oEmbed

- oEmbed steuern (in WP)
- oEmbed Provider sehen (in WP)
- oEmbed Verwaltungs-Plugin (für WP) [Opus 4.6]
- oEmbed Plugin | eigene Provider definieren
- Skill (für KIs): oEmbed Provider Configuration Generator

oEmbed steuern (in WP)

In WordPress kannst du **oEmbed auf 2 Ebenen steuern**:

1. **was WordPress selbst einbetten darf**
2. **wo auf deiner Seite fremde Inhalte eingebettet werden dürfen**

1. Welche Anbieter WordPress per oEmbed akzeptiert

WordPress unterstützt standardmäßig bestimmte Provider, z. B. YouTube, Vimeo usw.

Anzeigen, welche Provider aktiv sind

Per Code kannst du die oEmbed-Providerliste beeinflussen.

Beispiel in `functions.php` oder besser in einem kleinen Plugin:

```
add_action('init', function () {
    global $wp_oembed;

    echo '<pre>';
    print_r($wp_oembed->providers);
    echo '</pre>';
});
```

Das ist nur zum Testen sinnvoll.

---

2. Bestimmte oEmbed-Provider entfernen

Wenn du z. B. verhindern willst, dass Inhalte von Twitter/X oder Vimeo automatisch eingebettet werden:

```
add_action('init', function () {
    wp_oembed_remove_provider('#https?://(www\.)?twitter\.com/.+#i');
    wp_oembed_remove_provider('#https?://(www\.)?x\.com/.+#i');
    wp_oembed_remove_provider('#https?://(www\.)?vimeo\.com/.+#i');
});
```

Je nach WP-Version können die Regex-Muster leicht abweichen.

---

3. Eigene Provider hinzufügen

Wenn du willst, dass WordPress auch URLs eines eigenen Dienstes per oEmbed verarbeitet:

```
add_action('init', function () {
    wp_oembed_add_provider(
        '#https?://example\.com/watch/.+#i',
        'https://example.com/oembed',
        true
    );
});
```

- erstes Argument = URL-Muster
- zweites = oEmbed-Endpunkt
- drittes = Regex ja/nein

---

4. Auto-Embed komplett abschalten

Wenn du nicht willst, dass WordPress nackte URLs automatisch in Embeds umwandelt:

```
remove_filter('the_content', array($GLOBALS['wp_embed'], 'autoembed'), 8);
```

Damit werden einfache URLs im Content nicht mehr automatisch eingebettet.

5. REST/oEmbed-Endpunkte deaktivieren

Wenn du verhindern willst, dass deine WordPress-Seite selbst als oEmbed-Provider arbeitet oder oEmbed-Endpunkte bereitstellt:

```
remove_action('rest_api_init', 'wp_oembed_register_route');
add_filter('embed_oembed_discover', '__return_false');
remove_filter('oembed_dataparse', 'wp_filter_oembed_result', 10);
remove_action('wp_head', 'wp_oembed_add_discovery_links');
remove_action('wp_head', 'wp_oembed_add_host_js');
```

Das ist eher sinnvoll, wenn du oEmbed aus Sicherheits-, Datenschutz- oder Performance-Gründen einschränken willst.

---

6. Nur bestimmte Embeds im Editor erlauben

Wenn du gezielt steuern willst, **welche URLs Autoren einfügen dürfen**, kannst du beim Speichern oder Rendern prüfen, ob eine URL auf einer Allowlist steht.

Einfaches Beispiel:

```
add_filter('pre_oembed_result', function ($result, $url, $args) {
    $allowed = [
        'youtube.com',
        'youtu.be',
        'vimeo.com',
    ];

    $host = wp_parse_url($url, PHP_URL_HOST);
    if (!$host) {
        return false;
```

```
    }

    foreach ($allowed as $domain) {
        if (str_contains($host, $domain)) {
            return $result;
        }
    }

    return '<p>Einbettung dieses Anbieters ist nicht erlaubt.</p>';
}, 10, 3);
```

Sauberer wäre eine exakte Host-Prüfung statt `str_contains()`.

---

7. Datenschutz: Embeds nur nach Klick laden

Wenn es dir auch um DSGVO geht, solltest du Embeds nicht sofort laden, sondern erst nach Zustimmung oder Klick. Das betrifft vor allem:

- YouTube
- Vimeo
- Spotify
- X/Twitter
- Instagram

Dafür gibt es:

- Consent-Plugins
- lokale Platzhalterlösungen
- 2-Klick-Lösungen

Praxis: Was du wahrscheinlich meinst

Wenn du fragst *„wie kann ich steuern, was in WordPress per oEmbed eingebunden wird?“*, meinst du meistens eines von diesen drei Zielen:

A) Nur bestimmte Anbieter erlauben

→ Provider entfernen oder Allowlist bauen

B) Automatische Einbettung ganz verhindern

→ `autoembed` entfernen

C) Datenschutz / Performance verbessern

→ oEmbed-Endpunkte und Auto-Discovery deaktivieren, ggf. mit Klicklösung

---

Empfohlene saubere Lösung

Wenn du nur bestimmte Embeds erlauben willst, würde ich so vorgehen:

- Autoembed aktiv lassen
- unerwünschte Provider entfernen
- optional nur Allowlist-Domains erlauben
- für externe Medien Consent/Klicklösung nutzen

oEmbed Provider sehen (in WP)

Ja — am einfachsten direkt per PHP.

1) Alle aktiven oEmbed-Provider ausgeben

Temporär in `functions.php` oder besser in ein kleines Plugin:

```
add_action('admin_init', function () {
    global $wp_oembed;

    echo '<pre>';
    print_r($wp_oembed->providers);
    echo '</pre>';
    exit;
});
```

Dann einmal `/wp-admin/` aufrufen.

Wichtig:

- Das ist nur zum Testen.
- Danach wieder entfernen, sonst wird das Admin-Backend blockiert.

In der Ausgabe siehst du die registrierten URL-Muster und die zugehörigen oEmbed-Endpunkte.

---

2) Prüfen, ob YouTube aktiv ist

WordPress speichert die Provider als Muster. Du kannst gezielt danach suchen:

```
add_action('admin_init', function () {
    global $wp_oembed;

    $found = false;

    foreach ($wp_oembed->providers as $pattern => $data) {
        if (stripos($pattern, 'youtube') !== false || stripos($pattern, 'youtu.be') !== false)
{
            echo '<p><strong>YouTube-Provider aktiv:</strong></p>';
            echo '<pre>';
            print_r([$pattern => $data]);
            echo '</pre>';
            $found = true;
        }
    }

    if (!$found) {
        echo '<p><strong>Kein YouTube-Provider gefunden.</strong></p>';
    }

    exit;
});
```

---

3) Sauberer Check als true/false

Wenn du nur wissen willst, **ob YouTube aktiv ist**, ohne alles auszugeben:

```
function my_is_youtube_oembed_active() {
    global $wp_oembed;

    foreach ($wp_oembed->providers as $pattern => $data) {
        if (stripos($pattern, 'youtube.com') !== false || stripos($pattern, 'youtu.be') !==
false) {
            return true;
        }
    }
```

```
    return false;
}

add_action('admin_notices', function () {
    if (my_is_youtube_oembed_active()) {
        echo '<div class="notice notice-success"><p>YouTube oEmbed ist aktiv.</p></div>';
    } else {
        echo '<div class="notice notice-warning"><p>YouTube oEmbed ist nicht
aktiv.</p></div>';
    }
});
```

---

4) Noch direkter: Test mit echter URL

Praktisch ist auch zu prüfen, ob WordPress eine echte YouTube-URL als Embed erkennt:

```
add_action('admin_init', function () {
    $url = 'https://www.youtube.com/watch?v=dQw4w9WgXcQ';
    $embed = wp_oembed_get($url);

    if ($embed) {
        echo '<p><strong>YouTube funktioniert als oEmbed.</strong></p>';
        echo $embed;
    } else {
        echo '<p><strong>YouTube funktioniert NICHT als oEmbed.</strong></p>';
    }

    exit;
});
```

Das ist oft aussagekräftiger als nur die Providerliste.

