



- [\[link\]](#)
- [\[link\]](#)
- [\[link\]](#) API
- [\[link\]](#) API
- [\[link\]](#)



WordPress API WordPress

[API](#)

Options [API](#)/



API[API](#)



WordPress 2.7 设置 API 函数

add_settings_section()

add_settings_field()

设置 API

wp-admin/options.php

manage_options

设置 API

设置 API 函数 设置 API 函数

设置

设置 API 函数 WordPress 设置

设置

设置 API 函数 WordPress 设置 API 函数 WordPress 设置

设置

设置 WordPress API 函数 API 函数

- 设置 WordPress 设置 \$_POST 设置
- 设置
- 设置 WordPress 设置

设置

函数/方法	函数/方法
add_settings_section() add_settings_field()	add_settings_section() add_settings_field()
函数	函数
add_settings_error() get_settings_errors() settings_errors()	add_settings_error() get_settings_errors() settings_errors()

WP API

Settings

`register_setting()` `{ $wpdb->prefix }_options`

`add_settings_section()`

`add_settings_field()`

`register_setting()` `add_settings_*` `admin_init`

Example

```
register_setting(
    string $option_group,
    string $option_name,
    callable $sanitize_callback = ''
);
```

`register_setting()`

Example

```
add_settings_section(
    string $id,
    string $title,
    callable $callback,
    string $page
);
```

WordPress

`add_settings_section()`



```
add_settings_field(
    string $id,
    string $title,
    callable $callback,
    string $page,
    string $section = 'default',
    array $args = []
);
```

[add_settings_field\(\)](#)



```
function wporg_settings_init() {
    // register a new setting for "reading" page
    register_setting('reading', 'wporg_setting_name');

    // register a new section in the "reading" page
    add_settings_section(
        'wporg_settings_section',
        'WPOrg Settings Section', 'wporg_settings_section_callback',
        'reading'
    );

    // register a new field in the "wporg_settings_section" section, inside the "reading" page
    add_settings_field(
        'wporg_settings_field',
        'WPOrg Setting', 'wporg_settings_field_callback',
        'reading',
        'wporg_settings_section'
    );
}

/**
 * register wporg_settings_init to the admin_init action hook
 */
```

```

add_action('admin_init', 'wporg_settings_init');

/**
 * callback functions
 */

// section content cb
function wporg_settings_section_callback() {
    echo '<p>WPOrg Section Introduction.</p>';
}

// field content cb
function wporg_settings_field_callback() {
    // get the value of the setting we've registered with register_setting()
    $setting = get_option('wporg_setting_name');
    // output the field
    ?>
    <input type="text" name="wporg_setting_name" value="<?php echo isset( $setting ) ? esc_attr( $setting ) : "";
    ?>">
    <?php
}

```



[get_option\(\)](#) 



```

// Get the value of the setting we've registered with register_setting()
$setting = get_option('wporg_setting_name');

```

API

WordPress 1.0 API WordPress

{ \$wpdb->prefix }_options \$wpdb->prefix \$table_prefix wp-config.php

WordPress

```
// add a new option
add_option('wporg_custom_option', 'hello world!');

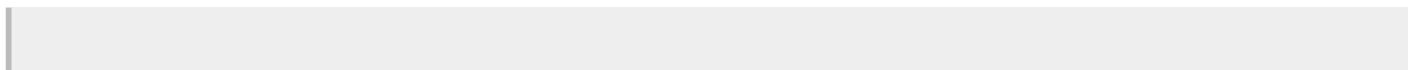
// get an option
$option = get_option('wporg_custom_option');
```

```
// array of options
$data_r = array('title' => 'hello world!', 1, false );

// add a new option
add_option('wporg_custom_option', $data_r);

// get an option
$options_r = get_option('wporg_custom_option');

// output the title
echo esc_html($options_r['title']);
```

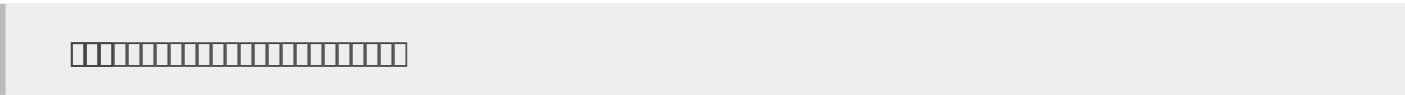




□□□□	□□□□	□□□□	□□□□
add_option()	get_option()	update_option()	delete_option()
add_site_option()	get_site_option()	update_site_option()	delete_site_option()



API



WPOrg wporg_options API API CRUD

```
/**
 * @internal never define functions inside callbacks.
 * these functions could be run multiple times; this would result in a fatal error.
 */

/**
 * custom option and settings
 */
function wporg_settings_init() {
    // Register a new setting for "wporg" page.
    register_setting( 'wporg', 'wporg_options' );

    // Register a new section in the "wporg" page.
    add_settings_section(
        'wporg_section_developers',
        __( 'The Matrix has you.', 'wporg' ), 'wporg_section_developers_callback',
        'wporg'
    );

    // Register a new field in the "wporg_section_developers" section, inside the "wporg" page.
    add_settings_field(
        'wporg_field_pill', // As of WP 4.6 this value is used only internally.
        // Use $args' label_for to populate the id inside the callback.
        __( 'Pill', 'wporg' ),
```

```

[]'wporg_field_pill_cb',
[]'wporg',
[]'wporg_section_developers',
[]array(
[]'label_for'      => 'wporg_field_pill',
[]'class'          => 'wporg_row',
[]'wporg_custom_data' => 'custom',
[])
);
}

/**
 * Register our wporg_settings_init to the admin_init action hook.
 */
add_action( 'admin_init', 'wporg_settings_init' );

/**
 * Custom option and settings:
 * - callback functions
 */

/**
 * Developers section callback function.
 *
 * @param array $args The settings array, defining title, id, callback.
 */
function wporg_section_developers_callback( $args ) {
    <?>
}

