GABRIELE ROMANATO

WordPress: visualizzare i post più letti senza plugin

Il problema maggiore quando si realizza un contatore di visite per i post di WordPress è eliminare dal conteggio gli hits da parte degli spider dei motori di ricerca. Fortunatamente con jQuery ed AJAX la soluzione è a portata di mano.

Definiamo in `functions.php` un callback AJAX utilizzando le API di WordPress:

```
function my_count_post_visits() {

        $raw_id = intval( $_POST['id'] );
        $id = 0;

        if( filter_var( $raw_id, FILTER_VALIDATE_INT ) ) {
                $id = $raw_id;
                $views = get_post_meta( $id,
'my_post_viewed', true );

                if( $views == '' ) {
                        update_post_meta( $id,
'my_post_viewed', '1' );
                } else {
                        $views_no = intval( $views );
                        update_post_meta( $id,
'my_post_viewed', ++$views_no );
                }
        }

        echo '';
        exit();
}

add_action( 'wp_ajax_my_count_visits', 'my_count_post_visits'
);
```

La tecnica è semplice: viene effettuata una richiesta POST in AJAX utilizzando come parametro l'ID del post. Come si può notare, l'ID viene convalidato poiché è necessario che sia un numero intero. Quindi viene aggiornato il custom field `my_post_viewed` con un valore incrementale.

jQuery entra in gioco non solo per effettuare la richiesta AJAX, ma anche per estrarre l'ID del post dall'attributo `class` dell'elemento `body`.

```
(function( $ ) {
        $(function() {

                var body = document.body,
                        bodyClass = body.className;

                if( $( "body" ).hasClass( "single" ) ) {

                        var postID = bodyClass.match(
/postid-(\d+)/ );
                        var id = postID[1];
                        var ajaxURL = "http://" +
location.host + "/wp-admin/admin-ajax.php";

                        $.ajax({
                                type: "POST",
                                dataType: "text",
                                url: ajaxURL,
                                data: {
                                        action:
"my_count_visits",
                                        id: id
                                },
                                success: function() {}
                        });


                }

        });
```

```
})( jQuery );
```

A questo punto possiamo creare un Loop personalizzato nel tema di WordPress:

```
$popular_posts_args = array(
        'posts_per_page' => 3,
        'meta_key' => 'my_post_viewed',
        'orderby' => 'meta_value_num',
        'order'=> 'DESC'
);

$popular_posts_loop = new WP_Query( $popular_posts_args );

while( $popular_posts_loop->have_posts() ):
        $popular_posts_loop->the_post();
        // Continua il Loop
endwhile;
wp_reset_query();
```

Come si può notare si tratta di una soluzione estremamente semplice da implementare.