ac_datagrid

Documentation technique v3.14.0

Grille de données PHP/JS — CRUD, édition cellule, import CSV, formatage conditionnel

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

1. Présentation

ac_datagrid est une classe PHP qui génère une grille de données interactive complète : affichage, tri, filtres, pagination, CRUD (ajout, édition, suppression), export, import CSV, formatage conditionnel et affichage de vignettes images. Tout est généré côté serveur en PHP et exécuté côté client en JavaScript vanilla pur, sans dépendance externe.

La classe est propriétaire (artisan-code.fr) et fait partie de la Classothèque.

Depuis la v3.12.0, ac_datagrid partage son noyau JavaScript (rendu, tri, pagination, édition, export, sous-grilles) avec ac_datagrid_pro, la version JS pure pour les projets sans backend PHP (Laravel, Symfony, API REST). Les deux classes sont compilées depuis les mêmes sources via un script de build commun ; seul le transport réseau diffère. Cette documentation ne couvre que ac_datagrid — voir la documentation dédiée pour ac_datagrid_pro.

1.1 Caractéristiques principales

- PHP/JavaScript vanilla — zéro dépendance externe
- Modes d'édition : inline, modal, cell (double-clic cellule)
- Import CSV avec mapping colonnes, prévisualisation et rapport d'erreurs
- Formatage conditionnel numérique (couleur texte selon seuil)
- Affichage vignettes images avec lightbox
- Duplication de ligne avec avertissement doublons
- Suppression des lignes épinglées (suppression multiple)
- Sous-grilles imbriquées (maître/détail)
- Colonnes fixes (sticky), redimensionnables, masquables, réordonnables
- Filtres par colonne, recherche globale, pagination serverside
- Export XLSX (Excel/LibreOffice)
- Totaux par colonne, totaux généraux serverside
- Cookies de personnalisation (ordre, colonnes, filtres, tris)
- CSRF intégré, versioning, gestion d'erreurs unifiée

1.2 Changelog v3

Version	Fonctionnalité
`3.0.0`	edit_mode:'cell' — édition cellule par double-clic
`3.1.0`	Duplication de ligne — bouton ⧉, avertissement doublons
`3.1.1`	Renommage is_admin→delete, bouton_ajouter→ajout. Correctif grille_id dans avant/après-suppression
`3.2.0`	Import CSV — 3 étapes : upload, mapping colonnes, prévisualisation + rapport
`3.2.1`	Suppression des lignes épinglées — bouton toolbar + événements
`3.3.0`	Formatage conditionnel numérique + affichage vignettes images avec lightbox
`3.4.0`	Accessibilité WCAG 2.1 AA — ARIA, navigation clavier
`3.5.0`	45 tests unitaires, documentation API HTML + ODT
`3.5.1`	Correction ajout_mode + coloration ligne en édition
`3.5.2`	Event acgr:row-click, sélection visuelle de ligne
`3.5.3`	e.detail simplifié à {id, grille_id, selected}

