ac_big_select

Widget Classothèque — sélection à fort volume avec recherche AJAX

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

1. Présentation

ac_big_select est un widget JavaScript 100% autonome qui remplace un <select> natif à fort volume d'options (plusieurs milliers de lignes) par un champ de recherche à la demande. Conçu pour s'intégrer nativement comme widget de colonne dans ac_datagrid (>= 3.9.0), mais utilisable de façon autonome. CSS auto-injecté une seule fois au premier appel — aucun fichier externe requis.

Caractéristiques principales :

- Classe JS pure — zéro dépendance (pas de jQuery, pas de framework)
- CSS auto-injecté — une seule fois, jamais de doublon
- Recherche à la demande — requête AJAX à partir d'un seuil de caractères configurable (2 par défaut), jamais de préchargement complet de la liste
- Anti-rebond (250 ms par défaut) — une seule requête après une pause de frappe, pas une par caractère
- Jeton de requête — ignore une réponse serveur devenue obsolète si une frappe plus récente a eu lieu entre-temps
- Navigation clavier complète — flèches haut/bas, Entrée, Échap
- Sélection à la souris sur un résultat de la liste
- Bouton d'effacement (✕) pour revenir en mode recherche
- Messages explicites — invite à taper au moins N lettres, message dédié si aucun résultat
- Liste de résultats en position fixed, attachée à document.body — jamais rognée par un conteneur à défilement contraint (modale, tableau avec défilement horizontal)
- Intégration ac_datagrid native (>= 3.9.0) — contrat « widget » standard, déclare ses modes d'édition supportés via une propriété statique

2. Installation

Fichier	Destination
ac_big_select.js	/assets/classes/ac_big_select.js

Inclusion dans la page :

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

Ancre HTML requise (générée et gérée automatiquement par ac_datagrid ; à créer soi-même uniquement pour un usage hors grille) :

```
<div id="mon-ancre"></div>
```

3. Utilisation minimale (hors grille)

```
const sel = new ac_big_select("mon-ancre", {
    _acgr_url: "/mon_endpoint_ajax.php",
    _acgr_id: "ma_grille",
    _champ: "id_beneficiaire",
});
```

```
// Pré-sélectionner une valeur existante
sel.setValeur(42, "Martin Sophie");

// Lire la sélection après un choix de l'utilisateur
const id = sel.getValeur(); // "42"
const libelle = sel.getLibelle(); // "Martin Sophie"
```

4. Format attendu d'une option (réponse serveur)

Propriété	Type	Requis	Description
id	any	oui	Identifiant retourné par getValeur() une fois l'option sélectionnée
libelle	string	oui	Texte affiché dans la liste de résultats et dans le champ une fois sélectionné

5. Options du constructeur

Option	Type	Défaut	Description
placeholder	string	"Rechercher…"	Texte affiché dans le champ avant toute saisie
seuil	number	2	Nombre minimum de caractères avant de déclencher une recherche
debounce_ms	number	250	Délai d'anti-rebond avant l'envoi de la requête, en millisecondes
limite	number	50	Déclarée mais non transmise à la requête serveur dans cette version — la limite réelle (LIMIT 50) est imposée côté serveur par ac_datagrid, indépendamment de cette valeur
vide_msg	string	"Aucun résultat"	Message affiché quand la recherche ne renvoie aucune option
invite_msg	string	auto	Message affiché tant que le seuil n'est pas atteint — calculé automatiquement (« Tapez au moins N lettres ») à partir de seuil si non fourni
_acgr_url	string	null	URL de l'endpoint AJAX — injectée automatiquement par ac_datagrid ; à fournir soi-même hors grille
_acgr_id	string	null	Identifiant de la grille — injecté automatiquement par ac_datagrid ; à fournir soi-même hors grille
_champ	string	null	Nom du champ de colonne — injecté automatiquement par ac_datagrid ; à fournir soi-même hors grille

6. Mécanisme de recherche — contrat serveur

Dès le seuil de caractères atteint, le widget envoie une requête POST vers _acgr_url :

```
acgr_id = _acgr_id
acgr_action = "options_recherche"
champ = _champ
terme = <texte saisi>
```

Réponse JSON attendue :

```
{ "ok": true, "options": [ { "id": 42, "libelle": "Martin Sophie" }, ... ] }
```

Le seuil de caractères est également vérifié côté serveur (défense en profondeur) et la limite de résultats (LIMIT 50) est imposée indépendamment de toute configuration client.

