

# CORSO DI WEB DESIGN I FORM



**VENETO  
FORMAZIONE**  
*innovazione continua*



Organismo  
di Formazione  
accreditato  
dalla Regione  
del Veneto

## 9

## I FORM

## I Form

I form sono dei moduli che permettono all'utente di interagire con il sito web iscrivendosi ad esempio a mailing list, newsletter o lasciando il proprio messaggio su un forum o su un sistema di commenti di un determinato sito.

**L'invio dei dati 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, inserire un'immagine;
- **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 una pagina scritta in php, oppure in qualche altro linguaggio simile.

In questa lezione verrà trattata solo la prima parte del form, la seconda verrà trattata in un secondo momento.

La sintassi di base per inserire un form nella pagina web è la seguente:

```
<form name="prova_form" action="dati_form.php" method="post">  
...  
</form>
```

Come si può notare il primo tag da inserire per la creazione di un modulo è

"<form ...>" che deve avere sempre almeno tre attributi:

- **name:** che serve per indicare il nome del form
- **action:** che indica la pagina alla quale verranno inviati i dati e che li processerà.
- **method:** che indica il metodo di invio dei dati e che può essere di due tipi: GET e POST che ora si vedranno nel dettaglio.



## IL METODO GET

Con il metodo GET i dati vengono inviati alla pagina indicata con l'attributo "action". Nell'URL della pagina indicata potremo allora vedere tutti i parametri nella barra degli indirizzi secondo questa forma:

```
dati_form.php?nome=Mario&cognome=Rossi&datilnviati=12345
```

Come si può notare dopo la pagina "dati\_form-php" compare un punto di domanda di seguito al quale vengono mostrati in successione tutti i dati inviati attraverso il form. Nonostante l'esempio indicato il metodo "GET" è fortemente sconsigliato per l'invio di dati sensibili proprio perché sono visibili da chiunque attraverso l'URL.

Alcuni server hanno inoltre delle limitazioni per quel che riguarda il metodo GET e non consentono di inviare form con valori superiori a 255 caratteri complessivi.

## IL METODO POST

Con il metodo POST invece l'invio dei dati avviene in due step distinti: prima viene contattata la pagina sul server che deve processare i dati (quella indicata con l'attributo "action"), successivamente vengono inviati i dati.

Per questo motivo i parametri non compaiono nella barra degli indirizzi. Questo metodo è preferibile se non si desidera che i parametri siano mostrati all'utente. In questo caso non ci sono limiti sulla lunghezza dei caratteri.

### Creazione di un modulo di registrazione: il tag input

Nella creazione di un form, è fondamentale il tag "input" che crea la casella che permette di inserire i dati nel modulo. All'interno del tag "input" poi, si deve obbligatoriamente inserire l'attributo "type" che permette di determinare il tipo di dati che si andranno ad inserire nel modulo.



I valori che si possono indicare all'interno dell'attributo "type" sono:

- **text:** permette di inserire qualsiasi tipo di carattere;
- **password:** utile per inserire le password facendo sì che al momento dell'inserimento compaiano solo dei pallini nascondendo quello che si sta realmente scrivendo;
- **checkbox:** permette di inserire delle opzioni multiple di scelta per l'utente, dove si potranno spuntare quelle di interesse;
- **radio:** permette di inserire delle opzioni tra le quali l'utente può sceglierne una soltanto;
- **file:** permette l'invio di un file allegato ai dati;
- **submit:** crea il pulsante per l'invio dei dati alla pagina indicata con l'attributo "action"
- **reset:** crea un pulsante che permette di cancellare tutti i dati inseriti nel modulo;
- **hidden:** permette l'invio di un dato particolare senza che questo venga visualizzato dall'utente.



Vediamone un esempio considerando un form di registrazione utenti:

```
<body>
  <form name="prova_form" action="dati_form.php" method="post">
    <p>
      <label>Nome</label>
      <input type="text" name="nome">
    </p>
    <p>
      <label>Cognome</label>
      <input type="text" name="cognome">
    </p>
    <p>
      <label>Indirizzo</label>
      <input type="text" name="indirizzo">
    </p>
    <p>
      <label>Provincia</label>
      <input type="text" name="provincia">
    </p>
    <p>
      <label>Sesso</label>
      Maschile<input type="radio" name="sesso"
value="maschile">
      Femminile<input type="radio" name="sesso"
value="femminile">
    </p>
    <label>Hobby</label><br>
    <input type="checkbox" name="informatica"
value="informatica" checked>Informatica<br>
    <input type="checkbox" name="viaggi" value="viaggi"
checked>Viaggi<br>
    <input type="checkbox" name="sport" value="sport"
checked>Sport<br>
    <input type="checkbox" name="altro" value="altro"
checked>Altro<br>
    <p>
      <label>Password</label>
      <input type="password" name="password" maxlength="10">
    </p>
    <input name="invia_dati" type="submit" value="Invia dati">
  </form>
</body>
```

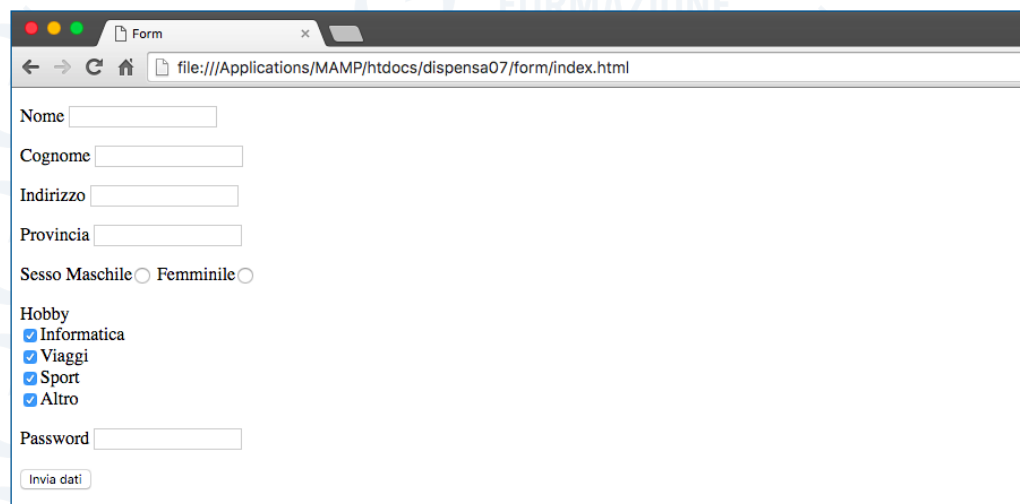


Si può notare nelle righe di codice descritte che sono stati inseriti alcuni attributi all'interno del tag "<input>".

- **name:** questo attributo indica un nome univoco per il campo. Esso diventa fondamentale quando si ha a che fare con linguaggi tipo php;
- **value:** particolarmente utile nell'uso del checkbox e radio. Permette di stabilire il valore del campo che sarà inviato alla pagina che processerà i dati. Serve inoltre per inserire il testo all'interno dei pulsanti (type="submit") che saranno visualizzati nella pagina;
- **checked:** questo attributo viene utilizzato per gli input di tipo "radio" e "checkbox" e permette di preselezionare una casella o un pulsante radio;
- **maxlength:** permette di stabilire il numero massimo di caratteri che si potranno inserire dentro alla casella.

Inoltre è stato inserito anche il tag "<label>" che permette di indicare come etichetta il nome del campo.