Version	Fonctionnalité
`3.5.4`	Corrections read_only — initModeCell, étoile, suppression groupée
`3.5.5`	Suppression outline sur .acgr-tr-selected
`3.5.6`	Suppression double outline mode cell
`3.5.7`	Filtres externes de type date (date_debut / date_fin)
`3.5.8`	Debounce recherche serverside, registre window.acGrilleInstances, subgrid_param séparé
`3.5.9`	masquerColonne / reafficherColonne appellent charger() en serverside
`3.5.10`	Correction focus perdu à la frappe en clientside, touche Échap pour vider la recherche
`3.5.11`	Option vide «—» en tête des selects en modale d'ajout
`3.5.12`	Scroll vers la grille fille après row-click (setTimeout 100ms)
`3.5.13`	Formatage conditionnel date, miniatures images avec lightbox, duplication configurable
`3.5.14`	castValeur() retourne null pour select FK vide (fix Incorrect integer value)
`3.5.15`	Correction boucle ajouterLigne mode inline (PK écrasée → mauvais événement émis)
`3.5.16`	Migration page Classothèque ac_datagrid — bandeau formatage conditionnel
`3.5.17`	Correction cookies : filtres auto et recherche globale non sauvegardés
`3.5.18`	Fix transparence colonnes fixées (sticky) sur lignes sélectionnées/en édition
`3.5.19`	Selects en cascade : pré-filtrage automatique à l'ouverture de l'édition + sélection de la valeur courante (tentative non fonctionnelle, corrigée en v3.7.5)
`3.5.20`	Cookies : sauvegarde/restauration nb lignes par page ; restauration correcte des largeurs de colonnes (table-layout:fixed après lock des largeurs auto) ; bouton ↺ visible si nb lignes modifié
`3.6.0`	Export XLSX natif (remplace ODS) — même API, .xlsx compatible Excel et LibreOffice ; option epingle:false pour masquer la colonne étoile (ignoré si read_only:true) ; tri multi-colonnes par Shift+clic avec indicateurs visuels ▲▼ et numéro de priorité, persistance localStorage ; cascade en mode cell (AJAX si col.parent) ; nouvelle ligne en tête page 1 (ajout + duplication) ; pageCourante forcé après INSERT modal
`3.6.2`	Option insert:false sur une colonne — empêche son insertion en BDD (utile pour les colonnes FK issues d'un JOIN utilisées comme parent de cascade)
`3.6.3`	Préférences grille (tri, colonnes masquées, largeurs, pagination) migrées de cookies vers localStorage — élimine les erreurs 400 Bad Request liées au dépassement de limite du header Cookie (~8 KB Apache)
`3.6.4`	Option edit_mode:'none' — grille sans aucune édition, sans bouton Valider/Annuler ; bloque nativement toute tentative de sauvegarde
`3.6.5`	Mode cell : blur sur select et checkbox déclenche la validation immédiate (cohérent avec text/number/date/textarea)
`3.6.6`	Largeurs de colonnes déclarées en configuration prises en compte dès l'initialisation (mode table-layout:auto qui respecte les style.width posés sur chaque <th>). Persistance des largeurs dans le localStorage DÉSACTIVÉE pour contourner un bug d'interaction navigateur/CSS sur les grilles sans colonne sticky. Le drag de redimensionnement reste fonctionnel pendant la session ; les autres préférences (ordre, masquage, filtres, tris, recherche, pagination) restent persistées.
`3.6.7`	Bug fix : longueur de pagination. Lors d'un changement de configuration (`longueur_defaut`), le localStorage périmé n'écrase plus la nouvelle valeur de

