



---

# Mastering the Art of WordPress: 13 Years of Advanced Tips and Tricks



Stanko Metodiev  
WordCamp Sofia, 2024

---

# About Stanko

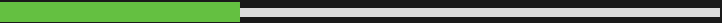
- CTO @ DevriX
- WordPress Core and community contributor
- (Co) Leader of the first WordPress meetup in Bulgaria - WordPress Sofia (WPBGUG)
- WordCamp Organizer and Lead Organizer of WordCamp Sofia 2023
- More than 13 years in developing complex SaaS platforms and multisite solutions
- Proud dad and a husband of princesses



Stanko Metodiev



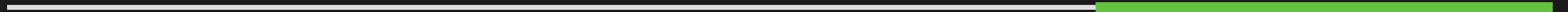
# Table of Contents

- A little bit of a backstory
  - Some code snippets here and there
  - Some WordPress tips and tricks
  - Some tools
  - Some personal thoughts and my journey so far
  - And I hope you'll learn something new today
- 

---

# The personal growth

From simply the blog owner to becoming a CTO



# The personal growth

★ Started university as both a joke and an experiment

# The personal growth

- ★ Started university as both a joke and an experiment
- ★ Met some amazing people and made connections

# The personal growth

- ★ Started university as both a joke and an experiment
- ★ Met some amazing people and made connections
- ★ Demonstrated organizational skills

# The personal growth

- ★ Started university as both a joke and an experiment
- ★ Met some amazing people and made connections
- ★ Demonstrated organizational skills
- ★ Wasn't the best student :D (not an actual advice)



# The personal growth

- ★ Started university as both a joke and an experiment
- ★ Met some amazing people and made connections
- ★ Demonstrated organizational skills
- ★ Wasn't the best student :D (not an actual advice)
- ★ Started as an intern

# The personal growth

- ★ Started university as both a joke and an experiment
- ★ Met some amazing people and made connections
- ★ Demonstrated organizational skills
- ★ Wasn't the best student :D (not an actual advice)
- ★ Started as an intern
- ★ Worked hard and learned daily

# The personal growth

- ★ Started university as both a joke and an experiment
- ★ Met some amazing people and made connections
- ★ Demonstrated organizational skills
- ★ Wasn't the best student :D (not an actual advice)
- ★ Started as an intern
- ★ Worked hard and learned daily
- ★ Made mistakes and improved by doing them

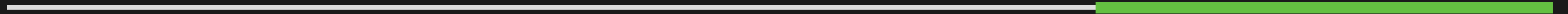
# The personal growth

- ★ Started university as both a joke and an experiment
- ★ Met some amazing people and made connections
- ★ Demonstrated organizational skills
- ★ Wasn't the best student :D (not an actual advice)
- ★ Started as an intern
- ★ Worked hard and learned daily
- ★ Made mistakes and improved by doing them
- ★ Didn't give up

Stanko Metodiev

---

# Beyond the Code: Professional Growth Path



# Open Source Community

---

- Build and share
- Write (technical) articles and share knowledge

- Mentor others
- Contribute to WordPress and Open-Source

- Speak at WordCamps and conferences
- Organize local meetups

# Technical Excellence

---

- Master your craft
- Technical expertise
- Balance perfection with pragmatism

- Strategic thinking
- Learning routine
- Time management

- Develop leadership skills
- Write *maintainable, scalable* code



**I Am Devloper**

@iamdevloper



I don't need you to tell me how  ~~fucking~~  good my code is, okay? I'm the one who writes it, I know how good it is. When designers code, they write  ~~shit~~ . Me, I write the gourmet scalable stuff because when I deploy it, I want it to work.



Let's get some (actual) advice,  
shall we?


