Come implementare le Web Push Notifications in WordPress senza plugin

Implementare le web push notifications in WordPress senza l'uso di plugin o servizi di terze parti richiede una buona conoscenza di JavaScript, PHP, e delle API REST di WordPress. In questa guida, ti mostrerò come utilizzare i custom post type e i custom field per gestire le iscrizioni, integrando la logica del modulo WebPush all'interno di un endpoint delle API REST di WordPress.

Crea un file denominato `sw.js` nella root del tuo sito WordPress. Questo file gestirà le notifiche push:

```
self.addEventListener('push', function(event) {
    const data = event.data.json();
    const options = {
        body: data.body,
        icon: data.icon,
        badge: data.badge,
        data: {
            url: data.url
        }
    };
    event.waitUntil(
        self.registration.showNotification(data.title,
options)
    );
});

self.addEventListener('notificationclick', function(event) {
    event.notification.close();
    event.waitUntil(
        clients.openWindow(event.notification.data.url)
```

```
    );
});
```

Registra il service worker nel front-end del tuo sito, aggiungendo questo codice in un file JavaScript definito in WordPress con la funzione `wp_register_script()`:

```
if ('serviceWorker' in navigator) {
    navigator.serviceWorker.register('/sw.js')
    .then(function(registration) {
        console.log('Service Worker registrato con
successo:', registration);
    }).catch(function(error) {
        console.log('Service Worker registrazione fallita:',
error);
    });
}
```

Per salvare le iscrizioni ai push notifications, utilizzeremo un Custom Post Type (CPT) chiamato "push_subscriptions".

```
// functions.php

function create_push_subscriptions_cpt() {
    $labels = [
        'name' => 'Push Subscriptions',
        'singular_name' => 'Push Subscription',
        'menu_name' => 'Push Subscriptions',
        'name_admin_bar' => 'Push Subscription',
    ];

    $args = [
        'labels' => $labels,
        'public' => false,
        'has_archive' => false,
```

```
        'show_ui' => true,
        'show_in_menu' => true,
        'supports' => ['title','custom-fields'],
        'capability_type' => 'post',
    ];

    register_post_type('push_subscription', $args);
}
add_action('init', 'create_push_subscriptions_cpt');
```

Utilizza i custom field per memorizzare le informazioni di iscrizione:

```
// functions.php

function save_subscription_data($subscription, $post_id) {
    foreach ($subscription as $key => $value) {
        update_post_meta($post_id, $key,
sanitize_text_field($value));
    }
}
```

Per inviare notifiche, devi prima chiedere il permesso agli utenti. Aggiungi questo script nel front-end:

```
function askPermission() {
    return new Promise(function(resolve, reject) {
        const permissionResult =
Notification.requestPermission(function(result) {
            resolve(result);
        });

        if (permissionResult) {
            permissionResult.then(resolve, reject);
        }
    }).then(function(permissionResult) {
        if (permissionResult !== 'granted') {
```

```
            throw new Error('Permesso per le notifiche non
concesso.');
        }
    });
}

askPermission().then(function() {
    console.log('Permesso concesso per le notifiche.');
});
```

Ora devi iscrivere l'utente al servizio push e salvare l'iscrizione utilizzando un endpoint REST personalizzato.

```
// functions.php

add_action('rest_api_init', function () {
    register_rest_route('push/v1', '/save-subscription', [
        'methods' => 'POST',
        'callback' => 'save_subscription',
    ]);
});

function save_subscription(WP_REST_Request $request) {
    $subscription_data = $request->get_json_params();

    $post_id = wp_insert_post([
        'post_type' => 'push_subscription',
        'post_title' => 'Subscription - ' . uniqid(),
        'post_status' => 'publish',
    ]);

    if ($post_id) {
        save_subscription_data($subscription_data, $post_id);
        return new WP_REST_Response('Iscrizione salvata',
200);
    }

    return new WP_REST_Response('Errore nel salvataggio
```

```
dell\'iscrizione', 500);
}
```

