

 marketful <small>fulfillment</small>	Documentación del archivo: class-admin-product-entries.php
	Autor: Mauricio Alcala
	Fecha: 13 de agosto de 2018
	Versión: 1.01
	Autoriza: Adolfo Yanes

Descripción general:

Este contiene la clase admin-product-entries, tiene las funciones de crear y ejecutar la query para solicitar todos los registros que se van a mostrar dentro de las diferentes secciones del plugin. Además, define el nombre del plugin, la versión y dos ejemplos de hooks para cómo poner en cola la hoja de estilo específica de administrador y JavaScript.

Funciones PHP

```

/*****
 * PRODUCT METADATA MercadoLibre
 * Se declaran constantes para referirse a campos en la base de datos
 */
if (!defined('ML_META_TITLE'))
    define('ML_META_TITLE', 'titulo_ml');
if (!defined('ML_META_STATUS'))
    define('ML_META_STATUS', 'mercadolibre');
if (!defined('ML_META_STOCK'))
    define('ML_META_STOCK', 'inventario_ml');
if (!defined('ML_META_EXPOSITION_TYPE'))
    define('ML_META_EXPOSITION_TYPE', 'exposicion_ml');
if (!defined('ML_META_DELIVERY_TYPE'))
    define('ML_META_DELIVERY_TYPE', 'tipo_de_envio_ml');
if (!defined('ML_META_STORE_RECALL'))
    define('ML_META_STORE_RECALL', 'retiro_en_tienda_ml');
if (!defined('ML_META_WARRANTY_TIME'))
    define('ML_META_WARRANTY_TIME', 'tiempo_garantia_ml');
if (!defined('ML_META_WARRANTY_UNIT_TIME'))
    define('ML_META_WARRANTY_UNIT_TIME', 'ut_garantia_ml');
if (!defined('ML_META_PRICE'))
    define('ML_META_PRICE', 'regular_price');
if (!defined('ML_META_CATEGORIES'))
    define('ML_META_CATEGORIES', 'categories_ml');
if (!defined('ML_META_LAST_CATEGORY'))
    define('ML_META_LAST_CATEGORY', 'last_category_ml');
if (!defined('ML_META_PRECIO_ML'))
    define('ML_META_PRECIO_ML', 'precio_ml');
*****/

/*****
 * @Clase MKF_ProductEntry que @hereda los metodos y atributos de MKF_DBCore
 *
 * Se declaran las variables de la clase. Ademas se definen las propiedades de la clase
 * ($instance @string, $plg_id @string y $version @string)
 *
 */
class MKF_ProductEntry extends MKF_DBCore
{
    private static $instance = NULL;
    private $plg_id;
    private $version;
    private $meta_title;
    private $meta_stock;
    private $meta_store;

```

```

private $meta_status;
private $meta_exp;
private $meta_wtime;
private $meta_utime;
private $meta_price;
private $meta_cat;
private $meta_lcat;
private $meta_precio_ml;
/*****/

/*****
 * @función __construct(@string = (PLUGIN_GSNAME), @string = (PLUGIN_GVERSION))
 *
 * Recibe el valor del $plg_id y $version, en caso de que no se envíen se asigna el
 * valor del plugin_gsname y plugin_version.
 *
 * Posteriormente, asigna a las variables que se definieron en un principio los valores de los
 * diferentes campos de ML.
 */
public function __construct( $plg_id = PLUGIN_GSNAME, $version = PLUGIN_GVERSION)
{
    $this->plg_id = $plg_id;
    $this->version = $version;
    $this->meta_title = ML_META_TITLE;
    $this->meta_status = ML_META_STATUS;
    $this->meta_stock = ML_META_STOCK;
    $this->meta_store = ML_META_STORE_RECALL;
    $this->meta_exp = ML_META_EXPOSITION_TYPE;
    $this->meta_ship = ML_META_DELIVERY_TYPE;
    $this->meta_wtime = ML_META_WARRANTY_TIME;
    $this->meta_utime = ML_META_WARRANTY_UNIT_TIME;
    $this->meta_price = ML_META_PRICE;
    $this->meta_cat = ML_META_CATEGORIES;
    $this->meta_lcat = ML_META_LAST_CATEGORY;
    $this->meta_precio_ml = ML_META_PRECIO_ML;
}
/*****/

/*****
 * @función GetInstance()
 *
 * Es una función estática que accede al objeto $instance y busca que sea nulo,
 * en cuyo caso crea el objeto.
 * En caso contrario retorna el objeto.
 */
// función para traer objetos de la clase
public static function GetInstance()
{
    if ( is_null( self::$instance ) )
    {
        self::$instance = new self;
    }
    return self::$instance;
}
/*****/

/*****
 * @función prefix_admin:add_metadata_to_product_entry()
 *
 * Primero tomamos el valor de extr_prefix_all en caso de no encontrar ningún valor le
 * asignamos una "p"
 * Posteriormente con la función update_post_meta de WP actualizamos los valores
 * del producto $p_product_id.
 * Finaliza redirigiendonos hacia admin.php en la sección mkf-product-entries.
 */
// función para guardar la data q viene del formulario
public function prefix_admin_add_metadata_to_product_entry()
{