```

```

/**
 * @internal never define functions inside callbacks.
 * these functions could be run multiple times; this would result in a fatal error.
 */

/**

```

```

* custom option and settings
*/

function wporg_settings_init() {
    // Register a new setting for "wporg" page.
    register_setting( 'wporg', 'wporg_options' );

    // Register a new section in the "wporg" page.
    add_settings_section(
        'wporg_section_developers',
        __( 'The Matrix has you.', 'wporg' ), 'wporg_section_developers_callback',
        'wporg'
    );

    // Register a new field in the "wporg_section_developers" section, inside the "wporg" page.
    add_settings_field(
        'wporg_field_pill', // As of WP 4.6 this value is used only internally.
        // Use $args' label_for to populate the id inside the callback.
        __( 'Pill', 'wporg' ),
        'wporg_field_pill_cb',
        'wporg',
        'wporg_section_developers',
        array(
            'label_for' => 'wporg_field_pill',
            'class' => 'wporg_row',
            'wporg_custom_data' => 'custom',
        )
    );
}

/**
 * Register our wporg_settings_init to the admin_init action hook.
 */
add_action( 'admin_init', 'wporg_settings_init' );

/**
 * Custom option and settings:
 * - callback functions
 */

```

```

/**
 * Developers section callback function.
 *
 * @param array $args The settings array, defining title, id, callback.
 */
function wporg_section_developers_callback( $args ) {
    <?>
    <p id="<?php echo esc_attr( $args['id'] ); ?>"><?php esc_html_e( 'Follow the white rabbit.', 'wporg' ); ?></p>
    <?php
}

/**
 * Pill field callback function.
 *
 * WordPress has magic interaction with the following keys: label_for, class.
 * - the "label_for" key value is used for the "for" attribute of the <label>.
 * - the "class" key value is used for the "class" attribute of the <tr> containing the field.
 * Note: you can add custom key value pairs to be used inside your callbacks.
 *
 * @param array $args
 */
function wporg_field_pill_cb( $args ) {
    // Get the value of the setting we've registered with register_setting()
    $options = get_option( 'wporg_options' );
    <?>
    <select
        <div id="<?php echo esc_attr( $args['label_for'] ); ?>"
        <div data-custom="<?php echo esc_attr( $args['wporg_custom_data'] ); ?>"
        <div name="wporg_options[<?php echo esc_attr( $args['label_for'] ); ?>]">
            <option value="red" <?php echo isset( $options[ $args['label_for'] ] ) ? ( selected( $options[ $args['label_for'] ], 'red', false ) ) : ( '' ); ?>>
            <div <?php esc_html_e( 'red pill', 'wporg' ); ?>
        </option>
        <div <option value="blue" <?php echo isset( $options[ $args['label_for'] ] ) ? ( selected( $options[ $args['label_for'] ], 'blue', false ) ) : ( '' ); ?>>
            <div <?php esc_html_e( 'blue pill', 'wporg' ); ?>
        </option>
    </select>
    <p class="description">

```

```

<?php esc_html_e( 'You take the blue pill and the story ends. You wake in your bed and you believe whatever
you want to believe.', 'wporg' ); ?>

</p>
<p class="description">
<?php esc_html_e( 'You take the red pill and you stay in Wonderland and I show you how deep the rabbit-hole
goes.', 'wporg' ); ?>

</p>
<?php
}

/**
 * Add the top level menu page.
 */
function wporg_options_page() {
    add_menu_page(
        'WPOrg',
        'WPOrg Options',
        'manage_options',
        'wporg',
        'wporg_options_page_html'
    );
}

/**
 * Register our wporg_options_page to the admin_menu action hook.
 */
add_action( 'admin_menu', 'wporg_options_page' );

/**
 * Top level menu callback function
 */
function wporg_options_page_html() {
    // check user capabilities
    if ( ! current_user_can( 'manage_options' ) ) {
        return;
    }

    // add error/update messages

```

```

// check if the user have submitted the settings
// WordPress will add the "settings-updated" $_GET parameter to the url
if ( isset( $_GET['settings-updated'] ) ) {
    // add settings saved message with the class of "updated"
    add_settings_error( 'wporg_messages', 'wporg_message', __( 'Settings Saved', 'wporg' ), 'updated' );
}

// show error/update messages
settings_errors( 'wporg_messages' );

?>
<div class="wrap">
    <h1><?php echo esc_html( get_admin_page_title() ); ?></h1>
    <form action="options.php" method="post">
        <?php
            // output security fields for the registered setting "wporg"
            settings_fields( 'wporg' );
            // output setting sections and their fields
            // (sections are registered for "wporg", each field is registered to a specific section)
            do_settings_sections( 'wporg' );
            // output save settings button
            submit_button( 'Save Settings' );
        ?>
    </form>
</div>
<?php
}

```