Esegui l'iscrizione nel front-end:

```
navigator.serviceWorker.ready.then(function(registration) {
    registration.pushManager.subscribe({
        userVisibleOnly: true,
        applicationServerKey:
urlBase64ToUint8Array('LA_TUA_CHIAVE_VAPID_PUBBLICA')
    }).then(function(subscription) {
        console.log('Iscritto con successo:', subscription);

        // Salva l'iscrizione tramite l'endpoint REST
        fetch('/wp-json/push/v1/save-subscription', {
            method: 'POST',
            body: JSON.stringify(subscription),
            headers: {
                'Content-Type': 'application/json'
            }
        });
    }).catch(function(error) {
        console.log('Iscrizione fallita:', error);
    });
});

function urlBase64ToUint8Array(base64String) {
    const padding = '='.repeat((4 - base64String.length % 4)
% 4);
    const base64 = (base64String + padding).replace(/-/g,
'+').replace(/_/g, '/');
    const rawData = window.atob(base64);
    const outputArray = new Uint8Array(rawData.length);
    for (let i = 0; i < rawData.length; ++i) {
        outputArray[i] = rawData.charCodeAt(i);
    }
    return outputArray;
}
```

Per inviare notifiche push, avrai bisogno di generare delle chiavi VAPID (Voluntary Application Server Identification).

Puoi generare le chiavi VAPID usando la libreria PHP `WebPush`. Prima, installala tramite Composer:

```
composer require minishlink/web-push
```

Poi, genera le chiavi:

```
use Minishlink\WebPush\VAPID;

$keys = VAPID::createVapidKeys();
echo json_encode($keys); // publicKey, privateKey
```

Salva queste chiavi da qualche parte in modo sicuro, preferibilmente in un file di configurazione.

L'invio delle notifiche verrà gestito all'interno di un altro endpoint REST, integrando il modulo WebPush.

```
// functions.php

add_action('rest_api_init', function () {
    register_rest_route('push/v1', '/send-notification', [
        'methods' => 'POST',
        'callback' => 'send_notification',
    ]);
});

function send_notification(WP_REST_Request $request) {
    $title = sanitize_text_field($request-
>get_param('title'));
```

```
    $body = sanitize_text_field($request->get_param('body'));
    $icon = esc_url($request->get_param('icon'));
    $badge = esc_url($request->get_param('badge'));
    $url = esc_url($request->get_param('url'));

    $args = [
        'post_type' => 'push_subscription',
        'post_status' => 'publish',
        'posts_per_page' => -1,
    ];

    $subscriptions = new WP_Query($args);

    $auth = [
        'VAPID' => [
            'subject' => 'mailto:your-email@example.com',
            'publicKey' => 'LA_TUA_CHIAVE_VAPID_PUBBLICA',
            'privateKey' => 'LA_TUA_CHIAVE_VAPID_PRIVATA'
        ],
    ];

    $webPush = new Minishlink\WebPush\WebPush($auth);

    if ($subscriptions->have_posts()) {
        while ($subscriptions->have_posts()) {
            $subscriptions->the_post();
            $subscription =
json_decode(get_post_meta(get_the_ID(), 'subscription',
true), true);
            $subscription =
Minishlink\WebPush\Subscription::create($subscription);

            $webPush->sendNotification(
                $subscription,
                json_encode([
                    'title' => $title,
                    'body' => $body,
                    'icon' => $icon,
                    'badge' => $badge,
                    'url' => $url,
                ])
            );
        }
```

```
        wp_reset_postdata();

        foreach ($webPush->flush() as $report) {
            $endpoint = $report->getRequest()->getUri()->__toString();
            if ($report->isSuccess()) {
                echo "[v] Notifica inviata con successo a {$endpoint}.\n";
            } else {
                echo "[x] Notifica fallita a {$endpoint}: {$report->getReason()}\n";
            }
        }
    } else {
        return new WP_REST_Response('Nessuna iscrizione trovata', 404);
    }

    return new WP_REST_Response('Notifiche inviate', 200);
}
```

Conclusione

Implementare le web push notifications in WordPress utilizzando custom post type e custom field per gestire le iscrizioni offre una soluzione flessibile e completamente integrata. Seguendo questa guida, potrai gestire e inviare notifiche direttamente dal tuo sito WordPress senza dipendere da plugin o servizi esterni, mantenendo il pieno controllo sui dati e sulla logica del sistema.