

# Logica e stile dei form html

**Marco Deseri**

Come realizzare un form html accessibile e esteticamente personalizzabile, attraverso l'uso dei css.

## Realizzare il form html

In questo tutorial impareremo a realizzare un form accessibile per l'immissione di dati in un sito: in particolare, realizzeremo un modulo ipotetico per l'immissione di notizie all'interno di un sistema di gestione dei contenuti.

Inseriremo questi campi nel modulo:

- il periodo di pubblicazione della notizia (che verrà immesso usando campi select e un campo input text);
- il titolo della notizia (campo di testo);
- la categoria della notizia (radio button);
- il testo (textarea).

I campi sono stati scelti a fini didattici, con l'intenzione di mostrare quasi tutti gli elementi che possono comparire nei form. Non è pensato come soluzione reale per l'inserimento delle notizie.

## Il codice html del form

Nel form di esempio conosceremo gli elementi html per creare moduli accessibili. Prima di cominciare, potete visualizzare l'esempio (`codice_esempio_css_html.html`) e salvarlo.

Non basta un buon html per avere moduli accessibili, è importante anche il linguaggio. Per cui ci sforzeremo di:

- Essere chiari, evitando gerghi e tecnicismi;
- Eliminare tutte le possibili ambiguità;

Per aumentare l'usabilità del form, faremo bene a segnalare agli utenti le sezioni logiche in cui è organizzato. Nel nostro caso, distingueremo tra una sezione dedicata all'*inserimento delle date* per la pubblicazione (da quando a quando la notizia sarà visibile) e una dedicata al *corpo della notizia* vera e propria.

Html mette a disposizione uno strumento per marcare la struttura dei form: il tag *fieldset*, che i browser rendono in un modo speciale.

In concreto, la struttura logica del nostro form sarà la seguente:

```
<form name="newseditor" action="azionedelform.php" method="post">
  <fieldset>
    <legend>Periodo di pubblicazione:</legend>
    <!-- qui inseriremo gli elementi di input della data -->
  </fieldset>

  <fieldset>
    <legend>Notizia da pubblicare: </legend>
    <!-- qui inseriremo gli elementi di input della notizia -->
  </fieldset>

  <fieldset>
    <input type="submit" value="invia" title="invia"/>
  </fieldset>
</form>
```

Come avrete notato, oltre al tag *fieldset*, abbiamo usato il tag *legend*. Mentre il primo serve a marcare logicamente le sezioni del modulo, il secondo consente di dar loro un titolo. Ora che abbiamo definito la struttura, passiamo a popolare il form con gli input veri e propri.

Per quello che riguarda la scelta della data, useremo dei *select*, sia per i giorni che per i mesi. Useremo, invece, un campo di testo per far scegliere l'anno. Ecco il codice:

```
<fieldset>
  <legend>La notizia sarà pubblicata dal:</legend>

  <label for="giornodal">giorno:</label>
  <select id="giornodal" name="giornodal">
    <option value="1">1</option>
    <option value="2">2</option>
    <option value="3">3</option>
    [...]
  </select>

  <label for="mesedal">mese:</label>
  <select id="mesedal" name="mesedal">
    <option value="1">Gennaio</option>
    <option value="2">Febbraio</option>
    <option value="3">Marzo</option>
    [...]
  </select>

  <label for="annodal">anno:</label>
  <input type="text" id="annodal" name="annodal" size="4" maxlength="4"/>

</fieldset>
```

In queste righe, abbiamo un elemento nuovo: l'elemento *label*. Questo tag consente di associare logicamente un'etichetta ad un campo di input. Per farlo è necessario che il valore dell'attributo "for" di *label* coincida con il valore dell'attributo "id" di *input*: in questo modo si stabilisce un legame logico tra i due elementi, e questo legame può essere sfruttato dai browser in fase di rendering. Non bisogna dimenticare di inserire l'attributo "name" nell'elemento di *input*: è infatti "name", e non "id", a dare il nome alla variabile che viene usata dallo script che processa il form.

Torniamo alla pratica. Per inserire la parte del modulo relativa alla data di fine pubblicazione, possiamo copiare il pezzo di codice appena visto, modificando opportunamente le variabili. In ogni caso, è possibile scaricare il codice completo dell'esempio.

Passiamo all'inserimento del corpo della notizia.

```
<label for="titolo">Titolo:</label>
<input type="text" id="titolo" name="titolo" size="60"/><br/>

<label for="categoria">Categoria:</label><br/>
<input type="radio" id="categoria" name="categoria" /> Cronaca <br/>
<input type="radio" id="categoria" name="categoria" /> Politica <br/>
<input type="radio" id="categoria" name="categoria" /> Sport <br/>

<label for="testo">Testo</label>
<textarea id="testo" name="testo" rows="10" cols="55">Inserisci qui il corpo della notizia</textarea>
```

Fatto! Abbiamo ormai tutti gli elementi, dobbiamo solo dare uno stile al nostro modulo.

