Guida allo sviluppo di plugin per wordpress: La gestione degli script

Nel precedente articolo siamo arrivati a fare in modo che il nostro plugin stampi correttamente il box della licenza alla fine di ogni post.
In questo articolo svilupperemo la parte che permetterà al box di essere chiuso con un effetto di animazione; in questo modo potremo vedere da vicino come WordPress gestisce l'utilizzo degli script.

Come includere jQuery in un plugin?

Svilupperemo il nostro script in jQuery, dunque dovremo fare in modo di rendere questa libreria disponibile. Ora, a qualcuno potrebbe venire in mente di farlo in questo modo:

Nulla di più sbagliato!
Pensate se ogni plugin che necessita di jQuery includesse la libreria nell'header. Rischieremmo di trovarcelo xx volte!

Fortunatamente WordPress mette a disposizione delle funzioni in grado di gestire l'inclusione di script.

wp_enqueue_script e wp_register_script

Grazie a *wp_register_script()* é possibile fare una mappatura degli script che ci servono e delle loro dipendenze per poi inviarli in modo sicuro nell'header tramite la funzione *wp_enqueue_script()*.
Vediamo subito un esempio. Immaginiamo di avere bisogno di includere *script-3.js* che dipende da *script-1.js* e *script-2.js*.

Dunque *wp_register_script* passa cinque parametri che sono nell'ordine:

1. Nome univoco dello script.
2. Posizione dello script.
3. Dipendenze (in forma di array).
4. Versione (tramite il quale é possibile assegnare un numero di versione allo script).
5. Footer (lo valorizzeremo con true se desideriamo che il nostro script sia incluso nel gangio *wp_footer*, altrimenti di default sarà incluso nell'header).

Dunque per includere correttamente i nostri script, procederemo in questo modo

In questo modo *script_3.js* verrà incluso nell'header dopo le sue dipendenze (*script_1.js* e *script_2.js*).

E' infine possibile utilizzare un altra forma più contratta. Infatti *wp_enqueue_script* ci permette comunque di indicare percorso e dipendenze, quindi potremmo fare anche in questo modo:

Dunque torniamo al nostro problema, ovvero includere jQuery. Basterà scaricare la libreria, salvarla nella cartella del nostro plugin, registrarla e passarla come dipendenza del nostro script.
No, non ci siamo ancora.

WordPress ha una lunga serie di **librerie ed estensioni registrate nativamente**, trovi la lista alla fine di questa pagina.

Come puoi vedere, ovviamente jQuery ne fa parte (come pure numerose delle sue user interface). Per richiamarla sarà sufficiente passare il suo handle nell'array delle dipendenze dello script:

Così facendo é come se dicessimo "*includi jQuery nell'header se non lo ha già fatto qualcun altro*". Di fatto funziona come require_once.

Le librerie e le estensioni registrate nativamente da WordPress sono automaticamente gestite. Se ci dovesse ad esempio servire ui-dialog, che dipende da jquery, ui.core, ui.widget, ui.position e altre non dovremo indicarle in questo modo

… array('jquery','jquery-ui-core','jquery-ui-widget', …..)

Ma basterà fare così

… array('jquery-ui-dialog');

E Wordpress includerà tutto quanto necessario.

Molto bene, ora che abbiamo visto nel dettaglio il funzionamento dell'inclusione degli script, torniamo al nostro plugin ed iniziamo a scrivere il codice che farà in modo che il box della licenza venga chiuso con un effetto di animazione.

Sviluppare lo script

Nota importantissima prima di iniziare:
Quando WordPress include jQuery, lo fa omettendo di mettere a disposizione il suo alias($), al suo posto dovremo utilizzare jQuery. Questo per evitare confitti con altre librerie che utilizzano $ come

alias.
Dunque un espressione del genere

Non ha nessun senso, come pure non avrà senso utilizzare $ all'interno di quella funzione.
Per ovviare a questo, é sufficiente passare $ come argomento della funzione.

Ed ecco che all'interno di quella funzione, $ avrà ancora valore di alias di jQuery.
Bene. Ora crea la cartella *js* all'interno della cartella del nostro plugin e crea il file *custom.js*, con questo contenuto

Molto semplice come puoi vedere; all'evento *click* sul link di chiusura, viene applicato il metodo *slideUp* al contenitore del nostro box licenza risalendo il DOM tramite il metodo *parent()*.

Ora torniamo al plugin e facciamo in modo che questo script venga incluso nell'header del nostro sito applicando le procedure che abbiamo visto all'inizio di questo articolo.

A questo punto il plugin é funzionante anche nell'effetto di chiusura.

Ed ecco come dovrebbe presentarsi

Conclusione

In questo articolo abbiamo visto la procedura corretta per inlcudere degli script e delle librerie js e siamo arrivati ad un primo traguardo: Il plugin funziona correttamente sul frontend utente.
A partire dal prossimo articolo di questa serie ci occuperemo della creazione del pannello amministrativo introducendo alcune opzioni.

Capitoli di questa guida

****

L'immagine principale dell'articolo è stata fornita da @Fotolia