```

```

extract($_REQUEST, EXTR_PREFIX_ALL, "p");
// update_post_meta(intval($p_product_id), $this->meta_title, empty($p_entry_title) ? null :
$p_entry_title);
update_post_meta(intval($p_product_id), $this->meta_stock, $p_stock);
// update_post_meta(intval($p_product_id), $this->meta_store, $p_store_recall);
update_post_meta(intval($p_product_id), $this->meta_status, $p_status_post);
update_post_meta(intval($p_product_id), $this->meta_exp, $p_exposition);
// update_post_meta(intval($p_product_id), $this->meta_wtime, $p_time_warranty);
// update_post_meta(intval($p_product_id), $this->meta_cat, json_encode($p_ml_categories,
JSON_FORCE_OBJECT));
// update_post_meta(intval($p_product_id), $this->meta_lcat,
$p_ml_categories['child'][count($p_ml_categories['child']) - 1]);
// update_post_meta(intval($p_product_id), $this->meta_precio_ml, $p_precio_ml);
header("Location: admin.php?page=mkf-product-entries&success");
}
/*****

/*****
* @función get_product_list()
*
* Esta función define la query que se requiere hacer y la ejecuta para traer todos los datos
* que requiere el plugin.
*
* Primero se define un array que posteriormente mostrara toda la información resultante
* llamado $out.
* Despues utilizando array_push() agregamos al final del array out() el resultado que devuelve
* ejecutar la query.
* Despues retorna el valor de $out
*/
// función para traer los productos
public function get_product_list()
{
    $out = array();

    $sql = "SELECT tmp.ID,
        tmp.sku,
        IFNULL(tmp.titulo_ml, tmp.title) title,
        CASE WHEN tmp.mercadolibre = 'A' THEN 'Activo'
              WHEN tmp.mercadolibre = 'I' THEN 'Inactivo'
        ELSE tmp.mercadolibre
        END status,
        CASE WHEN tmp.exposicion_ml = 'C' THEN 'Clasica'
              WHEN tmp.exposicion_ml = 'P' THEN 'Premium'
        ELSE tmp.exposicion_ml
        END exposicion,
        tmp.precio_ml as price,
        IFNULL(tmp.ml_stock, tmp.stock) stock,
        IFNULL(tmp.link_publicacion, tmp.wp_url) url
    FROM
    (
        SELECT p.ID,
            pm1.meta_value sku,
            (SELECT meta_value
             FROM {$this->getPostMetaTableName()}
             WHERE post_id = p.ID AND meta_key = '{$this->meta_title}') titulo_ml,
            p.post_title title,
            (SELECT meta_value
             FROM {$this->getPostMetaTableName()}
             WHERE post_id = p.ID AND meta_key = '{$this->meta_status}') mercadolibre,
            (SELECT meta_value
             FROM {$this->getPostMetaTableName()}
             WHERE post_id = p.ID AND meta_key = '{$this->meta_exp}') exposicion_ml,
            p.guid wp_url,
            (SELECT meta_value
             FROM {$this->getPostMetaTableName()}
             WHERE post_id = p.ID AND meta_key = 'link_publicacion') link_publicacion,

            pm3.meta_value stock,
            (SELECT meta_value
             FROM {$this->getPostMetaTableName()}