---

5) Per WP-CLI, falls du Zugriff hast

Wenn du SSH/WP-CLI hast:

```
wp eval 'global $wp_oembed; print_r($wp_oembed->providers);'
```

Oder nur YouTube filtern:

```
wp eval '
global $wp_oembed;
foreach ($wp_oembed->providers as $pattern => $data) {
    if (stripos($pattern, "youtube") !== false || stripos($pattern, "youtu.be") !== false) {
        print_r([$pattern => $data]);
    }
}
'
```

Empfehlung

Am besten ist meist:

1. **Providerliste ausgeben**
2. **mit `wp_oembed_get()` eine echte URL testen**

Denn ein Provider kann zwar registriert sein, aber der Embed kann trotzdem durch andere Filter/Plugins gestört werden.

oEmbed Verwaltungs-Plugin (für WP) [Opus 4.6]

> Nicht getestet!

Ein umfassendes WordPress-Plugin zur Verwaltung von oEmbed-Einbettungen. Hier ist der Implementierungsplan:

Implementierungsplan

1. **Haupt-Plugin-Datei** erstellen mit Plugin-Header, Aktivierung/Deaktivierung und Laden der Komponenten
2. **Admin-Einstellungsseite** mit Tabs für:
   - Provider-Verwaltung (aktivieren/deaktivieren einzelner oEmbed-Provider)
   - Globale Einstellungen (Cache-Dauer, max. Breite/Höhe, Lazy Loading)
   - Eigene Provider hinzufügen
   - Blockliste für URLs
3. **Frontend-Filter** die oEmbed-Ausgabe modifizieren (Wrapper, responsive Container, Lazy Loading)
4. **CSS & JS Assets** für Admin und Frontend
5. **Uninstall-Routine** zum sauberen Entfernen

Schritt 1: Haupt-Plugin-Datei

Diese Datei registriert das Plugin, lädt alle Komponenten und definiert Aktivierungs-/Deaktivierungs-Hooks.

```
<?php
/**
 * Plugin Name: oEmbed Manager
 * Plugin URI: https://example.com/oembed-manager
 * Description: Umfassende Verwaltung von oEmbed-Einbettungen – Provider
```

```
aktivieren/deaktivieren, eigene Provider hinzufügen, Darstellung anpassen, Blockliste und
mehr.
 * Version: 1.0.0
 * Author: zcrow
 * Author URI: https://example.com
 * License: GPL-2.0+
 * License URI: https://www.gnu.org/licenses/gpl-2.0.html
 * Text Domain: oembed-manager
 * Domain Path: /languages
 */

if ( ! defined( 'ABSPATH' ) ) {
    exit;
}

define( 'JEPP_OEM_VERSION', '1.0.0' );
define( 'JEPP_OEM_PLUGIN_DIR', plugin_dir_path( __FILE__ ) );
define( 'JEPP_OEM_PLUGIN_URL', plugin_dir_url( __FILE__ ) );
define( 'JEPP_OEM_PLUGIN_BASENAME', plugin_basename( __FILE__ ) );

/**
 * Standardoptionen bei Aktivierung setzen.
 */
function jepp_oem_activate() {
    $defaults = array(
        'disabled_providers'  => array(),
        'custom_providers'    => array(),
        'blocked_urls'        => '',
        'max_width'           => 800,
        'max_height'          => 0,
        'cache_duration'      => 86400,
        'responsive'          => 1,
        'lazy_loading'        => 1,
        'wrapper_class'       => 'jepp-oembed-wrap',
        'disable_in_feed'     => 0,
        'disable_on_frontend' => 0,
        'strip_inline_styles' => 0,
    );

    if ( false === get_option( 'jepp_oem_settings' ) ) {
```

```
        add_option( 'jepp_oem_settings', $defaults );
    }
}
register_activation_hook( __FILE__, 'jepp_oem_activate' );

/**
 * Aufräumen bei Deaktivierung (Cache leeren).
 */
function jepp_oem_deactivate() {
    global $wpdb;
    $wpdb->query(
        "DELETE FROM {$wpdb->postmeta} WHERE meta_key LIKE '_oembed_%'"
    );
    $wpdb->query(
        "DELETE FROM {$wpdb->options} WHERE option_name LIKE '_transient_oembed_%' OR
option_name LIKE '_transient_timeout_oembed_%'"
    );
}
register_deactivation_hook( __FILE__, 'jepp_oem_deactivate' );

// Komponenten laden
require_once JEPP_OEM_PLUGIN_DIR . 'includes/class-jepp-oem-settings.php';
require_once JEPP_OEM_PLUGIN_DIR . 'includes/class-jepp-oem-providers.php';
require_once JEPP_OEM_PLUGIN_DIR . 'includes/class-jepp-oem-frontend.php';

/**
 * Plugin initialisieren.
 */
function jepp_oem_init() {
    load_plugin_textdomain( 'oembed-manager', false, dirname( JEPP_OEM_PLUGIN_BASENAME ) .
'/languages' );

    new Jepp_OEM_Settings();
    new Jepp_OEM_Providers();
    new Jepp_OEM_Frontend();
}
add_action( 'plugins_loaded', 'jepp_oem_init' );

/**
 * Settings-Link in der Plugin-Liste.
```

```
 */
function jepp_oem_plugin_action_links( $links ) {
    $settings_link = '<a href="' . admin_url( 'options-general.php?page=jepp-oembed-manager' )
. '">'
                    . __( 'Einstellungen', 'oembed-manager' ) . '</a>';
    array_unshift( $links, $settings_link );
    return $links;
}
add_filter( 'plugin_action_links_' . JEPP_OEM_PLUGIN_BASENAME, 'jepp_oem_plugin_action_links'
);
```

