Linguaggi di Programmazione per il Web – Parte 7

Interazione tra HTML e PHP

I tag FORM e TABLE

Autore

Prof. **Rio Chierego**
riochierego@libero.it

Siti Utili

http://www.riochierego.it/mobile

http://www.html.it/

http://www.mrwebmaster.it

http://www.php.net/

http://php.html.it/

HTML: IL TAG FORM

Uno dei fattori che ha decretato il successo del Web è senz'altro la possibilità di **interagire** con esso

Esempio: la possibilità cioè di iscriversi a servizi di vario tipo (ad esempio mailing list), o effettuare ricerche impostando opportunamente i criteri.

Per organizzare questo genere di servizi è necessario raccogliere in qualche modo i **dati dell'utente**: per farlo si utilizzano, in maniera molto semplice, i **moduli** o **form**.

L'invio dei dati in un **form** è solitamente organizzato in due parti:

- una **pagina principale** che contiene i vari campi dei form, che consentono all'utente di effettuare delle scelte, scrivere del testo, impostare delle opzioni, etc.
- una **pagina secondaria** che viene richiamata dalla principale e che effettua "il lavoro" vero e proprio di processare e raccogliere i dati. Di norma si tratta di una pagina di programmazione che si trova sul server.

Può essere un cgi, oppure una pagina asp, **php**, jsp o altro

IL TAG FORM

La struttura di un modulo o form in HTML può essere sintetizzato come segue:

<form name = "…." action = "…." method ="…." target = "…." >
// campi di testo (tag INPUT, tag TEXTAREA)
// check button (tag INPUT)
// radio button esclusivi (oppure non esclusivi) (tag INPUT)
// menù di scelta semplice (tag SELECT)
// menù di scelta multipla (tag SELECT)
// bottoni standard di conferma, annulla e reset (tag INPUT, tag IMAGE)
</form>

I TAG fieldset e legend

Per la loro natura di "raccoglitori di informazioni", i moduli tendono a ingigantirsi e diventare lunghissimi. Per questo, a partire dall'HTML 4 sono stati introdotti dei tag per fare un po' d'ordine all'interno dei form.

Grazie al tag **<fieldset>** …. **</fieldset>** possiamo creare delle macro-aree all'interno dei form, e grazie al tag **<legend>…. </legend>**, possiamo indicare il nome di ciascuna macro-area.

Esempio: Poniamo ad esempio di dover raccogliere i dati di un utente, raccogliendo dati anagrafici, residenza, domicilio e reperibilità sul lavoro.

Possiamo farlo con la seguente sintassi:

```
<form name="datiUtenti“ action="paginaRisposta.php” method=“POST”>
  <fieldset>
    <legend> Dati Anagrafici </legend>
        …….
   </fieldset>

  <fieldset>
    <legend> Residenza </legend>
        …….
    </fieldset>
    …….
</form>
```

GLI ATTRIBUTI DEL TAG FORM

Vediamo più in dettaglio i possibili attributi del tag FORM:

name : serve ad indicare il nome del form per un eventuale richiamo di css o script (declassato a favore di id =“…”) ;

id : serve ad indicare il nome del form per un eventuale richiamo di css o script ;

action : indica l'URL del programma o della pagina di risposta che processerà i dati;

method : indica il metodo di invio dei dati del form al file indicato in “name”. Quando creiamo un form possiamo scegliere due metodi di invio: GET e POST;

target : stabilisce come far aprire i dati inviati dal form in una pagina differente rispetto a quella corrente (o in una determinata parte di un frameset);

enctype *: i*ndica il tipo di contenuto del form quando method=“POST“.

Può valere application/x-www-form-urlencoded valore di default (se non specificato nel tag *form*) per inviare testo o valori comuni oppure multipart/form-data per inviare grandi quantità di dati (file, immagini ecc.) oppure del testo che contenga alcuni caratteri non ascii;

class : specifica il nome della classe css che ne regola la grafica (è possibile utilizzare anche style=" ").

L'ATTRIBUTO method

Con il metodo **GET** la pagina di risposta viene contattata e i dati vengono inviati in un unico step. Nell'URL della pagina di risposta potremo allora vedere tutti i parametri nella barra degli indirizzi (più precisamente nella "query string", cioè nella "stringa di interrogazione") secondo una forma prestabilita:

paginaRisposta.php?par1=val1&par2=val2&…&parn=valn

Alcuni server hanno tuttavia hanno delle limitazioni per quel che riguarda il metodo GET e non consentono di inviare form con valori superiori a 255 caratteri complessivi. Il metodo GET è dunque particolarmente indicato per form con pochi campi e pochi dati da inviare.

Nel metodo **POST** invece l'invio dei dati avviene in due step distinti: prima viene contattata la pagina sul server che deve processare i dati, e poi vengono inviati i dati stessi. Per questo motivo i parametri non compaiono nella query string (dunque se non si desidera che i parametri siano mostrati all'utente questo metodo è preferibile).

In questo caso non ci sono limiti sulla lunghezza dei caratteri.

IL TAG label

Un altro tag particolarmente utile - si può utilizzare con ogni tipo di campo che vedremo d'ora in poi - è il tag **<label>… </label>,** che permette di indicare un'etichetta per il nome del campo.

Ovviamente il campo su cui si vogliono dare delle indicazioni deve essere compreso all'interno del tag **<label>** stesso.