Il risultato nel browser sarà il seguente:



The screenshot shows a web browser window with the title 'Form'. The address bar displays the file path: file:///Applications/MAMP/htdocs/dispensa07/form/index.html. The form contains the following elements:

- Nome:
- Cognome:
- Indirizzo:
- Provincia:
- Sesso: ☐ Maschile ☐ Femminile
- Hobby: ☒ Informatica, ☒ Viaggi, ☒ Sport, ☒ Altro
- Password:
- Invia dati:

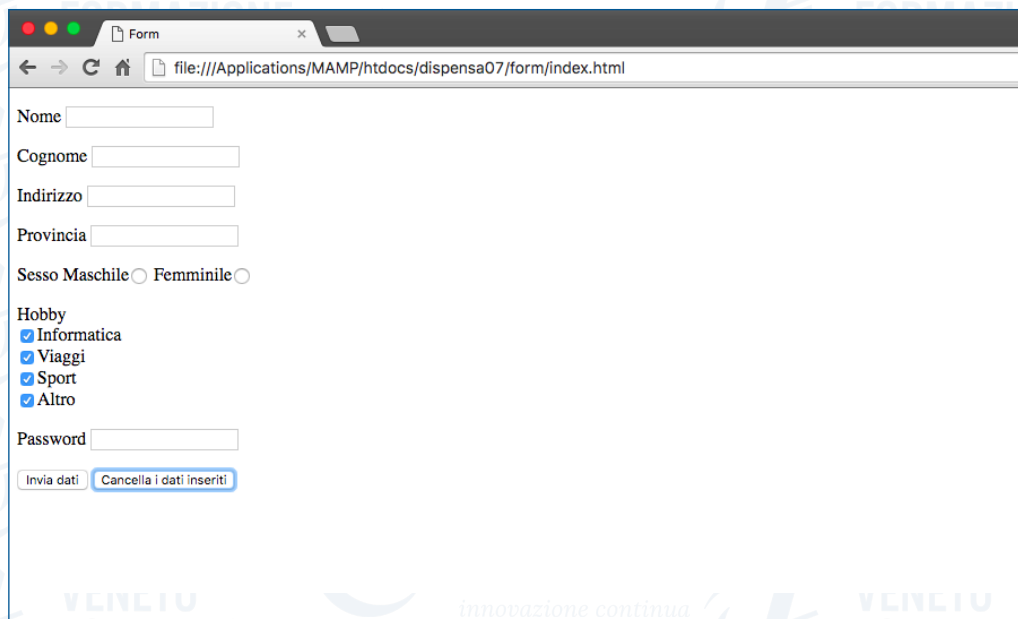
Figura 1 - Risultato del codice utilizzato per il modulo.



Proviamo ora ad inserire, dopo il pulsante “invia dati”, un pulsante che permetta di azzerare tutti i dati inseriti nel form, modificando il file “index.html” in questo modo.

```
<body>
  <form name="prova_form" "action="dati_form.php"
  method="post">
    ...
    <input name="invia_dati" type="submit" value="Invia dati">
    <input type="reset" value="Cancella i dati inseriti">
  </form>
</body>
```

Il browser ora visualizzerà il nuovo pulsante:



The screenshot shows a web browser window titled "Form" with the address bar displaying "file:///Applications/MAMP/htdocs/dispensa07/form/index.html". The form contains the following elements:

- Nome
- Cognome
- Indirizzo
- Provincia
- Sesso ☐ Maschile ☐ Femminile
- Hobby
  - ☒ Informatica
  - ☒ Viaggi
  - ☒ Sport
  - ☒ Altro
- Password
- Buttons:

Figura 2 - visualizzazione del pulsante nel browser.





Per dare la possibilità all'utente di inviare un file insieme ai dati, possiamo scrivere il codice che segue:

```
<body>
  <form name="prova_form" action="dati_form.php" method="post"
  enctype="multipart/form-data">
    ...
    <p>
      <label>Inserisci un'immagine</label>
      <input type="file" name="immagine">
    </p>
    ...
  </form>
</body>
```

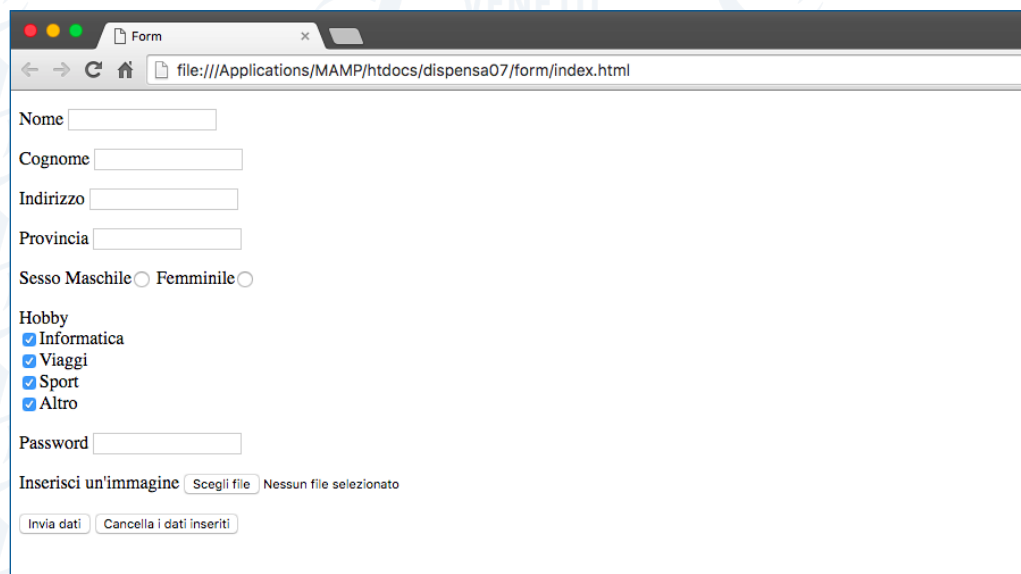
## NOTA

Se si vuole dare la possibilità di inviare un documento si deve inserire l'attributo `enctype="multipart/form-data"` all'interno del tag `<form>`





Nel browser si vedrà il seguente risultato:



The screenshot shows a web browser window with the address bar displaying `file:///Applications/MAMP/htdocs/dispensa07/form/index.html`. The form contains the following fields and controls:

- Nome:
- Cognome:
- Indirizzo:
- Provincia:
- Sesso: ☐ Maschile ☐ Femminile
- Hobby:
  - ☒ Informatica
  - ☒ Viaggi
  - ☒ Sport
  - ☒ Altro
- Password:
- Inserisci un'immagine:  Nessun file selezionato
- 

Figura 3 - Nel form sarà ora possibile inviare un documento.

## IL TAG TEXTAREA

Il tag “textarea” viene utilizzato quando si ha la necessità di inserire un campo con molto testo. Viene utilizzato, ad esempio, per dare la possibilità all’utente di inviare un messaggio in un modulo contatti o per indicare le regolamentazioni sulla privacy che l’utente deve accettare per iscriversi ad un determinato servizio. La sintassi di base è la seguente:

```
<textarea>
...
</textarea>
```



Aggiungiamo del nostro form il seguente codice:

```
<body>
<form name="prova_form" action="dati_form.php" method="post"
enctype="multipart/form-data">
  ...
  <p>
    <label>Informativa sulla privacy</label><br>
    <textarea name="informativa" cols="40" rows="6"
readonly>
In conformità con la vigente normativa, la informiamo che
ai sensi del Decreto Legislativo 30 giugno 2003 n.196 (e
successive modificazioni) i dati da Lei forniti saranno oggetto
di trattamento, sempre e comunque, avendo riguardo agli
obblighi della normativa sopra indicata.
  </textarea><br>
  <input type="checkbox" name="privacy" value="privacy">
    Accetto l'informativa sulla privacy
  </p>
  ...
</form>
</body>
```

Come si può notare al tag “<textarea>” sono stati aggiunti tre attributi oltre all’attributo name:

- **cols:** permette di specificare la larghezza della textarea. Il numero indicato rappresenta le colonne di cui il campo dovrà essere composto.
- **rows:** specifica il numero di righe (e quindi l’altezza) che dovrà contenere la textarea
- **readonly:** permette di rendere il campo disponibile solo per la lettura infatti all’utente sarà impossibile modificare il testo inserito. Questo attributo è utilizzabile anche per tutti i tag “<input>” visti finora.

Da segnalare l’utilizzo della proprietà CSS3 “**resize**” applicata al tag <textarea> che risulta ridimensionabile di default sia in altezza che in larghezza.