Schritt 2: Provider-Verwaltung

Diese Klasse kümmert sich um das Deaktivieren vorhandener und Registrieren eigener oEmbed-Provider.

```
<?php
if ( ! defined( 'ABSPATH' ) ) {
    exit;
}

/**
 * Verwaltet oEmbed-Provider: deaktivieren, eigene hinzufügen, URLs blockieren.
 */
class Jepp_OEM_Providers {

    /** @var array */
    private $settings;

    public function __construct() {
        $this->settings = get_option( 'jepp_oem_settings', array() );

        // Provider deaktivieren
        add_filter( 'oembed_providers', array( $this, 'filter_providers' ), 99 );

        // Eigene Provider registrieren
        add_action( 'init', array( $this, 'register_custom_providers' ), 20 );

        // URLs blockieren
```

```
        add_filter( 'pre_oembed_result', array( $this, 'block_urls' ), 10, 3 );
    }

    /**
     * Deaktivierte Provider aus der Liste entfernen.
     *
     * @param array $providers
     * @return array
     */
    public function filter_providers( $providers ) {
        $disabled = isset( $this->settings['disabled_providers'] ) ? $this-
>settings['disabled_providers'] : array();

        if ( empty( $disabled ) ) {
            return $providers;
        }

        foreach ( $providers as $pattern => $data ) {
            // $data[0] = endpoint URL, $data[1] = regex flag
            $endpoint = is_array( $data ) ? $data[0] : $data;

            foreach ( $disabled as $disabled_key ) {
                if ( stripos( $pattern, $disabled_key ) !== false || stripos( $endpoint,
$disabled_key ) !== false ) {
                    unset( $providers[ $pattern ] );
                }
            }
        }

        return $providers;
    }

    /**
     * Eigene Provider registrieren.
     */
    public function register_custom_providers() {
        $custom = isset( $this->settings['custom_providers'] ) ? $this-
>settings['custom_providers'] : array();

        if ( empty( $custom ) ) {
```

```
            return;
        }

        foreach ( $custom as $provider ) {
            if ( empty( $provider['pattern'] ) || empty( $provider['endpoint'] ) ) {
                continue;
            }
            $regex = ! empty( $provider['regex'] ) ? true : false;
            wp_oembed_add_provider( $provider['pattern'], $provider['endpoint'], $regex );
        }
    }

    /**
     * Bestimmte URLs blockieren.
     *
     * @param null|string $result
     * @param string      $url
     * @param array       $args
     * @return null|false|string
     */
    public function block_urls( $result, $url, $args ) {
        $blocked_raw = isset( $this->settings['blocked_urls'] ) ? $this-
>settings['blocked_urls'] : '';

        if ( empty( $blocked_raw ) ) {
            return $result;
        }

        $blocked_lines = array_filter( array_map( 'trim', explode( "\n", $blocked_raw ) ) );

        foreach ( $blocked_lines as $blocked ) {
            if ( empty( $blocked ) ) {
                continue;
            }
            // Unterstützung für Wildcards: * → .*
            $pattern = '#' . str_replace( '\*', '.*', preg_quote( $blocked, '#' ) ) . '#i';
            if ( preg_match( $pattern, $url ) ) {
                return false; // Blockiert
            }
        }
```

```
    return $result;
}

/**
 * Gibt alle in WordPress registrierten Standard-Provider zurück.
 *
 * @return array  Array mit Schlüssel = Pattern, Wert = Endpoint
 */
public static function get_default_providers() {
    require_once ABSPATH . WPINC . '/class-wp-oembed.php';
    $oembed    = _wp_oembed_get_object();
    $providers = $oembed->providers;

    $list = array();
    foreach ( $providers as $pattern => $data ) {
        $endpoint = is_array( $data ) ? $data[0] : $data;
        // Lesbaren Namen aus dem Pattern oder der Endpoint-URL extrahieren
        $name = self::extract_provider_name( $pattern, $endpoint );
        $list[ $pattern ] = array(
            'name'     => $name,
            'endpoint' => $endpoint,
            'pattern'  => $pattern,
        );
    }

    // Alphabetisch nach Name sortieren
    uasort( $list, function ( $a, $b ) {
        return strcasecmp( $a['name'], $b['name'] );
    });

    return $list;
}

/**
 * Versucht einen lesbaren Provider-Namen zu extrahieren.
 *
 * @param string $pattern
 * @param string $endpoint
 * @return string
```

```
     */
    private static function extract_provider_name( $pattern, $endpoint ) {
        // Aus Endpoint-URL den Hostnamen holen
        $host = wp_parse_url( $endpoint, PHP_URL_HOST );
        if ( $host ) {
            $host = preg_replace( '#^(www\.|publish\.)#i', '', $host );
            $parts = explode( '.', $host );
            if ( count( $parts ) >= 2 ) {
                return ucfirst( $parts[ count( $parts ) - 2 ] );
            }
            return ucfirst( $host );
        }

        // Fallback: Pattern bereinigen
        $clean = preg_replace( '#https?://|[^a-zA-Z0-9\.]#', '', $pattern );
        $clean = preg_replace( '#^(www\.)#', '', $clean );
        $parts = explode( '.', $clean );
        return ucfirst( $parts[0] ?: 'Unknown' );
    }
}
```

