzi.

Dopo che i dati vengono inseriti nei vari componenti di INPUT, diventano valori di variabili di ambiente, possono a questo punto essere prelevati dagli Scripts ed essere quindi elaborati, approfondiremo i due metodi più avanti.

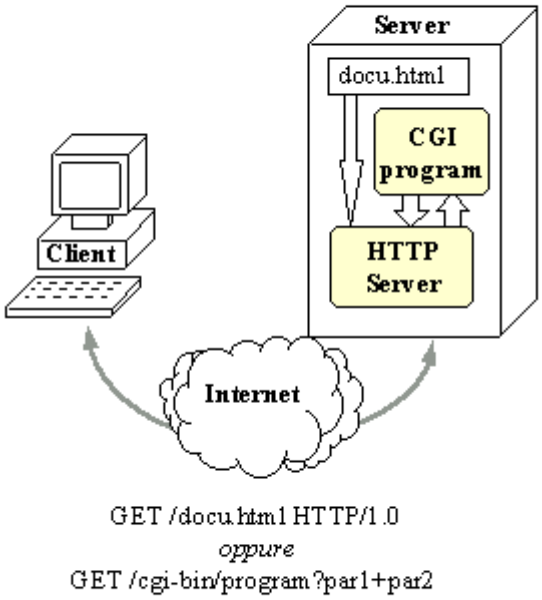
Una form, è sempre delimitata dai TAG <form> e </form>. Il Client accede alla pagina come ad una qualsiasi pagina HTML, eseguendo una GET su di essa, e ignorando a priori se questa contiene o meno una form (freccia 1 nella figura). Quando riceve la pagina, però, (freccia 2) incontra i marcatori che lo informano che la pagina contiene un modulo, e che specificano:

- che il Client, una volta sottoposto il modulo all'utente, deve "consegnarlo" al Server (specificato tramite l'attributo **action**).
- che per rimandare i dati al Server il Client deve eseguire un POST o un GET (specificato tramite l'attributo **method**).
- Altri marcatori all'interno della form specificheranno che tipo di dati l'utente deve inserire nel modulo, come e dove.

Una volta compilato il modulo, il Client rimanda i soli dati in esso inseriti al Server con il metodo opportuno (freccia 3), e riceve una risposta da mostrare all'utente (freccia 4).

Un Client può accedere ad una pagina HTML comunicando al Server il nome della pagina che desidera. Quando il Client esegue un GET a quell'indirizzo il Server legge il file e lo rimanda al Client. Tuttavia c'è un altro modo per ottenere informazioni dal Server: quello di chiedere al Server stesso di lanciare una applicazione particolare (chiamata Common Gateway), e di rimandare al Client il risultato della computazione (una pagina HTML composta dinamicamente dal programma).

CGI: Common Gateway Interface



GET /docu.html HTTP/1.0
oppure
GET /cgi-bin/program?par1+par2

Come fa il Server, una volta ricevuto una GET su un file, a capire se deve rimandare il file al Client, o se deve "eseguire" il file e mandare al Client solo il risultato ?

Il sistema è molto semplice : tutti i file presenti in una opportuna directory devono essere eseguiti, gli altri devono essere inoltrati così come sono.

La facoltà di ottenere dal Server l'ouput di un programma piuttosto che un file predefinito consente in un certo qual modo la confezione di pagine che siano su misura per il Client, e non le stesse pagine sempre e comunque per tutti. Si parla, in questo caso, di pagine generate dinamicamente, rispetto alle usuali pagine statiche.

E' evidente dunque l'utilità di poter passare al programma dei parametri. Come si vede in figura ciò è possibile specificando i parametri in coda all'indirizzo del programma, preceduti da un punto interrogativo e separati dal segno "+".

I tag utilizzati per creare le Form

Una volta definito un form mediante l'omonimo tag sarà necessario "popolarlo" con una serie di tag al suo interno.

Attraverso questi ultimi, sarà possibile creare l'interazione con l'utente.