Version	Fonctionnalité
	configuration. La config est désormais prioritaire si elle a changé depuis la dernière session. Les préférences utilisateur (localStorage) ne sont restaurées que si la config n'a pas changé.
3.6.8	Scrollbar horizontale miroir au-dessus de la grille (synchronisée, affichée automatiquement en cas de débordement). Colonne Actions masquée si aucun bouton possible (edit_mode:'none' + duplication:false + non admin). Hooks publics JS : container.acgrRefreshScroll(), événement acgr:refresh-scroll (ciblé), acgr:refresh-scroll-all (global — toutes les grilles) pour recalculer la scrollbar miroir depuis l'extérieur, notamment lors de l'activation d'un onglet.
3.7.0	Event acgr:log dispatché sur document après chaque écriture BDD confirmée (CREATE/UPDATE/DELETE). Permet aux projets hôtes de logger les actions sans modifier la classe. La clé table doit être présente dans la config JS. Les projets qui n'écoutent pas cet event ne sont pas perturbés.
3.7.1	Option total:false par colonne — exclut une colonne number des totaux automatiques (utile pour les champs numériques non sommables, ex : une année). Patch PHP (SUM serverside), buildConfigJS (transmission PHP→JS) et buildTfoot (calcul JS).
3.7.2	Symbole Σ/∑ toujours visible en pied de grille. Quand la colonne étoile est absente (epingle:false ou read_only:true), le symbole se loge dans la première colonne non-totalisée (l'ID en pratique) via un flag labelPose dans buildLigneTotaux. Aucune régression sur les grilles avec épinglage actif.
3.7.3	Mode lecture d'une colonne checkbox : affichage d'une vraie case à cocher désactivée (au lieu du texte brut 0/1), avec pointer-events:none pour que le double-clic traverse jusqu'au <td> et garde la cellule éditable. Comparaison de valeur cohérente entre lecture et édition (val est une string issue du JSON).
3.7.4	Séparation des listes d'options d'une colonne select : options (alimentée en priorité par sql_options) pour le <select> d'édition/création, et options_display (alimentée en priorité par sql_options_display) pour résoudre le libellé d'une valeur déjà enregistrée en lecture. Permet un dropdown de saisie filtré (ex : masquer des valeurs standards) tout en gardant l'affichage des lignes existantes correct. Rétrocompatible : si une seule requête est fournie, les deux listes retombent dessus ; si les deux sont identiques, elles ne sont exécutées qu'une fois.
3.7.5	Correction du pré-remplissage des selects en cascade (parent + sql_options avec :parent) à l'ouverture de l'édition en edit_mode:'inline'. La tentative précédente comparait la valeur à resélectionner avec input.value, relu sur un <select> qui vient d'être créé sans ses vraies options (elles dépendent du parent et ne sont jamais chargées à l'avance) — la comparaison se faisait donc toujours contre une chaîne vide, et aucune option n'était jamais présélectionnée. Le correctif utilise directement ligne[col.champ] (la valeur réelle transmise par le serveur) et cible uniquement le select concerné, sans recharger à tort les colonnes sœurs partageant le même parent. Même principe que la cascade en mode cell (v3.5.21), qui n'a jamais eu ce défaut. Aucun changement de config requis côté page.
3.7.6	Correction d'un affichage intermittent d'ID brut au lieu du libellé, juste après l'ajout d'une ligne référençant une FK créée après le chargement de la page (ex : une inscription à un cours ajoutée dans la même session, puis une participation saisie pour ce bénéficiaire sans recharger). Cause : options_display est chargée une seule fois au rendu PHP de la page, contrairement à options d'une colonne cascade, rechargée en direct via AJAX à chaque édition. Le correctif mémorise le libellé affiché du <select> au moment de la saisie (validerLigne()) et l'ajoute à options_display après succès de l'écriture (nouvelle fonction enrichirOptionsDisplay()), avant le re-rendu de la ligne. Transparent pour toutes les grilles existantes, aucun changement de config requis côté page.
3.8.0	Nouvelle option show_nbr_lines (défaut true) — masque à la source le sélecteur « Afficher X lignes » quand false, sans générer ni la div acgr-select-wrap ni sa règle CSS. Remplace un contournement CSS !important côté projet hôte devenu inutile.