IL TAG input

Il tag **<input>** associato all'attributo tipo (type="…") dà vita a numerosi formati per raccogliere informazioni dal navigatore (caselle di testo, pulsanti, password, immagini, file ecc.).

La sintassi è molto semplice, anche se ogni tipo ha la propria specifica ed è caratterizzato dal fatto di non avere tag di chiusura:

< **input** name="…." type="…." id="…." etc. etc.>

Attributi comuni più importanti:

name = "…" indica il nome del campo. Servirà per richiamarlo con script o altre procedure.

type = "…" i più importanti **text**, **password**, **checkbox**, **radio**, **hidden**, **submit**, **reset**

id = "…" indica il nome per un'eventuale richiamo css o script.

IL TAG input : type “text” e “password”

Il **type =“text”** consente di inserire il testo dentro un campo composto da una sola riga (a differenza del tag textarea).

Se <input type = "text“ ….. > allora sono utilizzabili i seguenti attributi:

maxlength = “…" numero max di caratteri che possiamo immettere.

size = “…" lunghezza effettiva del controllo misurata in numero di caratteri.

value = “…" eventuale valore iniziale.

readonly se inseriamo questo attributo in un campo text precompilato (input type="text" value="frase precompilata"), il testo può solo essere letto e NON modificabile dal navigatore

Il **type =“password”** consente di inserire del testo coperto da asterischi.

Se <input type =“password“ …. > allora sono utilizzabili i seguenti attributi:

maxlength = “…" numero max di caratteri che possiamo immettere.

size = “…" lunghezza effettiva del controllo misurata in numero di caratteri.

IL TAG input : type “checkbox” e “radio”

Il **type="checkbox"** consente di inserire una casella di spunta all'interno del form. A differenza del type="radio" le caselle checkbox possono essere attivate senza un numero max di limite.

Se <input type =“checkbox“ …. > allora sono utilizzabili i seguenti attributi:

checked = “…" questo attributo, inserito senza alcun valore (=" ") crea una casella non spuntata

value = “…" indica il valore da associare ad una determinata *check box* .

Il **type="radio"** inserisce una serie di caselle radio dove soltanto una può essere selezionata. Con l'attributo checked si preseleziona in automatico una casella. Di default (se non è presente checked) è selezionato il primo radio del form. Si suggerisce di inserire sempre l'attributo checked perchè non tutti i browser agiscono di default.

Se <input type = “radio“ …. > allora sono utilizzabili i seguenti attributi:

name = “…" nel tipo radio l'attributo *name* è importante perché, se uguale, raggruppa tutte le caselle radio che compongono il form .

value = “…" indica il valore da associare ad un determinato *radio* .

checked = “…" indica quello preselezionato in automatico dal form .

by Prof. Rio Chierego

IL TAG input : type “hidden” e “file”

Il **type=“hidden"** inseriamo un campo nascosto nel form per la trasmissione dei dati. La variabile che generiamo sarà utile per eventuali controlli o trasmissioni di dati che altrimenti andrebbero persi con l'invio del form.

Se <input type =“hidden“ …. > allora sono utilizzabili i seguenti attributi:

name = “…" questo attributo, indica una variabile passata in modo nascosto

value = “…" valore assegnato alla variabile da passare senza nessuna azione diretta dell’utente.

Il **type=“file"** consente di selezionare un file (.txt immagine od altro) dal computer del navigatore per inviarlo tramite il form. In questo caso nella riga iniziale del form si deve utilizzare il metodo **POST** ed ***enctype="multipart/form-data***".

Se <input type = “file“ …. > allora sono utilizzabili i seguenti attributi:

multiple se presente si specifica che è possibile scegliere più file contemporaneamente

L'utente può scegliere più file dal selettore di file in qualsiasi modo consentito dalla piattaforma scelta (ad es. Tenendo premuto Maiusc o Controllo , quindi facendo clic). Se si desidera che l'utente scelga un solo singolo file per <input> basta ometterlo .

HTML5: I nuovi TAG input

- **Validare l'input** e **migliorare l'esperienza utente** sulle pagine grazie ai nuovi elementi di tipo input: dagli attributi per la validazione agli slider alle date e ai color picker

- Tra le tante novità introdotte, una delle più importanti è sicuramente caratterizzata dai nuovi elementi di tipo input. Sono infatti molti i nuovi tipi di input, tra cui **date**, **datetime**, **mail**, **month**, **number**, **range**, **search**, **tel**, **url**, **time** e **week**

- Pur non essendo ancora in versione stabile, le novità introdotte dalle specifice dell'HTML5 prendono piede all'interno di progetti reali. Son sempre più i dispositivi e i browser moderni che iniziano a supportare le nuove feature e, di conseguenza, migliorare l'esperienza utente

HTML5: I nuovi TAG input

Prima di analizzare in dettaglio le funzionalita dei nuovi elementi di input è doveroso introdurre due nuovi attributi molto importanti: **required** e **pattern**

Questi due attributi ci consentono di simulare due aspetti molto importanti da tenere in considerazione quando si progetta una form: **la validazione dei dati**.

L'attributo required

- L'attributo required, come ci suggerisce il nome, ci consente di impedire l'invio del modulo se i campi obbligatori non sono stati compilati. Il suo funzionamento è davvero molto semplice:

Esempio: <input type="text" name="username" **required** />

L'attributo pattern

- L'attributo pattern può essere utilizzato in combinazione con l'attributo required appena visto. Esso, infatti, consente di definire delle espressioni regolari (**nel formato supportato da JavaScript**) per validare i dati dei moduli.

