Exprimiendo al máximo los hooks (actions, filters) a partir de casos reales

Sulema Rocha Betancur

Sulema Rocha Betancur

Full-stack developer

Freelancer alrededor de 10 años, WordPress, joomla, prestashop y desarrollo a medida (Laravel, Symfony)

/sulema-rocha-betancur

@suledev

@sulecita

#WCBCN

2023 WC BCN

Hooks
(Ganchos)

Son puntos dentro del core de WordPress o de un plugin, donde podemos agregar, modificar o eliminar un comportamiento.

Tipos de hooks

Actions

No modifica datos internos, pero hace uso de estos, se ejecuta según el orden de prioridad.

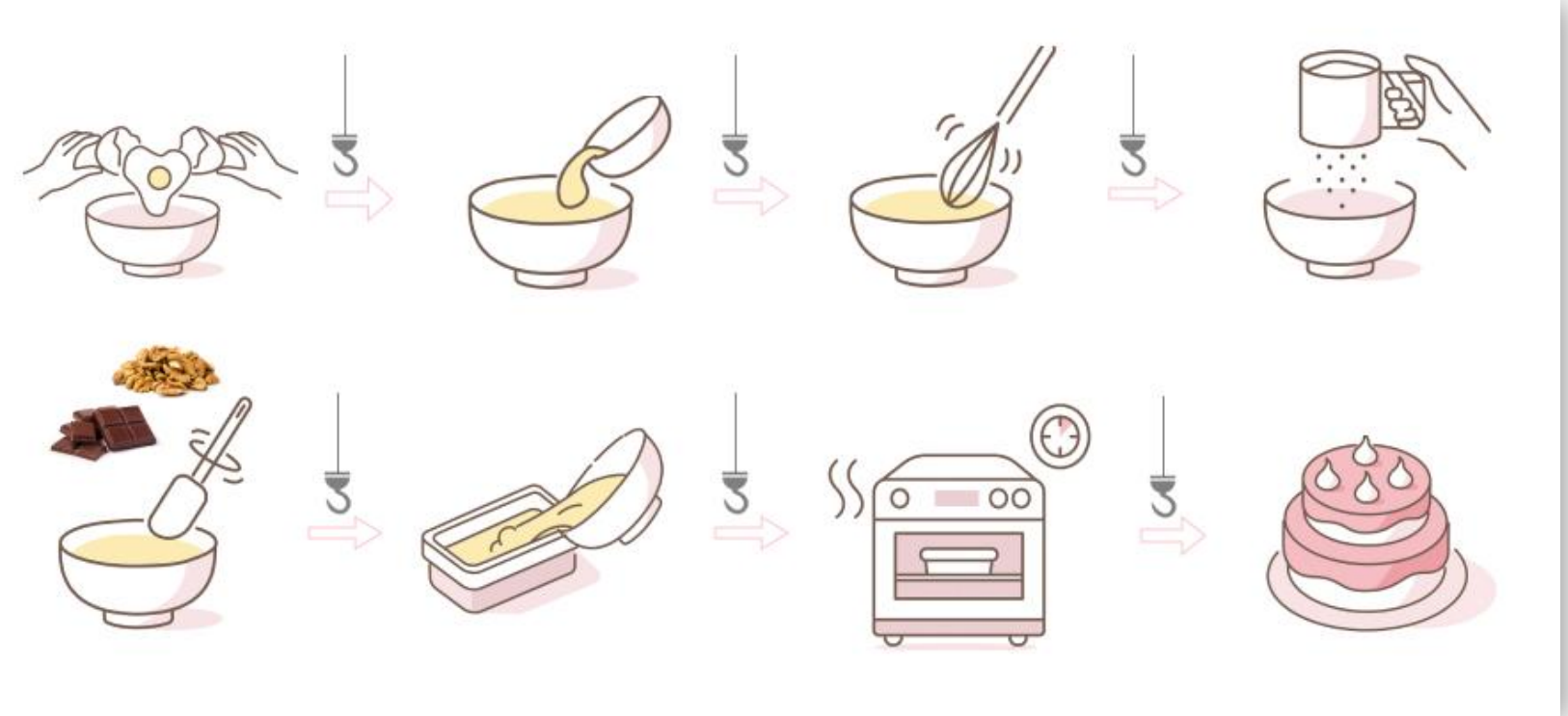
- *add_action*
- *do_action*
- *remove_action*

Filters

Modificar datos internos en tiempo de ejecución según el orden de prioridad.