In un documento html possono esserci più moduli ma questi dovranno essere tutti indipendenti fra di loro, nel senso che il tag **<form>** non può essere annidato (un modulo all'interno di un altro modulo) come per le tabelle o per altri tags di html. L'elemento form necessita di alcuni attributi per funzionare: *action*, *method* ed *enctype* e necessita del suo elemento di chiusura **</form>**

action

L'attributo **action** è l'azione da compiere nel momento dell'invio, di solito un URL che specifica la locazione del file o dello script al quale vengono inviati i dati del modulo, può essere anche un indirizzo di posta elettronica nel caso in cui i dati debbano essere inviati tramite il client di posta di chi lo compila. Se manca l'attributo **action** viene assunto per default lo stesso URL in cui si trova il modulo.

method

L'attributo **method** specifica il metodo per accedere all'URL dichiarato in action, due le possibilità:**post** o **get**. Il metodo **get** viene preferito per quei moduli che non necessitano di elaborazioni esterne. Per tutti gli altri casi si adopera **post**

enctype

L'attributo **enctype** specifica il tipo di media utilizzato per codificare i dati del modulo, per default è di tipo MINE.

L'elemento **<input>** rappresenta il campo più importante di un modulo. Questo campo consente all'utente di introdurre o modificare dati in diversi modi a seconda del tipo (**type**) di input adoperato.

Oltre al tipo di dati ve ne sono molti altri ma qui vi descrivo soltanto quelli realmente adoperati.

- ✓ **input** - genera la maggior parte degli elementi dei form HTML, a seconda del **type** specificato. Gli input utilizzati possono essere:
  - ➢ **text** - è utilizzato per creare *caselle di testo* in cui l'utente può scrivere del contenuto su "singola linea";
  - ➢ **file** - è utilizzato per creare *caselle di selezione di file in locale* al fine di poterli trasmettere al server remoto;
  - ➢ **radio** - permette di creare un gruppo di opzioni al cui interno deve essere fatta una scelta (non ammette scelte multiple);
  - ➢ **checkbox** - permette di creare un gruppo di opzioni al cui interno devono essere fatta delle scelte (ammette scelte multiple);
  - ➢ **button** - permette di creare bottoni "neutri" ai quali, cioè, può essere associata un'azione mediante Javascript;
  - ➢ **submit** - permette di creare bottoni di invio attraverso i quali viene, appunto inviato e processato il form;
  - ➢ **image** - permette di inserire immagini "attive" all'interno del modulo che fungeranno da bottoni;
  - ➢ **reset** - permette di creare bottoni per il reset del form (in sostanza vengono cancellate le scelte effettuate dall'utente ed il modulo torna al suo stato iniziale).
- ✓ **select** - crea una casella di riepilogo a scorrimento, chiamata in gergo **selectbox**;
- ✓ **textarea** - genera un'area di testo in cui è possibile andare a capo e viene utilizzata per permettere di inserire descrizioni, commenti o comunque testi piuttosto lunghi.

<label>...</label>

L'elemento **<label>** è una etichetta per descrivere un campo input.

```
<label for="nome"> Nome:</label><input id="nome" type="text">
```