Esempio: <input type="text" name="url" **required pattern="https?://.+"** />

Questo campo di input accetterà al suo interno solo valori che iniziano per **http://** oppure **https://**

HTML5: I nuovi TAG input - EXTRA

Personalizzare lo stile dei campi validi/non validi

- Una tecnica interessantissima che possiamo sfruttare per segnalare all'utente la validità o l'errore nella compilazione dei form consiste nel definire regole CSS ad hoc sfruttando gli **pseudo-selettori *:required*, *:valid*, *:invalid***.
- Con CSS3, infatti, viene introdotto anche il supporto a questi nuovi pseudo-selettori, grazie ai quali non abbiamo più bisogno di Javascript per assegnare nuove classi agli elementi in base ai dati da esso contenuti.

Facciamo un esempio pratico utilizzando uno sprite per mostrare quali tra i campi inseriti sono validi o presentano errori:

HTML5: I nuovi TAG input - pattern

Come costruire il pattern: la sintassi delle espressioni regolari

Veniamo adesso alla parte più difficile dell'argomento espressioni regolari, ovvero la definizione dei **pattern** (sequenze di caratteri di controllo), dei costrutti sintattici e semantici.

Una volta entrati nei vari meccanismi che regolano la definizione di un'espressione regolare, più o meno semplice o generica, risulterà molto semplice risolvere problemi apparentemente diffici, risolvibili altrimenti con lunghe routine di controllo attraverso il linguaggio di programmazione o di scripting utilizzato.

Esempio: pattern = espressione = "**^[a-z 0-9]+$"**

- Analizziamo la sintassi utilizzata all'interno di questa espressione regolare:
- **^** indica che la stringa deve iniziare con...;
- **[** e **]** indicano qualsiasi carattere tra quelli al loro interno;
- **a-z** e **0-9** indica che la stringa può contenere tutti i caratteri che vanno da "a" a "z" e da 0 a 9;
- **+** indica che la stringa deve contenere almeno uno dei caratteri tra le parentesi quadre;
- **$** indica il termine della stringa;

HTML5: I nuovi TAG input - pattern

- Ipotizziamo che uno dei caratteri da controllare in un'espressione regolare sia, accidentalmente, un metacarattere delle espressioni regolari o altro carattere speciale (come ad esempio una parentesi quadra).
- Per farlo utilizzeremo il carattere **back-slash** (\) per isolarlo.

Facciamo un esempio supponendo di dover verificare che una data stringa contenga il carattere "^":

"\^"

Se il carattere da controllare è proprio un back-slash, utilizzeremo un doppio back-slash:

"\\"

Il primo back-slash fungerà da **literal escape** ed il secondo verrà interpretato come carattere semplice e non come speciale.

HTML5: I nuovi TAG input - pattern

Vediamo di seguito un riepilogo dei caratteri speciali, dei metacaratteri e delle strutture sintattiche utilizzati nelle espressioni regolari:

Marcatori di inizio e termine stringa:

- **^** verifica l'inizio di una stringa (attenzione! se utilizzato all'interno di una parentesi quadra indica una negazione);
- **$** verifica la fine di una stringa;

Metacaratteri:

- **.** qualsiasi carattere ad esclusione del ritorno a capo;
- **\w** - verifica che una stringa sia alfanumerica, minuscola
- **\W** - verifica che una stringa sia alfanumerica, maiuscola
- **\d** - indica qualsiasi cifra numerica;
- **\D** - indica qualsiasi carattere che non sia una cifra numerica;
- **\s** - indica uno spazio vuoto;
- **\S** - indica qualsiasi carattere che non sia uno spazio vuoto;
- **\t** - indica una tabulazione;
- **\r** - indica un *carriage return*;
- **\n** - indica un *linefeed*;

HTML5: I nuovi TAG input - pattern

Parentesi quadre:

- **[...]** le parentesi quadre identificano *range* e classi di caratteri;
- **[abc]** identifica qualsiasi carattere tra quelli tra parentesi quadra;
- **[^abc]** identifica qualsiasi carattere ad esclusione di quelli tra quelli tra parentesi quadra (^ funge da negazione);
- **[a-z]** o **[0-9]** il trattino all'interno della parentesi quadra identifica un *range* di caratteri;

Parentesi tonde:

- **(...)** le parentesi tonde identificano dei gruppi di caratteri (i gruppi diventano fondamentali se si deve effettuare una sostituzione mediante espressioni regolari);
- **(a|b)** il simbolo | identifica un alternanza (o "a" o "b") all'interno del gruppo;

Ripetizioni di caratteri:

- **{x}** indica il carattere precedente per x volte
- **{x,}** indica il carattere precedente per x o più volte
- **{x,y}** il carattere precedente si ripetere x volte ma non più di y
- **?** indica il carattere precedente per 0 o 1 occorrenza
- ***** equivale a {0,}
- **+** equivale a {1,}

HTML5: I nuovi TAG input - EXTRA

Definiamo anche le due regole CSS:

```css
input:required:invalid,
input:focus:invalid
{
  background-image: url(/images/form_validation.png);
  background-position: right bottom;
  background-repeat: no-repeat;
}
input:required:valid
{
  background-image: url(/images/form_validation.png);
  background-position: right top;
  background-repeat: no-repeat;
}
```

Come tipico per gli sprite, modifichiamo la posizione dell'immagine di background del campo di input a seconda della sua validità.