- *add_filter*
- *remove_filter*

Receta para tarta



Casos Reales

- Proyecto 1. Web Vinícola
- Proyecto 2. Web para socios de una librería donde se actualiza la información mediante API
- Proyecto 3. Sistema de fidelización de clientes usando API

Proyecto 1: Vinícola

Vinícola que tiene su web informativa con una tienda online para vender sus productos
(woocommerce + WPML)

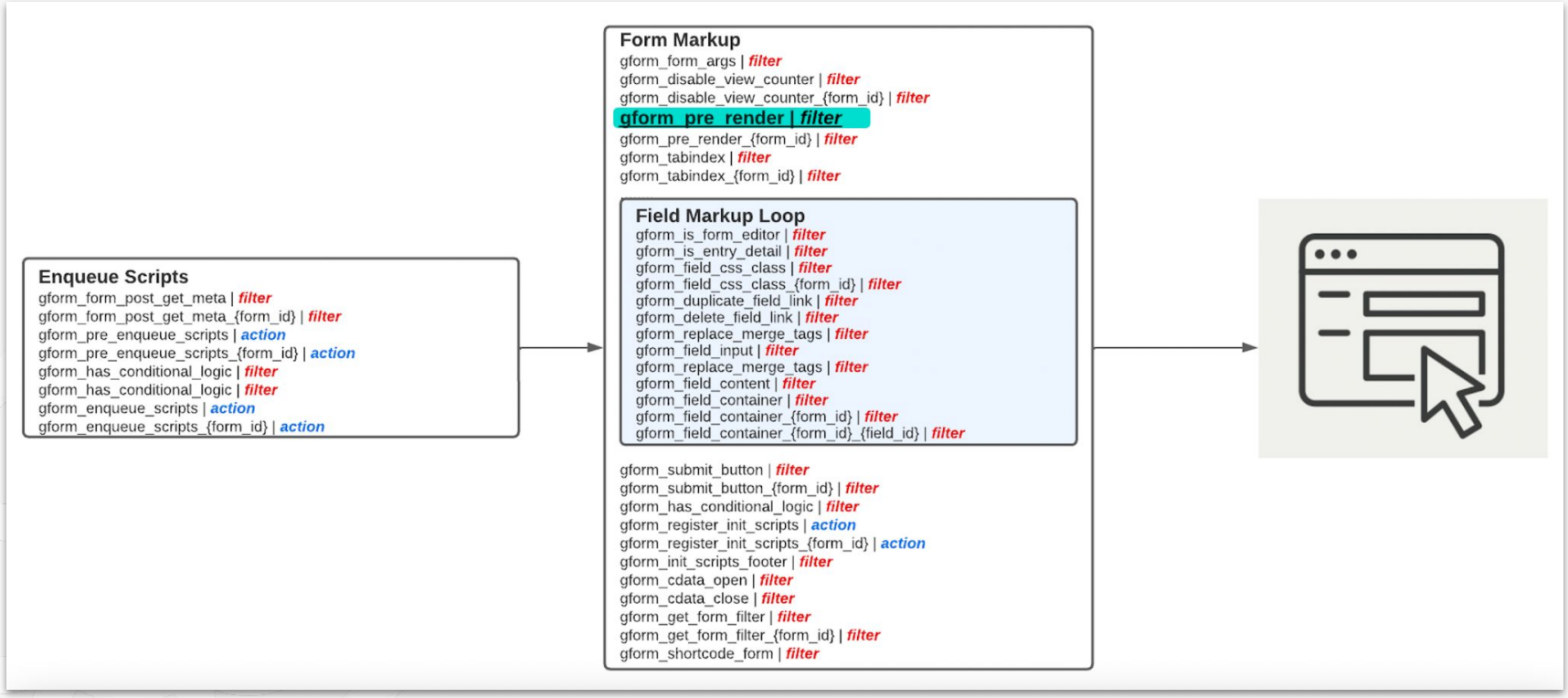
Proyecto 1: Vinícola
Problemas/Requerimientos

- Vender actividades/eventos.
- Permitir tener actividades sin fecha.
- Permitir regalar una actividad.
- Tener las reservas en un calendario.
- Notificación de recordatorio(1 día antes) y agradecimiento(1 día después).
- Exportar reportes de actividades realizadas o pendientes.
- Enviar información del usuario directamente a mailchimp.

