Plugin Directory

Changeset 3486202 for debugger-troubleshooter/trunk/debug-troubleshooter.php

Timestamp: 03/19/2026 07:55:39 AM (12 days ago)
Author: jhimross

Message: version 1.4.0

File: 1 edited

debugger-troubleshooter/trunk/debug-troubleshooter.php (15 diffs)

Unmodified Added Removed

debugger-troubleshooter/trunk/debug-troubleshooter.php — Tabular | Unified

r3459095 r3486202

```
4  4   * Plugin URI:        https://wordpress.org/plugins/debugger-troubleshooter
5  5   * Description:       A WordPress plugin for debugging and troubleshooting, allowing simulated plugin deactivation and theme switching without affecting the live site.
6      * Version:           1.3.2
   6   * Version:           1.4.0
7  7   * Author:            Jhimross
8  8   * Author URI:        https://profiles.wordpress.org/jhimross
...
22 22  * Define plugin constants.
23 23  */
24     define('DBGTBL_VERSION', '1.3.2');
   24  define('DBGTBL_VERSION', '1.4.0');
25 25  define('DBGTBL_DIR', plugin_dir_path(__FILE__));
26 26  define('DBGTBL_URL', plugin_dir_url(__FILE__));
...
80 80          add_action('admin_notices', array($this, 'troubleshooting_mode_notice'));
81 81          add_action('admin_bar_menu', array($this, 'admin_bar_exit_simulation'), 999);
   82
   83          // Include exit simulation script if active.
   84          add_action('wp_footer', array($this, 'print_exit_simulation_script'));
   85          add_action('admin_footer', array($this, 'print_exit_simulation_script'));
82 86      }
83 87
...
474 478     {
475 479         if (isset($_COOKIE[self::TROUBLESHOOT_COOKIE])) {
476             // phpcs:ignore WordPress.Security.ValidatedSanitizedInput.InputNotSanitized
477             $this->troubleshoot_state = json_decode(wp_unslash($_COOKIE[self::TROUBLESHOOT_COOKIE]), true);
478
479             if (!empty($this->troubleshoot_state)) {
   480           $token = sanitize_text_field(wp_unslash($_COOKIE[self::TROUBLESHOOT_COOKIE]));
   481           $sessions = get_option('dbgtbl_sessions', array());
   482
   483           if (isset($sessions[$token]) && is_array($sessions[$token])) {
   484               $this->troubleshoot_state = $sessions[$token];
   485
480 486               // Define DONOTCACHEPAGE to prevent caching plugins from interfering.
```

r3459095 r3486202

```
481 487                 if (!defined('DONOTCACHEPAGE')) {
... ...
486 492                 nocache_headers();
487 493
488     -               // Filter active plugins.
489     -               add_filter('option_active_plugins', array($this, 'filter_active_plugins'));
    494 +               // Filter active plugins. Note: The actual plugin deactivation happens via the MU plugin.
    495 +               add_filter('option_active_plugins', array($this, 'filter_active_plugins'), 0);
490 496                 if (is_multisite()) {
491     -                   add_filter('site_option_active_sitewide_plugins', array($this, 'filter_active_sitewide_plugins'));
    497 +                   add_filter('site_option_active_sitewide_plugins', array($this, 'filter_active_sitewide_plugins'), 0);
492 498                 }
493 499
... ...
695 701                 'timestamp' => time(),
696 702             );
    703 +
    704 +           $token = wp_generate_password(64, false);
    705 +           $sessions = get_option('dbgtbl_sessions', array());
    706 +           $sessions[$token] = $state;
    707 +           update_option('dbgtbl_sessions', $sessions);
    708 +
    709 +           // Create MU plugin drop-in to intercept early plugin loading
    710 +           $this->install_mu_plugin();
    711 +
697 712             // Set cookie with HttpOnly flag for security, and secure flag if site is HTTPS.
698     -           setcookie(self::TROUBLESHOOT_COOKIE, wp_json_encode($state), array(
    713 +           setcookie(self::TROUBLESHOOT_COOKIE, $token, array(
699 714                 'expires' => time() + DAY_IN_SECONDS,
700 715                 'path' => COOKIEPATH,
... ...
706 721             wp_send_json_success(array('message' => __('Troubleshooting mode activated.', 'debugger-troubleshooter')));
707 722         } else {
    723 +           $token = isset($_COOKIE[self::TROUBLESHOOT_COOKIE]) ? sanitize_text_field(wp_unslash($_COOKIE[self::TROUBLESHOOT_COOKIE])) : false;
    724 +           if ($token) {
    725 +               $sessions = get_option('dbgtbl_sessions', array());
    726 +               unset($sessions[$token]);
    727 +               update_option('dbgtbl_sessions', $sessions);
    728 +
    729 +               if (empty($sessions)) {
    730 +                   $this->remove_mu_plugin();
    731 +               }
    732 +           }
    733 +
708 734             // Unset the cookie to exit troubleshooting mode.
709 735             setcookie(self::TROUBLESHOOT_COOKIE, '', array(
... ...
761 787         );
762 788
763     -       // Set cookie with HttpOnly flag for security, and secure flag if site is HTTPS.
764     -       setcookie(self::TROUBLESHOOT_COOKIE, wp_json_encode($state), array(
765     -           'expires' => time() + DAY_IN_SECONDS,
766     -           'path' => COOKIEPATH,
767     -           'domain' => COOKIE_DOMAIN,
768     -           'samesite' => 'Lax',
769     -           'httponly' => true,
770     -           'secure' => is_ssl(),
```