---

# `wp_enqueue*` and Asset Optimizations

---

```
function enqueue_my_assets() {  
    wp_enqueue_style( 'single-post',  
        get_template_directory_uri() . '/css/single.min.css',  
        [],  
        filemtime( get_template_directory() . '/css/single.min.css' )  
    );  
  
    wp_enqueue_script( 'non-critical-script' );  
  
    // Preload critical assets  
    add_action( 'wp_head', function() {  
        echo '<link rel="stylesheet" href="critical-font.woff2">';  
    } );  
}  
add_action( 'wp_enqueue_scripts', 'enqueue_my_assets' );
```

```
function enqueue_my_assets() {  
    // Conditional loading  
    if ( is_single() ) {  
        wp_enqueue_style( 'single-post',  
            get_template_directory_uri() . '/css/single.min.css',  
            [],  
            filemtime( get_template_directory() . '/css/single.min.css' )  
        );  
    }  
  
    // Defer non-critical JS  
    wp_script_add_data( 'non-critical-script', 'defer', true );  
  
    // Preload critical assets  
    add_action( 'wp_head', function() {  
        echo '<link rel="preload" href="critical-font.woff2" as="font" type="font/woff2" crossorigin>';  
    } );  
}  
add_action( 'wp_enqueue_scripts', 'enqueue_my_assets' );
```



---

# The `query_posts()` trap

---



```
/**
 * BAD EXAMPLE: Using query_posts()
 * Why it's bad:
 * 1. Replaces the main query
 * 2. Breaks pagination
 * 3. Impacts performance
 * 4. Affects global variables
 */
query_posts( array(
    'post_type'      => 'project',
    'posts_per_page' => 10,
    'orderby'        => 'date',
    'order'           => 'DESC',
) );

if ( have_posts() ) {
    while ( have_posts() ) {
        the_post();
        // Loop content.
    }
}

wp_reset_query(); // Required, but still not good practice.
```

# Why **NOT** `query_posts()`?

- ★ Performance impact - replaces the primary query, `query_posts` results in two queries—one for the original and one for the override—leading to unnecessary database load
- ★ Risk of unexpected behavior - accidentally alter the main query in ways that interfere with themes and plugins expecting the default query structure, causing compatibility issues
- ★ Limited control: unlike `pre_get_posts`, `query_posts` doesn't allow developers to refine or manipulate the existing query
- ★ Best practice: WordPress recommends using `pre_get_posts` or custom `WP_Query` instances for safe and effective query modifications

---

`pre_get_posts()`

---



```
/**
 * Use Case: Modifying Main Query Properly
 * Instead of query_posts(), use pre_get_posts
 */
function modify_main_query( $query ) {
// Ensure we're not in admin and this is the main query.
    if ( ! is_admin() && $query->is_main_query() ) {
        if ( is_post_type_archive( 'project' ) ) {
            $query->set( 'posts_per_page', 10 );
            $query->set( 'orderby', 'date' );
            $query->set( 'order', 'DESC' );
        }
    }
}
add_action( 'pre_get_posts', 'modify_main_query' );
```

# Why `pre_get_posts()`?

- ★ Improves performance
- ★ Reduce unnecessary queries
- ★ Increased efficiency - load only relevant data, improving page speed
- ★ Tailor query results
- ★ It's just nice tool to have
- ★ There are other `pre_get_<something>`, check them out

---

# get\_posts()

---

```
$posts_args = array(  
    'post_type'      => 'project',  
    'posts_per_page' => 10,  
    'orderby'        => 'date',  
    'order'          => 'DESC',  
);  
  
$posts = get_posts( $posts_args );  
  
foreach ( $posts as $post ) {  
    setup_postdata( $post );  
    // Loop content.  
}  
  
wp_reset_postdata(); // Reset after using setup_postdata().
```

# Why and when to use `get_posts()`?

- ★ Best for simple queries
- ★ When you need just the post data
- ★ Better performance with small queries
- ★ Simple syntax
- ★ Always use `wp_reset_postdata()` after custom queries

---

# WP\_Query()

---



```
$custom_query_args = array(
    'post_type'      => 'project',
    'posts_per_page' => 10,
    'orderby'        => 'date',
    'order'          => 'DESC',
);

$custom_query = new WP_Query( $custom_query_args );

if ( $custom_query->have_posts() ) {
    while ( $custom_query->have_posts() ) {
        $custom_query->the_post();
        // Loop content.
    }
}

wp_reset_postdata(); // Reset custom query.
```

# Why and when to use **WP\_Query()**?

- ★ Full control over queries
- ★ Built-in compatibility
- ★ Flexible data retrieval
- ★ Enhanced performance options
- ★ Pagination support
- ★ And many more
- ★ Always use `wp_reset_postdata()` after custom queries



