ac_tags

Classe JS autonome — Saisie de tags avec badges

Jean-Noël Vidallet · artisan-code.fr · Mars 2026

1. Présentation

ac_tags est une classe JavaScript autonome qui transforme un input hidden en composant de saisie de tags. Les tags s'affichent sous forme de badges colorés avec suppression au clic. Elle supporte l'autocomplétion AJAX et le mode lecture seule avec liens cliquables.

Cas d'usage

- Catégorisation des articles de blog d'un auto-entrepreneur
- Mots-clés pour la recherche et le filtrage de contenus
- Génération de pages virtuelles par tag (URL rewriting)
- Affichage de tags cliquables sur une page vitrine

Caractéristiques principales

- Classe JS unique — CSS embarqué, zéro dépendance
- Validation à l'Entrée ou à la virgule
- Suppression au clic sur ✕ ou Backspace sur champ vide
- Détection de doublons côté client
- Autocomplétion AJAX optionnelle (dès 2 caractères)
- Navigation clavier dans le dropdown (flèches, Entrée, Echap)
- Normalisation optionnelle en minuscules
- Mode lecture seule avec liens cliquables vers pages de tags
- Couleur des badges configurable
- Copyright discret cliquable vers artisan-code.fr

2. Installation et fichiers

Fichier	Destination
ac_tags.js	/assets/progiciel/classes/js/ac_tags.js

Inclusion dans la page (avant </body>) :

```
<script src="/assets/progiciel/classes/js/ac_tags.js"></script>
```

3. Utilisation minimale

Dans le HTML, placer un input hidden avec un id unique :

```
<input type="hidden" id="mes_tags" name="mes_tags">
```

Instancier la classe :

```
const tags = new ac_tags('mes_tags');
```

Les tags saisis sont stockés dans l'input hidden, séparés par une virgule. C'est cet input qui est envoyé au serveur lors du submit du formulaire.

4. Options de configuration

Option	Type	Défaut	Description
max	int	10	Nombre maximum de tags autorisés
normaliser	bool	true	Convertir automatiquement en minuscules
separateur	string	`','`	Caractère de séparation dans l'input hidden
placeholder	string	`'Ajouter un tag…'`	Texte indicatif du champ de saisie
largeur	string	`'100%'`	Largeur CSS du composant
couleur	string	`'#5a7a62'`	Couleur de fond des badges
suggestions_url	string	null	URL endpoint AJAX autocomplétion. null = pas de suggestions
url_tag	string	null	Pattern URL en lecture seule. Ex: '/blog/tag/{tag}'
lecture_seule	bool	false	true = affichage uniquement, pas de saisie
callback	function	null	fn(tags) — appelée à chaque modification. Reçoit le tableau des tags.

5. Comportements clavier

Touche	Action
Entrée	Valide le tag en cours de saisie
Virgule	Valide le tag en cours de saisie
Backspace	Sur champ vide : supprime le dernier tag
Flèche bas	Sélectionne l'item suivant dans le dropdown
Flèche haut	Sélectionne l'item précédent dans le dropdown
Echap	Ferme le dropdown de suggestions

6. Autocomplétion

Quand suggestions_url est renseigné, l'autocomplétion se déclenche dès 2 caractères saisis. Un appel AJAX est envoyé avec le paramètre q.

Format de réponse attendu du serveur

Le endpoint doit retourner un tableau JSON de chaînes :

```
["yoga", "yogi", "yoyotte"]
```

Ou un tableau d'objets avec label :

```
[{"id": 1, "label": "yoga"}, {"id": 2, "label": "yogi"}]
```

Exemple endpoint PHP

```php
<?php
header('Content-Type: application/json');
$q = strtolower(trim($_GET['q'] ?? ''));
$stmt = $pdo->prepare('SELECT nom FROM tags WHERE nom LIKE ? ORDER BY nom LIMIT 10');
$stmt->execute([$q . '%']);
echo json_encode($stmt->fetchAll(PDO::FETCH_COLUMN));
```

⚠️ *Les tags déjà saisis sont automatiquement exclus des suggestions côté client — pas besoin de les*

filtrer côté serveur.

7. Mode lecture seule

En mode lecture seule, les tags s'affichent sans champ de saisie ni bouton de suppression. Si url_tag est renseigné, chaque badge devient un lien cliquable.

Sans lien

```
new ac_tags('mes_tags', {
    lecture_seule : true,
    couleur       : '#6366f1'
});
```

Avec liens cliquables

```
new ac_tags('mes_tags', {
    lecture_seule : true,
    url_tag       : '/blog/tag/{tag}',
    couleur       : '#0891b2'
});
```

Le paramètre {tag} est remplacé par la valeur du tag encodée pour l'URL. Exemple : "bien-être" → /blog/tag/bien-%C3%AAtre.

> ⚠️ *url_tag n'a d'effet qu'en mode lecture seule. En mode édition les badges ne sont jamais des liens.*

8. Structure MySQL recommandée

Pour exploiter pleinement les tags (recherche, filtrage, pages de catégorie), il est recommandé de ne pas les stocker en chaîne dans une colonne TEXT mais dans des tables dédiées.

```
-- Table des tags
CREATE TABLE tags (
    id_tag  INT AUTO_INCREMENT PRIMARY KEY,
    nom     VARCHAR(100) NOT NULL UNIQUE
);

-- Table de liaison articles <-> tags
CREATE TABLE articles_tags (
    id_article INT NOT NULL,
    id_tag     INT NOT NULL,
    PRIMARY KEY (id_article, id_tag)
);
```

> ⚠️ *C'est côté PHP que les tags reçus (chaîne séparée par des virgules) sont éclatés et insérés dans ces tables. ac_tags ne gère que la partie saisie/affichage.*

9. Méthodes publiques

Méthode	Retour	Description
getTags()	array	Retourne le tableau des tags saisis
setTags(array)	void	Charge une liste de tags — utile en mode édition

Méthode	Retour	Description
addTag(tag)	bool	Ajoute un tag programmatiquement. Retourne false si doublon ou max atteint.
clear()	void	Supprime tous les tags

Exemple

```
const tags = new ac_tags('mes_tags', {
    couleur  : '#5a7a62',
    callback : function(t) { console.log('Tags :', t); }
});

tags.setTags(['yoga', 'méditation', 'bien-être']);
tags.addTag('pranayama');  // retourne true
tags.addTag('yoga');       // retourne false (doublon)
tags.getTags();            // ['yoga', 'méditation', 'bien-être', 'pranayama']
tags.clear();              // vide tout
```

10. Lecture PHP côté serveur

L'input hidden envoie les tags sous forme de chaîne séparée par des virgules. C'est le PHP qui les éclate et les traite.

```
// Réception
$tags_str = $_POST['mes_tags'] ?? '';  // 'yoga,méditation,bien-être'
$tags     = array_filter(array_map('trim', explode(',', $tags_str)));

// Insertion en base
foreach ($tags as $tag) {
    // Insérer dans tags si inexistant
    $stmt = $pdo->prepare('INSERT IGNORE INTO tags (nom) VALUES (?)');
    $stmt->execute([$tag]);
    $id_tag = $pdo->lastInsertId() ?: $pdo->query(
        'SELECT id_tag FROM tags WHERE nom = ' . $pdo->quote($tag))->fetchColumn();
    // Lier à l'article
    $stmt = $pdo->prepare('INSERT IGNORE INTO articles_tags VALUES (?, ?)');
    $stmt->execute([$id_article, $id_tag]);
}
```