Di seguito elenchiamo gli attributi che servono a gestire il resize per gli elementi delle pagine web:

- **none:** l'elemento non è ridimensionabile (consigliato con l'utilizzo del tag `<textarea>`);
- **horizontal:** l'elemento può essere ridimensionato solo orizzontalmente;
- **vertical:** l'elemento può essere ridimensionato solo verticalmente;
- **both:** l'elemento può essere ridimensionato sia verticalmente che orizzontalmente;

Ecco un esempio:

```
/*impedisco il resize delle textarea*/  
  
textarea { resize: none; }
```

Il risultato che verrà visualizzato nella pagina è il seguente:

Nome

Cognome

Indirizzo

Provincia

Sesso Maschile ☐ Femminile ☐

Hobby

- ☒ Informatica
- ☒ Viaggi
- ☒ Sport
- ☒ Altro

Password

Inserisci un'immagine  Nessun file selezionato

**Informativa sulla privacy**

In conformità con la vigente normativa, la informiamo che ai sensi del Decreto Legislativo 30 giugno 2003 n.196 (e successive modificazioni) i dati da Lei forniti saranno oggetto di trattamento, sempre e comunque, avendo riguardo agli obblighi della normativa

☐ Accetto l'informativa sulla privacy

Figura 4 - Inserimento del tag “`<textarea>`”.



## IL TAG SELECT

Il tag “select” permette di creare un menù a tendina che presenta diverse opzioni di scelta. Ciascuna di queste deve essere racchiusa all’interno del tag “option” e il valore deve essere specificato attraverso l’attributo “value”.

Vediamone un esempio inserendo il seguente codice nel file “index.html”:

```
<form name="prova_form" action="dati_form.php" method="post">
  <p>
    <label>In quale città vorresti andare in vacanza?</label>
    <select name="città">
      <option value="">--Scegli--</option>
      <option value="roma">Roma</option>
      <option value="parigi">Parigi</option>
      <option value="nizza">Nizza</option>
      <option value="londra">Londra</option>
      <option value="berlino">Berlino</option>
      <option value="monaco">Monaco</option>
      <option value="barcellona">Barcellona</option>
      <option value="madrid">Madrid</option>
    </select>
  </p>
  <input name="invia_dati" type="submit" value="Invia dati">
</form>
```

Il risultato sarà il seguente:

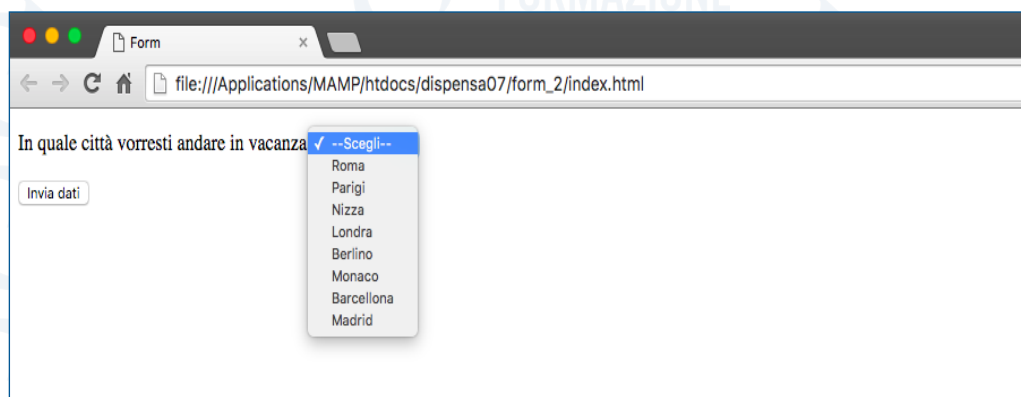


Figura 5 - esempio di una select nel browser.



Attraverso l'attributo "selected", inserito all'interno del tag "option", è possibile preimpostare una delle opzioni disponibili come prima scelta visualizzata nella pagina.

```
<form name="prova_form" action="dati_form.php" method="post">
  <p>
    <label>In quale città vorresti andare in vacanza?</label>
    <select name="città">
      <option value="">--Scegli--</option>
      <option value="roma">Roma</option>
      <option value="parigi" selected="selected">
        Parigi
      </option>
      <option value="nizza">Nizza</option>
      <option value="londra">Londra</option>
      <option value="berlino">Berlino</option>
      <option value="monaco">Monaco</option>
      <option value="barcellona">Barcellona</option>
      <option value="madrid">Madrid</option>
    </select>
  </p>
  <input name="invia_dati" type="submit" value="Invia dati">
</form>
```

Ora il browser mostrerà Parigi come prima scelta.

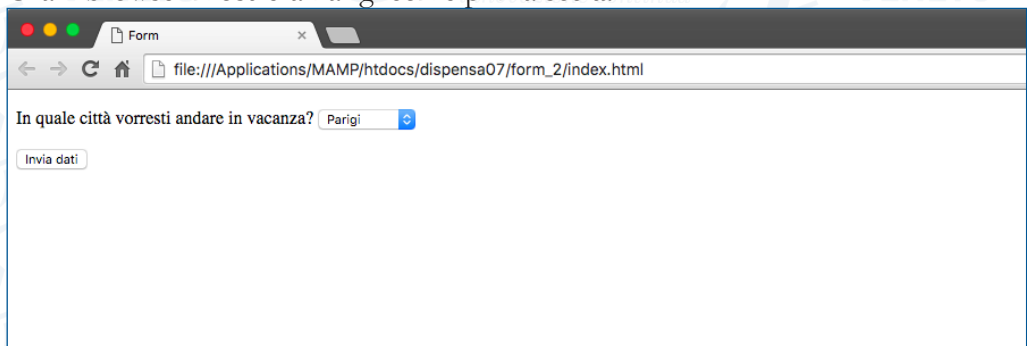


Figura 6 - Risultato dell'attributo "selected".



Inoltre, con l'attributo "multiple" possiamo visualizzare tutte le opzioni una dopo l'altra dando all'utente la possibilità di vederle tutte fin da subito e con l'attributo "size" specificare quante voci possono essere immediatamente visualizzate.

```
<form name="prova_form" action="dati_form.php" method="post">
  <p>
    <label>In quale città vorresti andare in vacanza?</
label><br>
    <select multiple="multiple" size="8">
      <option value="roma">Roma</option>
      <option value="parigi">Parigi</option>
      <option value="nizza">Nizza</option>
      <option value="londra">Londra</option>
      <option value="berlino">Berlino</option>
      <option value="monaco">Monaco</option>
      <option value="barcellona">Barcellona</option>
      <option value="madrid">Madrid</option>
    </select>
  </p>
  <input name="invia_dati" type="submit" value="Invia dati">
</form>
```

Il risultato infatti sarà quello visualizzato nella figura sottostante:

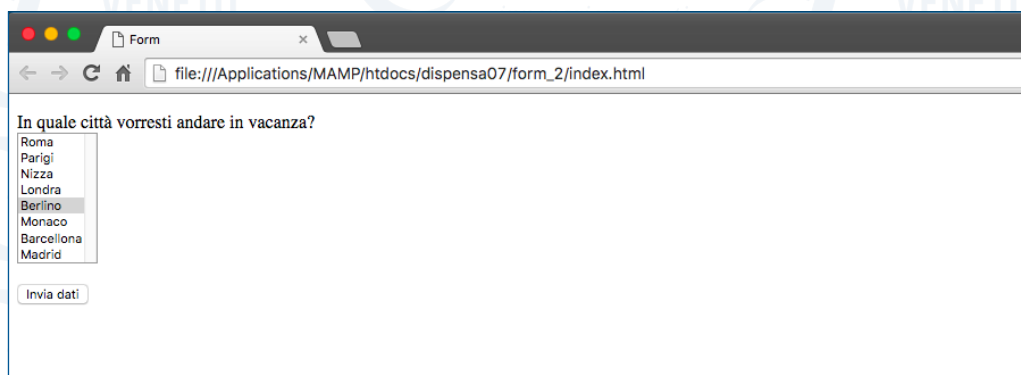


Figura 7 - Risultato visibile sul browser.





