Advanced Example

目录

- 文件夹结构
- Plugin Files
- 结果

For a more advanced example we will create a widget with a repeater field, to allow the user to generate a list with items. It will also have a conditional field display when choosing "other" marker type, to display a new control that let's the user enter a custom marker.

文件夹结构

该插件将包含两个文件：一个用于小部件，另一个主文件用于注册小部件。

```
elementor-list-widget/
|
├─ widgets/
|  └─ list-widget.php
|
└─ elementor-list-widget.php
```

Plugin Files

elementor-list-widget.php

```
<?php
/**
 * Plugin Name: Elementor List Widget
 * Description: List widget for Elementor.
```

```
 * Plugin URI:  https://elementor.com/
 * Version:     1.0.0
 * Author:      Elementor Developer
 * Author URI:  https://developers.elementor.com/
 * Text Domain: elementor-list-widget
 *
 * 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 List 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_list_widget( $widgets_manager ) {

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

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

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

widgets/list-widget.php

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

```
/**
 * Elementor List Widget.
 *
 * Elementor widget that inserts an embbedable content into the page, from any given URL.
 *
 * @since 1.0.0
 */
class Elementor_List_Widget extends \Elementor\Widget_Base {

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

	/**
	 * Get widget title.
	 *
	 * Retrieve list widget title.
	 *
	 * @since 1.0.0
	 * @access public
	 * @return string Widget title.
	 */
	public function get_title(): string {
		return esc_html__( 'List', 'elementor-list-widget' );
	}

	/**
	 * Get widget icon.
	 *
```

```
	 * Retrieve list widget icon.
	 *
	 * @since 1.0.0
	 * @access public
	 * @return string Widget icon.
	 */
	public function get_icon(): string {
		return 'eicon-bullet-list';
	}

	/**
	 * Get widget categories.
	 *
	 * Retrieve the list of categories the list 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 list widget belongs to.
	 *
	 * @since 1.0.0
	 * @access public
	 * @return array Widget keywords.
	 */
	public function get_keywords(): array {
		return [ 'list', 'lists', 'ordered', 'unordered' ];
	}

	/**
	 * 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/';
	}

	/**
	 * Get widget promotion data.
	 *
	 * Retrieve the widget promotion data.
	 *
	 * @since 1.0.0
	 * @access protected
	 * @return array Widget promotion data.
	 */
	protected function get_upsale_data(): array {
		return [
			'condition' => true,
			'image' => esc_url( ELEMENTOR_ASSETS_URL . 'images/go-pro.svg' ),
			'image_alt' => esc_attr__( 'Upgrade', 'elementor-list-widget' ),
			'title' => esc_html__( 'Promotion heading', 'elementor-list-widget' ),
			'description' => esc_html__( 'Get the premium version of the widget with additional styling
capabilities.', 'elementor-list-widget' ),
			'upgrade_url' => esc_url( 'https://example.com/upgrade-to-pro/' ),
			'upgrade_text' => esc_html__( 'Upgrade Now', 'elementor-list-widget' ),
		];
	}

	/**
	 * 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 list 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__( 'List Content', 'elementor-list-widget' ),
				'tab' => \Elementor\Controls_Manager::TAB_CONTENT,
			]
		);

		/* Start repeater */
```

```
□□$repeater = new \Elementor\Repeater();

□□$repeater->add_control(
□□□'text',
□□□[
□□□□'label' => esc_html__( 'Text', 'elementor-list-widget' ),
□□□□'type' => \Elementor\Controls_Manager::TEXT,
□□□□'placeholder' => esc_html__( 'List Item', 'elementor-list-widget' ),
□□□□'default' => esc_html__( 'List Item', 'elementor-list-widget' ),
□□□□'label_block' => true,
□□□□'dynamic' => [
□□□□□'active' => true,
□□□□],
□□□]
□□);

□□$repeater->add_control(
□□□'link',
□□□[
□□□□'label' => esc_html__( 'Link', 'elementor-list-widget' ),
□□□□'type' => \Elementor\Controls_Manager::URL,
□□□□'dynamic' => [
□□□□□'active' => true,
□□□□],
□□□]
□□);

