Come sviluppare un widget per WordPress

Dopo aver lasciato passare un po' di tempo per darvi la possibilità di metabolizzare il corso sui plugins, eccomi a tornare ancora a parlare di sviluppo su piattaforma WordPress ed in particolare di **widgets**.
I widgets sono un caso particolare di plugin ed hanno lo scopo di realizzare un'interfaccia che verrà visualizzata nella sidebar. Il tutto è veramente molto semplice; unica precondizione è di possedere almeno le basi della programmazione ad oggetti. Darò inoltre per scontato che tu abbia seguito il corso sui plugin. Sei pronto?

Scenario

In questo tutorial svilupperemo un widget semplicissimo e senza nessuna pretesa, quello che ci interessa è l'aspetto della programmazione.
Grazie a questo widget potremo visualizzare nella sidebar il nostro numero di telefono, cellulare e fax. il risultato finale sarà questo.

Anatomia di un widget

Il sistema per realizzare widgets messo a punto dagli sviluppatori di WordPress è a dir poco esemplare.
La classe *WP_Widget* si occupa di gestire le tre componenti fondamentali di un widget ovvero:

- La visualizzazone sul front-end tramite il metodo *widget()*
- Il form nel quale è possibile gestire le opzioni tramite il metodo *form()*
- Ed infine la ridefinizione delle opzioni con il metodo *update()*

Un widget non è altro che un'estensione della classe *WP_Widget.*
Nel processo di sviluppo, come vedremo tra poco, saremo estremamente facilitati dalla perizia con la quale il team di WordPress ha sviluppato questa componente.
Ma adesso iniziamo????

Warm up

Nella cartella *plugin* (wp_content) crea la cartella *yiw-numbers* ed al suo interno il file *yiw.numbers.widget.php*
Come ho detto all'inizio, un widget è un caso particolare di plugin, dunque dovremo iniziare con l'inserire un'intestazione identica a quella che si utilizza per i plugin appunto.

Estendere WP_Widget

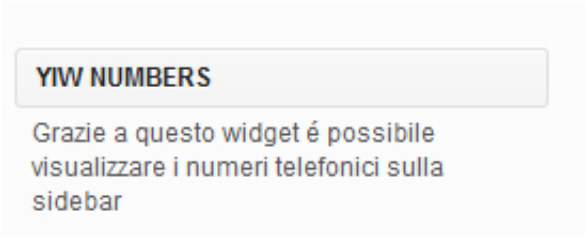
Iniziamo ora con il dichiarare un'estensione della classe *WP_Widget* (*yiw_numbers_widget*) con al suo interno i metodi che abbiamo visto precedentemente.

Il costruttore di classe

Nella maggior parte dei casi, il costruttore di classe risulterà piuttosto semplice. Basterà invocare il costruttore della classe parent passandogli tre parametri:

- Il nome univoco del widget
- Il titolo
- Un array associativo nel quale noi utilizzeremo solo la voce "description".

Il titolo e la descrizione verranno visualizzati nel pannello dei widget in questo modo



Il form

Come prima cosa dobbiamo fare passare la variabile *$instance* (che contiene una serie di informazioni sul nostro widget) come argomento del metodo *form*

Il nostro form conterrà quattro campi

- Il titolo del widget
- Il numero di telefono
- Il numero di cellulare
- il numero di fax

Iniziamo con il definire i valori di default di questi quattro campi.
Lo faremo con un array associativo che chiameremo *$defaults*

A differenza di quanto abbiamo visto per i plugin, **non dobbiamo creare dei nomi complessi per le opzioni**. Nel caso dei widget infatti le opzioni sono salvate in un'unica stringa JSON per widget ed il nome del campo sarà quanto passato al costruttore nel primo parametro.
Dunque nel nostro caso, nella tabella *wp_options* troveremo un campo *widget_yiw_numbers* con un contenuto simile a questo:
a:3:{i:2;a:0:{}i:3;a:4:{s:5:"title";s:8:"Contatti";s:5:"phone";s:0:"";s:6:"mobile";s:0:"";s:3:"fax";s:0:"";}s:12:"_multiwidget";i:1;}
L'unica cosa alla quale dovremo prestare attenzione è l'id del widget (ed il nome della classe ovviamente).