## IL TAG <OPTGROUP>

Siccome i menù di scelta tendono a diventare particolarmente lunghi, nell'HTML 4 è stato introdotto il tag "<optgroup>" che consente di suddividere le varie possibilità di scelta in gruppi tramite l'utilizzo di apposite etichette. Ecco l'esempio:

```
<form name="prova_form" action="dati_form.php" method="post">
  <p>
    <label>In quale città vorresti andare in vacanza?</label><br>
    <select>
      <option value="">--Scegli--</option>
      <optgroup label="Italia">
        <option value="roma">Roma</option>
        <option value="venezia">Venezia</option>
      </optgroup>
      <optgroup label="Francia">
        <option value="parigi">Parigi</option>
        <option value="nizza">Nizza</option>
      </optgroup>
      <optgroup label="Inghilterra">
        <option value="londra">Londra</option>
      </optgroup>
      <optgroup label="Germania">
        <option value="berlino">Berlino</option>
        <option value="monaco">Monaco</option>
      </optgroup>
      <optgroup label="Spagna">
        <option value="barcellona">Barcellona</option>
        <option value="madrid">Madrid</option>
      </optgroup>
    </select>
  </p>
  <input name="invia_dati" type="submit" value="Invia dati">
</form>
```





E il risultato visualizzato nel browser:

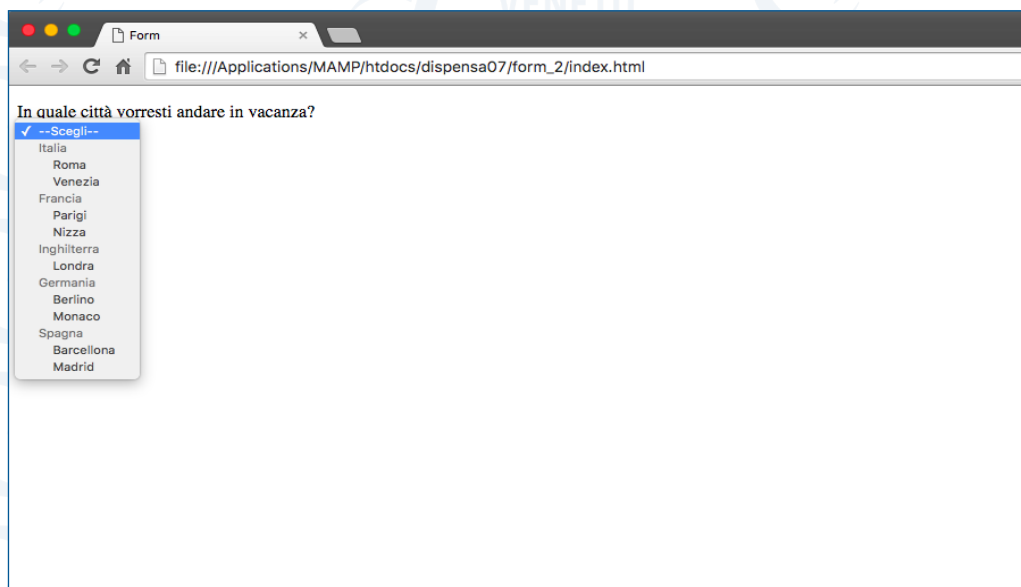


Figura 8 - Risultato nel browser dell'elemento "<optgroup>".

Con l'avvento di HTML 5 sono state aggiunte nuove funzionalità ai form che in precedenza si potevano ottenere soltanto attraverso l'utilizzo di altri linguaggi.

Infatti, in passato, la compilazione dei form avveniva senza che ci fosse una verifica sulla correttezza dei dati inseriti dall'utente costringendo gli sviluppatori ad inserire dei controlli scritti con linguaggi lato server come php.

Ad esempio quando l'utente andava a compilare un campo dove veniva richiesto un indirizzo e-mail, i dati venivano trasmessi ad una ulteriore pagina che effettuava il controllo sulla sintassi per accertare che si trattasse di un indirizzo e-mail valido. Con HTML 5 questo tipo di controllo avviene nel momento stesso dell'inserimento dei dati nel form rendendo superflue le verifiche nelle pagine successive.

Vediamo ora in dettaglio le funzionalità introdotte da HTML 5.



### IL TAG <INPUT TYPE="TEL">

È possibile utilizzare l'elemento input con type=tel per creare un campo adatto all'inserimento di numeri di telefono. Questo tipo di input non impone un particolare formato. Ciò è intenzionale. In pratica, i campi di numero di telefono sono campi liberi, perché, a livello intenzionale, i numeri possono essere scritti in diversi modi. Questo input può risultare utile soprattutto per i dispositivi mobili che possono presentare direttamente la tastiera che permette di inserire i numeri anziché quella alfanumerica che costringerebbe l'utente a passare manualmente a quella numerica. Un esempio del tag input con type=tel è il seguente:

```
<form name="prova_form" action="#" method="post">
  <p>
    <label>Inserisci il numero di telefono</label>
    <input type="tel" name="tel">
  </p>
  <input name="invia_dati" type="submit" value="Invia dati">
</form>
```

### IL TAG <INPUT TYPE="SEARCH">

È possibile utilizzare l'elemento input con type=search per creare un campo di ricerca. Nella maggior parte dei browser non c'è alcuna differenza tra un campo di tipo text e un campo search, ma in Safari e su Chrome se scriviamo qualcosa nel campo compare una piccola X sulla destra che, se cliccata, lo svuota. Su Mac OS X, inoltre visivamente l'input ha i bordi arrotondati. Un esempio del tag input con type=search è il seguente:

```
<form name="prova_form" action="#" method="post">
  <p>
    <label>Inserisci il testo da cercare</label>
    <input type="search" name="cerca">
  </p>
  <input name="invia_dati" type="submit" value="Invia">
</form>
```



## IL TAG <INPUT TYPE="URL">

L'elemento input con type=url permette di creare un campo destinato all'inserimento di un indirizzo web.

Il tipo url, se specificato, dovrebbe rappresentare l'inserimento di un URL assoluto, ovvero nel formato http://www.sito.com/etc.... Nel caso in cui il valore inserito non sia valido, il browser restituisce un messaggio che invita ad inserire un indirizzo web corretto. I dispositivi mobili possono presentare tastiere personalizzate per facilitare l'inserimento. Ad esempio iPhone modifica la sua tastiera eliminando la barra spaziatrice e mettendo il punto, lo slash e l'estensione “.com”. Un esempio del tag input con type=url è il seguente:

```
<form name="prova_form" action="#" method="post">
  <p>
    <label>Inserisci il tuo sito web</label>
    <input type="url" name="url">
  </p>
  <input name="invia_dati" type="submit" value="Invia">
</form>
```

## IL TAG <INPUT TYPE="EMAIL">

L'elemento input con type=email viene usato per creare un campo per inserire un indirizzo e-mail. L'input con tipo email, se specificato, dovrebbe rappresentare l'inserimento di indirizzi e-mail. Una fondamentale condizione di validità, dunque, sarà rappresentata dalla presenza del simbolo @. Nel caso in cui il valore inserito non sia valido il browser mostra un messaggio che invita ad inserire un indirizzo email valido. I dispositivi mobili possono presentare, anche in questo caso, tastiere ad hoc. iPhone, ad esempio modifica la sua tastiera mostrando la chiocciola e il punto. Un esempio del tag input con type=email è il seguente:

```
<form name="prova_form" action="#" method="post">
  <p>
    <label>Inserisci la tua E-mail</label>
    <input type="email" name="email">
  </p>
  <input name="invia_dati" type="submit" value="Invia">
</form>
```



## I TAG INPUT CHE PERMETTONO LA GESTIONE DELLE DATE

HTML5 mette a disposizione nuovi tipi di input concepiti per facilitare il compito dei programmatori nell'inserimento di date da parte degli utenti.

Dal momento che ci sono diversi tipi di date di cui potremmo aver bisogno, ecco che sono stati implementati nella specifica diverse tipologie.

**In particolare:**

- **datetime-local:** gestisce sia la data che l'ora (senza fuso orario);
- **date:** gestisce le date;
- **month:** gestisce i mesi;
- **week:** gestisce le settimane;
- **time:** gestisce l'ora.

**Per tutti questi input abbiamo degli attributi specifici che sono:**

- **min:** che rappresenta il minimo valore accettato;
- **max:** che rappresenta il massimo valore accettato;
- **step:** che rappresenta la granulosità dei valori accettati.