---

# Query optimizations


---

```
// Bad get_posts() Query
$all_posts = get_posts( array(
    'post_type'      => 'post',
    'posts_per_page' => -1, // Never do this*
));

// Bad WP_Query() Query
$all_posts_with_wp_query = new WP_Query( array(
    'post_type'      => 'post',
    'posts_per_page' => -1, // Never do this*
));
```

```
// Bad get_posts() Query
$all_posts = get_posts( array(
    'post_type'      => 'post',
    'posts_per_page' => -1, // Never do this*
));
```

```
// Bad WP_Query() Query
$all_posts_with_wp_query = new WP_Query( array(
    'post_type'      => 'post',
    'posts_per_page' => -1, // Never do this*
));
```



# Why you should be careful with **-1**?

- ★ Retrieve all posts without limitation
- ★ Avoids pagination
- ★ Useful for non-public data \*

Caution: be mindful when using **-1** on high-traffic, public-facing pages with a large dataset, as it can impact performance.

```
/**
 * PERFORMANCE TIP: Optimized WP_Query for better performance
 * Using specific fields and disabling unnecessary counts
 */
$optimized_query = new WP_Query( array(
    'post_type'           => 'project',
    'posts_per_page'      => 10,
    'orderby'             => 'date',
    'order'               => 'DESC',
    'no_found_rows'       => true, // Improves performance when no pagination.
    'fields'              => 'ids', // If you only need post IDs.
    'update_post_meta_cache' => false, // If you don't need meta data.
    'update_post_term_cache' => false, // If you don't need taxonomy terms.
) );
```



# WP\_Query optimizations

- ★ use `posts_per_page`
- ★ use `post_status` when applicable
- ★ use `no_found_rows` when you don't need pagination
- ★ use `update_post_meta_cache` argument if you don't need meta data
- ★ use `update_post_term_cache` argument when do don't need taxonomy
- ★ use `fields` when you need just IDs\*