r3459095 r3486202

```
771                 ));
    789         $token = isset($_COOKIE[self::TROUBLESHOOT_COOKIE]) ? sanitize_text_field(wp_unslash($_COOKIE[self::TROUBLESHOOT_COOKIE])) : false;
    790         if (!$token) {
    791             wp_send_json_error(array('message' => __('Troubleshooting session not found.', 'debugger-troubleshooter')));
    792         }
    793
    794         $sessions = get_option('dbgtbl_sessions', array());
    795         if (isset($sessions[$token])) {
    796             $sessions[$token] = $state;
    797             update_option('dbgtbl_sessions', $sessions);
    798         } else {
    799             wp_send_json_error(array('message' => __('Invalid troubleshooting session.', 'debugger-troubleshooter')));
    800         }
    801
772 802         wp_send_json_success(array('message' => __('Troubleshooting state updated successfully. Refreshing page...', 'debugger-troubleshooter')));
773 803     }
...
798 828     {
799 829         if (isset($_COOKIE[self::SIMULATE_USER_COOKIE])) {
800                 $this->simulated_user_id = (int) $_COOKIE[self::SIMULATE_USER_COOKIE];
801
802                 // Hook into determine_current_user to override the user ID.
803                 // Priority 20 ensures we run after most standard authentication checks.
804                 add_filter('determine_current_user', array($this, 'simulate_user_filter'), 20);
    830             $token = sanitize_text_field(wp_unslash($_COOKIE[self::SIMULATE_USER_COOKIE]));
    831             $sim_users = get_option('dbgtbl_sim_users', array());
    832
    833             if (isset($sim_users[$token])) {
    834                 $this->simulated_user_id = (int) $sim_users[$token];
    835
    836                 // Hook into determine_current_user to override the user ID.
    837                 // Priority 20 ensures we run after most standard authentication checks.
    838                 add_filter('determine_current_user', array($this, 'simulate_user_filter'), 20);
    839             }
805 840         }
806 841     }
...
878 913                 ),
879 914             ));
880
881                 // Add inline script for the exit action since we might be on the frontend
882                 // where our admin.js isn't enqueued, or we need a global handler.
883                 add_action('wp_footer', array($this, 'print_exit_simulation_script'));
884                 add_action('admin_footer', array($this, 'print_exit_simulation_script'));
885 915         }
886 916     }
...
891 921     public function print_exit_simulation_script()
892 922     {
    923         if (!$this->is_simulating_user()) {
    924             return;
    925         }
    926
    927         $nonce = wp_create_nonce('debug_troubleshoot_nonce');
    928         $exit_url = admin_url('admin-ajax.php?action=debug_troubleshoot_toggle_simulate_user&enable=0&nonce=' . $nonce);
893 929         ?>
894 930         <script type="text/javascript">
```