Le jeton de requête (compteur incrémenté à chaque nouvelle recherche) permet d'ignorer une réponse arrivée après qu'une frappe plus récente a relancé une autre recherche — la réponse la plus ancienne des deux est silencieusement écartée.

7. États du champ

7.1 Recherche (avant sélection)

Le focus sans saisie affiche le message d'invite. Dès le seuil atteint, l'anti-rebond puis la requête se déclenchent. Les résultats s'affichent dans une liste, navigable au clavier (flèches, Entrée, Échap) ou à la souris.

7.2 Sélection effectuée

Le champ se referme sur le libellé choisi, le bouton d'effacement (✕) apparaît pour revenir en recherche. Un événement ac-bigselect:change est émis (bubbles) à chaque sélection ou effacement, avec e.detail.valeur et e.detail.libelle.

> ⚠️ *Choix volontairement simple : la liste de résultats est fermée (plutôt que repositionnée en continu) au moindre défilement ou redimensionnement de la fenêtre — une liste en position fixed ne peut pas suivre le défilement d'un conteneur interne (modale, tableau).*

8. API publique

Méthode	Retour	Description
getValeur()	string	Retourne l'identifiant sélectionné, chaîne vide si aucune sélection
getLibelle()	string	Retourne le libellé affiché de la sélection courante
setValeur(id, libelle)	void	Définit la sélection par programmation (utilisée par ac_datagrid au chargement d'une ligne existante)

Événement DOM émis sur l'ancre (bubbles) :

ac-bigselect:change

e.detail.valeur et e.detail.libelle — émis à chaque sélection ou effacement.

9. Exemple complet — intégration ac_datagrid

```
['champ' => 'id_beneficiaire',
 'type' => 'select',
 'widget' => 'ac_big_select',
 'champ_libelle' => 'lib_beneficiaire',
 'sql_options_ajax' => 'SELECT id_beneficiaire AS id, CONCAT(nom, " ", prenom) AS
libelle
                        FROM dat_beneficiaires
                        WHERE id_asso = :id_asso AND actif = 1
                          AND CONCAT(nom, " ", prenom) LIKE :terme
                        ORDER BY nom, prenom',
 'sql_options' => 'SELECT id_beneficiaire AS id, CONCAT(nom, " ", prenom) AS libelle
                        FROM dat_beneficiaires WHERE id_asso = :id_asso']
```

ac_datagrid injecte automatiquement _acgr_url, _acgr_id et _champ dans widget_options au moment d'instancier le widget — rien à fournir explicitement côté config de colonne pour ces trois clés.

10. Intégration avec ac_datagrid

ac_datagrid (>= 3.9.0) interroge la propriété statique MODES_SUPPORTES du widget avant de l'instancier. ac_big_select déclare ['modal', 'inline'] : actif dans ces deux modes d'édition. En edit_mode:'cell', le <select> natif reste utilisé — par choix, pas par limite technique.

> ⚠️ *Un widget qui ne déclare pas static MODES_SUPPORTES reste actif en modal uniquement (comportement historique, aucune régression pour les widgets existants comme ac_carte ou ac_telephone_multipays).*

Détail cosmétique connu et volontairement laissé tel quel : un double cadre visuel peut apparaître autour du champ (anneau de focus natif du navigateur superposé à la bordure du widget) — accepté comme repère visuel pour identifier un champ ac_big_select en un coup d'œil.