```

```

        WHERE post_id = p.ID AND meta_key = '{$this->meta_stock}') m1_stock,
        (SELECT meta_value
        FROM {$this->getPostMetaTableName()}
        WHERE post_id = p.ID AND meta_key = '{$this->meta_precio_ml}') precio_ml
    FROM {$this->getPostTableName()} p
    INNER JOIN {$this->getPostMetaTableName()} pm1 ON pm1.post_id = p.ID and pm1.meta_key = '_sku'
    INNER JOIN {$this->getPostMetaTableName()} pm2 ON pm2.post_id = p.ID and pm2.meta_key =
'_regular_price'
    INNER JOIN {$this->getPostMetaTableName()} pm3 ON pm3.post_id = p.ID and
    (pm3.meta_key = '_stock' or pm3.meta_key like
'%stock_quantity')
    WHERE p.post_type = 'product') tmp";

    array_push($out, array("data"=> $this->execute_custom_query($sql)));
    return $out;
}
/*****

/*****
* @función publica get_ml_metadata(@string = (NULL))
*
* Recibe el valor de el $post_id y en caso de no encontrarlo le asigna valor nulo.
* Crea en una variable @string la query que se requiere ejecutar.
* Posteriormente con la función array_push() se ejecuta la query y se guarda dentro del
* array $out
* Para finalizar retornando el valor de $out.
*/
public function get_ml_metadata($post_id = null)
{
    $out = array();

    $post_id = is_null($post_id) ? 0 : intval($post_id);

    $sql = "SELECT pm1.meta_value title,
        pm2.meta_value status,
        pm3.meta_value inventory,
        pm4.meta_value exposicion,
        pm5.meta_value delivery_type,
        pm6.meta_value store_recall,
        pm7.meta_value time_warranty,
        pm8.meta_value utime_warranty,
        pm9.meta_value price,
        pm10.meta_value categories,
        pm11.meta_value precio_ml
    FROM {$this->getPostTableName()} p
    LEFT JOIN {$this->getPostMetaTableName()} pm1 ON pm1.post_id = p.ID and pm1.meta_key = '{$this->meta_title}'
    LEFT JOIN {$this->getPostMetaTableName()} pm2 ON pm2.post_id = p.ID and pm2.meta_key = '{$this->meta_status}'
    LEFT JOIN {$this->getPostMetaTableName()} pm3 ON pm3.post_id = p.ID and pm3.meta_key = '{$this->meta_stock}'
    LEFT JOIN {$this->getPostMetaTableName()} pm4 ON pm4.post_id = p.ID and pm4.meta_key = '{$this->meta_exp}'
    LEFT JOIN {$this->getPostMetaTableName()} pm5 ON pm5.post_id = p.ID and pm5.meta_key = '{$this->meta_ship}'
    LEFT JOIN {$this->getPostMetaTableName()} pm6 ON pm6.post_id = p.ID and pm6.meta_key = '{$this->meta_store}'
    LEFT JOIN {$this->getPostMetaTableName()} pm7 ON pm7.post_id = p.ID and pm7.meta_key = '{$this->meta_wtime}'
    LEFT JOIN {$this->getPostMetaTableName()} pm8 ON pm8.post_id = p.ID and pm8.meta_key = '{$this->meta_utime}'
    LEFT JOIN {$this->getPostMetaTableName()} pm9 ON pm9.post_id = p.ID and pm9.meta_key = '{$this->meta_price}'
    LEFT JOIN {$this->getPostMetaTableName()} pm10 ON pm10.post_id = p.ID and pm10.meta_key = '{$this->meta_cat}'
    LEFT JOIN {$this->getPostMetaTableName()} pm11 ON pm11.post_id = p.ID and pm11.meta_key = '{$this->meta_precio_ml}'
    WHERE p.ID = {$post_id}";

    array_push($out, array("data"=> $this->execute_custom_query($sql)));
    return $out;
}

```

```

/*****/

/*****
 * @función get_product_edit_form_title(@tring = (NULL), @string = (NULL))
 *
 * Dentro de esta función se retorna el valor del titulo dependiendo si es nulo o simplemente esta vacío,
 * así como el valor del SKU.
 */
public static function get_product_edit_form_title($title = null, $sku = null)
{
    return ((is_null($title) || empty($title)) ? "" : $title) .
           ((is_null($sku) || empty($sku)) ? "" : " // " . $sku);
}
/*****/

/*****
 * @función product_edit_form_title_to_presenter(@string = (NULL), @tring = (NULL))
 *
 * Hace echo al valor $title y $Sku del objeto en cuestión.
 */
public static function product_edit_form_title_to_presenter($title = null, $sku = null)
{
    echo self::get_product_edit_form_title($title, $sku);
}
/*****/

/*****
 * @función product_edit_form_title_to_presenter(@string = (NULL), @tring = (NULL))
 *
 * Hace echo al valor $title y $Sku del objeto en cuestión.
 */
public static function product_edit_form_title_to_presenter($title = null, $sku = null)
{
    echo self::get_product_edit_form_title($title, $sku);
}
/*****/

```