---

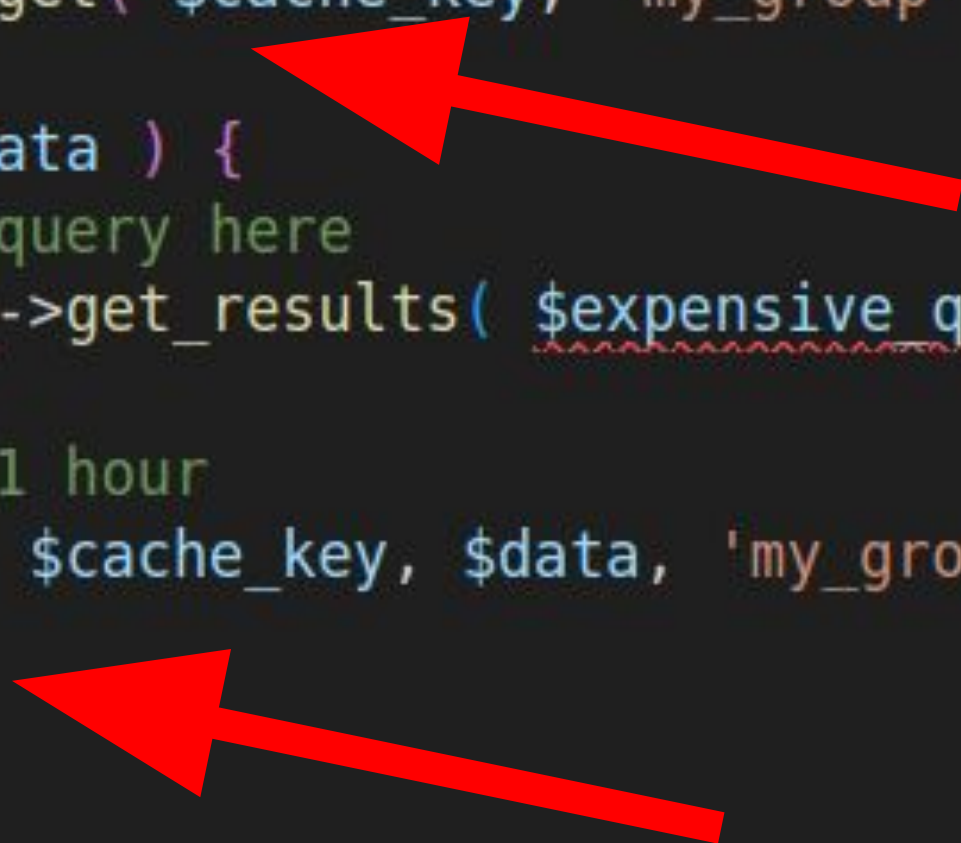
**Object Cache** is your friend

---

```
function get_expensive_data() {  
    $cache_key = 'expensive_data_' . md5( serialize( $args ) );  
  
    // Try to get cached data  
    $data = wp_cache_get( $cache_key, 'my_group' );  
  
    if ( false === $data ) {  
        // Expensive query here  
        $data = $wpdb->get_results( $expensive_query );  
  
        // Cache for 1 hour  
        wp_cache_set( $cache_key, $data, 'my_group', HOUR_IN_SECONDS );  
    }  
  
    return $data;  
}
```



```
function get_expensive_data() {  
    $cache_key = 'expensive_data_' . md5( serialize( $args ) );  
  
    // Try to get cached data  
    $data = wp_cache_get( $cache_key, 'my_group' );  
  
    if ( false === $data ) {  
        // Expensive query here  
        $data = $wpdb->get_results( $expensive_query );  
  
        // Cache for 1 hour  
        wp_cache_set( $cache_key, $data, 'my_group', HOUR_IN_SECONDS );  
    }  
  
    return $data;  
}
```



# Why and when to use **Object Cache**?

- ★ Improves performance - reduces the number of expensive database queries
- ★ Scales better with high traffic - caching frequently requested data prevents database overload during peak traffic
- ★ Increases user experience - faster response times improve the user experience
- ★ Flexible expiration

