Come ordinare elementi con jQuery (2/2)

Nel precedente articolo abbiamo visto come il metodo sortable permetta l'ordinamento di una lista di elementi tramite il trascinamento. Ne abbiamo visto però solo l'implementazione "visiva". **Ma se vogliamo che questo ordinamento sia durevole, quindi che rimanga tale anche se ricarichiamo la pagina, dovremo fare un passo in più.**
Nel caso pratico, gli elementi li estrarremo da una tabella del database nella quale salveremo l'id, il nome dell'elemento ed il suo ordine.
La modifica dell'ordine dovrà poi essere intercettata ed inviata con una chiamata asincrona ad un file che si occuperà di aggiornare il database.
Come puoi vedere nell'esempio, se modifichi l'ordine, questo rimarrà inalterato anche ricaricando la pagina (a meno che ci sia un altro utente che sta utilizzando l'esempio) in quanto viene salvato nel database così come verrà illustrato in questo articolo.

Creare il database

Creiamo un database e chiamiamalo *sortable*.
In questo database creiamo una tabella chiamata lista con tre campi: *id, item_name e item_order*.
Puoi eseguire questa query:

Ora popoliamo questa tabella con gli elementi che saranno visualizzati:

Ed infine creiamo il file con i parametri di connessione *db_config.php*:

Creare il supporto per le operazioni lato server

Bene, a questo punto possiamo iniziare a sviluppare la classe che si occuperà di gestire le operazioni lato server, la chiameremo *sortableSupport.php*.

Iniziamo con l'includere il file con i parametri di connessione al database e a dichiarare la classe e la proprietà *conn* che rappresenta la risorsa di connessione.

Scriviamo ora il metodo per la connessione al database sul quale c'è poco da dire:

E dichiariamo questo metodo nel costruttore di classe visto che qualunque operazione che questa classe dovrà svolgere necessita della connessione al database.

Ora possiamo sviluppare il metodo grazie al quale visualizzeremo la lista degli elementi e che andrà a sostituire l'elenco statico nella pagina index.php:

Come vedi vengono estratti i dati ordinati in base al campo *item_order*; in seguito andiamo a creare l'elenco in modo dinamico. L'id dell'elemento sarà: *item_valoreDellIdPrelevatoDalDatabase*. Non ci resta che istanziare la classe. A questo punto il nostro script dovrebbe essere così (non è ancora finito):

Ora apriamo il file *index.php* che abbiamo utilizzato per il precedente articolo. In questo file dovremo sostituire la lista degli elementi con il metodo *showItem()*.

Quindi questa parte di codice:

Andrà sostituito con questa:

In questo modo la nostra lista verrà inserita dinamicamente.

Individuare il cambiamento di ordine

Ora siamo ad un passaggio cruciale. Dobbiamo in qualche modo **individuare il cambiamento di ordine ed inviarlo al server in modo che sia possibile aggiornare il database**.

Iniziamo con il riprendere il nostro script jQuery che avevamo lasciato così:

A questo punto **passeremo come parametro il gestore dell'evento update** con il quale definiamo appunto le procedure da svolgere quando la lista viene modificata. Lo faremo come sempre tramite una funzione, in questo modo:

Con questa funzione dovremo **leggere l'ordine della lista ed inviarla al server con una richiesta ajax**.
Per leggere l'ordine degli elementi non dovremo fare altro che serializzare la lista. Ti ricordi dell'articolo che ho scritto sul metodo *serialize*?
Ebbene, anche *sortable* dispone del metodo *serialize* che utilizzeremo semplicemente in questo modo:

Nel precedente articolo avevo insistito sull'importanza del formato dell'id degli elementi della lista: *nomeSempreUguale_numeroSempreDiverso*
Nel concreto

Il motivo di questo formato particolare è il seguente:
Serialize si aspetta questo formato e restituirà un array il cui **nome sarà quello che c'è prima del trattino, mentre i valori saranno quello che c'è dopo il trattino**.
I valori di questo array saranno ordinati nello stesso modo nel quale è ordinata la lista.
Quindi se sposteremo l'elemento 2 prima dell'elemento 1, l'array che ne risulterà sarà così

2,1,3,4,5

O più precisamente, se teniamo conto anche delle chiavi:

item[0] -> 2
item[1] -> 1
item[2] -> 3
item[3] -> 4
item[4] -> 5

Adesso non ci resta che spedire questo array al server con una semplice richiesta ajax:

Naturalmente *order.php* dobbiamo ancora scriverlo, ma lo faremo presto.
Il nostro codice jQuery è ora pronto, anzi l'intera pagina *index.php* è pronta e si presenta così:

Come aggiornare l’ordinamento nel database?

Ora abbiamo il nostro array in viaggio per il server. Iniziamo con lo sviluppare un nuovo metodo della classe *sortableSupport* che sia in grado di aggiornare l’ordinamento nel database.

Per farlo scorreremo l’array tramite il costrutto *foreach* grazie al quale preleveremo la chiave di ciascun elemento (che rappresenterà l’ordine) e il valore che corrisponderà all’id dell’elemento. Dunque per ogni elemento avremo l’id ed il numero di ordinamento. Non dovremo fare altro che aggiornare la tabella, in questo modo:

Come vedi modifichiamo il campo item_order in corrispondenza dell’id.

A questo punto non resta che creare il semplicissimo file order.php, nel quale dovremo unicamente eseguire il metodo appena visto.

Per finire ti riporto il codice della classe sortableSupport completa:

Conclusione

L’articolo è venuto un po’ lungo ma volevo spiegare tutto nei dettagli. Come hai potuto vedere sortable fornisce tutti i supporti per poter implementare in modo semplice l’ordinamento di oggetti. E tu, hai già utilizzato una funzionalità del genere? In che occasione?