简单示例

目录

- 文件夹结构_____
- Plugin Files
- The Result

让我们构建一个为 Elementor 添加新控件的插件。该控件将显示一个可选择货币的下拉字段。

文件夹结构

该插件将包含三个文件。主文件将注册控件和 JS 文件。为了测试控件，我们将创建一个简单的测试小部件。

```
elementor-currency-control/
|
├─ controls/
|  └─ currency.php
|
├─ widgets/
|  └─ currency-widget.php
|
└─ elementor-currency-control.php
```

Plugin Files

elementor-currency-control.php

```
<?php
/**
 * Plugin Name: Elementor Currency Control
 * Description: Add new Elementor control for currencies selection.
```

```
 * Plugin URI:  https://elementor.com/
 * Version:     1.0.0
 * Author:      Elementor Developer
 * Author URI:  https://developers.elementor.com/
 * Text Domain: elementor-currency-control
 *
 * Requires Plugins: elementor
 * Elementor tested up to: 3.25.0
 * Elementor Pro tested up to: 3.25.0
 */

if ( ! defined( 'ABSPATH' ) ) {
	exit; // Exit if accessed directly.
}

/**
 * Register Currency Control.
 *
 * Include control file and register control class.
 *
 * @since 1.0.0
 * @param \Elementor\Controls_Manager $controls_manager Elementor controls manager.
 * @return void
 */
function register_currency_control( $controls_manager ) {

	require_once( __DIR__ . '/controls/currency.php' );

    $controls_manager->register( new \Elementor_Currency_Control() );

}
add_action( 'elementor/controls/register', 'register_currency_control' );

/**
 * Register Currency Widget.
 *
 * Include widget file and register widget class.
 *
 * @since 1.0.0
 * @param \Elementor\Widgets_Manager $widgets_manager Elementor widgets manager.
```

```
 * @return void
 */
function register_currency_widget( $widgets_manager ) {

	require_once( __DIR__ . '/widgets/currency-widget.php' );

	$widgets_manager->register( new \Elementor_Currency_Widget() );

}
add_action( 'elementor/widgets/register', 'register_currency_widget' );
```

controls/currency.php

```
<?php
/**
 * Elementor currency control.
 *
 * A control for displaying a select field with the ability to choose currencies.
 *
 * @since 1.0.0
 */
class Elementor_Currency_Control extends \Elementor\Base_Data_Control {

	/**
	 * Get currency control type.
	 *
	 * Retrieve the control type, in this case `currency`.
	 *
	 * @since 1.0.0
	 * @access public
	 * @return string Control type.
	 */
	public function get_type(): string {
		return 'currency';
	}

	/**
	 * Get currencies.
	 *
	 * Retrieve all the available currencies.
```

```
	 *
	 * @since 1.0.0
	 * @access public
	 * @static
	 * @return array Available currencies.
	 */
	public static function get_currencies(): array {
		return [
			'USD' => 'USD ($)',
			'EUR' => 'EUR (€)',
			'GBP' => 'GBP (£)',
			'JPY' => 'JPY (¥)',
			'ILS' => 'ILS (₪)',
		];
	}

	/**
	 * Get currency control default settings.
	 *
	 * Retrieve the default settings of the currency control. Used to return
	 * the default settings while initializing the currency control.
	 *
	 * @since 1.0.0
	 * @access protected
	 * @return array Currency control default settings.
	 */
	protected function get_default_settings(): array {
		return [
			'currencies' => self::get_currencies()
		];
	}

	/**
	 * Get currency control default value.
	 *
	 * Retrieve the default value of the currency control. Used to return the
	 * default value while initializing the control.
	 *
	 * @since 1.0.0
	 * @access public
```

```
	 * @return array Currency control default value.
	 */
	public function get_default_value(): string {
		return 'EUR';
	}

	/**
	 * Render currency control output in the editor.
	 *
	 * Used to generate the control HTML in the editor using Underscore JS
	 * template. The variables for the class are available using `data` JS
	 * object.
	 *
	 * @since 1.0.0
	 * @access public
	 */
	public function content_template(): void {
		$control_uid = $this->get_control_uid();
		?>
		<div class="elementor-control-field">

			<# if ( data.label ) {#>
			<label for="<?php echo $control_uid; ?>" class="elementor-control-title">{{{ data.label
			}}}</label>
			<# } #>

			<div class="elementor-control-input-wrapper">
				<select id="<?php echo $control_uid; ?>" data-setting="{{ data.name }}">
					<option value=""><?php echo esc_html__( 'Select currency', 'elementor-currency-control' );
					?></option>
					<# _.each( data.currencies, function( currency_label, currency_value ) { #>
					<option value="{{ currency_value }}">{{{ currency_label }}}</option>
					<# } ); #>
				</select>
			</div>

		</div>

		<# if ( data.description ) { #>
		<div class="elementor-control-field-description">{{{ data.description }}}</div>
```

```
		<# } #>
		<?php
	}

}
```