---

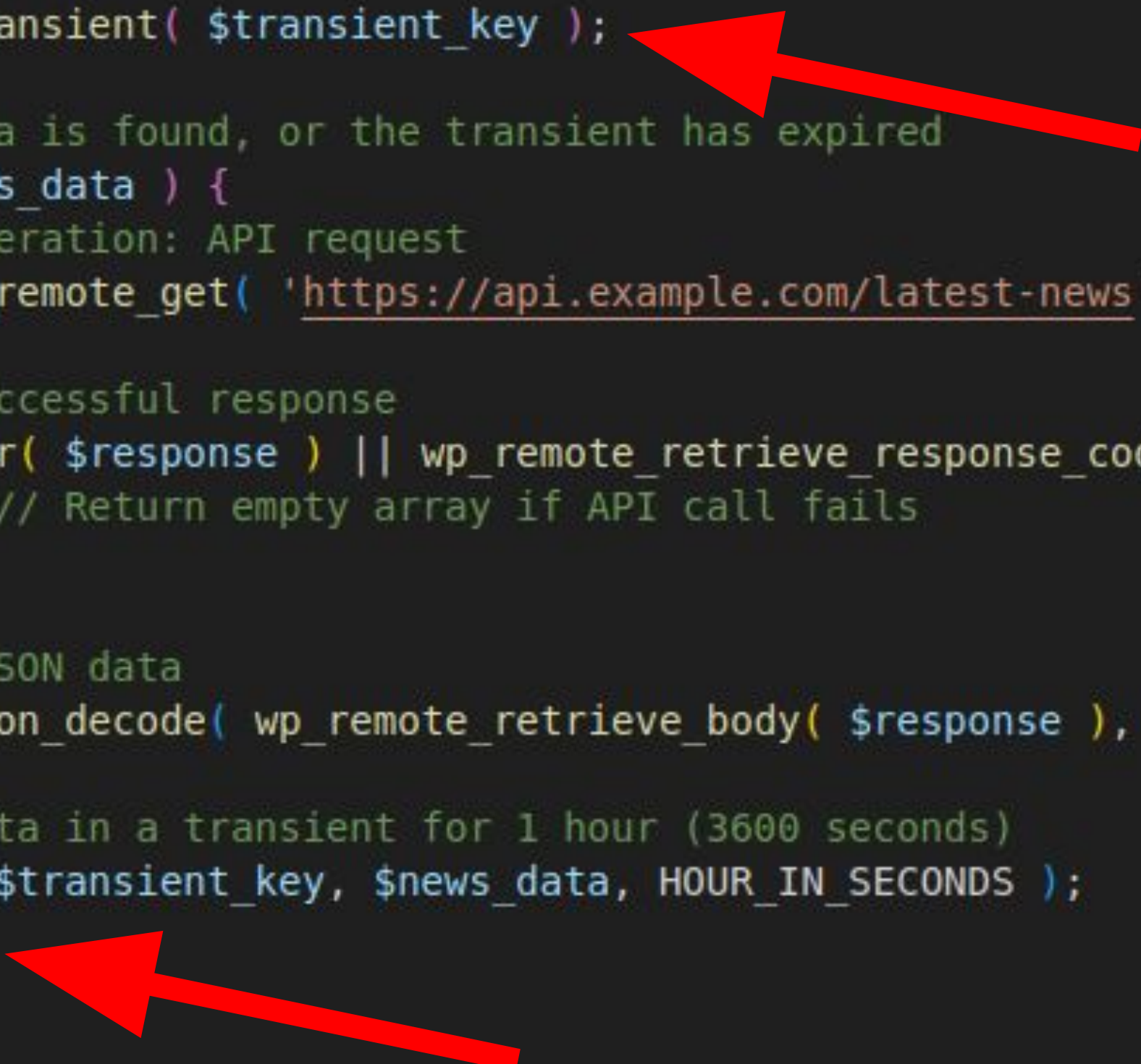
**Transients** are your friends

---

```
function get_latest_news() {  
    // Define a unique key for the transient  
    $transient_key = 'latest_news_data';  
  
    // Try to get cached data  
    $news_data = get_transient( $transient_key );  
  
    // If no cached data is found, or the transient has expired  
    if ( false === $news_data ) {  
        // Expensive operation: API request  
        $response = wp_remote_get( 'https://api.example.com/latest-news' );  
  
        // Check for successful response  
        if ( is_wp_error( $response ) || wp_remote_retrieve_response_code( $response ) !== 200 ) {  
            return []; // Return empty array if API call fails  
        }  
  
        // Decode the JSON data  
        $news_data = json_decode( wp_remote_retrieve_body( $response ), true );  
  
        // Cache the data in a transient for 1 hour (3600 seconds)  
        set_transient( $transient_key, $news_data, HOUR_IN_SECONDS );  
    }  
  
    return $news_data;  
}
```



```
function get_latest_news() {  
    // Define a unique key for the transient  
    $transient_key = 'latest_news_data';  
  
    // Try to get cached data  
    $news_data = get_transient( $transient_key );  
  
    // If no cached data is found, or the transient has expired  
    if ( false === $news_data ) {  
        // Expensive operation: API request  
        $response = wp_remote_get( 'https://api.example.com/latest-news' );  
  
        // Check for successful response  
        if ( is_wp_error( $response ) || wp_remote_retrieve_response_code( $response ) !== 200 ) {  
            return []; // Return empty array if API call fails  
        }  
  
        // Decode the JSON data  
        $news_data = json_decode( wp_remote_retrieve_body( $response ), true );  
  
        // Cache the data in a transient for 1 hour (3600 seconds)  
        set_transient( $transient_key, $news_data, HOUR_IN_SECONDS );  
    }  
  
    return $news_data;  
}
```



```
// Usage in a template or function
$latest_news = get_latest_news();
if ( ! empty( $latest_news ) ) {
    foreach ( $latest_news as $news_item ) {
        echo '<h2>' . esc_html( $news_item['title'] ) . '</h2>';
        echo '<p>' . esc_html( $news_item['summary'] ) . '</p>';
    }
}
```

# Why and when to use **Transients**?

- ★ transients cache expensive data (e.g., API responses) to prevent repetitive processing
- ★ Reduces server load
- ★ Efficient caching control and Ideal for temporary Data
- ★ Flexible storage - transients can be stored in the database or in-memory using object caching systems like Redis or Memcached
- ★ Better user experience - faster load times improve the user experience