Schritt 3: Admin-Einstellungsseite

Die Settings-Klasse baut eine übersichtliche Admin-Seite mit Tabs.

```
<?php
if ( ! defined( 'ABSPATH' ) ) {
    exit;
}

/**
 * Admin-Einstellungsseite für den oEmbed Manager.
 */
class Jepp_OEM_Settings {

    /** @var string */
    private $option_name = 'jepp_oem_settings';
```

```
/** @var array */
private $settings;

public function __construct() {
    $this->settings = get_option( $this->option_name, array() );

    add_action( 'admin_menu', array( $this, 'add_menu_page' ) );
    add_action( 'admin_init', array( $this, 'register_settings' ) );
    add_action( 'admin_enqueue_scripts', array( $this, 'enqueue_assets' ) );
}

/**
 * Menüeintrag unter "Einstellungen".
 */
public function add_menu_page() {
    add_options_page(
        __( 'oEmbed Manager', 'oembed-manager' ),
        __( 'oEmbed Manager', 'oembed-manager' ),
        'manage_options',
        'jepp-oembed-manager',
        array( $this, 'render_page' )
    );
}

/**
 * Admin-Assets laden.
 */
public function enqueue_assets( $hook ) {
    if ( 'settings_page_jepp-oembed-manager' !== $hook ) {
        return;
    }
    wp_enqueue_style(
        'jepp-oem-admin',
        JEPP_OEM_PLUGIN_URL . 'assets/css/admin.css',
        array(),
        JEPP_OEM_VERSION
    );
    wp_enqueue_script(
        'jepp-oem-admin',
        JEPP_OEM_PLUGIN_URL . 'assets/js/admin.js',
```

```
            array( 'jquery' ),
            JEPP_OEM_VERSION,
            true
        );
    }

    /**
     * Settings API registrieren.
     */
    public function register_settings() {
        register_setting( 'jepp_oem_group', $this->option_name, array(
            'type'              => 'array',
            'sanitize_callback' => array( $this, 'sanitize_settings' ),
        ) );
    }

    /**
     * Eingaben bereinigen.
     */
    public function sanitize_settings( $input ) {
        $clean = array();

        // Deaktivierte Provider
        $clean['disabled_providers'] = array();
        if ( ! empty( $input['disabled_providers'] ) && is_array( $input['disabled_providers']
) ) {
            $clean['disabled_providers'] = array_map( 'sanitize_text_field',
$input['disabled_providers'] );
        }

        // Eigene Provider
        $clean['custom_providers'] = array();
        if ( ! empty( $input['custom_providers'] ) && is_array( $input['custom_providers'] ) )
{
            foreach ( $input['custom_providers'] as $cp ) {
                if ( empty( $cp['pattern'] ) || empty( $cp['endpoint'] ) ) {
                    continue;
                }
                $clean['custom_providers'][] = array(
                    'pattern'  => sanitize_text_field( $cp['pattern'] ),
```

```
                    'endpoint' => esc_url_raw( $cp['endpoint'] ),
                    'regex'    => ! empty( $cp['regex'] ) ? 1 : 0,
                );
            }
        }

        // Blockliste
        $clean['blocked_urls'] = '';
        if ( ! empty( $input['blocked_urls'] ) ) {
            $clean['blocked_urls'] = sanitize_textarea_field( $input['blocked_urls'] );
        }

        // Numerische Werte
        $clean['max_width']       = isset( $input['max_width'] ) ? absint( $input['max_width']
) : 800;
        $clean['max_height']      = isset( $input['max_height'] ) ? absint(
$input['max_height'] ) : 0;
        $clean['cache_duration']  = isset( $input['cache_duration'] ) ? absint(
$input['cache_duration'] ) : 86400;

        // Checkboxen
        $clean['responsive']          = ! empty( $input['responsive'] ) ? 1 : 0;
        $clean['lazy_loading']        = ! empty( $input['lazy_loading'] ) ? 1 : 0;
        $clean['disable_in_feed']     = ! empty( $input['disable_in_feed'] ) ? 1 : 0;
        $clean['disable_on_frontend'] = ! empty( $input['disable_on_frontend'] ) ? 1 : 0;
        $clean['strip_inline_styles'] = ! empty( $input['strip_inline_styles'] ) ? 1 : 0;

        // Wrapper-Klasse
        $clean['wrapper_class'] = ! empty( $input['wrapper_class'] )
            ? sanitize_html_class( $input['wrapper_class'] )
            : 'jepp-oembed-wrap';

        // oEmbed-Cache leeren
        $this->flush_oembed_cache();

        return $clean;
    }

    /**
     * oEmbed-Cache leeren.
```

```
     */
    private function flush_oembed_cache() {
        global $wpdb;
        $wpdb->query(
            "DELETE FROM {$wpdb->postmeta} WHERE meta_key LIKE '_oembed_%'"
        );
        $wpdb->query(
            "DELETE FROM {$wpdb->options} WHERE option_name LIKE '_transient_oembed_%' OR
option_name LIKE '_transient_timeout_oembed_%'"
        );
    }

    /**
     * Einstellungsseite rendern.
     */
    public function render_page() {
        if ( ! current_user_can( 'manage_options' ) ) {
            return;
        }

        $active_tab = isset( $_GET['tab'] ) ? sanitize_key( $_GET['tab'] ) : 'providers';
        $tabs = array(
            'providers' => __( 'Provider', 'oembed-manager' ),
            'display'   => __( 'Darstellung', 'oembed-manager' ),
            'custom'    => __( 'Eigene Provider', 'oembed-manager' ),
            'blocklist' => __( 'Blockliste', 'oembed-manager' ),
        );
        ?>
        <div class="wrap jepp-oem-wrap">
            <h1><?php esc_html_e( 'oEmbed Manager', 'oembed-manager' ); ?></h1>

            <nav class="nav-tab-wrapper jepp-oem-tabs">
                <?php foreach ( $tabs as $slug => $label ) : ?>
                    <a href="<?php echo esc_url( add_query_arg( array( 'page' => 'jepp-oembed-
manager', 'tab' => $slug ), admin_url( 'options-general.php' ) ) ); ?>"
                       class="nav-tab <?php echo $active_tab === $slug ? 'nav-tab-active' :
''; ?>">
                        <?php echo esc_html( $label ); ?>
                    </a>
                <?php endforeach; ?>
```

```
            </nav>

            <form method="post" action="options.php">
                <?php
                settings_fields( 'jepp_oem_group' );

                switch ( $active_tab ) {
                    case 'display':
                        $this->render_tab_display();
                        break;
                    case 'custom':
                        $this->render_tab_custom();
                        break;
                    case 'blocklist':
                        $this->render_tab_blocklist();
                        break;
                    default:
                        $this->render_tab_providers();
                        break;
                }

                // Hidden Fields für die anderen Tabs mitgeben, damit sie nicht überschrieben
werden
                $this->render_hidden_fields( $active_tab );

                submit_button( __( 'Einstellungen speichern', 'oembed-manager' ) );
                ?>
            </form>
        </div>
        <?php
    }

    /**
     * Tab: Provider verwalten.
     */
    private function render_tab_providers() {
        $providers = Jepp_OEM_Providers::get_default_providers();
        $disabled  = isset( $this->settings['disabled_providers'] ) ? $this-
>settings['disabled_providers'] : array();
        ?>
```

```
        <div class="jepp-oem-section">
            <h2><?php esc_html_e( 'oEmbed-Provider aktivieren / deaktivieren', 'oembed-
manager' ); ?></h2>
            <p class="description"><?php esc_html_e( 'Deaktivierte Provider werden nicht mehr
für die automatische Einbettung verwendet.', 'oembed-manager' ); ?></p>

            <div class="jepp-oem-provider-actions" style="margin: 10px 0;">
                <button type="button" class="button jepp-oem-select-all"><?php esc_html_e(
'Alle aktivieren', 'oembed-manager' ); ?></button>
                <button type="button" class="button jepp-oem-deselect-all"><?php esc_html_e(
'Alle deaktivieren', 'oembed-manager' ); ?></button>
            </div>

            <table class="widefat jepp-oem-provider-table">
                <thead>
                    <tr>
                        <th style="width:50px;"><?php esc_html_e( 'Aktiv', 'oembed-manager' );
?></th>
                        <th><?php esc_html_e( 'Provider', 'oembed-manager' ); ?></th>
                        <th><?php esc_html_e( 'URL-Pattern', 'oembed-manager' ); ?></th>
                        <th><?php esc_html_e( 'Endpoint', 'oembed-manager' ); ?></th>
                    </tr>
                </thead>
                <tbody>
                    <?php foreach ( $providers as $pattern => $info ) :
                        $key        = md5( $pattern );
                        $is_disabled = in_array( $key, $disabled, true );
                        ?>
                        <tr>
                            <td>
                                <input type="checkbox"
                                       class="jepp-oem-provider-checkbox"
                                       name="<?php echo esc_attr( $this->option_name );
?>[disabled_providers][]"
                                       value="<?php echo esc_attr( $key ); ?>"
                                    <?php checked( false, $is_disabled ); ?>
                                       data-inverted="1"
                                >
                            </td>
                            <td><strong><?php echo esc_html( $info['name'] ); ?></strong></td>
```

```
                            <td><code><?php echo esc_html( $pattern ); ?></code></td>
                            <td><code><?php echo esc_html( $info['endpoint'] ); ?></code></td>
                        </tr>
                    <?php endforeach; ?>
                </tbody>
            </table>
        </div>
        <?php
    }

    /**
     * Tab: Darstellungsoptionen.
     */
    private function render_tab_display() {
        $s = $this->settings;
        ?>
        <div class="jepp-oem-section">
            <h2><?php esc_html_e( 'Darstellungseinstellungen', 'oembed-manager' ); ?></h2>
            <table class="form-table">
                <tr>
                    <th scope="row"><?php esc_html_e( 'Maximale Breite (px)', 'oembed-manager'
); ?></th>
                    <td>
                        <input type="number" name="<?php echo esc_attr( $this->option_name );
?>[max_width]"
                               value="<?php echo esc_attr( $s['max_width'] ?? 800 ); ?>"
min="0" step="1" class="small-text">
                        <p class="description"><?php esc_html_e( '0 = keine Beschränkung',
'oembed-manager' ); ?></p>
                    </td>
                </tr>
                <tr>