IL TAG input : type “date”, “datetime” e “color”

Il **type= "date"** si occupa di mostrare un **date picker**, ovvero un piccolo tool con cui selezionare una data, scegliendola da un piccolo calendario navigabile; il tutto nativamente all’interno del browser

Se <input type =“date“ …. > allora sono utilizzabili i seguenti attributi:

min= “…" questo attributo indica la data più piccola che può essere inserita

max= “…" questo attributo la data massima che può essere inserita.

step= “…" questo attributo permette di definire un valore di intervallo rispetto al giorno attuale.

Ad esempio, supponendo di essere al giorno 10 del mese e di impostare uno step di 2, saranno validi solo i giorni 12, 14, 16, e così via.

Il **type= "datetime"** ha lo stesso comportamento del tipo **date** con in più la possibilità di far scegliere all’utente oltre alla data anche l’orario, incluso il **timezone**

Il **type= “color"** : spesso si ha la necessità di far personalizzare all’utente il colore di un determinato elemento e, finora, abbiamo sempre dovuto includere uno script Javascript che inserisse un color picker nelle nostre pagine. Questa proprietà ci consente di sfruttare direttamente la finestra dei colori del sistema operativo

IL TAG input : type “mail” e “month”

Il **type= “mail"** è un classico campo di input che consente in aggiunta di validare automaticamente il valore inserito in esso. In parole più semplici mostra un avviso nel caso in cui il valore inserito non è un indirizzo email valido

Se <input type =“mail“ …. > allora sono utilizzabili i seguenti attributi:

pattern = “…" questo attributo permette di definire un’espressione regolare per validare i valori inseriti.

N.B. Per maggiori dettagli su questo attributo si rimanda alla slide n. 16

Il **type= “month"** è del tutto uguale al date o il datetime con la sola differenza che consente di scegliere solo mese e anno, senza il giorno

Se <input type =“month“ …. > allora sono utilizzabili i seguenti attributi:

min= “…" questo attributo indica il mese più piccolo che può essere selezionato

max= “…" questo attributo indica il mese più grande che può essere selezionato

step= “…" questo attributo permette di definire un valore di intervallo rispetto al mese attuale. Ad esempio, supponendo di essere nel mese di Febbraio e di impostare uno step di 2, saranno validi solo i mesi di Aprile, Giugno e così via.

by Prof. Rio Chierego

IL TAG input : type “number”

Il **type= “number"** è molto probabilmente uno dei più interessanti e, allo stato attuale, più utilizzato.

Come è facilmente intuibile dal nome, grazie ad esso è possibile far accettare al campo di input solo valori numerici. Attraverso gli attributi min e max, poi, è anche possibile definire l’intervallo in cui deve essere contenuto il valore. Alcuni browser, inoltre, forniscono anche delle freccette con cui aumentare e diminuire il valore del campo senza il bisogno di inserirlo da tastiera.

Se <input type =“number“ …. > allora sono utilizzabili i seguenti attributi:

min= “…“ per definire il valore minore inseribile

max= “…" per definire il valore maggiore inseribile

step= “…" per definire l’intervallo di avanzamento.

Molti browser, infatti, forniscono due piccole freccette per aumentare o diminuire il valore contenuto nel campo. Utilizzando l’attributo step i valori aumenteranno o diminuiranno in funzione di questo valore. Ad esempio, supponendo di aver definito uno step di valore 5, il contenuto del campo di input varierà di 5 unità ogni volta.

IL TAG input : type “range” e “search”

Il **type= “range"** ha lo stesso funzionamento del tipo number con una differente visualizzazione grafica. Il range, infatti, consente di far selezionare all’utente un valore numerico compreso tra due estremi di un intervallo numerico attraverso l’utilizzo di uno slider..

Se <input type =“range“ …. > allora sono utilizzabili i seguenti attributi:

min= “…“ per definire il valore minore inseribile

max= “…" per definire il valore maggiore inseribile

step= “…" per definire l’intervallo di avanzamento.

Il **type= “search“** è utilizzato per i campi di ricerca.

Esso è a tutti gli effetti un classico campo di testo che alcuni browser interpretano aggiungendo una semplice x al suo interno con cui è possibile svuotarne il contenuto

Se <input type =“search“ …. > allora sono utilizzabili i seguenti attributi:

pattern = “…" questo attributo permette di definire un’espressione regolare per validare i valori inseriti.

N.B. Per maggiori dettagli su questo attributo si rimanda alla slide n. 16

IL TAG input : type “tel” e “url”

Il **type= “tel"** : secondo la specifica del W3C questo campo può essere utilizzato per contenere numeri telefonici. Per quanto ancora nessun browser desktop fornisca una sua interpretazione, è consigliabile utilizzare questo tipo quando si devono gestire numeri telefonici poiché nei dispositivi mobili viene fornita una tastiera con valori numerici che, ovviamente, facilita l’inserimento da parte dell’utente.

Se <input type =“tel“ …. > allora sono utilizzabili i seguenti attributi:

pattern = “…" questo attributo permette di definire un’espressione regolare per validare i valori inseriti.