r3459095 r3486202

```
895 931             function debugTroubleshootExitSimulation() {
896 932                 if (confirm('<?php echo esc_js(__('Are you sure you want to exit User Simulation?', 'debugger-troubleshooter')); ?>')) {
897                     var data = new FormData();
898                     data.append('action', 'debug_troubleshoot_toggle_simulate_user');
899                     data.append('enable', '0');
900                     // We might not have the nonce available globally on frontend, so we rely on cookie check in backend mostly,
901                     // but for AJAX we need it. If we are on frontend, we might need to expose it.
902                     // For simplicity in this MVP, we'll assume admin-ajax is accessible.
903                     // SECURITY NOTE: In a real scenario, we should localize the nonce on wp_enqueue_scripts as well if we want frontend support.
904                     // For now, let's try to fetch it from a global if available, or just rely on the cookie clearing which is less secure but functional for a dev tool.
905                     // BETTER APPROACH: Use a dedicated endpoint or just a simple GET parameter that we intercept on init to clear the cookie.
906
907                     // Let's use a simple redirect to a URL that handles the exit.
908                     window.location.href = '<?php echo esc_url(admin_url('admin-ajax.php?action=debug_troubleshoot_toggle_simulate_user&enable=0')); ?>';
    933                 window.location.href = <?php echo wp_json_encode($exit_url); ?>;
909 934                 }
910 935             }
... ...
918 943     public function ajax_toggle_simulate_user()
919 944     {
920         // Note: For the "Exit" action via GET request (from Admin Bar), we might not have a nonce.
921         // Since this is a dev tool and we are just clearing a cookie, the risk is low, but ideally we'd check a nonce.
922         // For the "Enter" action (POST), we definitely check the nonce.
923
924         $is_post = isset($_SERVER['REQUEST_METHOD']) && 'POST' === $_SERVER['REQUEST_METHOD'];
925         if ($is_post) {
926             check_ajax_referer('debug_troubleshoot_nonce', 'nonce');
927         }
    945     check_ajax_referer('debug_troubleshoot_nonce', 'nonce');
928 946
929 947         if (!current_user_can('manage_options') && !$this->is_simulating_user()) {
... ...
935 953         $enable = isset($_REQUEST['enable']) ? (bool) $_REQUEST['enable'] : false;
936 954         $user_id = isset($_REQUEST['user_id']) ? (int) $_REQUEST['user_id'] : 0;
    955     $is_post = isset($_SERVER['REQUEST_METHOD']) && 'POST' === $_SERVER['REQUEST_METHOD'];
937 956
938 957         if ($enable && $user_id) {
    958             $token = wp_generate_password(64, false);
    959             $sim_users = get_option('dbgtbl_sim_users', array());
    960             $sim_users[$token] = $user_id;
    961             update_option('dbgtbl_sim_users', $sim_users);
    962
939 963             // Set cookie
940                 setcookie(self::SIMULATE_USER_COOKIE, $user_id, array(
    964             setcookie(self::SIMULATE_USER_COOKIE, $token, array(
941 965                 'expires' => time() + DAY_IN_SECONDS,
942 966                 'path' => COOKIEPATH,
... ...
946 970                 'secure' => is_ssl(),
947 971             ));
948                 wp_send_json_success(array('message' => __('User simulation activated. Reloading...', 'debugger-troubleshooter')));
    972             wp_send_json_success(array(
    973                 'message'  => __('User simulation activated. Redirecting...', 'debugger-troubleshooter'),
    974                 'redirect' => admin_url()
```

r3459095 r3486202

```
            ));
        } else {
            $token = isset($_COOKIE[self::SIMULATE_USER_COOKIE]) ? sanitize_text_field(wp_unslash($_COOKIE[self::SIMULATE_USER_COOKIE])) : false;
            if ($token) {
                $sim_users = get_option('dbgtbl_sim_users', array());
                unset($sim_users[$token]);
                update_option('dbgtbl_sim_users', $sim_users);
            }

            // Clear cookie
            setcookie(self::SIMULATE_USER_COOKIE, '', array(
...
        }
    }

    /**
     * Installs the MU plugin used to intercept active plugins before standard plugins are loaded.
     */
    private function install_mu_plugin()
    {
        $mu_dir = WPMU_PLUGIN_DIR;
        if (!is_dir($mu_dir)) {
            @mkdir($mu_dir, 0755, true);
        }

        $mu_file = $mu_dir . '/debugger-troubleshooter-mu.php';

        $mu_content = "<?php
/**
 * Plugin Name: Debugger & Troubleshooter (MU Plugin)
 * Description: Intercepts active plugins to apply troubleshooting mode correctly.
 * Version: 1.0
 * Author: Jhimross
 */

if (!defined('ABSPATH')) {
    exit;
}

// Ensure the token from cookie exists and maps to an active session.
if (isset(\$_COOKIE['wp_debug_troubleshoot_mode'])) {
    \$token = sanitize_text_field(wp_unslash(\$_COOKIE['wp_debug_troubleshoot_mode']));
    \$sessions = get_option('dbgtbl_sessions', array());

    if (isset(\$sessions[\$token]) && is_array(\$sessions[\$token])) {
        // Replace active plugins for this request
        add_filter('option_active_plugins', function (\$plugins) use (\$sessions, \$token) {
            if (isset(\$sessions[\$token]['plugins'])) {
                return \$sessions[\$token]['plugins'];
            }
            return \$plugins;
        }, 0);

        if (is_multisite()) {
            add_filter('site_option_active_sitewide_plugins', function (\$plugins) use (\$sessions, \$token) {
                if (isset(\$sessions[\$token]['sitewide_plugins'])) {
                    \$new_plugins = array();
                    foreach (\$sessions[\$token]['sitewide_plugins'] as \$plugin_file) {
                        \$new_plugins[\$plugin_file] = time();
                    }
```

r3459095 r3486202

```
                    return \$new_plugins;
                }
                return \$plugins;
            }, 0);
        }
    }
}
";
        @file_put_contents($mu_file, $mu_content);
    }

    /**
     * Removes the MU plugin when no longer needed.
     */
    private function remove_mu_plugin()
    {
        $mu_file = WPMU_PLUGIN_DIR . '/debugger-troubleshooter-mu.php';
        if (file_exists($mu_file)) {
            @unlink($mu_file);
        }
    }
}
```