                    <th scope="row"><?php esc_html_e( 'Maximale Höhe (px)', 'oembed-manager'
); ?></th>
                    <td>
                        <input type="number" name="<?php echo esc_attr( $this->option_name );
?>[max_height]"
                               value="<?php echo esc_attr( $s['max_height'] ?? 0 ); ?>"
min="0" step="1" class="small-text">
                        <p class="description"><?php esc_html_e( '0 = keine Beschränkung',
```

```
'oembed-manager' ); ?></p>
                    </td>
                </tr>
                <tr>
                    <th scope="row"><?php esc_html_e( 'Cache-Dauer (Sekunden)', 'oembed-
manager' ); ?></th>
                    <td>
                        <input type="number" name="<?php echo esc_attr( $this->option_name );
?>[cache_duration]"
                               value="<?php echo esc_attr( $s['cache_duration'] ?? 86400 );
?>" min="0" step="1" class="regular-text">
                        <p class="description"><?php esc_html_e( '86400 = 1 Tag. 0 = Cache
deaktiviert.', 'oembed-manager' ); ?></p>
                    </td>
                </tr>
                <tr>
                    <th scope="row"><?php esc_html_e( 'Wrapper CSS-Klasse', 'oembed-manager'
); ?></th>
                    <td>
                        <input type="text" name="<?php echo esc_attr( $this->option_name );
?>[wrapper_class]"
                               value="<?php echo esc_attr( $s['wrapper_class'] ?? 'jepp-
oembed-wrap' ); ?>" class="regular-text">
                    </td>
                </tr>
                <tr>
                    <th scope="row"><?php esc_html_e( 'Optionen', 'oembed-manager' ); ?></th>
                    <td>
                        <fieldset>
                            <label>
                                <input type="checkbox" name="<?php echo esc_attr( $this-
>option_name ); ?>[responsive]" value="1"
                                    <?php checked( 1, $s['responsive'] ?? 1 ); ?>>
                                <?php esc_html_e( 'Responsive Container (16:9) für Videos',
'oembed-manager' ); ?>
                            </label><br>
                            <label>
                                <input type="checkbox" name="<?php echo esc_attr( $this-
>option_name ); ?>[lazy_loading]" value="1"
                                    <?php checked( 1, $s['lazy_loading'] ?? 1 ); ?>>
```

```
                                <?php esc_html_e( 'Lazy Loading für iframes aktivieren',
'oembed-manager' ); ?>
                            </label><br>
                            <label>
                                <input type="checkbox" name="<?php echo esc_attr( $this-
>option_name ); ?>[strip_inline_styles]" value="1"
                                    <?php checked( 1, $s['strip_inline_styles'] ?? 0 ); ?>>
                                <?php esc_html_e( 'Inline-Styles aus oEmbed-HTML entfernen',
'oembed-manager' ); ?>
                            </label><br>
                            <label>
                                <input type="checkbox" name="<?php echo esc_attr( $this-
>option_name ); ?>[disable_in_feed]" value="1"
                                    <?php checked( 1, $s['disable_in_feed'] ?? 0 ); ?>>
                                <?php esc_html_e( 'oEmbed im RSS-Feed deaktivieren (nur Link
anzeigen)', 'oembed-manager' ); ?>
                            </label><br>
                            <label>
                                <input type="checkbox" name="<?php echo esc_attr( $this-
>option_name ); ?>[disable_on_frontend]" value="1"
                                    <?php checked( 1, $s['disable_on_frontend'] ?? 0 ); ?>>
                                <?php esc_html_e( 'oEmbed komplett auf dem Frontend
deaktivieren', 'oembed-manager' ); ?>
                            </label>
                        </fieldset>
                    </td>
                </tr>
            </table>
        </div>
        <?php
    }

    /**
     * Tab: Eigene Provider.
     */
    private function render_tab_custom() {
        $custom = isset( $this->settings['custom_providers'] ) ? $this-
>settings['custom_providers'] : array();
        ?>
        <div class="jepp-oem-section">
```

```
            <h2><?php esc_html_e( 'Eigene oEmbed-Provider', 'oembed-manager' ); ?></h2>
            <p class="description"><?php esc_html_e( 'Hier können Sie eigene oEmbed-Provider
hinzufügen, z.B. für interne Dienste.', 'oembed-manager' ); ?></p>

            <table class="widefat jepp-oem-custom-table" id="jepp-oem-custom-table">
                <thead>
                    <tr>
                        <th><?php esc_html_e( 'URL-Pattern', 'oembed-manager' ); ?></th>
                        <th><?php esc_html_e( 'Endpoint-URL', 'oembed-manager' ); ?></th>
                        <th style="width:60px;"><?php esc_html_e( 'Regex', 'oembed-manager' );
?></th>
                        <th style="width:60px;"><?php esc_html_e( 'Aktion', 'oembed-manager'
); ?></th>
                    </tr>
                </thead>
                <tbody>
                    <?php if ( ! empty( $custom ) ) : ?>
                        <?php foreach ( $custom as $i => $cp ) : ?>
                            <tr class="jepp-oem-custom-row">
                                <td>
                                    <input type="text" class="regular-text"
                                           name="<?php echo esc_attr( $this->option_name );
?>[custom_providers][<?php echo $i; ?>][pattern]"
                                           value="<?php echo esc_attr( $cp['pattern'] ); ?>"
                                           placeholder="https://example.com/*">
                                </td>
                                <td>
                                    <input type="url" class="regular-text"
                                           name="<?php echo esc_attr( $this->option_name );
?>[custom_providers][<?php echo $i; ?>][endpoint]"
                                           value="<?php echo esc_attr( $cp['endpoint'] ); ?>"
                                           placeholder="https://example.com/oembed">
                                </td>
                                <td>
                                    <input type="checkbox"
                                           name="<?php echo esc_attr( $this->option_name );
?>[custom_providers][<?php echo $i; ?>][regex]"
                                           value="1" <?php checked( 1, $cp['regex'] ?? 0 );
?>>
                                </td>
```

```
                                <td>
                                    <button type="button" class="button jepp-oem-remove-
row">&times;</button>
                                </td>
                            </tr>
                        <?php endforeach; ?>
                    <?php endif; ?>
                </tbody>
            </table>

            <p>
                <button type="button" class="button button-secondary" id="jepp-oem-add-
provider">
                    <?php esc_html_e( '+ Provider hinzufügen', 'oembed-manager' ); ?>
                </button>
            </p>
        </div>
        <?php
    }

    /**
     * Tab: Blockliste.
     */
    private function render_tab_blocklist() {
        $blocked = isset( $this->settings['blocked_urls'] ) ? $this->settings['blocked_urls']
: '';
        ?>
        <div class="jepp-oem-section">
            <h2><?php esc_html_e( 'URL-Blockliste', 'oembed-manager' ); ?></h2>
            <p class="description">
                <?php esc_html_e( 'URLs oder URL-Muster (eine pro Zeile), die nicht
eingebettet werden sollen. Wildcards (*) sind erlaubt.', 'oembed-manager' ); ?>
            </p>
            <textarea name="<?php echo esc_attr( $this->option_name ); ?>[blocked_urls]"
                      rows="10" class="large-text code"
                      placeholder="https://example.com/private/*&#10;https://badsite.com/*"
            ><?php echo esc_textarea( $blocked ); ?></textarea>
        </div>
        <?php
    }
```

```
    /**
     * Versteckte Felder rendern, damit andere Tabs nicht verloren gehen.
     *
     * @param string $active_tab
     */
    private function render_hidden_fields( $active_tab ) {
        $s = $this->settings;

        if ( 'providers' !== $active_tab ) {
            // Deaktivierte Provider als hidden fields
            $disabled = isset( $s['disabled_providers'] ) ? $s['disabled_providers'] :
array();
            foreach ( $disabled as $d ) {
                echo '<input type="hidden" name="' . esc_attr( $this->option_name ) .
'[disabled_providers][]" value="' . esc_attr( $d ) . '">';
            }
        }

        if ( 'display' !== $active_tab ) {
            $fields = array( 'max_width', 'max_height', 'cache_duration', 'wrapper_class' );
            foreach ( $fields as $f ) {
                $val = isset( $s[ $f ] ) ? $s[ $f ] : '';
                echo '<input type="hidden" name="' . esc_attr( $this->option_name ) . '[' . $f
. ']" value="' . esc_attr( $val ) . '">';
            }
            $checkboxes = array( 'responsive', 'lazy_loading', 'strip_inline_styles',
'disable_in_feed', 'disable_on_frontend' );
            foreach ( $checkboxes as $cb ) {
                if ( ! empty( $s[ $cb ] ) ) {
                    echo '<input type="hidden" name="' . esc_attr( $this->option_name ) . '['
. $cb . ']" value="1">';
                }
            }
        }

        if ( 'custom' !== $active_tab ) {
            $custom = isset( $s['custom_providers'] ) ? $s['custom_providers'] : array();
            foreach ( $custom as $i => $cp ) {
                echo '<input type="hidden" name="' . esc_attr( $this->option_name ) .
```

```
'[custom_providers][' . $i . '][pattern]" value="' . esc_attr( $cp['pattern'] ) . '">';
                echo '<input type="hidden" name="' . esc_attr( $this->option_name ) .
'[custom_providers][' . $i . '][endpoint]" value="' . esc_attr( $cp['endpoint'] ) . '">';
                if ( ! empty( $cp['regex'] ) ) {
                    echo '<input type="hidden" name="' . esc_attr( $this->option_name ) .
'[custom_providers][' . $i . '][regex]" value="1">';
                }
            }
        }

        if ( 'blocklist' !== $active_tab ) {
            $blocked = isset( $s['blocked_urls'] ) ? $s['blocked_urls'] : '';
            echo '<input type="hidden" name="' . esc_attr( $this->option_name ) .
'[blocked_urls]" value="' . esc_attr( $blocked ) . '">';
        }
    }
}
```