È importante sottolineare che al momento questi tipi di input sono supportati pienamente solamente da Opera. Vediamoli nel dettaglio:

### DATE TIME

Serve per permettere l'inserimento di date e ore in un solo colpo. Visivamente nei browser che lo supportano abbiamo la generazione di un datepicker in cui abbiamo la possibilità di selezionare un giorno e con l'opzione di mettere anche l'ora, come nell'immagine qui sotto. Eccone un esempio:

```
<form name="prova_form" action="dati_form.php" method="post">
  <p>
    <label>Inserisci la data e l'ora corrente</label>
    <input type="datetime" name="data">
  </p>
  <input name="invia_dati" type="submit" value="Invia dati">
</form>
```



Gli attributi in questo caso devono essere inseriti nel seguente modo:

- **min:** il valore di questo attributo deve essere una data e ora con il fuso orario valida. Ad esempio inserendo il valore `min="2013-05-05T10:00Z"` non si potranno selezionare date minori al 05/05/2013 e con un orario inferiore alle 10:00
- **max:** il valore di questo attributo deve essere una data e ora con il fuso orario valida e deve essere maggiore del valore dell'attributo `min` se specificato. Ad esempio inserendo il valore `max="2013-05-05T10:00Z"` non si potranno selezionare date maggiori al 05/05/2013 e con un orario maggiore delle 10:00
- **step:** il valore di questo attributo deve essere un intero e rappresenta i secondi. Il valore di default è di 60 secondi. Ad esempio inserendo l'attributo e il valore `step="300"` i minuti incrementeranno sempre di cinque.

## DATETIME-LOCAL

È del tutto simile a `datetime`, con l'unica differenza che non vengono passate informazioni sul fuso orario.

E il risultato visto nel browser:

Inserisci la data e l'ora corrente

luglio 2016

lun	mar	mer	gio	ven	sab	dom
27	28	29	30	1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Figura 9 - Risultato dell'uso del tag `input` con `type=datetime`.





Gli attributi in questo caso devono essere inseriti nel seguente modo:

- **min:** il valore di questo attributo deve essere una data e ora valida. Ad esempio inserendo il valore `min="2013-05-05T10:00"` non si potranno selezionare date minori al 05/05/2013 e con un orario inferiore alle 10:00
- **max:** il valore di questo attributo deve essere una data e ora valida e deve essere maggiore del valore dell'attributo `min` se specificato. Ad esempio inserendo il valore `max="2013-05-05T10:00"` non si potranno selezionare date maggiori al 05/05/2013 e con un orario maggiore delle 10:00
- **step:** il valore di questo attributo deve essere un intero e rappresenta i secondi. Il valore di default è di 60 secondi. Ad esempio inserendo l'attributo e il valore `step="350"` l'ora si incrementerà sempre di cinque minuti

## DATE

Serve per inserire una data. Nei browser che lo supportano si ottiene un calendario in cui abbiamo la possibilità di selezionare un giorno. Il risultato dell'utilizzo del tag `input` con `type=date` è del tutto simile a quello precedente soltanto che per quest'ultimo è previsto l'inserimento della sola data e non dell'ora.

Gli attributi in questo caso devono essere inseriti nel seguente modo:

- **min:** il valore di questo attributo deve essere una data valida. Ad esempio inserendo il valore `min="2013-05-05"` non si potranno selezionare date minori al 05/05/2013
- **max:** il valore di questo attributo deve essere una data valida e deve essere maggiore del valore dell'attributo `min` se specificato. Ad esempio inserendo il valore `max="2013-05-05"` non si potranno selezionare date maggiori al 05/05/2013
- **step:** il valore di questo attributo deve essere espresso in giorni. Il valore di default è di 1 giorno. Ad esempio inserendo il valore `step="3"` si potranno selezionare le date con intervalli di tre giorni

## MONTH

Serve per permettere di selezionare un mese dell'anno. Nei browser che lo supportano si ottiene un calendario in cui abbiamo la possibilità di selezionare un mese.



Il risultato dell'utilizzo del tag input con type=month è del tutto simile a quelli visti in precedenza soltanto che per quest'ultimo è previsto l'inserimento del solo mese.

Gli attributi in questo caso devono essere inseriti nel seguente modo:

- **min:** il valore di questo attributo deve essere un mese valido. Ad esempio inserendo il valore min="2013-02" non si potranno selezionare mesi inferiori a febbraio 2013
- **max:** il valore di questo attributo deve essere un mese valido e deve essere maggiore del valore dell'attributo min se specificato. Ad esempio inserendo il valore max="2013-02" non si potranno selezionare mesi maggiori di febbraio 2013
- **step:** il valore di questo attributo deve essere espresso in mesi. Il valore di default è di 1 mese. Ad esempio inserendo il valore step="3" si potranno selezionare i mesi ad intervalli di tre

## WEEK

Viene usato per la selezione di una determinata settimana dell'anno (composta da anno – numero di settimana). Nei browser che lo supportano si ottiene un calendario in cui abbiamo la possibilità di selezionare una sola settimana. Il risultato dell'utilizzo del tag input con type=week è del tutto simile a quelli visti in precedenza soltanto che per quest'ultimo è previsto l'inserimento di una sola settimana selezionabile dal calendario.

Gli attributi in questo caso devono essere inseriti nel seguente modo:

- **min:** il valore di questo attributo deve essere una settimana valida. Ad esempio inserendo il valore min="2013-02W7" non si potranno selezionare settimane inferiori alla diciassettesima settimana del 2013
- **max:** il valore di questo attributo deve essere una settimana valida e deve essere maggiore del valore dell'attributo min se specificato. Ad esempio inserendo il valore max="2013-02W7" non si potranno selezionare settimane maggiori alla diciassettesima settimana del 2013
- **step:** il valore di questo attributo deve essere espresso in settimane. Il valore di default è di 1 settimana. Ad esempio inserendo il valore step="3" si potranno selezionare settimane ad intervalli di tre.





## TIME

Serve per selezionare e inserire una determinata ora del giorno attraverso due frecce che compaiono a destra del campo.

Gli attributi in questo caso devono essere inseriti nel seguente modo:

- **min:** il valore di questo attributo deve essere un orario valido. Ad esempio inserendo il valore `min="00:02"` non si potrà selezionare un orario inferiore alle 00:02
- **max:** il valore di questo attributo deve essere un orario valido e deve essere maggiore del valore dell'attributo `min` se specificato. Ad esempio inserendo il valore `max="00:02"` non si potrà selezionare un orario maggiore alle 00:02
- **step:** il valore di questo attributo deve essere un intero e rappresenta i secondi. Il valore di default è di 60 secondi. Ad esempio inserendo l'attributo e il valore `step="300"` i minuti incrementeranno sempre di cinque.

## IL TAG <INPUT TYPE="NUMBER">

È possibile utilizzare l'elemento `input` con `type=number` per creare un campo destinato all'inserimento di un numero. I dispositivi mobili possono presentare tastiere personalizzate per facilitarne l'inserimento.

Anche per questo tipo di `input` troviamo i seguenti attributi:

- **min:** specifica il minimo valore permesso. La sintassi è semplice: `min="1"` permette solo l'inserimento di numeri positivi.
- **max:** specifica il massimo valore permesso. `max="10"` permette solo l'inserimento di numeri inferiori o uguali a 10. Il valore di questo attributo deve essere maggiore del valore dell'attributo `min` se specificato.
- **step:** L'attributo `step` limita i valori permessi. Il valore di `step` se specificato deve essere un numero (anche non intero) maggiore di zero oppure la stringa `"any"` (che equivale a non inserire l'attributo). La sintassi è anche in questo caso molto semplice: `step=3` influenza i valori permettendo valori come -3, 0, 3, 6 ma non -1 o 2.



