Skill: oEmbed JS Provider Configuration Assistant

Purpose: When a user wants to add a new oEmbed provider to the oEmbed JS WordPress plugin, help them determine the correct values for each field based on whatever information they provide (a URL, an iframe embed code, a provider website, an API documentation link, or just a service name).

Context: The oEmbed JS plugin allows users to configure custom oEmbed providers. Each provider has the following fields that need to be filled in:

Field	Description	Expected Format	Required
Name	Human-readable name of the provider (e.g., "YouTube", "Spotify")	Plain text	Yes
URL Pattern	A regex pattern that matches URLs from this provider that should be converted to embeds	Regular expression (without delimiters), case-insensitive matching is applied automatically	Yes
Endpoint URL	The oEmbed API endpoint URL where the plugin sends requests to retrieve embed data	A full HTTPS URL, must return JSON	Yes
Regex	Whether the URL Pattern field is a regular expression	Checkbox (true/false). Almost always true.	Yes
Enabled	Whether this provider is active	Checkbox (true/false)	Yes
Embed Type	The type of embed this provider returns	One of: "iframe" (for video/audio players), "rich" (for HTML embeds like tweets, posts), or "photo" (for images)	Yes
Max Width (override)	Per-provider maximum width override	CSS value (e.g., "640px", "100%") or leave empty to use global setting	No
Max Height (override)	Per-provider maximum height override	CSS value (e.g., "480px") or leave empty to use global setting	No
Fixed Width (override)	Per-provider fixed width override	CSS value or leave empty	No

Field	Description	Expected Format	Required
Fixed Height (override)	Per-provider fixed height override	CSS value or leave empty	No
Wrapper Class (override)	Per-provider CSS class override for the embed wrapper	CSS class name or leave empty to use global setting	No

How to derive the values from user input:

1. **If the user provides a URL they want to embed (e.g., "https://www.example.com/video/12345"):**
   - Identify the provider/service from the domain.
   - Search for whether this provider has a known oEmbed endpoint. Check oembed.com/providers.json or the provider's documentation.
   - Construct a regex URL pattern that matches all embeddable URLs from this provider. Use capturing groups and character classes as needed. Escape dots with `\\.` in the regex.
   - If no oEmbed endpoint exists, inform the user that this provider does not support oEmbed and suggest alternatives (e.g., using an iframe embed directly via a custom solution, or checking if the provider offers oEmbed through a third-party service).
2. **If the user provides an iframe embed code (e.g., `<iframe src="https://player.example.com/embed/12345">`)**:
   - Extract the iframe src URL to understand the embed URL structure.
   - Work backwards from the embed/player URL to identify the provider.
   - Look up whether the provider has an oEmbed endpoint.
   - Note: The oEmbed endpoint URL is NOT the iframe src. The endpoint is an API URL that accepts a content URL and returns embed HTML (which typically contains the iframe). These are different things.
   - If the provider has an oEmbed API, determine the correct content URL pattern (the URL a user would share, not the embed URL) and the API endpoint.
3. **If the user provides just a service name (e.g., "Vimeo"):**
   - Look up the provider in the known oEmbed providers directory.
   - Provide the standard configuration.
4. **If the user provides a provider's API documentation link:**
   - Read the documentation to extract the oEmbed endpoint URL and supported URL schemes.
   - Convert the URL schemes to regex patterns.

Converting oEmbed scheme patterns to regex:

Many providers document their supported URLs using wildcard patterns like `https://www.example.com/video/*`. Convert these to regex:

- Replace `*` with `[a-zA-Z0-9_-]+` for path segments, or `.+` for broader matching.
- Escape literal dots: `.` becomes `\\.`
- Use `https?://` to match both HTTP and HTTPS if both are supported.

- Use `(?:www\\.)?` to optionally match the www subdomain if appropriate.
- Combine multiple scheme patterns using `|` (alternation) within a non-capturing group `(?:...|...)` if needed, or use a single broader pattern if all schemes can be covered.

Choosing the Embed Type:

- Use **"iframe"** for video and audio players (YouTube, Vimeo, Spotify, Dailymotion, etc.) — services whose oEmbed response contains an `<iframe>` in the `html` field.
- Use **"rich"** for services that return complex HTML that is not just an iframe (Twitter/X posts, Instagram embeds, TikTok, Reddit, CodePen, etc.) — these often include `<blockquote>` elements with `<script>` tags.
- Use **"photo"** for services that return image data (Flickr photos, Giphy, etc.) — the oEmbed response will have `type: "photo"` and a `url` field pointing to the image.

Response format: Present the recommended configuration in a clear table. Example:

For a user who says: "I want to embed Vimeo videos"

Field	Value
Name	Vimeo
URL Pattern	`https?://(?:www\.)?vimeo\.com/(\d+)`
Endpoint URL	`https://vimeo.com/api/oembed.json`
Regex	☑ Checked
Enabled	☑ Checked
Embed Type	iframe
Max Width	*(leave empty — uses global setting)*
Max Height	*(leave empty — uses global setting)*
Fixed Width	*(leave empty)*
Fixed Height	*(leave empty)*
Wrapper Class	*(leave empty — uses global setting)*

Additional guidance to provide the user:

- After adding the provider, remind the user to click **"Save Settings"** in the plugin admin page.
- If the provider requires authentication (e.g., Instagram requires a Facebook App access token), warn the user and explain that the oEmbed endpoint may need an `access_token` parameter appended. The endpoint field can include query parameters, e.g., `https://graph.facebook.com/v18.0/instagram_oembed?access_token=YOUR_TOKEN`.
- If the provider is already in the plugin's built-in defaults or can be imported from the oEmbed Directory browser (the "Browse oEmbed Directory" button in the admin), suggest that simpler path instead of manual configuration.

- The URL Pattern should match the **user-facing/shareable URL** (the URL someone would copy from their browser address bar), NOT the embed/player URL.
- The Endpoint URL must point to the provider's oEmbed API endpoint that accepts a `url` parameter and returns JSON. The plugin will call this endpoint via its server-side proxy to avoid CORS issues.
- Test the configuration by placing a matching link inside a container that matches the plugin's configured CSS selectors.

When oEmbed is not available:

If the provider does not offer an oEmbed endpoint, inform the user clearly and suggest:

1. Check if a third-party oEmbed proxy service (like Iframely or Embedly) supports the provider.
2. The provider may support oEmbed discovery — look for `<link type="application/json+oembed">` in the HTML head of content pages.
3. If none of these work, this plugin cannot embed content from that provider, as it relies on the oEmbed protocol.

Revision #2
Created 2026-06-09 06:01:55 UTC by art10m
Updated 2026-06-09 06:47:11 UTC by art10m