Schritt 4: Frontend-Filter

Diese Klasse modifiziert die oEmbed-Ausgabe im Frontend (Wrapper, responsive, Lazy Loading etc.).

```
<?php
if ( ! defined( 'ABSPATH' ) ) {
    exit;
}

/**
 * Modifiziert die oEmbed-Ausgabe im Frontend.
 */
class Jepp_OEM_Frontend {

    /** @var array */
    private $settings;

    public function __construct() {
        $this->settings = get_option( 'jepp_oem_settings', array() );
```

```
    // oEmbed komplett deaktivieren
    if ( ! empty( $this->settings['disable_on_frontend'] ) && ! is_admin() ) {
        add_action( 'init', array( $this, 'disable_oembed_completely' ) );
        return;
    }

    // Max-Width/Height setzen
    add_filter( 'embed_defaults', array( $this, 'set_embed_defaults' ) );

    // Cache-Dauer setzen
    add_filter( 'oembed_ttl', array( $this, 'set_cache_ttl' ), 10, 4 );

    // HTML-Ausgabe modifizieren
    add_filter( 'embed_oembed_html', array( $this, 'modify_output' ), 99, 4 );

    // Im Feed deaktivieren
    if ( ! empty( $this->settings['disable_in_feed'] ) ) {
        add_filter( 'embed_oembed_html', array( $this, 'disable_in_feed' ), 100, 4 );
    }

    // Frontend-CSS laden
    add_action( 'wp_enqueue_scripts', array( $this, 'enqueue_frontend_styles' ) );
}

/**
 * oEmbed komplett deaktivieren.
 */
public function disable_oembed_completely() {
    // Auto-Discovery deaktivieren
    remove_action( 'wp_head', 'wp_oembed_add_discovery_links' );
    remove_action( 'wp_head', 'wp_oembed_add_host_js' );

    // oEmbed REST-API Endpoint deaktivieren
    remove_action( 'rest_api_init', 'wp_oembed_register_route' );

    // oEmbed-Filter entfernen
    remove_filter( 'pre_oembed_result', 'wp_filter_pre_oembed_result', 10 );

    // Auto-Embeds deaktivieren
    remove_filter( 'the_content', array( $GLOBALS['wp_embed'], 'autoembed' ), 8 );
```

```
}

/**
 * Standard-Embed-Dimensionen setzen.
 *
 * @param array $defaults
 * @return array
 */
public function set_embed_defaults( $defaults ) {
    $max_w = $this->settings['max_width'] ?? 800;
    $max_h = $this->settings['max_height'] ?? 0;

    if ( $max_w > 0 ) {
        $defaults['width'] = $max_w;
    }
    if ( $max_h > 0 ) {
        $defaults['height'] = $max_h;
    }

    return $defaults;
}

/**
 * Cache-TTL setzen.
 *
 * @param int    $ttl
 * @param string $url
 * @param array  $attr
 * @param int    $post_id
 * @return int
 */
public function set_cache_ttl( $ttl, $url, $attr, $post_id ) {
    $custom_ttl = $this->settings['cache_duration'] ?? 86400;
    return (int) $custom_ttl;
}

/**
 * oEmbed-HTML-Ausgabe modifizieren.
 *
 * @param string $html
```

```
     * @param string $url
     * @param array  $attr
     * @param int    $post_id
     * @return string
     */
    public function modify_output( $html, $url, $attr, $post_id ) {
        if ( empty( $html ) ) {
            return $html;
        }

        // Inline-Styles entfernen
        if ( ! empty( $this->settings['strip_inline_styles'] ) ) {
            $html = preg_replace( '/\s+style\s*=\s*"[^"]*"/i', '', $html );
            $html = preg_replace( "/\s+style\s*=\s*'[^']*'/i", '', $html );
        }

        // Lazy Loading für iframes
        if ( ! empty( $this->settings['lazy_loading'] ) ) {
            $html = $this->add_lazy_loading( $html );
        }

        // Wrapper-Container
        $wrapper_class = $this->settings['wrapper_class'] ?? 'jepp-oembed-wrap';
        $is_video      = $this->is_video_embed( $html, $url );
        $responsive    = ! empty( $this->settings['responsive'] ) && $is_video;

        $classes = esc_attr( $wrapper_class );
        if ( $responsive ) {
            $classes .= ' jepp-oembed-responsive';
        }

        $html = '<div class="' . $classes . '">' . $html . '</div>';

        return $html;
    }

    /**
     * Lazy Loading Attribut zu iframes hinzufügen.
     *
     * @param string $html
```

```
     * @return string
     */
    private function add_lazy_loading( $html ) {
        // Nur wenn ein iframe vorhanden ist und noch kein loading-Attribut gesetzt
        if ( stripos( $html, '<iframe' ) !== false && stripos( $html, 'loading=' ) === false )
{
            $html = str_ireplace( '<iframe', '<iframe loading="lazy"', $html );
        }
        return $html;
    }

    /**
     * Prüft ob es sich um ein Video-Embed handelt.
     *
     * @param string $html
     * @param string $url
     * @return bool
     */
    private function is_video_embed( $html, $url ) {
        $video_hosts = array(
            'youtube.com', 'youtu.be', 'vimeo.com', 'dailymotion.com',
            'dai.ly', 'tiktok.com', 'twitch.tv', 'facebook.com/watch',
            'wistia.com', 'videopress.com',
        );

        foreach ( $video_hosts as $host ) {
            if ( stripos( $url, $host ) !== false ) {
                return true;
            }
        }

        // Fallback: iframe mit typischen Video-Attributen prüfen
        if ( preg_match( '/<iframe[^>]+src=["\'][^"\']*(?:video|embed|player)/i', $html ) ) {
            return true;
        }

        return false;
    }

    /**
```

```
     * Im Feed nur den Link anzeigen.
     *
     * @param string $html
     * @param string $url
     * @param array  $attr
     * @param int    $post_id
     * @return string
     */
    public function disable_in_feed( $html, $url, $attr, $post_id ) {
        if ( is_feed() ) {
            return '<a href="' . esc_url( $url ) . '">' . esc_html( $url ) . '</a>';
        }
        return $html;
    }

    /**
     * Frontend-Styles laden.
     */
    public function enqueue_frontend_styles() {
        wp_enqueue_style(
            'jepp-oem-frontend',
            JEPP_OEM_PLUGIN_URL . 'assets/css/frontend.css',
            array(),
            JEPP_OEM_VERSION
        );
    }
}
```

Schritt 5: Admin-CSS

```
/* oEmbed Manager – Admin Styles */

.jepp-oem-wrap {
    max-width: 1100px;
}

.jepp-oem-tabs {
    margin-bottom: 20px;
```

```
}

.jepp-oem-section {
    background: #fff;
    border: 1px solid #ccd0d4;
    border-radius: 4px;
    padding: 20px 24px;
    margin-top: 15px;
}

.jepp-oem-section h2 {
    margin-top: 0;
    padding-top: 0;
    border-bottom: 1px solid #eee;
    padding-bottom: 10px;
}

/* Provider-Tabelle */
.jepp-oem-provider-table {
    margin-top: 10px;
}

.jepp-oem-provider-table td,
.jepp-oem-provider-table th {
    vertical-align: middle;
    padding: 10px 12px;
}

.jepp-oem-provider-table code {
    font-size: 12px;
    background: #f0f0f1;
    padding: 2px 6px;
    border-radius: 3px;
    word-break: break-all;
}

.jepp-oem-provider-table tbody tr:nth-child(even) {
    background: #f9f9f9;
}
```

```
.jepp-oem-provider-table tbody tr:hover {
    background: #f0f6fc;
}

/* Eigene Provider */
.jepp-oem-custom-table td {
    vertical-align: middle;
    padding: 8px 10px;
}

.jepp-oem-custom-table input[type="text"],
.jepp-oem-custom-table input[type="url"] {
    width: 100%;
}

.jepp-oem-remove-row {
    color: #a00 !important;
    border-color: #a00 !important;
    font-weight: bold;
    font-size: 16px;
    line-height: 1;
    padding: 2px 8px !important;
}

.jepp-oem-remove-row:hover {
    color: #dc3232 !important;
    border-color: #dc3232 !important;
}

/* Provider Actions */
.jepp-oem-provider-actions .button {
    margin-right: 8px;
}
```

Schritt 6: Frontend-CSS

```
/* oEmbed Manager – Frontend Styles */
```

```
.jepp-oembed-wrap {
    max-width: 100%;
    margin: 1.5em 0;
    clear: both;
}

/* Responsive Video Container (16:9) */
.jepp-oembed-responsive {
    position: relative;
    padding-bottom: 56.25%; /* 16:9 */
    height: 0;
    overflow: hidden;
}

.jepp-oembed-responsive iframe,
.jepp-oembed-responsive object,
.jepp-oembed-responsive embed,
.jepp-oembed-responsive video {
    position: absolute;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    border: 0;
}
```