Proyecto 1: Vinícola
Soluciones

- Actividades => **Woocommerce Bookings**.
- Actividades sin fecha => **Gravity Forms (hooks)**.
- Permitir regalar una actividad => **ACF (hooks), desarrollo para notificación de email**.
- Conectar con calendario => **Woocommerce Bookings**
- Notificaciones => **Woocommerce Bookings y desarrollo a medida, CRON Events (hooks)**.
- Exportar reportes de actividades => **Admin Columns (hooks).**
- Enviar información a mailchimp => **(hooks)**

Proyecto 1: Vinícola



Proyecto 1: Vinícola

```php
add_filter( 'gform_pre_render', 'populate_choices' );
function populate_choices( $form ) {
    session_start();
    if( !is_admin() ){
        if ( $form['id'] == get_id_form_contact_activity() ){
            foreach($form['fields'] as $field ){
                if( $field->id == get_id_field_name_activity() ){
                    global $product;
                    $field->defaultValue = $product->get_name();
                }
            }
        }
    }
    return $form;
}
```

```php
add_filter('mc4wp_integration_woocommerce_data', function($data, $order_id) {
    $order = wc_get_order( $order_id );
    if( $order ){
        $data['MMERGE5'] = $order->get_billing_country();
        $data['MMERGE6'] = apply_filters( 'wpml_current_language', null );
    }
    return $data;
}, 10, 2 );
```

Proyecto 1: Vinícola

```php
public function __construct() {
    add_filter( 'ac/export/values', array( $this, 'apply_filter_admin_columns_by_export'), 10, 3);
    add_filter( 'ac/column/value', array( $this, 'apply_filter_admin_columns' ), 10, 3 );
}
public function apply_filter_admin_columns_by_export( $value, AC\Column $column, $id ){
    if ( $column instanceof AC\Column\CustomField ) {
        if( '_booking_order_item_id' == $column->get_meta_key()){...}
        if( '_booking_persons' == $column->get_meta_key() ){...}
        if( '_booking_start' == $column->get_meta_key()){...}
        if( $column->get_meta_key() == "_booking_customer_id_country"){...}
        if( $column->get_meta_key() == "_booking_customer_id_city"){
            $booking = get_wc_booking( $id );
            if( $booking->get_order() ){
                $order_ = new WC_Order( $booking->get_order_id()  );
                $value = $order_->get_billing_city();
            }
        }

    }
    return $value;
}
```

Proyecto 2: Socios de una Librería

Conectar con el sistema central de datos de la librería mediante API

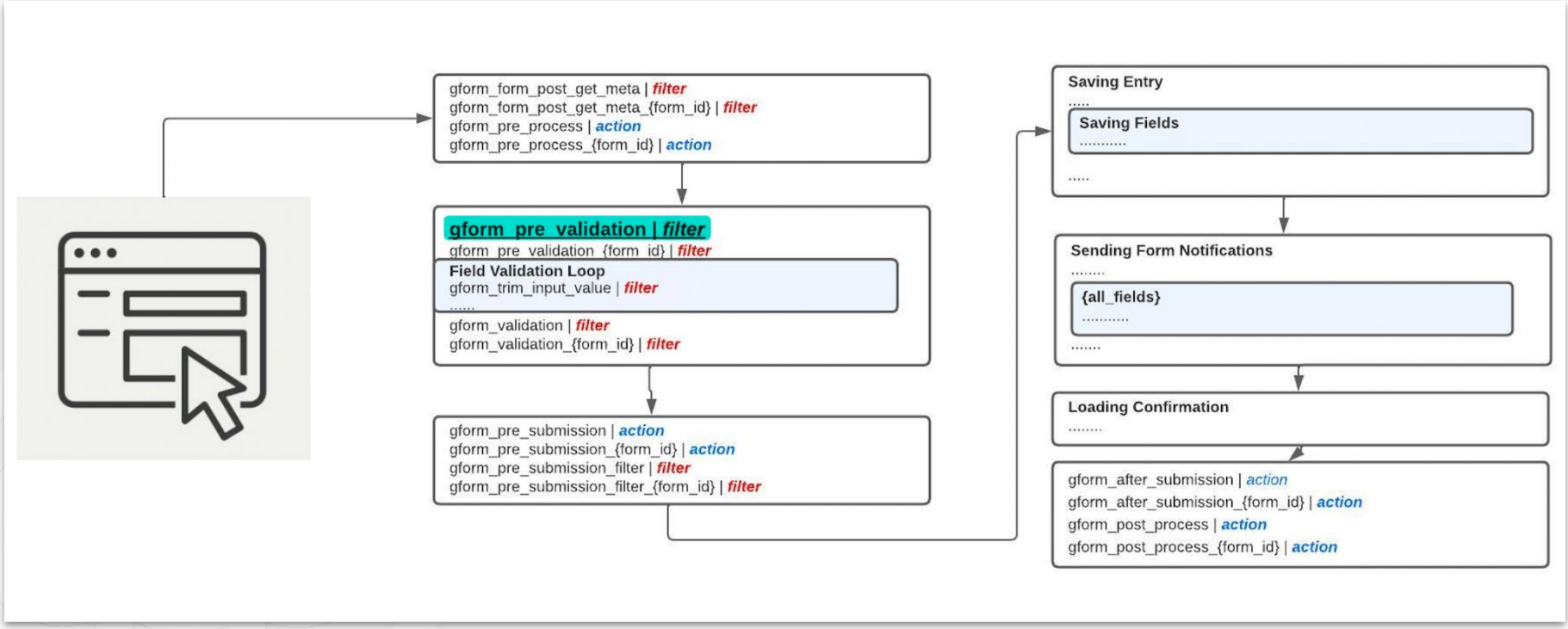
Proyecto 2: Socios de una Librería
Problemas/Requerimientos