In seguito scriviamo questo

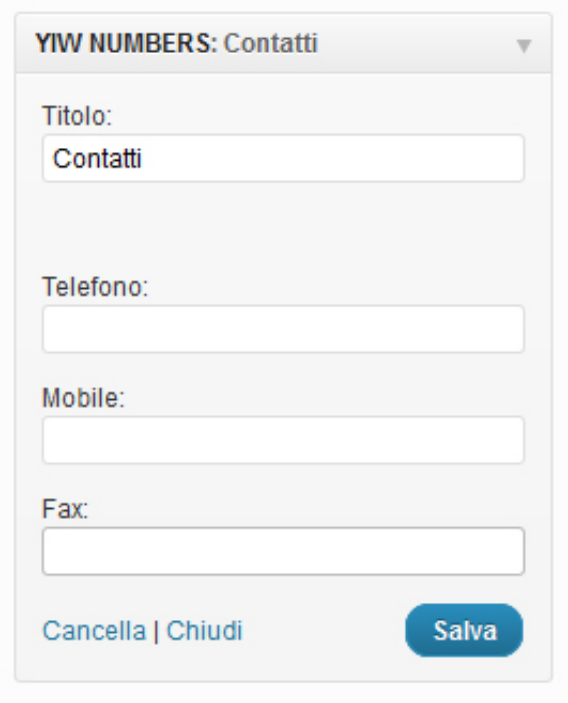
Grazie a questa riga di codice, aggiungiamo le informazioni contenute in *$defaults* a *$instance*.

Ora sviluppiamo il nostro form.
Iniziamo con il campo per il titolo:

Un aspetto molto importante da rilevare a questo punto è che gli attributi *id* e *name* dobbiamo farli gestire da WordPress tramite gli appositi metodi *get_field_id()* e *get_filed_name()*.
Il *value* del campo lo prenderemo invece da *$instance*.

Facciamo la stessa cosa con gli altri tre campi ed otteniamo infine il metodo *form* completo.

Il risultato sarà questo



Come hai potuto vedere, abbiamo dovuto definire poco di più che il markup del form. **Tutto il resto, ovvero la resa grafica, il fatto che sia draggabile, il fatto che sia ordinabile, il bottone salva, i link cancella e chiudi con relative funzionalità sono a carico della classe WP_Widget e non ce ne dobbiamo minimamente preoccupare**. Non ci dobbiamo nemmeno occupare, a differenza dei plugin, di come queste opzioni saranno salvate nel database. Non è fantastico?

Il metodo update

Con il metodo *update* ridefiniamo la variabile *$instance* con i nuovi valori assunti a seguito di una modifica delle opzioni, ovvero quando premiamo il pulsante *salva* del form.
Dovremo far passare al metodo le variabili *$new_istance* e *$old_istance*

Restituiremo quindi la variabile *$instance* nella quale sovrascriveremo i nuovi valori ottenuti con il salvataggio del form, in questo modo

Il metodo widget

Al metodo *widget* dovremo passare due argomenti: *$args* e *$instance*

Come ormai sappiamo, *$instance* contiene le informazioni del nostro widget, non da ultimo il titolo ed i numeri. *$args* invece contiene delle informazioni più generali sui widget come ad esempio quanto definito alla registrazione della sidebar (before_widget, after_widget, before_title, after_title).

Applichiamo dunque a *$args* la funzione *extract*, in modo da disporre delle variabili necessarie.
In seguito applichiamo gli eventuali filtri posti al titolo del widget.

A questo punto possiamo iniziare con l'output vero e proprio del nostro widget.
Partiamo dalla variabile *$before_widget* (generalmente contiene un tag
).
Proseguiamo con il titolo, preceduto da *$before_title* e seguito da *$after_title*, quindi il markup del widget ed infine la variabile *$after_widget*.

Ed anche il metodo widget, che si occupa dell'output è pronto

Registrare il widget

L'ultima operazione che ci resta da fare è **registrare il widget e passarlo al gancio di inizializzazione dei widget**.

Conclusa la nostra classe, scriviamo questa funzione all'interno della quale passiamo alla funzione *register_widget* il nome della classe che definisce il nostro widget.

Fatto questo non ci resta che passare questa funzione al gancio di inizializzazione dei widget

Il nostro widget ora è pronto. Non ci resta che aprire la sezione plugin del pannello amministrativo e attivarlo.
In seguito nel menù aspetto alla voce widget, lo potremo trascinare nella sidebar.

Conclusione

In questo tutorial abbiamo gettato le basi per lo sviluppo di widget; certamente avrai notato quanto questa operazione sia "developer-friendly".
È ovvio che il nostro widget è molto semplice, ma una volta capito il meccanismo non c'è limite se

non la nostra fantasia.
Chiaramente, con un livello di complessità maggiore, sarà bene sfruttare appieno le possibilità che ci offre la programmazione ad oggetti.
Ad esempio nessuno ci vieta, all'interno della classe, di creare dei metodi privati nei quali svolgere operazioni complesse o di appoggio.
Infine, se il tuo widget necessita di un foglio di stile particolare oppure di un plugin javascipt, la procedura per integrarli è la medesima di quanto abbiamo visto nel tutorial sui plugin (con le apposite funzioni *wp_enqueue_style()* e *wp_enqueue_script()*).
Tutto questo lo vedremo nei prossimi tutorial, nei quali vi mostrerò lo sviluppo di un widget complesso. A presto.