Version	Fonctionnalité
	Rétrocompatible : absent de la config, comportement inchangé. Refonte du bandeau read_only : texte par défaut plus explicite (« Consultation — ce tableau n'est pas modifiable avec votre rôle. » au lieu de « Consultation uniquement — modifications désactivées. »), icône œil au lieu du cadenas, style neutre piloté par les variables de thème au lieu d'un jaune/orange d'alerte codé en dur, et largeur ajustée au texte au lieu de pleine largeur. read_only_msg reste surchargeable comme avant.
`3.9.0`	Sélection à fort volume pour les colonnes select : nouvelle clé sql_options_ajax (recherche AJAX à partir de 2 caractères, LIMIT 50 côté serveur) et nouvelle clé champ_libelle (le libellé est fourni par une jointure dans le SQL principal plutôt que par une liste options_display préchargée). Nouvelle action AJAX options_recherche. Les widgets Classothèque peuvent désormais s'activer en mode inline, pas seulement modal : chaque widget déclare sa propre propriété statique MODES_SUPPORTES (repli sur ['modal'] si absente — aucun changement de comportement pour les widgets existants qui ne la déclarent pas). Nouveau widget Classothèque : ac_big_select. Design strictement additif : toutes les nouvelles clés sont optionnelles et absentes par défaut, zéro impact sur les grilles existantes.
`3.9.1`	Correction d'un bug préexistant (antérieur à cette version, invisible tant qu'ajout_mode:'modal' n'était pas utilisé sans widget) : la branche modale de l'ajout de ligne écrasait l'ID temporaire négatif de la clé primaire par sa valeur defaut en itérant toutes les colonnes, faute d'exclure explicitement la PK de cette boucle — contrairement à la branche inline, qui a toujours eu cette garde. Conséquence sur les tables dont l'injection de champs système dépend d'un ID strictement négatif : l'INSERT échouait par violation de clé étrangère. Corrigé en excluant la PK de la boucle d'initialisation, dans la branche modale comme dans la branche inline.
`3.9.2`	Nouveau type de colonne datetime, pour les colonnes DATETIME en base (à ne plus déclarer en date, qui tronque silencieusement toute valeur porteuse d'une heure). Rendu en <input type="datetime-local"> (sans les secondes). Le filtre automatique de colonne compare par jour (DATE(champ) côté serveur), l'heure n'entre pas en compte dans le filtrage. Additif : aucune colonne date existante n'est affectée.
`3.9.3`	Affichage des colonnes de type date au format jj/mm/aaaa en lecture.
`3.9.4`	Dédoublonnage de masquerColonne(), reafficherColonne() et ouvrirPanneauColonnes() — deux définitions concurrentes du même nom coexistaient, JavaScript ne conservait que la dernière déclarée. Ajout du préfixe cfg.id sur les id de checkbox du panneau de colonnes masquées, pour éviter toute collision entre grilles sur une même page.
`3.10.0`	Passage à une architecture de développement src/ + build.php : le code source est désormais réparti en fichiers PHP/JS/CSS distincts, recompilés en un seul fichier ac_datagrid.php par un script de build local (aucun changement pour les pages hôtes, qui continuent d'inclure un fichier unique sans dépendance). Fusion de render() et renderServerside() en une seule fonction, ce qui corrige au passage un oubli : les widgets inline (ex. ac_big_select) et la resynchronisation de la scrollbar miroir du haut, jusqu'ici actifs uniquement en mode client, le sont désormais aussi en serverside.
`3.10.1`	Fusion des branches modale et inline de ajouterLigne() en une fonction construireNouvelleLigne() commune. Corrige au passage : la valeur par défaut d'une colonne (option defaut) n'était appliquée qu'en ajout_mode:'modal', jamais en ajout_mode:'inline' — c'est désormais le cas dans les deux modes. Corrige aussi un bug de focus : après un ajout inline, le curseur ne se plaçait jamais dans le premier champ de la nouvelle ligne (parenthèses d'appel manquantes dans le setTimeout).
`3.10.2`	Deux chantiers. (1) build.php compacte désormais intégralement le fichier livrable : retrait de tous les commentaires PHP/JS/CSS, aucun retour à la ligne dans le JS, le CSS ni le PHP (les nowdoc sont convertis en chaînes PHP simple-quote au moment du build), bannière de fichier généré en tête, et une option --dev pour produire à la place un build lisible et diagnostiquable en sandbox. Aucun changement fonctionnel : les sorties générées à l'exécution sont identiques à l'octet près. (2) Dédoublonnage de validerChampsRequis() (partagée entre l'ajout modal et l'ajout inline, comportement de