- Validar datos de socio (API).
- Recuperar información (API) y mostrar dicha información al socio.
- Guardar información modificada por el socio en el sistema central (API).
- Enviar notificación al socio con los datos modificados.
- Mostrar una pantalla de agradecimiento después de la edición de datos.
- Seguimiento de las llamadas al sistema externo.

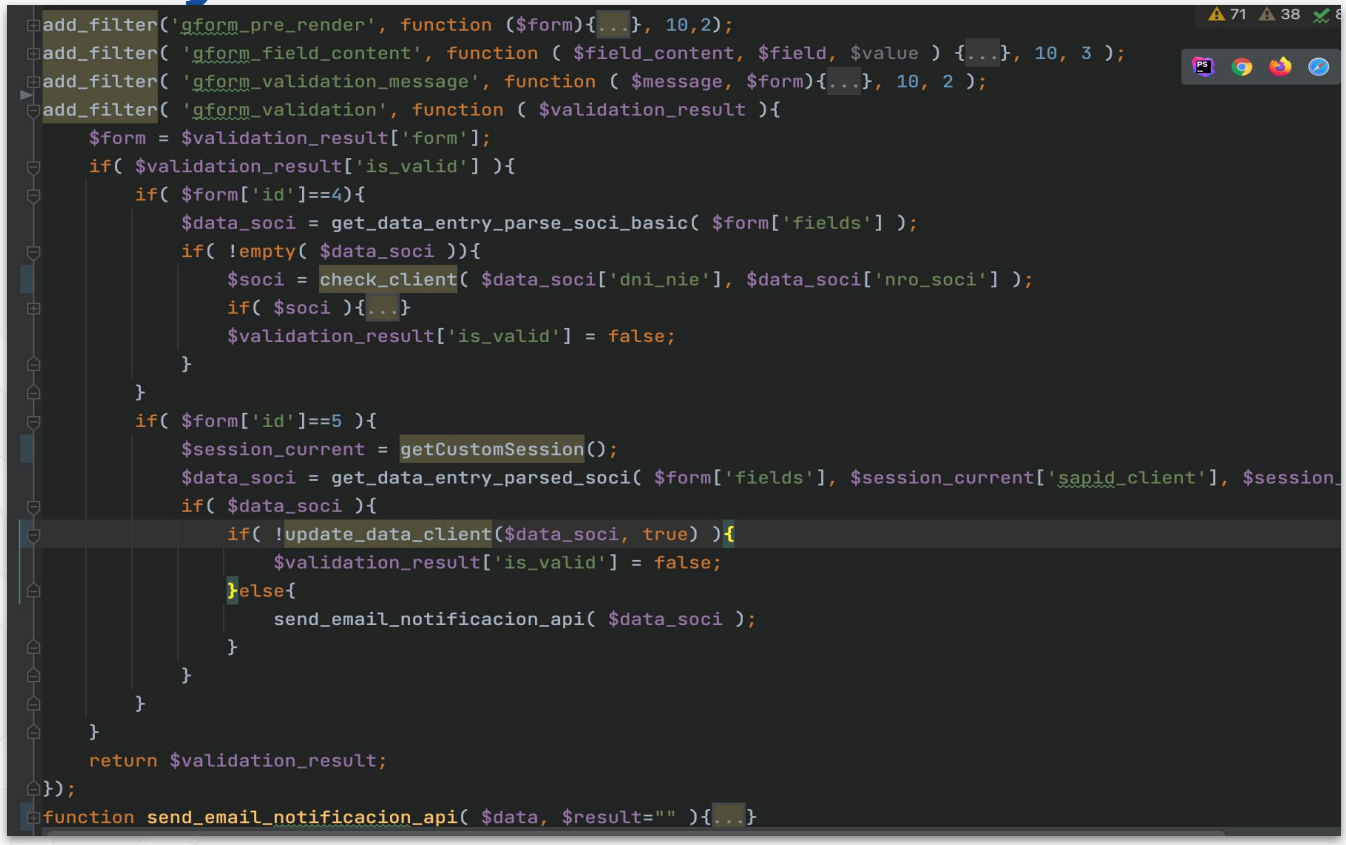
Proyecto 2: Socios de una Librería Soluciones