Esempio

La pagina mostrata è un classico esempio di ``*modulo elettronico*''. In questo caso il modulo serve per raccogliere impressioni e suggerimenti dagli utenti ed inviarli al server

```
<TITLE>
   APPREZZAMENTO SETTIMANA DI RECUPERO
</TITLE>
<HEAD>
<BODY>
<H1>
  Rilevo di dati statistici per la settimana di recupero
</H1>
<HR>
<FORM action="pagina.asp" METHOD="GET" >
<P> <H2>DATI PERSONALI</H2>
<B>Nome : </B>
<INPUT NAME="NOME" TYPE ="text">
<B> Cognome : </B>
<INPUT NAME="COGNOME" TYPE ="text">
<BR>
<B> Classe : </B> <INPUT NAME="Classe" TYPE ="text">
<B>Qualifica : </B>
<SELECT NAME ="QUALI">
<OPTION SELECTED> Dirigente
<OPTION SELECTED> DSGA
<OPTION SELECTED> Docente
<OPTION SELECTED> Bidello
<OPTION SELECTED> Impiegato Segreteria
<OPTION SELECTED> Collaboratore Tecnico
<OPTION SELECTED> Studente Biennio
<OPTION SELECTED> Studente Triennio
<OPTION SELECTED> Altro
</SELECT>
<B>Sesso : </B>
M<INPUT NAME="SEX" TYPE ="radio" VALUE="M">
F<INPUT NAME="SEX" TYPE ="radio" VALUE="F">
<HR>
<H2>Note</H2>
L'interruzione mi è sembrata
Utile<INPUT NAME="UTIL" TYPE ="CHECKBOX" VALUE="SI">
<BR>
Positiva<INPUT NAME="COMP" TYPE ="CHECKBOX" VALUE="SI">
<BR>
<BR>
Secondo il tuo parere, si potrebbe migliorare questo tipo di iniziativa ?
Si<INPUT NAME="ORI" TYPE ="radio" VALUE="SI">
No<INPUT NAME="ORI" TYPE ="radio" VALUE="NO">
<P>
Quali sarebbero, a tuo giudizio, le migliorie da attuare ?
(è possibile la scelta multipla)
<BR>
<SELECT NAME ="MIGLIORIE" MULTIPLE>
<OPTION > Diminuzione del numero degli alunni per corso
<OPTION > Assegnazione aule/Laboratori
<OPTION > Attivazioni di altri tipi di interventi
<OPTION > Aumento dei tempi di recupero
```

```
<OPTION > Diminuzione dei tempi di recupero
<OPTION > Estensione degli argomenti trattati
</SELECT>
<P>
<H2>Altro</H2>
<TEXTAREA NAME="POSI" ROWS=5 COLS=60></TEXTAREA>
<P>
<INPUT TYPE="RESET" VALUE ="Cancella">
<INPUT TYPE="SUBMIT" VALUE ="Invia">
</FORM>
<HR>
</BODY>
</HEAD>
```

Rilevo di dati statistici per settimana di recupero

DATI PERSONALI

Nome **Cognome**
Classe **Qualifica :** Altro **Sesso** M F

Note

L'interruzione mi è sembrata Utile
Positiva

Secondo il tuo parere, si potrebbe migliorare questo tipo di iniziativa ? Si No

Quali sarebbero, a tuo giudizio, le migliorie da attuare ? (è possibile la scelta multipla)

- Diminuzione del numero degli alunni per corso
- Assegnazione aule/Laboratori
- Attivazioni di altri tipi di interventi
- Aumento dei tempi di recupero

Altro

Cancella Invia

Nota

Utilizzando il TYPE="***password***", sarà possibile nascondere i caratteri durante la digitazione

Password: **<input** type="password" name="pwd"**>**

POST

```
POST /cgi-bin/my_prog HTTP/1.0
Accept: text/html
...  (tutti gli altri formati accettati)
User-Agent: ...
Content-type: ...
Content-length: ...
    (una linea vuota)
org=Roma%20Tre
&browser=netscape
```

GET

```
GET /cgi-bin/my_prog?org=Roma%20Tre
                    +brow=netscape HTTP/1.0
Accept: text/html
...  (tutti gli altri formati accettati)
User-Agent: ...
Content-type: ...
Content-length: ...
    (una linea vuota)
```

