

Forms

Forms

I form (o le form) sono lo strumento principale per permettere all'utente di inserire dati e di inviarli verso il server, permettendogli di effettuare elaborazioni parametriche.

Utilizziamo form in tantissimi contesti, per esempio:

- Login
- Ricerca
- Inserimento di dati anagrafici
- Inserimento di un bonifico
- Inserimento dei dati di un ordine
- ...

Forms

In HTML sono disponibili molteplici tag tramite i quali realizzare form di vario tipo.

Tutti quanti vanno però contenuti dentro il tag `<form>` che pertanto svolge il compito di contenitore generale.

Il tag form ha due attributi principali:

- **action:** identifica l'url verso il quale inviare i dati inseriti nella form.
- **method:** indica la modalità con cui trasferire i dati inseriti nella form. Può avere due valori:
 - **GET:** i dati sono accodati alla query string dell'url. E' il valore di default che viene utilizzato in caso di omissione.
 - **POST:** i dati sono inseriti nel corpo della richiesta

Forms – get e post



```
<form action="https://www.dominio.com/search">  
  <input type="text" name="q">  
</form>
```

Invierà la seguente richiesta al server:

```
https://www.dominio.com/search?q=html
```

Forms – get e post



```
<form action="https://www.dominio.com/search" method="post">  
  <input type="text" name="q">  
</form>
```

Invierà la seguente richiesta al server:

`https://www.dominio.com/search`

E il seguente parametro nel "corpo" della richiesta

`q = html`

Forms – get e post

Ci sono differenze sostanziali tra get e post. In particolare il metodo GET:

- Inserisce i dati della form nell'url e quindi nella barra degli indirizzi del browser
- I dati arrivano al server viaggiando in chiaro. L'url non è cifrato nelle connessione HTTPS.
- L'url può essere loggato da tutti i server che attraversa mentre viaggia verso il server.
- L'url può essere memorizzato come segnalibro del browser
- La lunghezza massima dell'url è limitata e dipende dal browser
- L'url non può essere utilizzato per l'invio di dati binari (per esempio immagini)
- Il protocollo HTML stabilisce che sia utilizzato per operazioni di lettura o operazioni idempotenti

Forms – get e post

Il metodo POST invece:

- Invia i dati nel corpo della richiesta, senza riportarli nell'url
- Non ha alcuna restrizione sulla quantità di dati che si possono inviare
- Può inviare anche dati binari
- Nelle connessione HTTPS i dati della form viaggiano cifrati
- Il protocollo HTML stabilisce che sia utilizzato per operazioni che comportano modifiche sul server

Form fields

All'interno del tag `<form>` possono essere utilizzati una serie di tag, i più frequenti sono:

- `input`
- `label`
- `select`
- `textarea`
- `button`

Per i dettagli di ogni tag si veda <https://htmlreference.io>

Form fields

Ogni tag ha i propri attributi. Tuttavia ce ne sono alcuni comuni alla maggior parte dei tag:

- **name:** Rappresenta il nome dell'elemento che verrà inviato al server. Deve essere univoco all'interno della form ed è obbligatorio.
- **required:** Permette di specificare un campo obbligatorio. E' un attributo senza valore.
- **readonly:** Permette di specificare un campo in sola lettura. E' un attributo senza valore.
- **disabled:** Permette di specificare un campo non abilitato. Verrà mostrato con una grafica differente. E' un attributo senza valore.
- **placeholder:** permette di specificare un testo da visualizzare quando il campo non è compilato

E' inoltre possibile utilizzare i classici attributi id e class.

Form fields – dimensioni e posizionamento

Gli elementi che compongono le form possono essere considerati elementi inline.

La larghezza viene specificata con l'attributo `size` oppure con l'attributo `width`. `Size` prevede di specificare la larghezza in numero di caratteri mentre `width` utilizza le logiche già viste.

L'altezza viene specificata con l'attributo `height` utilizzando le modalità già viste.

Il tag `form` e i tag dei singoli elementi possono essere utilizzati insieme ai tag già visti, in modo da ottenere layout complessi. Per esempio:

- Utilizzo di `table` per organizzare tag input
- Utilizzo di floating div per creare form responsive

Form in Javascript

E' possibile intercettare il submit della form per gestirlo tramite javascript, ottenendo:

1. Form gestite completamente da js senza comunicazione con il server
2. Comunicazione con il server gestita da js (differentemente da quanto previsto di default)

Per attivare questa modalità è necessario:

- Omettere gli attributi action e method sul tag form
- Gestire l'evento onsubmit sul tag form, nel quale chiamare la funziona js di gestione. L'evento deve sempre restituire false per inibire il submit standard
- (Eventualmente) Gestire gli eventi sui con controlli della form, per esempio onClick sui pulsanti o onchange sulle checkbox