- **Wordpress** => Web con las 3 pantallas.
- **Oxygen** => Para generar los templates.
- **Gravity Forms** => Formularios de validación y edición y notificación a los socios (**hooks**).

Proyecto 2: Socios de una Librería



Proyecto 2: Socios de una Librería



```php
add_filter('gform_pre_render', function ($form){...}, 10,2);
add_filter( 'gform_field_content', function ( $field_content, $field, $value ) {...}, 10, 3 );
add_filter( 'gform_validation_message', function ( $message, $form){...}, 10, 2 );
add_filter( 'gform_validation', function ( $validation_result ){
    $form = $validation_result['form'];
    if( $validation_result['is_valid'] ){
        if( $form['id']==4){
            $data_soci = get_data_entry_parse_soci_basic( $form['fields'] );
            if( !empty( $data_soci )){
                $soci = check_client( $data_soci['dni_nie'], $data_soci['nro_soci'] );
                if( $soci ){...}
                $validation_result['is_valid'] = false;
            }
        }
        if( $form['id']==5 ){
            $session_current = getCustomSession();
            $data_soci = get_data_entry_parsed_soci( $form['fields'], $session_current['sapid_client'], $session_
            if( $data_soci ){
                if( !update_data_client($data_soci, true) ){
                    $validation_result['is_valid'] = false;
                }else{
                    send_email_notificacion_api( $data_soci );
                }
            }
        }
    }
    return $validation_result;
});
function send_email_notificacion_api( $data, $result="" ){...}
```

Proyecto 3: Sistema de fidelización de clientes

Apartado en la web para que los clientes se registren y tenga un panel donde se muestre su información personal, cupones, cheques, historial de compras.