## I css per cambiare l'aspetto di un form

Anche i form possono essere personalizzati grazie ai css: principalmente, si tratta di agire sui bordi, sullo sfondo e sui font.

Per farlo, dobbiamo assegnare delle classi agli elementi nel file html: a queste classi, attraverso le regole espresse nel css, assegneremo delle regole di formattazione.

Dovremo andare nel file html e aggiungere, agli elementi cui vogliamo assegnare una classe, una stringa del tipo `class="nomedellaclassa"`. Nel dettaglio:

- Assegnamo a tutti gli elementi *input* di tipo "text" e a tutti i *select* la classe "testo". Basta aggiungere all'interno dei rispettivi tag questo: `class="testo"`.
- Al fieldset che contiene gli input relativi alla data di pubblicazione, assegnamo `class="pubblicazione"`.
- Al fieldset che contiene gli input per l'inserimento dei contenuti della notizia, invece, assegnamo la `class="notizia"`.
-

Assegnamo la classe "submit" al fieldset in cui è contenuto il pulsante per il submit della form.

Oltre a definire gli stili in base alle classi, è possibile farlo a livello di elementi. Questo significa ridefinire, all'interno del documento, il modo in cui il browser effettua il rendering di un determinato tag.

Useremo questa tecnica per definire la larghezza della form e il font con cui vogliamo che venga visualizzata.

La sintassi da usare per ridefinire un elemento (in pratica un tag) nei css è semplice: basta scrivere il nome dell'elemento e subito dopo inserire tra parentesi graffe le regole.

```
form
{
    font-family:Verdana,Arial,Helvetica,sans-serif;
    width:650px;
}
```

Le proprietà che abbiamo settato influenzeranno tutti gli elementi compresi tra i tag form e /form. Per approfondire il significato della prima istruzione, potete consultare il nostro tutorial su font e css (/tutorial/css/1/). La seconda istruzione, invece, setta a 650 pixel la larghezza dell'elemento form (e quindi 650 pixel è il limite massimo di larghezza per tutti gli elementi contenuti al suo interno).

In maniera simile, definiamo le proprietà relative all'elemento legend:

```
legend
{
    font-size:100%;
    border:1px solid #000000;
    background-color:#efefef;
    color:#cc0000;
    padding:3px;
}
```

La dimensione del testo è definita in termini relativi (in percentuale): i visitatori potranno ridimensionare facilmente il testo, usando l'apposita funzione del proprio browser.

Impostiamo il bordo: largo 1 pixel, solido (senza effetti di rientranza o sporgenza, né di tratteggio) e di colore nero. #000000 è il valore esadecimale del nero, avremmo anche potuto usare una scorciatoia e scrivere semplicemente *black*.

La proprietà background-color riguarda il colore di sfondo: un grigio chiaro (#efefef);

Il color, invece, è il colore delle scritte: #cc0000 è un rosso piuttosto scuro;

Infine, il padding è il margine interno, lo spazio tra il bordo e il contenuto dell'elemento: in questo caso è di 3 pixel.

Per definire l'aspetto degli input text useremo un'altra tecnica, combinando la selezione per classi e per elementi. La sintassi è più complessa, vediamo prima un modello di esempio:

```
elemento.classe elemento {regole}
```

Con questo selettore facciamo due distinzioni principali. Prima limitiamo l'applicazione delle regole per le *classi* di un *elemento* specifico. Poi, all'interno della nostra prima selezione, facciamo un'ulteriore distinzione: scegliamo un elemento, a questo applichiamo le regole.

In pratica scriveremo:

```
fieldset.pubblicazione input {margin-top:10px;}
```

Tradotto in linguaggio umano, questo significa che la regola verrà applicata agli elementi *input* contenuti all'interno degli elementi *fieldset* di classe "pubblicazione".

Nello specifico, abbiamo impostato un margine superiore di 10 pixel, per tutti gli elementi di input relativi all'inserimento della data.

Ora abbiamo visto i principi per lo styling di una form con i css. Potete approfondire analizzando il codice html dell'esempio (codice\_esempio\_css\_html.html) e il codice del foglio di stile css (stileform.css).

## Risorse

Sul codice html per scrivere le form:

- 

Moduli (Form) accessibili (<http://webaccessibile.org/argomenti/argomento.asp?cat=295>) su Webaccessibile.org.

- 

Better Accessible forms (<http://www.accessify.com/tutorials/better-accessible-forms.asp>) su accessify.com (in inglese).

Su come applicare gli stili css alle form:

- 

Style Web Forms Using CSS (<http://www.sitepoint.com/article/1166>) su sitepoint.com (in inglese).

- 

Moduli con stile: usare i CSS con i form ([http://pro.html.it/articoli/id\\_254/idcat\\_8/pro.html](http://pro.html.it/articoli/id_254/idcat_8/pro.html)) su pro.html.it.