Schritt 7: Admin-JavaScript

```
/**
 * oEmbed Manager – Admin JavaScript
 */
(function ($) {
    'use strict';

    $(document).ready(function () {

        // --- Provider Tab: Alle aktivieren / deaktivieren ---
```

```
        // Invertierte Logik: Checkbox AN = Provider aktiv (NICHT in disabled-Liste)
        // Beim Absenden müssen wir die NICHT angehakten als "disabled" senden.

        // "Alle aktivieren" → alle Checkboxen abhaken
        $('.jepp-oem-select-all').on('click', function () {
            $('.jepp-oem-provider-checkbox').prop('checked', true);
        });

        // "Alle deaktivieren" → alle Checkboxen abhaken entfernen
        $('.jepp-oem-deselect-all').on('click', function () {
            $('.jepp-oem-provider-checkbox').prop('checked', false);
        });

        // Invertierte Logik: Vor dem Submit die Werte umkehren
        $('form').on('submit', function () {
            $('.jepp-oem-provider-checkbox[data-inverted="1"]').each(function () {
                // Wenn NICHT gecheckt → soll als disabled gesendet werden
                // Wenn gecheckt → NICHT senden (Provider ist aktiv)
                if ($(this).is(':checked')) {
                    $(this).prop('disabled', true); // Nicht mitsenden
                } else {
                    $(this).prop('checked', true);  // Wert mitsenden
                    $(this).prop('disabled', false);
                }
            });
        });

        // --- Custom Provider Tab: Zeilen hinzufügen/entfernen ---

        var $table = $('#jepp-oem-custom-table tbody');

        $('#jepp-oem-add-provider').on('click', function () {
            var idx = $table.find('tr').length;
            var optName = $('input[name*="custom_providers"]').length > 0
                ?
$('input[name*="custom_providers"]').first().attr('name').replace(/\[custom_providers\]\[\d+\]
.*/, '[custom_providers]')
                : 'jepp_oem_settings[custom_providers]';

            var row = '<tr class="jepp-oem-custom-row">' +
```

```
                '<td><input type="text" class="regular-text" name="' + optName + '[' + idx +
'][pattern]" placeholder="https://example.com/*"></td>' +
                '<td><input type="url" class="regular-text" name="' + optName + '[' + idx +
'][endpoint]" placeholder="https://example.com/oembed"></td>' +
                '<td><input type="checkbox" name="' + optName + '[' + idx + '][regex]"
value="1"></td>' +
                '<td><button type="button" class="button jepp-oem-remove-
row">&times;</button></td>' +
                '</tr>';

            $table.append(row);
        });

        $(document).on('click', '.jepp-oem-remove-row', function () {
            $(this).closest('tr').remove();
            // Indizes neu nummerieren
            $table.find('tr').each(function (i) {
                $(this).find('input').each(function () {
                    var name = $(this).attr('name');
                    if (name) {
                        $(this).attr('name', name.replace(/\[custom_providers\]\[\d+\]/,
'[custom_providers][' + i + ']'));
                    }
                });
            });
        });
    });
})(jQuery);
```

Schritt 8: Uninstall-Routine

```
<?php
/**
 * Wird beim Löschen des Plugins ausgeführt.
 * Entfernt alle gespeicherten Optionen und den oEmbed-Cache.
 */

if ( ! defined( 'WP_UNINSTALL_PLUGIN' ) ) {
```

```
    exit;
}

// Plugin-Option entfernen
delete_option( 'jepp_oem_settings' );

// oEmbed-Cache entfernen
global $wpdb;
$wpdb->query(
    "DELETE FROM {$wpdb->postmeta} WHERE meta_key LIKE '_oembed_%'"
);
$wpdb->query(
    "DELETE FROM {$wpdb->options} WHERE option_name LIKE '_transient_oembed_%' OR option_name
LIKE '_transient_timeout_oembed_%'"
);
```

Dateistruktur

```
oEmbed Plugin/
├── oembed-manager.php              ← Haupt-Plugin-Datei
├── uninstall.php                   ← Aufräumen bei Deinstallation
├── includes/
│   ├── class-jepp-oem-settings.php ← Admin-Einstellungsseite
│   ├── class-jepp-oem-providers.php← Provider-Verwaltung
│   └── class-jepp-oem-frontend.php ← Frontend-Filter
├── assets/
│   ├── css/
│   │   ├── admin.css               ← Admin-Styles
│   │   └── frontend.css            ← Frontend-Styles
│   └── js/
│       └── admin.js                ← Admin-JavaScript
└── languages/                      ← Übersetzungsdateien
```

Funktionsübersicht

Feature	Beschreibung
Provider an/aus	Jeden WordPress-Standard-Provider einzeln deaktivieren
Eigene Provider	Neue oEmbed-Endpunkte mit URL-Pattern hinzufügen
URL-Blockliste	Bestimmte URLs/Muster von der Einbettung ausschließen
Max. Dimensionen	Maximale Breite und Höhe für Embeds festlegen
Responsive Container	Automatisch 16:9-Container für Video-Embeds
Lazy Loading	`loading="lazy"` für iframes aktivieren
Inline-Styles entfernen	Unerwünschte Style-Attribute aus dem HTML entfernen
Feed-Deaktivierung	Im RSS-Feed nur Links statt Embeds anzeigen
Komplett deaktivieren	oEmbed auf dem gesamten Frontend abschalten
Cache-Steuerung	Cache-Dauer anpassen, Cache wird bei Änderungen geleert

Das Plugin ist sofort einsatzbereit – einfach den Ordner `oEmbed Plugin` nach `wp-content/plugins/` kopieren und im WordPress-Backend aktivieren. Die Einstellungen finden sich dann unter **Einstellungen → oEmbed Manager**.

oEmbed Plugin | eigene Provider definieren

```
<iframe height="300" style="width: 100%;" scrolling="no" title="GSAP - Animation Triad Demo: Timing, Motion, Meaning" src="https://codepen.io/editor/ZcarecroW/embed/019da77a-6995-7501-9af8-4866e69b7f1c?default-tab=html%2Cresult" frameborder="no" loading="lazy" allowtransparency="true">
  See the Pen <a href="https://codepen.io/editor/ZcarecroW/pen/019da77a-6995-7501-9af8-4866e69b7f1c">
  GSAP - Animation Triad Demo: Timing, Motion, Meaning</a> by ZcarecroW (<a href="https://codepen.io/ZcarecroW">@ZcarecroW</a>)
  on <a href="https://codepen.io">CodePen</a>.
</iframe>
```

CodePen als oEmbed-Provider hinzufügen

CodePen unterstützt offiziell das oEmbed-Protokoll. Du kannst es direkt über den **Tab "Eigene Provider"** in deinem Plugin hinzufügen.

Über die Admin-Oberfläche

Gehe zu **Einstellungen → oEmbed Manager → Eigene Provider** und füge folgende Werte ein:

Feld	Wert
URL-Pattern	`https://codepen.io/*/pen/*`
Endpoint-URL	`https://codepen.io/api/oembed`
Regex	☐ (nicht angehakt)

Das Problem mit deiner URL

Deine aktuelle URL hat das Format:

```
https://codepen.io/editor/ZcarecroW/embed/019da77a-6995-7501-9af8-4866e69b7f1c
```

Das ist eine **Editor/Embed-URL**, keine Standard-Pen-URL. oEmbed von CodePen funktioniert nur mit dem kanonischen Pen-Format. Du musst also in deinem WordPress-Beitrag diese URL einfügen:

```
https://codepen.io/ZcarecroW/pen/019da77a-6995-7501-9af8-4866e69b7f1c
```

Dann erkennt WordPress die URL automatisch und holt über den oEmbed-Endpoint die Einbettung.

Erweiterte Variante: Auch Editor-URLs unterstützen

Falls du auch die `/editor/`-URLs und `/embed/`-URLs automatisch auflösen möchtest, brauchst du einen zweiten Provider-Eintrag mit Regex. Da das URL-Pattern `https://codepen.io/*/pen/*` nur die Standard-Pen-URLs matcht, kannst du zusätzlich einen Regex-Provider anlegen:

Feld	Wert
URL-Pattern	`#https?://codepen\.io/.+/(pen|embed|full)/.*#i`
Endpoint-URL	`https://codepen.io/api/oembed`
Regex	☑ (angehakt)

Allerdings ist der sauberste Weg einfach die **kanonische Pen-URL** zu verwenden.

Testen

Du kannst prüfen, ob der oEmbed-Endpoint funktioniert, indem du folgende URL im Browser aufrufst:

```
https://codepen.io/api/oembed?url=https://codepen.io/ZcarecroW/pen/019da77a-6995-7501-9af8-4866e69b7f1c&format=json
```

Das sollte eine JSON-Antwort mit dem Embed-HTML zurückliefern.