□□/* End repeater */

$this->add_control(
□□□'list_items',
□□□[
□□□□'label' => esc_html_□□□', 'elementor-list-widget' ),
□□□□'type' => \Elementor\Controls_Manager::REPEATER,
□□□□'fields' => $repeater->get_controls(),          □□*□□□□□□ */
□□□□'default' => [
□□□□□[
□□□□□□'text' => esc_html□□□(#1', 'elementor-list-widget' ),
□□□□□□'link' => '',
```

□□□□□□],
□□□□□□[
□□□□□□□'text' => esc_html□□□□(#2', 'elementor-list-widget'),
□□□□□□□'link' => '',
□□□□□□],
□□□□□□[
□□□□□□□'text' => esc_html□□□□(#3', 'elementor-list-widget'),
□□□□□□□'link' => '',
□□□□□□],
□□□□],
□□□□'title_field' => '{{{ text }}}',
□□□]
□□);

□□$this->end_controls_section();

□□$this->start_controls_section(
□□□'marker_section',
□□□[
□□□□'label' => esc_html_□□□□', 'elementor-list-widget'),
□□□□'tab' => \Elementor\Controls_Manager::TAB_CONTENT,
□□□]
□□);

□□$this->add_control(
□□□'marker_type',
□□□[
□□□□'label' => esc_html_□□□□', 'elementor-list-widget'),
□□□□'type' => \Elementor\Controls_Manager::CHOOSE,
□□□□'options' => [
□□□□□'ordered' => [
□□□□□□'title' => esc_html□□□□□', 'elementor-list-widget'),
□□□□□□'icon' => 'eicon-editor-list-ol',
□□□□□],
□□□□□'unordered' => [
□□□□□□'title' => esc_html□□□□□', 'elementor-list-widget'),
□□□□□□'icon' => 'eicon-editor-list-ul',
□□□□□],
□□□□□'other' => [

```
□□□□□□□'title' => esc_html□□□□('', 'elementor-list-widget' ),
□□□□□□□'icon' => 'eicon-edit',
□□□□□□],
□□□□□],
□□□□□'default' => 'ordered',
□□□□□'toggle' => false,
□□□□]
□□);

□□$this->add_control(
□□□□'marker_content',
□□□□[
□□□□□'label' => esc_html_□□('', 'elementor-list-widget' ),
□□□□□'type' => \Elementor\Controls_Manager::TEXT,
□□□□□'placeholder' => esc_html_□□('□□□□□□', 'elementor-list-widget' ),
□□□□□'default' □□,'
□□□□□'condition' => [
□□□□□□'marker_type[value]' => 'other',
□□□□□],
□□□□□'selectors' => [
□□□□□□'{{WRAPPER}} .elementor-list-widget-text::marker' => 'content: "{{VALUE}}";',
□□□□□],
□□□□]
□□);

□□$this->end_controls_section();

□□$this->start_controls_section(
□□□□'style_content_section',
□□□□[
□□□□□'label' => esc_html_□□('', 'elementor-list-widget' ),
□□□□□'tab' => \Elementor\Controls_Manager::TAB_STYLE,
□□□□]
□□);

□□$this->add_control(
□□□□'title_color',
□□□□[
□□□□□'label' => esc_html_□□('', 'elementor-list-widget' ),
```

```
□□□□'type' => \Elementor\Controls_Manager::COLOR,
□□□□'selectors' => [
□□□□□'{{WRAPPER}} .elementor-list-widget-text' => 'color: {{VALUE}};',
□□□□□'{{WRAPPER}} .elementor-list-widget-text > a' => 'color: {{VALUE}};',
□□□□],
□□□]
□□);

□□$this->add_group_control(
□□□\Elementor\Group_Control_Typography::get_type(),
□□□[
□□□□'name' => 'icon_typography',
□□□□'selector' => '{{WRAPPER}} .elementor-list-widget-text, {{WRAPPER}} .elementor-list-widget-
text > a',
□□□]
□□);

□□$this->add_group_control(
□□□\Elementor\Group_Control_Text_Shadow::get_type(),
□□□[
□□□□'name' => 'text_shadow',
□□□□'selector' => '{{WRAPPER}} .elementor-list-widget-text',
□□□]
□□);

$this->end_controls_section();

□□$this->start_controls_section(
□□□'style_marker_section',
□□□[
□□□□'label' => esc_html_□('□□□', 'elementor-list-widget' ),
□□□□'tab' => \Elementor\Controls_Manager::TAB_STYLE,
□□□]
□□);

□□$this->add_control(
□□□'marker_color',
□□□[
□□□□'label' => esc_html_□('□', 'elementor-list-widget' ),
```

```
        'type' => \Elementor\Controls_Manager::COLOR,
        'selectors' => [
            '{{WRAPPER}} .elementor-list-widget-text::marker' => 'color: {{VALUE}};',
        ],
    ]
);