---

**REST API** to the rescue

---

```

/**
 * Register custom REST API endpoint for post previews.
 *
 * @since 1.0.0
 *
 * @return void
 */
function register_post_preview_endpoint() {
    register_rest_route(
        'preview/v1',
        '/post/(?P<id>\d+)',
        array(
            'methods'           => WP_REST_Server::READABLE,
            'callback'          => 'get_post_preview_data',
            'permission_callback' => '__return_true',
            'args'              => array(
                'id' => array(
                    'validate_callback' => function( $param ) {
                        return is_numeric( $param );
                    },
                ),
            ),
        ),
    );
}

add_action( 'rest_api_init', 'register_post_preview_endpoint' );

```

```

/**
 * Get preview data for a specific post.
 *
 * @since 1.0.0
 *
 * @param WP_REST_Request $request Request object.
 * @return WP_REST_Response|WP_Error Response object or WP_Error.
 */
function get_post_preview_data( $request ) {
    $post_id = $request['id'];
    $post     = get_post( $post_id );

    if ( ! $post ) {
        return new WP_Error(
            'no_post',
            __( 'Post not found', 'text-domain' ),
            array( 'status' => 404 )
        );
    }

    $preview_data = array(
        'title'           => $post->post_title,
        'excerpt'         => wp_trim_words( get_the_excerpt( $post ), 20 ),
        'featured_image' => array(
            'full'         => get_the_post_thumbnail_url( $post_id, 'full' ),
            'thumbnail'    => get_the_post_thumbnail_url( $post_id, 'thumbnail' ),
            'medium'       => get_the_post_thumbnail_url( $post_id, 'medium' ),
        ),
        'author'          => get_the_author_meta( 'display_name', $post->post_author ),
        'date'            => get_the_date( 'F j, Y', $post ),
    );

    return rest_ensure_response( $preview_data );
}

```



← → ↻  https://domain.com/wp-json/preview/v1/post/39



▼ {

Raw

Parsed

```
{
  "title": "Deleniti quis sequi inventore dolorum.",
  "excerpt": "This is our content from the dashboard. Quas temporibus consequatur aut praesentium saepe cupiditate eligendi. Aut pariatur veritatis occaecati autem.&hellip;",
  "featured_image": {
    "full": "https://domain.com/wp-content/uploads/2024/10/wp_dummy_content_generator_56.jpg",
    "thumbnail": "https://domain.com/wp-content/uploads/2024/10/wp_dummy_content_generator_56-150x150.jpg",
    "medium": "https://domain.com/wp-content/uploads/2024/10/wp_dummy_content_generator_56-300x300.jpg"
  },
  "author": "Stanko Metodiev",
  "date": "October 23, 2024"
}
```

# Why and when to use **REST API**?

★ In our example:

- Building social media cards or creating preview widgets
- Creating link previews in headless WordPress setups
- Getting optimized post data for mobile apps