Un esempio del tag input con type=number è il seguente:

```
<form name="prova_form" action="dati_form.php"
method="post">
  <p>
    <label>Inserisci un numero</label>
    <input type="number" name="numero" min="3" max="30"
step="3">
  </p>
  <input name="invia_dati" type="submit" value="Invia dati">
</form>
```

### IL TAG <INPUT TYPE="RANGE">

Molto simile semanticamente all'input type=number, questo nuovo tipo di input permette agli utenti di inserire un numero tramite uno slider. Per capire meglio vediamo subito un esempio:

```
<form name="prova_form" action="dati_form.php"
method="post">
  <p>
    <label>Inserisci un numero</label>
    <input type="range" name="numero" min="3" max="30"
step="3">
  </p>
  <input name="invia_dati" type="submit" value="Invia dati">
</form>
```

Il risultato ottenuto con il browser Opera è il seguente:

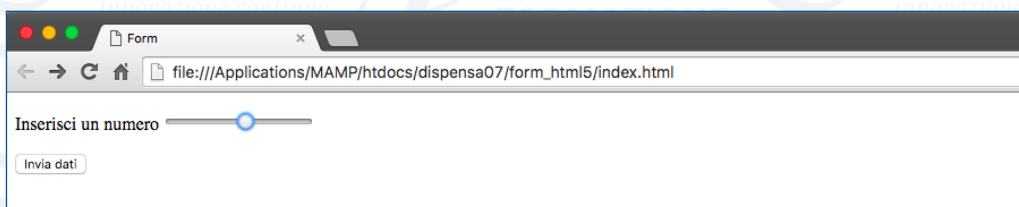


Figura 10 - Risultato ottenuto con il browser Opera.



## IL TAG <INPUT TYPE="COLOR">

L'elemento input con type=color dovrebbe creare un color picker, quel tipo particolare di widget utile per la selezione di un colore a partire da una palette di colori. Una volta selezionato il colore, il campo passa alla nostra pagina di ricezione un colore RGB esadecimale composto da 6 cifre. Nella maggior parte dei browser non c'è alcuna differenza tra un campo di tipo text e un campo color, ma su Opera abbiamo un comportamento particolare: infatti visivamente abbiamo una select particolare con i colori ma se clicchiamo abbiamo una scelta dei colori base. Inoltre se clicchiamo su "Altro" richiamiamo il color picker di sistema. Per capire meglio vediamone subito un esempio:

```
<form name="prova_form" action="dati_form.php"
method="post">
  <p>
    <label>Inserisci il tuo colore preferito</label>
    <input type="color" name="colore">
  </p>
  <input name="invia_dati" type="submit" value="Invia dati">
</form>
```

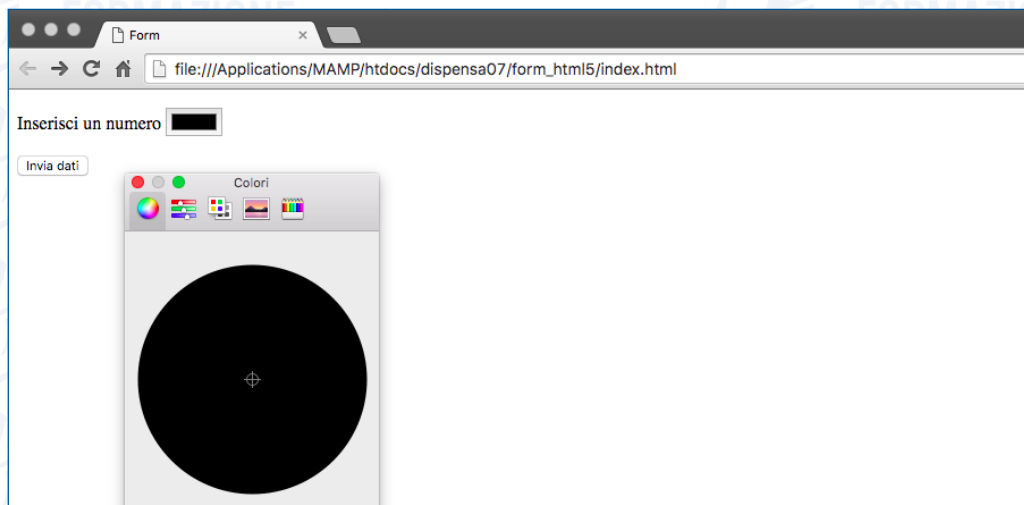


Figura 11 - Risultato ottenuto con il browser Opera dopo aver cliccato sul pulsante "altro..."



## IL TAG BUTTON

Il tag `<button>` offre la possibilità di creare dei pulsanti con un aspetto particolarmente ricco.

Il tag `<button>`, a differenza del tag `<input>`, dà la possibilità di inserire il testo del bottone tra l'apertura e la chiusura del tag medesimo. Questo ci consente di specificare anche del codice HTML all'interno del tag.

Il contenuto del tag `<button>` è costituito principalmente da testo, ma è possibile inserire anche immagini (cosa che rende questo tag di gran lunga più personalizzabile del tag `<input>`).

La sintassi è la seguente:

```
<button type="button">Testo del pulsante</button>
```

Ed ora vediamo un esempio inserendo il seguente codice nel documento HTML:

```
<form name="prova_form" action="#" method="post">  
  <button name="bottone generico" type="button">  
    testo del bottone generico  
  </button>  
  <br>  
  <button name="accetta" type="submit">  
    ACCETTA  
  </button>  
</form>
```

Ecco il risultato nel browser:

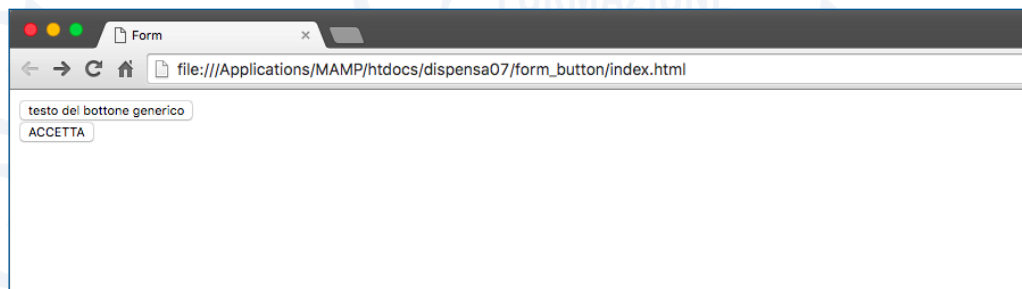


Figura 12 - Risultato ottenuto con il browser.



È consigliabile specificare sempre l'attributo `type`, per evitare che il pulsante si comporti in maniera imprevedibile: in caso di una mancata dichiarazione del tipo, Internet Explorer assegnerà di default `type="button"`, mentre tutti gli altri browser assegneranno `type="submit"`.

Di seguito gli attributi più importanti utilizzabili all'interno del tag `"button"`:

- **disabled:** con valore `"disable"` che specifica se il pulsante è disattivato;
- **name:** utile a specificare un nome per il pulsante;
- **type:** con i valori `"button"`, `"reset"`, `"submit"` utili a specificare la tipologia di appartenenza del pulsante
- **value:** per specificare un valore che verrà visualizzato come tooltip

## IL TAG LABEL

Il tag `<label>` definisce un testo da associare ad un campo di un form HTML; si usa in particolar modo con i campi di tipo radio e checkbox. Consente di selezionare un'opzione anche cliccando sul suo testo descrittivo, non solo sul pulsante di opzione vero e proprio grazie.

Supporta l'attributo **for** che riporta lo stesso valore dell'id del campo, che, allo scopo, si rende obbligatorio.

La sintassi è la seguente, eccone un esempio:

```
<input type="radio" name="sesso" id="uomo" value="maschio" />
<label for="uomo">Uomo</label>

<input type="radio" name="sesso" id="donna" value="femmina" />
<label for="donna">Donna</label>
```



## ALTRI ATTRIBUTI PER I FORM

### AUTOFOCUS

L'attributo autofocus serve a impostare il focus su uno specifico elemento del form appena la pagina è caricata. Un esempio è quello della homepage di Google: appena viene caricata possiamo immediatamente scrivere sul campo di ricerca senza necessariamente cliccarci dentro. È importante precisare che solo un elemento per pagina può avere questo attributo e che va utilizzato con un certo criterio. È per questo motivo che autofocus dovrebbe essere usato solo nelle pagine che contengono solamente, o principalmente, form come per esempio pagine di login o di ricerca.

### PLACEHOLDER

Il valore dell'attributo placeholder è visualizzato all'interno di un input, o di una textarea, fin quando il campo è vuoto e l'utente non ci clicca all'interno. Semanticamente l'attributo placeholder dovrebbe essere valorizzato con valori accettabili dal form, dovrebbe, in altre parole, contenere un esempio di ciò che l'utente andrà a scrivere nel campo anche se spesso il placeholder viene utilizzato erroneamente al posto del tag label descrivendo quindi cosa si dovrebbe inserire.

