Web design elements: come validare i form con jQuery?

Nelle scorse lezioni abbiamo visto come strutturare un form, e le scelte progettuali che ci sono alla base. Un aspetto importante che abbiamo trascurato è la gestione degli errori: hai mai compilato un lungo form per scoprire solo dopo averlo inviato di aver sbagliato qualcosa? Vediamo allora come JavaScript (e in particolare jQuery) può esserci d'aiuto!

Ci siamo passati tutti: compili un form, interpreti in un modo o nell'altro il captcha di turno, accetti la licenza e invio. Cinque secondi. L'username è già occupato. Ricompilare di nuovo il form. Captcha. Licenza. Invio. Cinque secondi. Le password non coincidono. Ricompila. Captcha. Licenza. Invio.

Il problema in queste iterazioni è il ritardo che si crea tra un invio e l'altro, dovuto all'interazione tra client (invio del form) e server (validazione e restituzione del risultato). L'esperienza utente ne soffre parecchio (anche la nostra pazienza), ma come risolvere? Una possibile soluzione è aggiungere alla validazione lato server quella lato client, tramite JavaScript. Nota che ho scritto aggiungere e non sostituire: la validazione lato server è fondamentale, e non deve mai mancare!

Il principale vantaggio della validazione lato client, è che avviene **in tempo reale**, permettendo quindi una correzione immediata di eventuali errori.

Reinventare la ruota o riutilizzare?

Anche se è relativamente semplice realizzare un sistema di validazione per form in jQuery partendo da zero, esistono diversi plugin realizzati dalla comunità che già ovviano a questo problema. I vantaggi di utilizzare questi plugin sono diversi:

- Non si deve reinventare la ruota.
- Offrono funzionalità che sono ben testate in tutti i maggiori browser.
- Si ha più tempo per concentrarsi su altri problemi, specifici alla nostra applicazione.

Detto questo, se si hanno delle necessità particolari nulla vieta di creare qualcosa di più adatto. Il form che utilizzeremo è simile a quello creato nella lezione precedente, a cui abbiamo aggiunto dei campi per l'immissione di username e password.

Per la validazione del nostro form di esempio, utilizzeremo il plugin validation dal repository di jQuery. Questo è uno dei plugin più vecchi (la documentazione parla di luglio 2006) e nel tempo si è arricchito di molte funzionalità che lo rendono molto versatile.

Una volta scaricato l'archivio ed estratto il contenuto, copiamo il file jquery.validate.js nella nostra directory di lavoro. Naturalmente abbiamo bisogno anche dell'ultima versione di jQuery e di un file vuoto su cui lavorare. Colleghiamo il tutto nel nostro file XHTML:

Methods e Rules

Il plugin validation si basa su due concetti fondamentali: methods e rules. I *methods* rappresentano la logica di validazione dei nostri campi: ad esempio il metodo email verifica che il testo inserito rappresenti effettivamente un'email. I metodi forniti con il plugin sono circa una ventina: il più importante è sicuramente il metodo required che permette di specificare se un campo deve essere obbligatorio o meno. Questa è la documentazione completa corredata da esempi.

Le *rules* invece collegano i methods ai vari campi, permettendo così la validazione del form. Come al solito, un esempio vale più di mille parole. Apriamo dunque il file validateYIW.js nell'editor:

Il plugin è attivato dalla funzione .validate, che viene applicata al form selezionato tramite l'ID. All'interno della funzione inseriamo l'elemento rules che contiene tutte le regole per il nostro form, campo per campo. Ogni regola è composta dall'attributo name del campo, in cui sono specificati i metodi che si applicano e il loro valore. Analizziamoli uno per uno:

- **required**: se impostato a true, imposta il campo come obbligatorio.
- **minlength**: imposta il numero minimo di caratteri che può contenere il campo.
- **maxlength**: imposta il numero massimo di caratteri per il campo.
- **email**: se impostato a true, verifica che il testo inserito sia un'email valida.
- **equalTo**: verifica se il testo immesso è uguale a quello del campo fornito come argomento.

Come puoi vedere la validazione è già funzionante: all'atto dell'invio per ogni errore il plugin aggiunge un messaggio accanto al campo interessato. Di default questi messaggi vengono racchiusi in un elemento label a cui è associata la classe "error". Per evidenziarli un po' ho aumentato le dimensioni del testo e li ho colorati di rosso:

Avrai notato, però, che tutti i messaggi sono in Inglese. Anche se il plugin fornisce già un file con la localizzazione in Italiano, la trovo troppo generica. Meglio quindi specificare un messaggio personalizzato per ogni tipo di errore: ciò è presto fatto tramite l'utilizzo dell'elemento messages:

Seguendo la stessa struttura di rules, ho specificato un messaggio per ogni errore, in modo da guidare l'utente nella risoluzione. Il form risultante già potrebbe andar bene: vediamo come possiamo migliorarlo.

Sarebbe utile poter evidenziare in modo più evidente i campi contenenti errori. Per questo scopo possiamo utilizzare gli elementi highlight e unhighlight che permettono di specificare delle azioni da intraprendere in presenza o assenza di errori, rispettivamente. La sintassi per entrambi è la seguente:

Come puoi vedere, entrambi gli elementi richiedono una funzione, a cui passare due argomenti:

- **element**, che rappresenta l'elemento del form con errori
- **errorClass**, che rappresenta il nome della classe d'errore (di default è "error")

Possiamo quindi pensare di aggiungere la classe "error" agli elementi li che contengono i vari elementi di input, in modo da applicare un feedback grafico tramite CSS.

Dato che l'elemento li contiene l'elemento di input, utilizziamo la funzione parent di jQuery per selezionarlo, aggiungendo poi la classe d'errore, sfruttando la concatenazione. Ora non resta che applicare qualche regola CSS, così da evidenziare meglio gli errori:

Il risultato finale è più che soddisfacente.

Il plugin validation permette largo margine di miglioramento: ad esempio si potrebbe aggiungere un box con il numero di errori posizionato sopra il form, così da fornire un maggior feedback grafico.

Conclusioni e considerazioni

In questo articolo abbiamo visto come sfruttare jQuery per migliorare l'usabilità dei nostri form, a partire dal semplice XHTML+CSS. E con questo chiudiamo la serie sulla progettazione dei form. In seguito vedremo come realizzare il collegamento con un database, tramite PHP.

Voglio lasciarti con un pensiero: i form, nonostante tutti gli abbellimenti grafici e i miglioramenti tramite JavaScript, non hanno avuto una grande evoluzione. Insomma si tratta sempre di riempire un modulo campo per campo! Pensi che esista o esisterà qualche tecnologia o approccio che possa limitare il loro uso, predominante nel web dei nostri giorni?

- I form: elementi di base
- I form: struttura e aspetto
- I form: jQuery e considerazioni
- Come realizzare un modulo contatti in PHP