$this->add_control(
    'marker_spacing',
    [
        'label' => esc_html__( 'Spacing', 'elementor-list-widget' ),
        'type' => \Elementor\Controls_Manager::SLIDER,
        'size_units' => [ 'px', 'em', 'rem', 'custom' ],
        'range' => [
            'px' => [
                'min' => 0,
                'max' => 100,
            ],
            'em' => [
                'min' => 0,
                'max' => 10,
            ],
            'rem' => [
                'min' => 0,
                'max' => 10,
            ],
        ],
        'default' => [
            'unit' => 'px',
            'size' => 40,
        ],
        'selectors' => [
            // '{{WRAPPER}} .elementor-list-widget' => 'padding-left: {{SIZE}}{{UNIT}};',
            '{{WRAPPER}} .elementor-list-widget' => 'padding-inline-start: {{SIZE}}{{UNIT}};',
        ],
    ]
);

$this->end_controls_section();
```

```
□}

□/**
□ *□□□□□□□□□□□□□□□□
□*
□ *□□ PHP □□□□□□□□□□□ HTML□
□*
□ * @since 1.0.0
□ * @access protected
□ */
□protected function render(): void {
□□$settings = $this->get_settings_for_display();
□□$html_tag = [
□□□'ordered' => 'ol',
□□□'unordered' => 'ul',
□□□'other' => 'ul',
□□];
□□$this->add_render_attribute( 'list', 'class', 'elementor-list-widget' );
□□?>
□□<<?php echo $html_tag[ $settings['marker_type'] ]; ?> <?php $this-
>print_render_attribute_string( 'list' ); ?>>
□□□<?php
□□□foreach ( $settings['list_items'] as $index => $item ) {
□□□□$repeater_setting_key = $this->get_repeater_setting_key( 'text', 'list_items', $index );
□□□□$this->add_render_attribute( $repeater_setting_key, 'class', 'elementor-list-widget-text' )
□□□□$this->add_inline_editing_attributes( $repeater_setting_key );
□□□□?>
□□□□<li <?php $this->print_render_attribute_string( $repeater_setting_key ); ?>>
□□□□□<?php
□□□□□$title = $settings['list_items'][$index]['text'];

□□□□□if ( ! empty( $item['link']['url'] ) ) {
□□□□□□$this->add_link_attributes( "link_{$index}", $item['link'] );
□□□□□□$linked_title = sprintf( '<a %1$s>%2$s</a>', $this->get_render_attribute_string(
"link_{$index}" ), $title );
□□□□□□echo $linked_title;
□□□□□} else {
□□□□□□echo $title;
```

□□□□□}
□□□□□?>
□□□□
□□□□<?php
□□□}
□□□?>
□□</<?php echo $html_tag[$settings['marker_type']]; ?>>
□□<?php
□}

/**
□ *□□□□□□□□□□□□□□□□□□□
□*
□ *□□ Backbone JavaScript □□□□□□□□□□□□□□□□
□*
□ * @since 1.0.0
□ * @access protected
□ */
□protected function content_template(): void {
□□?>
□□<#
□□html_tag = {
□□□'ordered': 'ol',
□□□'unordered': 'ul',
□□□'other': 'ul',
□□};
□□view.addRenderAttribute('list', 'class', 'elementor-list-widget');
□□#>
□□<{{{ html_tag[settings.marker_type] }}} {{{ view.getRenderAttributeString('list') }}}>
□□□<# _.each(settings.list_items, function(item, index) {
□□□□const repeater_setting_key = view.getRepeaterSettingKey('text', 'list_items', index);
□□□□view.addRenderAttribute(repeater_setting_key, 'class', 'elementor-list-widget-text');
□□□□view.addInlineEditingAttributes(repeater_setting_key);
□□□□#>
□□□□<li {{{ view.getRenderAttributeString(repeater_setting_key) }}}>
□□□□□<# const title = item.text; #>
□□□□□<# if (item.link) { #>
□□□□□□<# view.addRenderAttribute(`link_${index}`, item.link); #>
□□□□□□<a href="{{ item.link.url }}" {{{ view.getRenderAttributeString(`link_${index}`) }}}>

```
□□□□□□□□{{{title}}}
□□□□□□□</a>
□□□□□□<# } else { #>
□□□□□□□{{{title}}}
□□□□□□<# } #>
□□□□□</li>
□□□□<# } ); #>
□□</{{{ html_tag[ settings.marker_type ] }}}>
□□<?php
□}

}
```

结果

一个带有重复器字段的新 “列表 ”小部件:

版本号 #2
由 壳壳虫 创建于 2026-03-26 17:08:44 UTC
由 壳壳虫 更新于 2026-03-26 17:08:44 UTC