N.B. Per maggiori dettagli su questo attributo si rimanda alla slide n. 16

Il **type= “url"** è molto utile quando si devono far inserire all’utente degli indirizzi web. Anche questo tipo è un classico campo di testo che, all’invio della form, ci avvisa se il valore inserito non è un indirizzo valido

Se <input type =“url“ …. > allora sono utilizzabili i seguenti attributi:

pattern = “…" questo attributo permette di definire un’espressione regolare per validare i valori inseriti oltre alla validazione interna degli url che già è integrata al suo interno

IL TAG input : type “time” e “week”

Il **type= “time"** sempre in tema con i tipi relativi alla gestione delle date, il tipo time consente di far selezionare all’utente un orario

Se <input type =“time“ …. > allora sono utilizzabili i seguenti attributi:

min= “…" questo attributo indica l’orario più piccolo che può essere selezionato

max= “…" questo attributo indica l’orario più grande che può essere selezionato

step= “…" questo attributo permette di definire un valore di intervallo rispetto all’ora attuale.

Il **type= “week"** Ultimo tipo introdotto nell’HTML5 è il tipo week, anch’esso utilizzato per gestire le date. Quest ultimo ha lo stesso comportamento di date, datetime, ecc. con la differenza che è utilizzato per selezionare le settimane

Se <input type =“week“ …. > allora sono utilizzabili i seguenti attributi:

min= “…" questo attributo indica la settimana più piccola che può essere selezionata

max= “…" questo attributo indica la settimana più grande che può essere selezionata

step= “…" questo attributo permette di definire un valore di intervallo rispetto alla settimana.

by Prof. Rio Chierego

IL TAG input : type “submit”, “reset” e “button”

Il **type=“submit"** consente di inviare il form alla pagina o programma prestabilito all'inizio. Se cliccato invieremo tutte le informazioni che il navigatore ha inserito nel form.

.

Se <input type =“submit“ …. > allora sono utilizzabili i seguenti attributi:

value = “…" il testo che leggiamo sopra al pulsante.

Il **type=“reset"** consente di resettare il form allo stato iniziale. In pratica cancella tutti i dati inseriti dal navigatore e riporta tutto il form allo stato di partenza

Se <input type = “reset“ …. > allora sono utilizzabili i seguenti attributi:

value = “…" il testo che leggiamo sopra al pulsante.

Il **type=“button"** si utilizza quando il form può contenere una funzione javascript, magari per aiutare nel compilamento oppure per fare un controllo

.

Se <input type =“button“ …. > allora sono utilizzabili i seguenti attributi:

value = “…" il testo che leggiamo sopra al pulsante

onclick = “…" indica cosa fare se cliccato (javascript).

ondblclick = “…" indica cosa fare se cliccato due vlte (javascript)

by Prof. Rio Chierego

IL TAG input : type “image”

Il **type=“image"** crea un pulsante di tipo submit (per l'invio del form) con l'inserimento di un'immagine. Si consiglia sempre di inserire l'attributo alt=" " per consentire leggere un testo alternativo nel caso l'immagine non sia raggiungibile o non visibile dal browser.

L'invio dei dati comprende anche l'invio delle coordinate del click (x ed y) consentendo di applicare un'azione diversa a seconda della posizione del clic.
In pratica otterremo l'invio del form con due variabili chiamate:

nome.x *:* con ***nome*** che prende il valore dell'attributo ***name=“…".*** Questa variabile trasmette la coordinata del clic a partire dalla sinistra dell'immagine (valore in pixel).

nome.y *:* con ***nome*** che prende il valore dell'attributo ***name=“…".*** Questa variabile trasmette la coordinata del clic a partire dall'alto dell'immagine (valore in pixel).

Se <input type =“image“ …. > allora sono utilizzabili i seguenti attributi:

name = “…" il nome della variabile che sarà spedita con il clic e sarà associata al valore .x e .y delle coordinate .

src = “…" il percorso URL dell'immagine da visualizzare .

alt = “…" il testo che apparirà se l'immagine non è visibile .

IL TAG input : Tabella di compatibilità

by Prof. Rio Chierego

IL TAG textarea

Il **tag <textarea>…</textarea>** consente di inserire del testo da leggere all'interno di un form.

Se <textarea …. > allora sono utilizzabili i seguenti attributi:

name = "…" Il nome associato alla textarea identificativo per un richiamo css o script .

title = "…" Il titolo associato alla textarea .

rows = "…" Il numero di righe che compone la textarea. Inserire un numero intero senza unità di misura, che corrisponde al numero di caratteri che vogliamo vedere .

cols = "…" Il numero di colonne che compone la textarea. Inserire un numero intero senza unità di misura, che corrisponde al numero di caratteri che vogliamo vedere.

id = "…" Il nome identificativo per un richiamo css o script .

readonly = "…" Se inseriamo questo attributo in una textarea precompilata, il testo può solo essere letto e NON modificabile dal navigatore

wrap = "…" Imposta il tipo di tabulazione del testo. Può avere tre valori:

(-) ***virtual*** il testo va a capo ma viene spedito su una sola linea

(-) ***physical*** il testo viene spedito come è scritto

(-) ***off*** il testo non va a capo e viene spedito così

I TAG select e option

Il **tag <select>…</select>** consente al navigatore di scegliere fra più opzioni.
Le opzioni possono essere singole o multiple.
Il **tag select** al suo interno contiene i **tag <option>…</option>**, che in pratica sono le opzioni cliccabili disponibili

Se <select …. > allora sono utilizzabili i seguenti attributi:

name = "…" il nome che identifica le opzioni per un richiamo css o script .

title = "…" il titolo del tag *select* .

id = "…" il nome identificativo per un eventuale richiamo css o script.

size = "…" il numero max di righe da vedere

multiple se inseriamo questo valore si abilitano scelte multiple

Se <option …. > allora sono utilizzabili i seguenti attributi:

selected l'opzione visualizzata in partenza. Se omesso non si vedrà niente oppure il primo option (dipende dal tipo di browser).

value = "…" il valore dell'opzione.