Hinweis zum Provider-Matching

In deiner filter_providers-Methode werden Provider über `stripos` gematcht. Eigene Provider, die über register_custom_providers mit `wp_oembed_add_provider` registriert werden, sind davon nicht betroffen – sie werden direkt in WordPress registriert und funktionieren unabhängig vom Deaktivierungs-Filter.

> ich hab nun deine beiden definitionen eingetragen und auf einer wp page einen link eingefügt:
> https://codepen.io/editor/ZcarecroW/pen/019da77a-6995-7501-9af8-4866e69b7f1c
> leider ist es im frontend immer noch nur ein link.

CodePen oEmbed – Problemanalyse

Das Problem liegt an mehreren Stellen:

1. Die URL ist falsch

Die URL `https://codepen.io/editor/ZcarecroW/pen/...` ist **keine gültige CodePen-Pen-URL**. Das `/editor/` gehört dort nicht hin. Die korrekte URL lautet:

```
https://codepen.io/ZcarecroW/pen/019da77a-6995-7501-9af8-4866e69b7f1c
```

2. Das Provider-Matching hat ein Problem

In deiner filter_providers-Methode wird mit `md5($pattern)` als Key gearbeitet. Beim Deaktivieren von Providern wird der `md5`-Hash des Patterns als Vergleichswert gegen das Pattern und den Endpoint geprüft – das funktioniert korrekt für Standard-Provider. Aber das eigentliche Problem ist, dass **eigene Provider über `wp_oembed_add_provider` registriert werden**, was im `init`-Hook passiert – und der `oembed_providers`-Filter greift auf eine andere interne Liste zu. Das sollte eigentlich funktionieren.

3. Der eigentliche Grund: WordPress erkennt die URL nicht

WordPress macht bei oEmbed ein **Pattern-Matching** gegen die URL. Dein Pattern `https://codepen.io/*/pen/*` matcht nur URLs im Format:

```
https://codepen.io/{username}/pen/{id}
```

Aber **nicht** URLs mit `/editor/` darin.

Lösung

Schritt 1: Verwende die **richtige URL** in deinem WordPress-Beitrag. Füge in den Block-Editor einfach diese URL auf einer eigenen Zeile ein:

```
https://codepen.io/ZcarecroW/pen/019da77a-6995-7501-9af8-4866e69b7f1c
```

Schritt 2: Stelle sicher, dass dein eigener Provider korrekt eingetragen ist:

Feld	Wert
URL-Pattern	`https://codepen.io/*/pen/*`
Endpoint-URL	`https://codepen.io/api/oembed`
Regex	☐ (nicht angehakt)

Schritt 3: Leere den oEmbed-Cache. Speichere einfach nochmal die Plugin-Einstellungen – das löst flush_oembed_cache aus und löscht alle gecachten oEmbed-Einträge.

Verifikation

Du kannst prüfen, ob der CodePen oEmbed-Endpoint deine URL überhaupt akzeptiert. Öffne diese URL im Browser:

```
https://codepen.io/api/oembed?url=https://codepen.io/ZcarecroW/pen/019da77a-6995-7501-9af8-4866e69b7f1c&format=json
```

Wenn du eine JSON-Antwort mit `html`-Feld bekommst, funktioniert der Endpoint. Wenn du einen Fehler bekommst, liegt das Problem bei CodePen selbst (z.B. ist der Pen möglicherweise privat).

Optional: Auch Editor-URLs unterstützen

Falls du trotzdem auch `/editor/`-URLs unterstützen willst, brauchst du einen **zweiten** Provider-Eintrag mit Regex:

Feld	Wert
URL-Pattern	`#https?://codepen\.io/(?:editor/)?([^/]+)/(?:pen|embed|full)/(.+)#i`
Endpoint-URL	`https://codepen.io/api/oembed`
Regex	☑ (angehakt)

Wichtig: Auch mit Regex-Matching sendet WordPress die **originale URL** an den oEmbed-Endpoint. Wenn CodePen die `/editor/`-URL nicht als gültige Pen-URL akzeptiert, wird es trotzdem nicht funktionieren. Daher ist die sauberste Lösung: **Verwende die kanonische Pen-URL ohne `/editor/`**.

Skill (für KIs): oEmbed Provider Configuration Generator

> Skill: oEmbed Provider Configuration Generator

Role

You are an oEmbed provider configuration specialist. Your job is to analyze any given embed code (iframe, blockquote,
script tag, URL pattern, or API documentation) and produce the exact configuration values needed to register a custom
oEmbed provider in WordPress.

Output Format

For every provider you identify, return a table with exactly these three fields:

Field	Description
URL-Pattern	The URL scheme that matches embeddable content from this provider.
Endpoint-URL	The oEmbed API endpoint that returns embed metadata (JSON/XML).
Regex	`on` if the URL-Pattern is a regular expression, `off` if it uses simple wildcards (`*`).

Rules

1. **Analyze the input thoroughly.** The user may provide:
   - A raw `<iframe>` embed code
   - A `<blockquote>` + `<script>` embed snippet
   - A plain URL or set of example URLs
   - API documentation or a link to an oEmbed endpoint
   - A description of a service in natural language
2. **Determine the oEmbed endpoint.**
   - Check if the service is listed at https://oembed.com/providers.json — if so, use the official endpoint.
   - If the user provides an iframe `src`, extract the base domain and check for a well-known oEmbed endpoint at:
     - `https://{domain}/oembed`
     - `https://{domain}/api/oembed`
     - `https://{domain}/services/oembed`
   - If the service supports oEmbed auto-discovery (`<link rel="alternate" type="application/json+oembed" ...>`), mention this and extract the endpoint from the `href`.
   - If no oEmbed endpoint exists, clearly state: **"This service does not appear to support oEmbed natively. Consider using a proxy service like Iframely or Embedly, or implement a custom handler."**
3. **Construct the URL-Pattern.**
   - The pattern must match all embeddable content URLs from this provider (not the iframe src, but the canonical/public-facing URL users would paste into WordPress).
   - Prefer simple wildcard patterns using `*` (Regex = `off`) when possible. Examples:
     - `https://example.com/video/*`
     - `https://*.example.com/content/*`
   - Use a regex pattern (Regex = `on`) only when wildcard notation is insufficient, e.g.:
     - `#https?://(www\.)?example\.com/videos?/[\w-]+#i`
4. **Validate the endpoint URL.**
   - The endpoint must accept at least a `url` query parameter: `{endpoint}?url={content_url}&format=json`
   - Include `format=json` in the endpoint if the provider requires it.
   - Example: `https://example.com/oembed?format=json`
5. **Always provide a test instruction.** After the table, give the user a curl command or browser URL they can use to verify the endpoint works:

   ```
   Test: curl
   "https://example.com/oembed?url=https://example.com/video/12345&format=json"
   ```

6. **Handle edge cases:**
   - If the iframe contains no identifiable service, ask the user for more context (e.g., the public URL where this embed appears, or the service name).
   - If multiple providers are detected (e.g., a page embedding both YouTube and SoundCloud), return one table row per provider.

- If the embed uses a proprietary JavaScript SDK with no oEmbed support, explain alternatives (custom shortcode, manual iframe allowlisting).

7. **Output example for reference:**
   Input: `<iframe src="https://player.vimeo.com/video/76979871" width="640" height="360" frameborder="0"></iframe>`
   Output:

Field	Value
URL-Pattern	`https://vimeo.com/*`
Endpoint-URL	`https://vimeo.com/api/oembed.json`
Regex	off

> Note: The URL-Pattern uses the canonical Vimeo URL (`https://vimeo.com/76979871`), not the player/iframe URL. WordPress will send the canonical URL to the oEmbed endpoint.

```
Test: curl "https://vimeo.com/api/oembed.json?url=https://vimeo.com/76979871"
```

8. **Additional context the user may find helpful:**
   - Briefly explain the difference between the **canonical URL** (what users paste) and the **player/iframe URL** (what the embed code contains).
   - If the provider requires API keys or authentication for oEmbed, mention it.
   - If the provider returns `rich` type instead of `video` type, note this as it may affect responsive wrapper behavior.

Response Language

Always respond in the same language the user writes in. The configuration values themselves (URLs, patterns) remain as-is regardless of language.