★ Headless applications

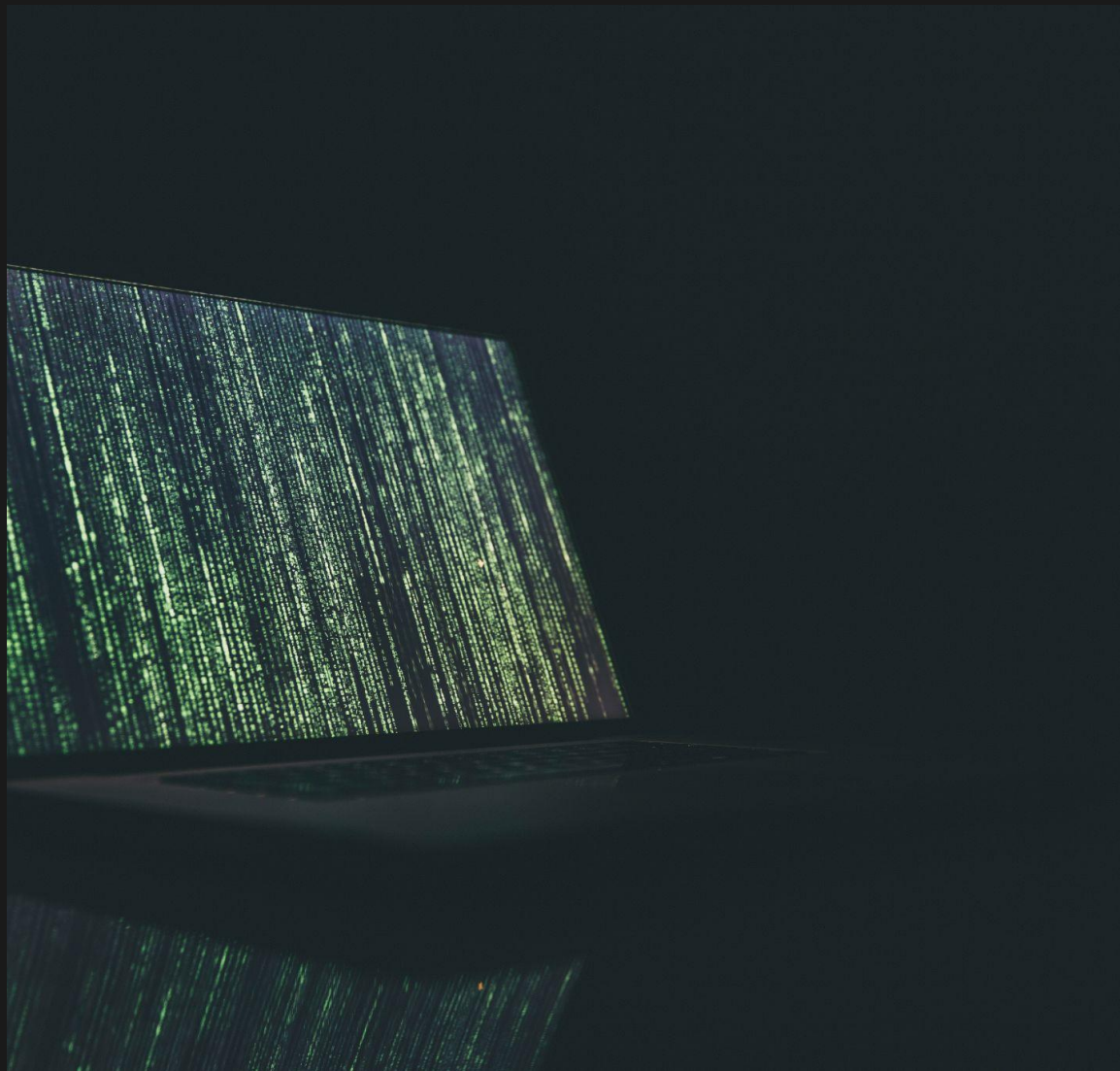
★ Data hub applications

★ 3rd party integrations

★ Many, many more

# What about security?

Stanko Metodiev



# Security 101

- Use **strong** and **unique** passwords. Full stop.
- Use 2FA/Passkeys
- Regular users audit
- Limited login attempts
- Solid hosting provider
- Regular updates
- Do not rely on **security plugins** \*
- Never ever **nulled** themes/plugins



```
/**
 * Disable WordPress Application Passwords
 * If you're not using the REST API
 */
add_filter( 'wp_is_application_passwords_available', '__return_false' );
```

```
/**
 * Remove WordPress Version
 * Hide version number from sources
 */
remove_action( 'wp_head', 'wp_generator' );
add_filter( 'the_generator', '__return_empty_string' );
```

```
/**
 * Disable XML-RPC and Pingbacks
 */
add_filter( 'xmlrpc_enabled', '__return_false' );
add_filter( 'wp_headers', function( $headers ) {
    unset( $headers['X-Pingback'] );
    return $headers;
} );
```

```
/**
 * Block Bad Queries
 * Prevent SQL injection attempts
 */
function block_bad_queries() {
    global $user_ID;

    if ( ! $user_ID ) {
        if ( preg_match( '/\..\./|\.\/|union|concat|echo|select|wp-config/i', $_SERVER['REQUEST_URI'] ) ) {
            wp_die( 'Invalid Request' );
        }
    }
}
add_action( 'init', 'block_bad_queries' );
```