N.B. Se inseriamo il valore **multiple** possiamo inserire più **selected** contemporaneamente. Se abbiamo più selected ma non abbiamo inserito multiple nel tag <select> creeremo un errore.

Il TAG optgroup

Il **tag <optgroup>…</optgroup>** si utilizza con il tag **select** solo se all'interno dobbiamo inserire dei gruppi di opzioni. Il nome del gruppo di opzione è inserito dall'attributo **label="…"**

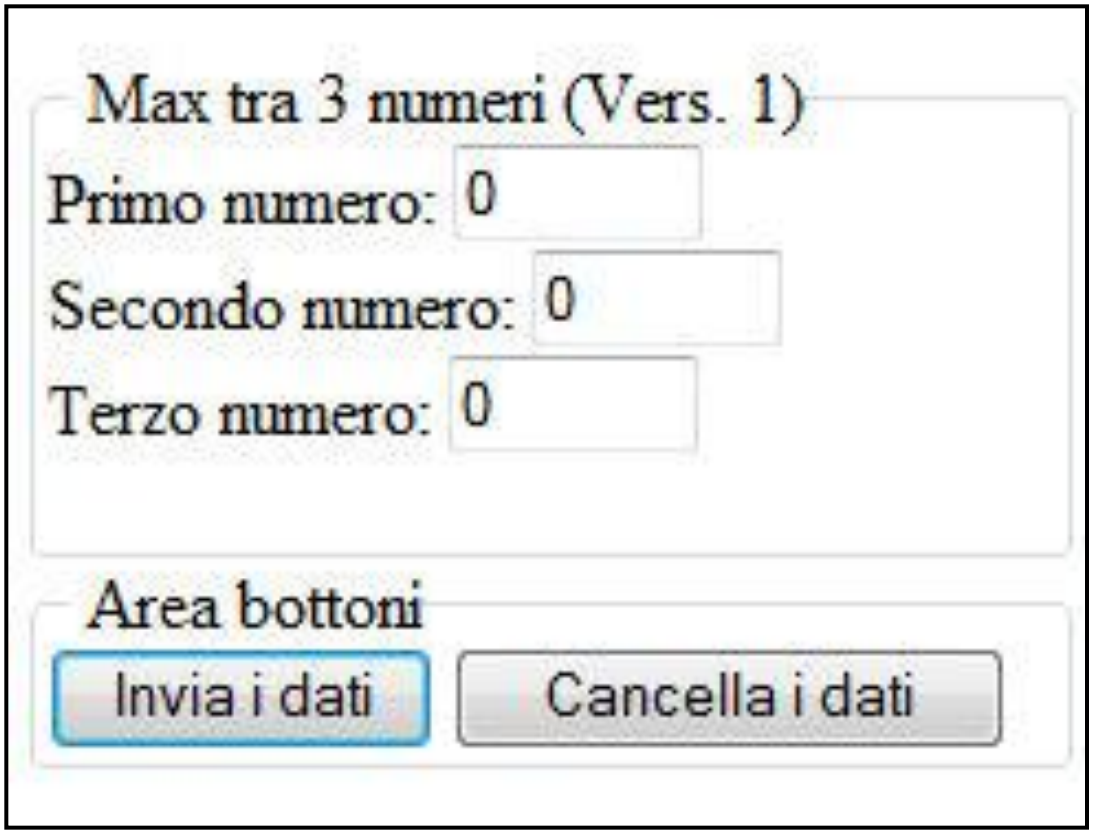
Se <optgroup …. > allora sono utilizzabili i seguenti attributi:

title = "…" il titolo del gruppo .

id = "…" il nome identificativo per un eventuale richiamo css o script.

label = "…" Il nome del gruppo di opzioni

Interazione HTML e PHP:
max tra 3 numeri



Layout browser

Interazione HTML e PHP: max tra 3 numeri

```
<form name="esempio1" action="esempio1.php" method="POST" target="_blank" >
 <fieldset>
  <legend>Max tra 3 numeri (Vers. 1)</legend>
    <label>Primo numero: <input type="text" name="a1" size='5' maxlength='5' value="0"
tabindex="1"><br>
    </label>
    <label>Secondo numero: <input type="text" name="b1" size='5' maxlength='5' value="0"
tabindex="2" ><br>
    </label>
    <label>Terzo numero: <input type="text" name="c1" size='5' maxlength='5' value="0"
tabindex="3"><br><br>
    </label>
 </fieldset>
  <fieldset>
   <legend>Area bottoni</legend>
     <input type="submit" value="Invia i dati" tabindex="4">
     <input type="reset"  value="Cancella i dati" tabindex="5">
  </fieldset>
 </form>
```