widgets/currency-widget.php

```
<?php
if ( ! defined( 'ABSPATH' ) ) {
	exit; // Exit if accessed directly.
}

/**
 * Elementor Currency Widget.
 *
 * Elementor widget that uses the currency control.
 *
 * @since 1.0.0
 */
class Elementor_Currency_Widget extends \Elementor\Widget_Base {

	/**
	 * Get widget name.
	 *
	 * Retrieve currency widget name.
	 *
	 * @since 1.0.0
	 * @access public
	 * @return string Widget name.
	 */
	public function get_name(): string {
		return 'currency';
	}

	/**
	 * Get widget title.
	 *
	 * Retrieve currency widget title.
```

```
	 *
	 * @since 1.0.0
	 * @access public
	 * @return string Widget title.
	 */
	public function get_title(): string {
		return esc_html__( 'Currency', 'elementor-currency-control' );
	}

	/**
	 * Get widget icon.
	 *
	 * Retrieve currency widget icon.
	 *
	 * @since 1.0.0
	 * @access public
	 * @return string Widget icon.
	 */
	public function get_icon(): string {
		return 'eicon-cart-medium';
	}

	/**
	 * Get widget categories.
	 *
	 * Retrieve the list of categories the currency widget belongs to.
	 *
	 * @since 1.0.0
	 * @access public
	 * @return array Widget categories.
	 */
	public function get_categories(): array {
		return [ 'general' ];
	}

	/**
	 * Get widget keywords.
	 *
	 * Retrieve the list of keywords the currency widget belongs to.
```

```
	 *
	 * @since 1.0.0
	 * @access public
	 * @return array Widget keywords.
	 */
	public function get_keywords(): array {
		return [ 'currency', 'currencies' ];
	}

	/**
	 * Get custom help URL.
	 *
	 * Retrieve a URL where the user can get more information about the widget.
	 *
	 * @since 1.0.0
	 * @access public
	 * @return string Widget help URL.
	 */
	public function get_custom_help_url(): string {
		return 'https://developers.elementor.com/docs/widgets/';
	}

	/**
	 * Whether the widget requires inner wrapper.
	 *
	 * Determine whether to optimize the DOM size.
	 *
	 * @since 1.0.0
	 * @access public
	 * @return bool Whether to optimize the DOM size.
	 */
	public function has_widget_inner_wrapper(): bool {
		return false;
	}

	/**
	 * Whether the element returns dynamic content.
	 *
	 * Determine whether to cache the element output or not.
```

```
	 *
	 * @since 1.0.0
	 * @access protected
	 * @return bool Whether to cache the element output.
	 */
	protected function is_dynamic_content(): bool {
		return false;
	}

	/**
	 * Register currency widget controls.
	 *
	 * Add input fields to allow the user to customize the widget settings.
	 *
	 * @since 1.0.0
	 * @access protected
	 */
	protected function register_controls(): void {

		$this->start_controls_section(
			'content_section',
			[
				'label' => esc_html__( 'Content', 'elementor-currency-control' ),
				'tab' => \Elementor\Controls_Manager::TAB_CONTENT,
			]
		);

		$this->add_control(
			'price',
			[
				'label' => esc_html__( 'Price', 'elementor-currency-control' ),
				'type' => \Elementor\Controls_Manager::NUMBER,
				'default' => 100,
			]
		);

		$this->add_control(
			'price_currency',
			[
```

```
        'label' => esc_html__( 'Currency', 'elementor-currency-control' ),
        'type' => 'currency',
    ]
  );

  $this->end_controls_section();

}

/**
 * Render currency widget output on the frontend.
 *
 * Written in PHP and used to generate the final HTML.
 *
 * @since 1.0.0
 * @access protected
 */
protected function render(): void {
  $settings = $this->get_settings_for_display();

  if ( empty( $settings['price_currency'] ) ) {
    return;
  }

  echo $settings['price_currency'] . ' ' . $settings['price'];
}

}
```

The Result

#2
 2026-03-26 17:08:44 UTC
 2026-03-26 17:08:44 UTC