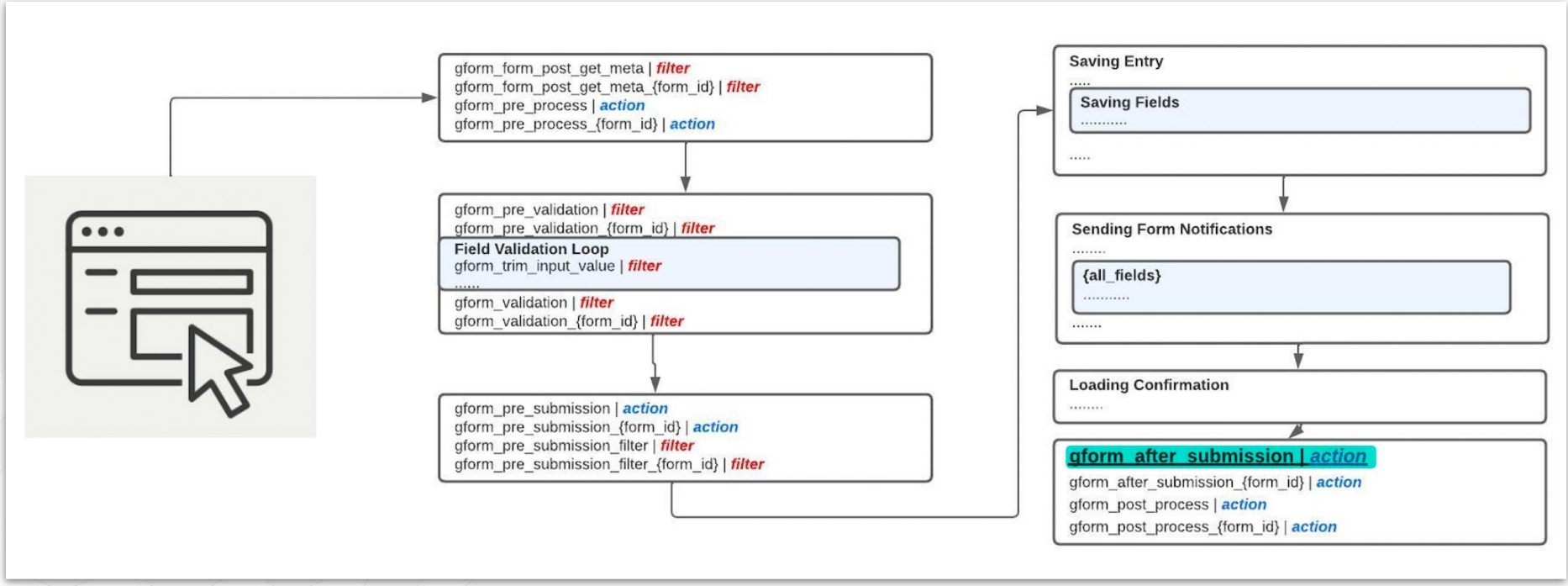
Proyecto 3: Sistema de fidelización de clientes
Problemas/Requerimientos

- El usuario de la web se registre desde un formulario, y este conecte con el sistema central (API).
- Recuperar información (API) y mostrar dicha información en el panel habilitado.
- Modificar información de código postal y datos personales en el sistema central (API).
- Enviar notificación de alta al usuario que se registró.

Proyecto 3: Sistema de fidelización de clientes Soluciones

- El usuario de la web se registre desde un formulario, y este conecte con el sistema central (API) => **Gravity Forms (hooks)**
- Recuperar información (API) y mostrar dicha información en el panel habilitado => **(shortcodes)**
- Modificar información de código postal y datos personales en el sistema central (API) => **Gravity Forms (hooks)**
- Enviar notificación de alta al usuario que se registró => **Gravity Forms (hooks)**

Proyecto 3: Sistema de fidelización de clientes



Proyecto 3: Sistema de fidelización de clientes

```php
add_action( 'gform_after_submission', 'register_user_submission', 10, 2 );
function register_user_submission( $entry, $form ) {
    if( $form['id'] == get_field('ebook_form', 'options') ){
        $field_register_user = get_field('name_checkbox_register_user_book_field', 'options');
        if( $entry[ $field_register_user ] ){...}
    }
    if( is_active_targeta_fidelity() ){
        if ( $form['id'] == get_field('formulari_activacio', 'options') )
        {
            $field_id_user = get_field('id_user_form_fidelity_field', 'options');
            if( $field_id_user ){...}
        }
    }
}

add_filter( 'wp_new_user_notification_email', 'custom_wp_new_user_notification_email', 10, 3 );
function custom_wp_new_user_notification_email( $wp_new_user_notification_email, $user, $blogname ) {
    $message = $wp_new_user_notification_email['message'];
    $wp_new_user_notification_email['message'] = str_replace( search: 'wp-login.php', replace: 'login', $message);
    return $wp_new_user_notification_email;
}
```

Buenas prácticas a la hora de programar con Hooks

- Desarrollar el código de la solución en un plugin.
- Utilizar git.
- Utilizar preprocesadores CSS.
- Código limpio.

Conclusiones

- Los **hooks** van cobrando importancia no solo en el core de WordPress sino que también en los plugins.
- Empezar a usar **hooks** en los nuevos desarrollos de plugins.
- Usar de manera moderada los plugins y hacer uso de **hooks** para evitar el uso de algunos plugins.
- Documentar los cambios de los **hooks**, para así tener una web mantenible.

Empezamos a integrar los hooks a la hora de implementar soluciones?

¡Gracias!

¿Preguntas?

#WCBCN