Codice HTML

by Prof. Rio Chierego

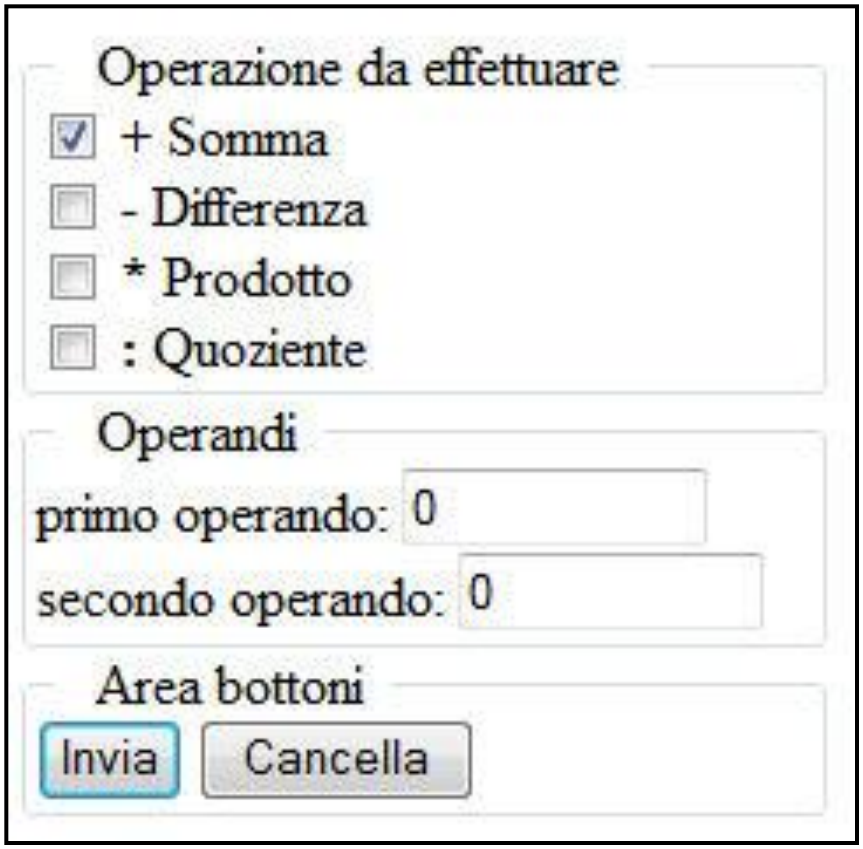
Interazione HTML e PHP:
max tra 3 numeri

```
 <?php
echo "
<html>
  <head> <title> Max tra 3 numeri</title>
  </head>
  <body>";
//METODO POST reperimento dati dal form di input
$a = $_POST["a1"];
$b = $_POST["b1"];
$c = $_POST["c1"];
// inizializzo minimo e massimo al primo valore
$max = $a;
$min = $a;
// controllo se il secondo dato è il massimo
if ($b > $max)
   {
   $max = $b;
   }
// controllo se il secondo dato è il minimo
if ($b < $min)
   {
   $min = $b;
   }
// controllo se il terzo dato è il massimo
if ($c > $max)
   {
   $max = $c;
    }
// controllo se il terzo dato è il minimo
if ($c < $min)
   {
   $min = $c;
   }

echo "minimo = ".$min."<br>";
echo "massimo = ".$max."<br>";

echo "
</body>
<html>";

?>
```

Codice PHP

by Prof. Rio Chierego

Interazione HTML e PHP:
calcolatrice con uso di check button



Layout browser

Interazione HTML e PHP:
calcolatrice con uso di check button

```
<form name="esempio2" action="esempio2.php" method="POST" target="_blank" >
 <fieldset>
   <legend>  Operazione da effettuare  </legend>
    <input type="checkbox" name="primo" value="CB-valore1" tabindex="1" checked="checked" > <b>+</b> Somma<br>
    <input type="checkbox" name="secondo" value="CB-valore2" tabindex="2"> <b>-</b> Differenza <br>
    <input type="checkbox" name="terzo" value="CB-valore3" tabindex="3" > <b>*</b> Prodotto <br>
    <input type="checkbox" name="quarto" value="CB-valore4" tabindex="4"> <b>:</b> Quoziente <br>
 </fieldset>
 <fieldset>
    <legend>  Operandi  </legend>
      <label>primo operando: <input type="text" name="a" maxlength="20" size="10" value="0" tabindex="5" ><br>
      </label>
      <label>secondo operando: <input type="text" name="b" maxlength="20" size="10" value="0" tabindex="6" ><br>
      </label>
  </fieldset>
   <fieldset>
    <legend>  Area bottoni  </legend>
     <input type="submit" value="Invia" tabindex="7">
     <input type="reset"  value="Cancella" tabindex="8">
  </fieldset>
</form>
```

