□□□□□□

□□

- Complete Example

□□□□□□□□□□□□□□□□□□□□ □□□□□□ □□□□□ □ □□□□ □□□□ API □ □□ □□ API□

[alert]□□□□□□□□□□□□□□□□□□□□□□□□□□□□ [/alert]

□□□□□□□□□□□□□□□□□□□

Complete Example

Complete example which adds a Top-Level Menu named `WPOrg`, registers a custom option named `wporg_options` and performs the CRUD (create, read, update, delete) logic using Settings API and Options API (including showing error/update messages).

```
/**
 * @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 callbakc 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
    id="<?php echo esc_attr( $args['label_for'] ); ?>"
    data-custom="<?php echo esc_attr( $args['wporg_custom_data'] ); ?>"
    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 ) ) : ( '' ); ?>>
      <?php esc_html_e( 'red pill', 'wporg' ); ?>
    </option>
    <option value="blue" <?php echo isset( $options[ $args['label_for'] ] ) ? ( selected(
$options[ $args['label_for'] ], 'blue', false ) ) : ( '' ); ?>>
      <?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
}
```

□□□ #2
□ □□□ □□□ 2026-03-26 17:08:39 UTC
□ □□□ □□□ 2026-03-26 17:08:39 UTC