

WORDPRESS HOOKS

Save & CRUD Hooks

Hooks in this category let you **react** when WordPress creates, saves, updates, publishes, transitions, or deletes content.



Control the post lifecycle **here.**



save_post

Fires when a post is saved.

```
<?php
add_action(
    'save_post',
    'my_handle_save_post'
);
```

01

Save content



wp_insert_post

Fires when a post is inserted or updated.

```
<?php
add_action(
    'wp_insert_post',
    'my_handle_insert_update'
);
```

02

Insert & update



transition_post_status

Fires when a post changes status.

```
<?php
add_action(
    'transition_post_status',
    'my_handle_status_change'
);
```

03

Status changes



before_delete_post

Fires just before a post is deleted.

```
<?php
add_action(
    'before_delete_post',
    'my_handle_before_delete'
);
```

04

Deletion lifecycle

4WP

4wp.dev/hooks/category/save-crud/

Save smarter. Change safely.

Use lifecycle hooks to control how WordPress content is saved and changed.

save_post()

The `save_post` action hook is fired when a `post`, `page`, or `custom post type` is saved in the database.

Fired after a post, page, or custom post is saved.



```
add_action( 'save_post',  
'my_save_post_data', 10, 3 );  
Save Custom Data
```



```
add_action( 'save_post',  
'my_update_post_views', 10, 3 );  
Update Post Views
```



```
add_action( 'save_post',  
'my_generate_excerpt', 10, 3 );  
Generate Excerpts
```



```
add_action( 'save_post',  
'my_clear_post_cache', 10, 3 );  
Clear Post Cache
```

```
<?php  
add_action( 'save_post',  
    'my_save_post_data', 10, 3 );  
  
function my_save_post_data( $post_id,  
    $post, $update ) {  
    // Avoid autosave, revisions, and  
    // unnecessary execution.  
    if ( wp_is_post_autosave( $post_id ) ||  
        wp_is_post_revision( $post_id ) ||  
        defined( 'DOING_AUTOSAVE' ) &&  
        DOING_AUTOSAVE ) {  
        return;  
    }  
    // Your code here.  
}  
?>
```



Save posts.

Run custom logic when content is saved. Ideal for storing `meta`, `syncing data`, and more.

wp_insert_post()

The `wp_insert_post` action hook is fired immediately after a `post`, `page`, or `custom post` type is inserted into the database, **before** any cache is cleaned.

*Fired after a post, page, or custom post type is **inserted** into the database.*



```
add_action( 'wp_insert_post',  
'my_update_post_meta', 10, 2 );
```

Update Post Meta



```
add_action( 'wp_insert_post',  
'my_log_new_post', 10, 2 );
```

Log New Posts



```
add_action( 'wp_insert_post',  
'my_set_custom_slug', 10, 2 );
```

Set Custom Slug



```
add_action( 'wp_insert_post',  
'my_sync_external_data', 10, 2 );
```

Sync External Data

```
<?php
```

```
add_action( 'wp_insert_post',  
'my_function', 10, 2 );
```

```
function my_function( $post_id, $post ) {  
    // $post_id: ID of the inserted post.  
    // $post: Post object.
```

```
    // Your code here.
```

```
}
```

```
?>
```

transition_post_status

The `transition_post_status` action hook is fired after a post's status changes from one to another, but **before** the change is saved in the database. It's perfect for reacting to any status transition.

*Fired after WordPress detects a status change but **before** it's saved.*



```
add_action( 'transition_post_status',  
'my_notify_status_change', 10, 3 );
```

Notify on Status Change



```
add_action( 'transition_post_status',  
'my_email_on_publish', 10, 3 );
```

Email on Publish



```
add_action( 'transition_post_status',  
'my_audit_status_change', 10, 3 );
```

Audit Status Changes



```
add_action( 'transition_post_status',  
'my_update_stats', 10, 3 );
```

Update Analytics / Stats



```
add_action( 'transition_post_status',  
'my_integration_hook', 10, 3 );
```

Trigger Integrations

```
<?php  
add_action( 'transition_post_status',  
    'my_notify_status_change', 10, 3 );  
  
function my_notify_status_change( $new_status,  
    $old_status, $post ) {  
    // Skip auto-saves and revisions.  
    if ( wp_is_post_autosave( $post ) ||  
        wp_is_post_revision( $post ) ) {  
        return;  
    }  
    // Act on specific transitions.  
    if ( 'publish' === $new_status &&  
        'draft' === $old_status ) {  
        // Your code here.  
    }  
}
```

React to status changes.

Use `transition_post_status` to run code **before** a new status is saved, whenever a post changes status.

before_delete_post()

Fires right before a post is permanently deleted.

The last chance to act while the post still exists.



Custom DB Cleanup

```
add_action( 'before_delete_post',  
    'my_db_cleanup', 10, 2 );
```

Database



Generated Files

```
add_action( 'before_delete_post',  
    'my_cleanup_files', 10, 2 );
```

Files



External Sync

```
add_action( 'before_delete_post',  
    'my_external_sync', 10, 2 );
```

Integrations



Audit Logging

```
add_action( 'before_delete_post',  
    'my_audit_log', 10, 2 );
```

Security



Related Data

```
add_action( 'before_delete_post',  
    'my_related_cleanup', 10, 2 );
```

Relationships



Last chance
before it's *gone*.

```
// In wp_delete_post()  
function wp_delete_post( $post_id, $force = false ) {  
    // ...  
    do_action( 'before_delete_post',  
        $post_id, get_post( $post_id ) );  
  
    // Continue with deletion...  
    // ...  
}
```

// Hook signature

```
do_action( 'before_delete_post',  
    int $post_id, WP_Post $post );
```



Permanent delete only –
not Trash.

Hook before it's *gone*.

Clean up related data and resources
before the post record disappears.