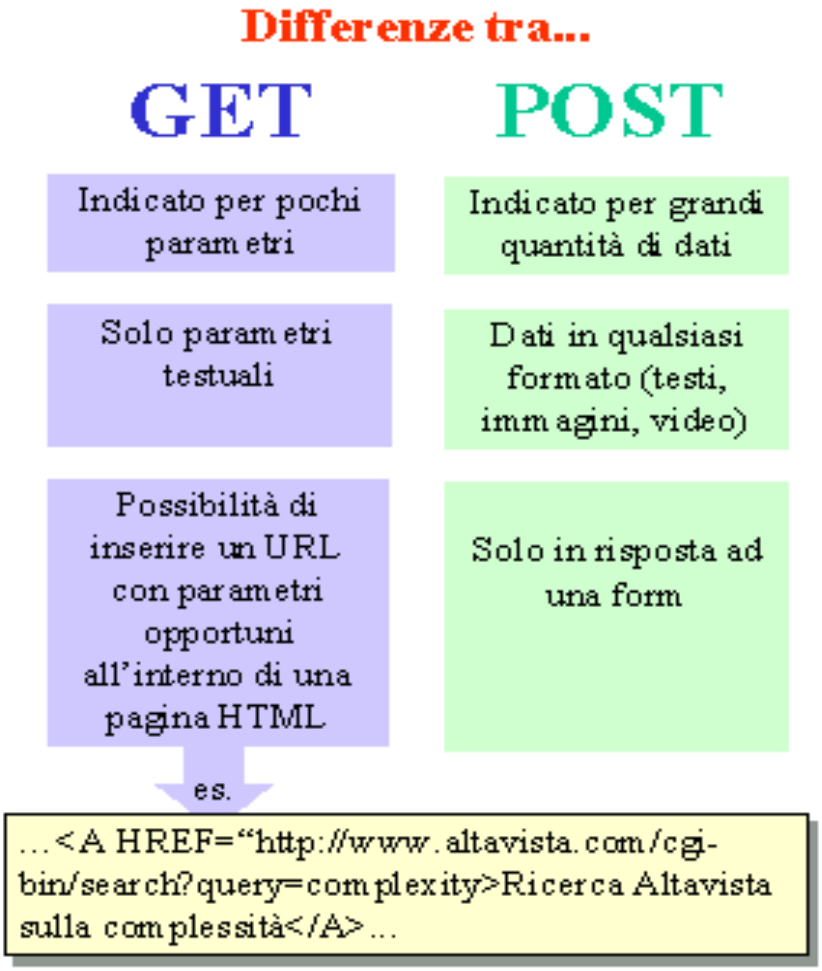
Direttive POST e GET

Quando l'utente preme il tasto *submit*, il Client rimanda i dati inseriti nel modulo al Server opportuno.

Se il METHOD selezionato era **POST** i dati vengono posti alla fine dell'header, in coppie label_variabile = valore_variabile, separate tra loro dal simbolo "**&**". Gli spazi vengono sostituiti da **%20** (il simbolo % indica che seguirà un numero esadecimale che individua il carattere nella tabella ASCII; in questo caso l'esadecimale 20 corrisponde al decimale 32, che nella tabella ASCII individua lo spazio).

Se il METHOD è uguale a **GET** i parametri (sempre in coppie label_variabile=valore_variabile) vengono tutti allineati sulla riga dell'indirizzo del programma CGI, separati da questo da un "**?**" e separati tra loro dal una segno "**+**".

Le Differenze tra GET e POST



1) Il metodo GET è più indicato del metodo POST quando sono in gioco pochi parametri. Se invece i parametri sono troppi o troppo lunghi, passarli sulla linea di comando, oltre che essere poco pratico, potrebbe addirittura essere impossibile, in quanto il sistema operativo del Server potrebbe avere una lunghezza massima per i comandi.

2) Sulla linea di comando si possono passare solamente parametri testuali, dunque il metodo GET è vincolato ad utilizzare solo tale formato, mentre con il metodo POST qualsiasi altro formato può essere adottato. Possono così essere scambiati tra Client e Server anche files con immagini, sonoro, video, ecc.

3) Una terza differenza tra il metodo GET ed il metodo POST è nel fatto che una chiamata GET viene eseguita normalmente sull'URL che segue l'attributo **HREF** di un marcatore **<A>** all'interno di una pagina HTML generica (come nell'esempio in figura), mentre un POST viene eseguito dal Client solo in risposta ad una Form.