be careful with the items you'll use here



# What about some tools?

# Tools

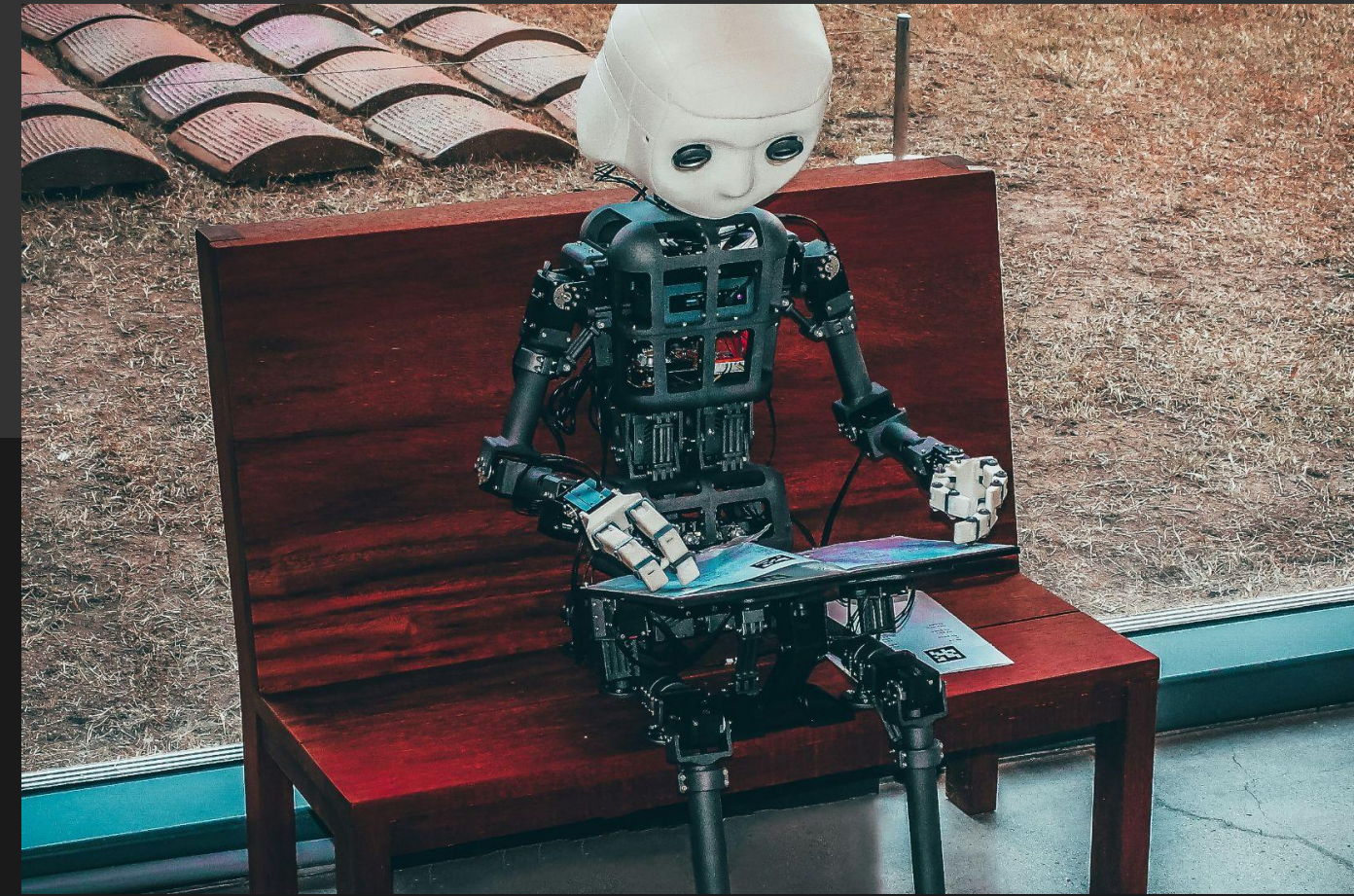
- ★ Personalized Text Editor/IDE
- ★ Shortcuts and alias
- ★ Trusted plugins
- ★ Templates for anything
- ★ Code snippets, libraries, and templates
- ★ Bash/SSH/Terminal
- ★ WP-CLI and Query Monitor
- ★ Chrome dev tools \*
- ★ Git/Version Control





# AI Tools

- ★ ChatGPT / Claude / Gemini
- ★ AI changes the way we work
- ★ AI can be your friend if you
- ★ know how to use it
- ★ AI website / WordPress builders
- ★ Make sure to check some of the AI-related talks today!







**Thank you!**

DevriX

Get in  
Touch

Twitter

@metodiew

LinkedIn

Stanko Metodiev

WordPress.org

metodiew

metodiew.com

Stanko Metodiev