### REQUIRED

required è un attributo che serve a rendere obbligatorio l'inserimento dei dati nel campo da parte dell'utente. La condizione viene valutata al submit del form.

### AUTOCOMPLETE

Spesso i browser riempiono i campi da inserire in maniera automatica. Nella maggior parte dei casi è un comportamento comodo ma in altri fastidioso. Si pensi per esempio ai campi password o ai campi del codice della banca: probabilmente non vogliamo che il browser li completi in automatico. In questi casi l'attributo autocomplete ci permette di decidere quali campi il browser può completare automaticamente e quali no.

In particolare i valori che accetta sono:

- **on:** indica che il valore non è particolarmente sensibile e che il browser può compilarlo in maniera automatica;
- **off:** indica che il valore è particolarmente sensibile o con un tempo di scadenza





- (il codice di attivazione di un servizio, per esempio) e che quindi l'utente deve inserirlo manualmente ogni volta che lo compila;
- **nessun valore:** indica in questo caso di usare il valore di default scelto dal browser (normalmente on).

## MULTIPLE

Questo attributo serve nel caso in cui l'utente abbia bisogno di inserire più valori per lo stesso input, per esempio se gli vengono chiesti gli indirizzi e-mail di amici a cui inviare un invito.

## Raggruppare i moduli con i tag `<fieldset>` e `<legend>`

Per la loro natura di “raccoltori di informazioni”, i moduli tendono ad ingigantirsi e diventare lunghissimi. Per questo, con l'HTML 4, sono stati introdotti dei tag per fare un po' d'ordine all'interno dei form.

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

Vediamo un esempio formattando innanzitutto il form con il codice CSS che segue:

```
form {width:300px;}  
input[type="reset"] {margin-top:20px;}
```

Come si può notare è stato dato al form una larghezza di 300 pixel. Inoltre, la particolare sintassi utilizzata nella riga 7, permette di dare un margine superiore solo a tutti i tag “input” di tipo “reset”.

### NOTA

La sintassi “input[type=“image”]” potrebbe non funzionare con le versioni più vecchie di Internet Explorer.





Nel file “index.html” inseriamo invece il seguente codice:

```
<form name="prova_form" action="dati_form.php" method="post"
enctype="multipart/form-data">

  <fieldset>
    <legend>Dati personali</legend>
    <p>
      <label>Nome</label>
      <input type="text" name="nome">
    </p>
    <p>
      <label>Cognome</label>
      <input type="text" name="cognome">
    </p>
    <p>
      <label>Indirizzo</label>
      <input type="text" name="indirizzo">
    </p>
    <p>
      <label>Provincia</label>
      <input type="text" name="provincia">
    </p>
    <p>
      <label>Sesso</label>
      Maschile<input type="radio" name="sesso"
value="maschile">
      Femminile<input type="radio" name="sesso"
value="femminile">
    </p>
  </fieldset>

  <fieldset>
    <legend>Dati per il login</legend>
    <label>Hobby</label><br>
    <input type="checkbox" name="informatica"
value="informatica" checked>Informatica<br>
    <input type="checkbox" name="viaggi" value="viaggi"
checked>Viaggi<br>
    <input type="checkbox" name="sport" value="sport"
checked>Sport<br>
    <input type="checkbox" name="altro" value="altro"
checked>Altro<br>
```



```
<p>  
  <label>Password</label>  
  <input type="password" name="password"  
    maxLength="10">  
</p>  
</fieldset>  
  
<input name="invia_dati" type="submit" value="Invia  
dati">  
<input type="reset" value="Cancella i dati inseriti">  
</form>
```

Il risultato che sarà visualizzato nel browser sarà il seguente:

Form

file:///Applications/MAMP/htdocs/dispensa07/form/index.html

Dati personali

Nome

Cognome

Indirizzo

Provincia

Sesso Maschile ☐ Femminile ☐

Dati per il login

Hobby

☒ Informatica

☒ Viaggi

☒ Sport

☒ Altro

Password

Invia dati Cancella i dati inseriti

Figura 13 - Con i tag “<fieldset>” e “<legend>” l’aspetto del form risulta più gradevole.

## NOTA

Per rendere il form più accessibile è stato aggiunto nel tag “<label>” l’attributo “for” che, associato all’id del tag “<input>” permette di selezionare un’opzione anche cliccando sul suo testo descrittivo, non solo sul pulsante di opzione vero e proprio.



## ALCUNI ESEMPI DI FORM

Nelle prossime pagine vedremo come realizzare dei semplici form, vedremo come partendo da una struttura molto semplice scritta in HTML potremmo ottenere dei risultati anche molto distanti a livello di aspetto grafico impostando differentermente i fogli di stile CSS.

Di seguito verrà fornito il codice HTML di base sulla quale lavoreremo:

```
<!doctype html>
<html>
<head>
  <meta charset="UTF-8">
  <meta name="author" content="il tuo nome" />
  <meta name="copyright" content="il tuo nome" />
  <meta name="keywords" content="Esercizi web design" />
  <meta name="description" content="" />
  <link rel="stylesheet" href="css/style.css" />
  <title>Form di esempio</title>
</head>
<body>
  <h1>Flat Design Form</h1>
  <form name="form_esempio" action="dati_form.php"
method="post" enctype="multipart/form-data">
    <h2>Iscriviti</h2>
    <input type="text" name="username"
placeholder="USERNAME">
    <input type="tel" name="telefono"
placeholder="TELEFONO">
    <input type="email" name="email"
placeholder="MAIL">
    <input type="password" name="password"
placeholder="PASSWORD">
    <input name="registrati" type="submit"
value="REGISTRATI">
  </form>
</body>
</html>
```



Questo è il risultato del browser senza applicare nessuna formattazione da parte del CSS:

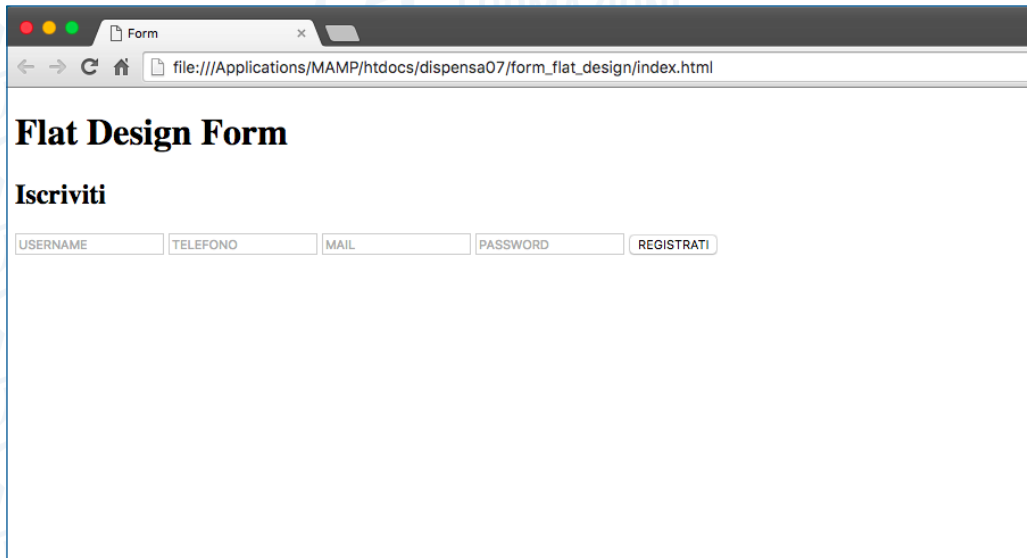


Figura 14 - Risultato ottenuto con il browser, struttura in html senza formattazione.