Codice HTML

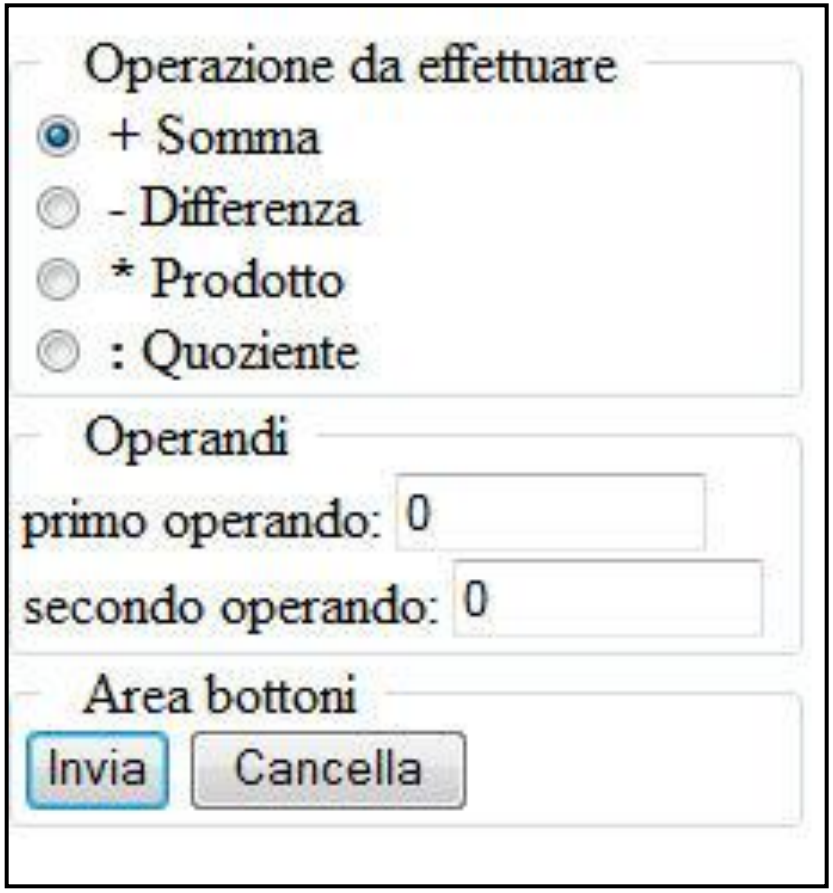
by Prof. Rio Chierego

Interazione HTML e PHP:
calcolatrice con uso di check button

```
<?php
$a = $_POST["a"];
$b = $_POST["b"];

if ( (is_numeric($a) == TRUE) && (is_numeric($b) == TRUE) )
  {
  if (isset($_POST["primo"]))
    {
    $ris = $a + $b;   /// somma
    echo  $ris."<br>";
    }
  if (isset($_POST[“secondo"]))
    {
    $ris = $a - $b;   /// differenza
    echo  $ris."<br>";
    }
  if (isset($_POST[“terzo"]))
    {
    $ris = $a * $b;   /// prodotto
    echo  $ris."<br>";
    }
  if (isset($_POST[“quarto"]))
    {
    if ($b != 0)
      {
      $ris = $a / $b;   // quoziente
      echo  $ris."<br>";
      }
     else
      {
      echo “ERRORE: divisore nullo!<br>";
      }
    }
  }
else
  {
  echo "<br>Errore di input: operandi non numerici";
  }
?>
```

Codice PHP

Interazione HTML e PHP:
calcolatrice con uso di radio button



Layout browser

Interazione HTML e PHP:
calcolatrice con uso di radio button

```
<form name="esempio3" action="esempio3.php" method="POST" target="_blank" >
 <fieldset>
   <legend>  Operazione da effettuare  </legend>
     <input type="radio" name="comune" value="RB-valore1" tabindex="1" checked="checked" ><b>+</b>Somma<br>
     <input type="radio" name="comune" value="RB-valore2" tabindex="2"> <b>-</b>Differenza<br>
     <input type="radio" name="comune" value="RB-valore3" tabindex="3" > <b>*</b> Prodotto<br>
     <input type="radio" name="comune" value="RB-valore4" tabindex="4"> <b>:</b> Quoziente<br>
 </fieldset>
 <fieldset>
    <legend>  Operandi  </legend>
      <label>primo operando: <input type="text" name="a" maxlength="20" size="10" value="0" tabindex="5" ><br>
      </label>
      <label>secondo operando: <input type="text" name="b" maxlength="20" size="10" value="0" tabindex="6" ><br>
      </label>
  </fieldset>
   <fieldset>
    <legend>  Area bottoni  </legend>
     <input type="submit" value="Invia" tabindex="7">
     <input type="reset"  value="Cancella" tabindex="8">
  </fieldset>
</form>
```

Codice HTML

Interazione HTML e PHP:
calcolatrice con uso di radio button

```
<?php
$a = $_POST["a"];
$b = $_POST["b"];

if ( (is_numeric($a) == TRUE) && (is_numeric($b) == TRUE) )
  {
  if (strcmp($_POST["comune"], "RB-valore1") == 0)
    {
    $ris = $a + $b; // somma
    echo "La somma dei due operandi e' : ".  $ris."<br>";
    }
  else if (strcmp($_POST["comune"], "RB-valore2") == 0)
    {
    $ris = $a - $b; / / differenza
    echo "La differenza dei due operandi e' : ". $ris."<br>";
     }
  else if (strcmp($_POST["comune"], "RB-valore3") == 0)
    {
    $ris = $a * $b;  // prodotto
    echo "Il prodotto dei due operandi e' : ". $ris."<br>";
    }
else if (strcmp($_POST["comune"], "RB-valore4") == 0)
    {
    if ($b != 0)
       {
       $ris = $a / $b; // quoziente
       echo "Il quoziente dei due operandi e' : ". $ris."<br>";
        }
     else
       {
      echo "ERRORE: divisore nullo!br>";
       }
    }
 else
    {
    echo "Impossibile che tu sia qui!!!!!<br>";
    }
else
  {
  echo "<br>Errore di input: operandi non numerici";
  }

?>
```

Codice PHP

by Prof. Rio Chierego