## FORM IN STILE: FLAT DESIGN

Ora andremo a scrivere all'interno del foglio di stile le seguenti regole:

```
* {margin:0;
padding:0;
font-family: Helvetica, Arial, sans-serif;
font-weight:100;}

body {background-image:url(..img/texture.jpg);
background-position:center;
background-repeat:no-repeat;
background-size:cover;}

form {background-color:hsla(0,0%,0%,0.30);
width:480px;
margin:0 auto;
padding:1px;
}
```



```
h1 {text-align:center;
padding-top:80px;
padding-bottom:40px;
color:white;
font-size:40px;}
```

```
h2 {text-align:center;
padding-top:60px;
padding-bottom:20px;
color:white;}
```

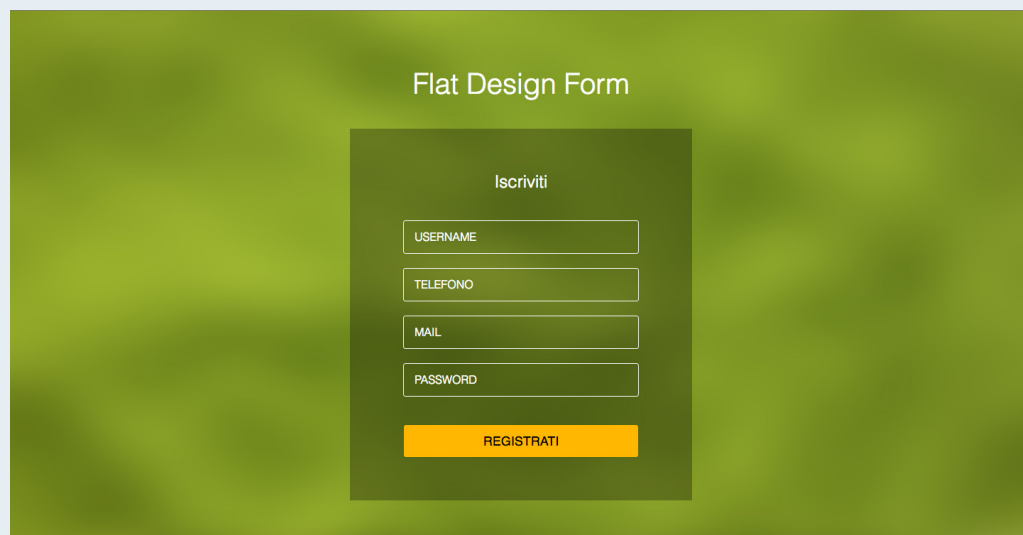
```
input[type="password"], input[type="text"],
input[type="tel"], input[type="email"],
input[type="submit"] {
display:block;
width:300px;
height:25px;
background-color:transparent;
margin:20px auto 20px auto;
padding:10px 15px 10px 15px;
border:1px solid white;
border-radius:3px;
color:white;
font-size:16px;
line-height:25px;
}
```

```
/*REGOLE PER CAMBIARE IL COLORE DEL TESTO VISIBILE GRAZIE A
PLACEHOLDER, DEFINITO ALL'INTERNO DEL TAG INPUT*/
::-webkit-input-placeholder{color:white;} /*webkit browser*/
::-moz-placeholder{color:white;} /*mozilla firefox 19+*/
:-moz-placeholder{color:white;} /*mozilla firefox 4/18*/
:-ms--input-placeholder{color:white;} /*internet explorer
10+*/
```



```
input[type="submit"] {  
    width:330px;  
    height:45px;  
    background-color:hsla(43,100%,50%,1.00);  
    border:hsla(43,100%,50%,1.00);  
    margin-top:40px;  
    margin-bottom:60px;  
    font-size:18px;  
    color:black;  
    font-weight:400;  
}  
  
input[type="submit"]:hover {  
    background-color:white;  
    border:white;  
}
```

Ecco il risultato della formattazione in stile flat design:



The image shows a web form titled "Flat Design Form" with a registration section. The form is centered on a light blue background. The registration section is a dark gray box with the title "Iscriviti" (Register). It contains four input fields: "USERNAME", "TELEFONO" (Phone), "MAIL", and "PASSWORD". Below these fields is a yellow button labeled "REGISTRATI" (Register).

Figura 15 - Risultato della formattazione del form in stile flat-design





## FORM IN STILE: MATERIAL DESIGN

Continuiamo con l'esercizio scrivendo all'interno di un altro foglio di stile le seguenti regole:

```
* {margin:0;  
padding:0;  
font-family: Helvetica, Arial, sans-serif;  
font-weight:100;}
```

```
body {background-color:white;}
```

```
form {background-color:#eee;  
width:480px;  
margin:0 auto;  
padding:1px;  
overflow:hidden;  
}
```

```
h1 {text-align:center;  
background-color:#eee;  
padding-top:15px;  
padding-bottom:15px;  
color:#999;  
font-size:20px;  
margin-bottom:60px;}
```

```
h2 {text-align:center;  
padding:30px 20px 30px 20px;  
color:white;  
background-color:#5f50e1;  
box-shadow:0 3px 5px #8D8D8D;  
margin-bottom:50px;}
```

```
input[type="password"], input[type="text"],  
input[type="tel"], input[type="email"],  
input[type="submit"] {  
display:block;  
width:330px;  
height:25px;  
background-color:transparent;
```





```
margin:20px auto 20px auto;
padding:10px 0px 10px 0px;
border:none;
border-bottom:2px solid #999;
color:#999;
font-size:16px;
line-height:25px;
}

::-webkit-input-placeholder{color:#999;} /*webkit browser*/
::-moz-placeholder{color:#999;} /*mozilla firefox 19+*/
:-moz-placeholder{color:#999;} /*mozilla firefox 4/18*/
:-ms--input-placeholder{color:#999;} /*internet explorer
10+*/

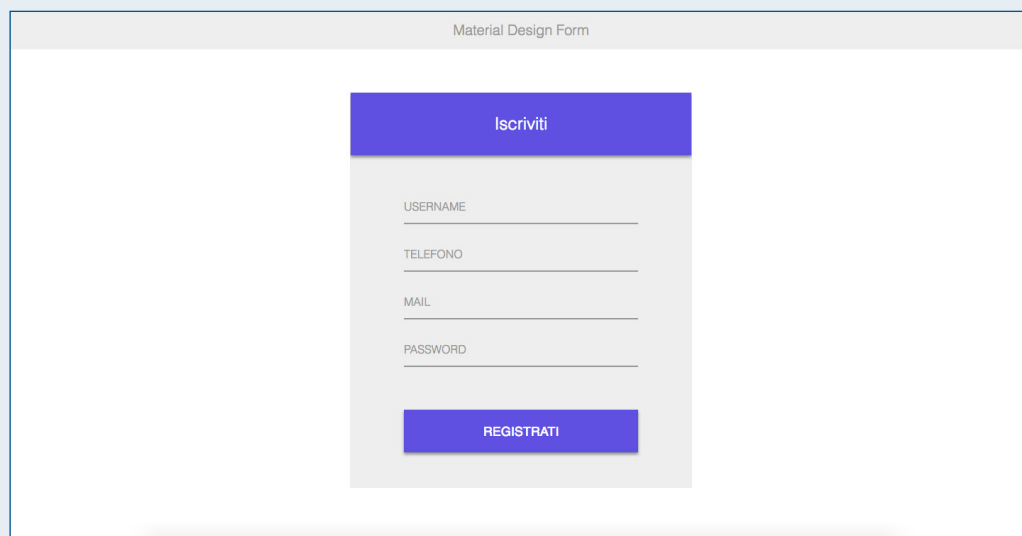
input[type="submit"] {
width:330px;
height:60px;
background-color:#5f50e1;
border:#5f50e1;
margin-top:60px;
margin-bottom:60px;
font-size:18px;
color:white;
font-weight:400;
box-shadow:0 3px 5px #8D8D8D;
}

input[type="submit"]:hover {
box-shadow:0 3px 8px #222;
}

input[type="password"]:focus, input[type="text"]:focus,
input[type="tel"]:focus, input[type="email"]:focus {border-
bottom:2px solid #5f50e1; color:#5f50e1;}
```



Osserviamo quindi il risultato della formattazione in stile material design:



The image shows a Material Design form titled "Material Design Form". It features a blue header bar with the text "Iscriviti". Below the header, there are four input fields labeled "USERNAME", "TELEFONO", "MAIL", and "PASSWORD". At the bottom of the form, there is a blue button labeled "REGISTRATI". The form is centered on a light gray background.

*Figura 16 - Risultato della formattazione del form in stile material-design*