Version	Fonctionnalité
	surlignage en modale préservé) et de attacherEcouteursCell() (partagée entre les cellules select et checkbox en edit_mode:'cell') — ce second dédoublonnage a révélé puis corrigé un bug d'édition introduit lors du même chantier : une structure if/else mal reconstituée faisait que toute cellule en édition affichait la valeur brute au lieu du libellé pour une colonne select, quel que soit son type.
`3.11.0`	Cinq chantiers, tous testés unitairement (batterie tests/run.php, 12 tests) et rétrocompatibles par défaut. (1) Export CSV : nouvelle clé export_formats (défaut ['xlsx'] — comportement inchangé pour toute grille existante) ; un tableau à 2 entrées transforme le bouton unique en menu déroulant. (2) Action custom : bouton optionnel dans la colonne Actions (action_custom, défaut false), tire l'événement acgr:action-custom avec {id, ligne, table, grille_id} — fonctionne même en read_only. (3) Agrégats multiples : col.agregat (somme | moyenne | min | max | nombre, défaut somme) remplace le SUM unique des totaux, en serverside (SQL) comme en clientside (JS) ; préfixe visuel par cellule (⌀ ▼ ▲ #) uniquement pour les agrégats non par défaut. (4) Hook serveur on_avant_sauvegarde : callable exécuté juste avant l'INSERT/UPDATE, reçoit tous les champs (y compris invisibles/FK), doit retourner le tableau, peut bloquer via throw new RuntimeException() — voir §12 Hooks serveur PHP. (5) Types email / url / tel / color : pattern de validation par défaut (overridable via col.pattern), ignoré si un widget est configuré sur la colonne. Correctif associé : validerChampsRequis() s'applique désormais aussi en edit_mode:'cell' (jusqu'ici seul le contrôle serveur s'appliquait dans ce mode — amélioration UX, aucun changement sur ce qui peut être enregistré).
`3.12.0`	Fusion architecturale avec ac_datagrid_pro (noyau JS commun). Nouveautés partagées : preset (mini/recherche/edition/edition_avancee/complet), format_conditionnel multi-règles, epingle_page_1, événement acgr:apres-chargement, e.detail.row sur acgr:row-click. Voir section 13.
`3.13.0`	Rétrocompatible par défaut (aucune grille existante en prod affectée sans configuration explicite). Correctifs : 'defaut' de colonne (option jusque-là ignorée par le rendu JS) ; collision du registre window.acGrilleInstances entre sous-grilles (alias sur cfg.id, cfg.acgr_id conservé en rétrocompatibilité). Ajouts colonne : placeholder, aide (édition + en-tête), min/max/step (number), maxlength (text/textarea), decimales (plafond de décimales à l'affichage, applicable à toute colonne number, indépendant de monetaire), pk (clé primaire explicite, prioritaire sur l'heuristique), editable:'creation' (saisissable uniquement à la création, verrouillé et visible en modification). Ajout grille : defaut_actif — active réellement la valeur defaut d'un filtre externe, avec persistance cookie. Hooks serveur : on_avant_suppression (nouveau, bloque une suppression avec message) ; on_log enrichi d'un 4e paramètre $donnees. Événements JS : acgr:before_delete (nouveau, annulable via preventDefault(), suppression simple et groupée) ; acgr:log enrichi de detail.donnees. Voir section 14.
`3.14.0`	Rétrocompatible par défaut (aucune grille existante en prod affectée, la clé formule n'ayant jamais existé avant cette version). formule (colonne) — colonne calculée, syntaxe type tableur ('({prix_ht} * {tva}) * {nbr_articles}'), {champ} référençant toute autre colonne de la même ligne. Force editable:false automatiquement, dans ac_datagrid.php comme dans ac_datagrid_pro. Recalculée en direct côté navigateur à chaque saisie d'un champ source (inline, modale, duplication) ; valeur vide tant qu'un champ source manque ou n'est pas numérique, jamais 0 par défaut. Recalculée et écrite côté serveur de façon autoritaire à chaque INSERT/UPDATE pour ac_datagrid.php (jamais lue depuis $_POST) ; pour ac_datagrid_pro avec un backend

Version	Fonctionnalité
	externe, c’est la valeur calculée côté navigateur qui est transmise. Évaluation protégée par whitelist stricte avant tout calcul — aucune exécution de code arbitraire possible. Chaînage entre colonnes calculées non supporté dans cette version (prévu 4.0.0). Voir section 15.

2. Installation et usage minimal

```
require_once 'ac_datagrid.php';

$grille = new ac_datagrid([
    'id'      => 'praticiens',
    'table'   => 'acgr_praticiens',
    'bdd'     => ['host'=>'localhost', 'db'=>'mabase', 'user'=>'user',
'pass'=>'pass'],
    'colonnes' => [
        ['champ'=>'id',  'titre'=>'ID',  'type'=>'number', 'editable'=>false],
        ['champ'=>'nom', 'titre'=>'Nom', 'type'=>'text',   'requis'=>true],
    ],
]);

if ($grille->estRequeteAjax()) { $grille->traiterAjax(); exit; }

echo $grille->genereCSS('sobre');
echo $grille->genereHTML();
echo '<script>' . $grille->getJSCore() . '</script>';
```

3. Options de configuration globales

Option	Type / Défaut	Description
`id`	string (requis)	Identifiant unique de la grille (alphanumérique + _)
`table`	string (requis)	Nom de la table SQL
`bdd`	array (requis)	Connexion BDD : host, db, user, pass
`colonnes`	array (requis)	Définition des colonnes — voir §4
`edit_mode`	string / inline	Mode d'édition : inline | modal | cell
`ajout_mode`	string / inline	Mode d'ajout quand edit_mode='cell' : inline | modal
`delete`	bool / false	Affiche le bouton supprimer (rôle admin)
`ajout`	bool / true	Affiche le bouton + Ajouter
`duplication`	bool / false	Affiche le bouton ⧉ duplication
`import_csv`	bool / false	Affiche le bouton ⬆️ CSV dans la toolbar
`csv_separateur`	string / ;	Séparateur CSV : ; ou ,
`serverside`	bool / false	Pagination et tri côté serveur (grandes tables)
`recherche`	bool / true	Active la recherche globale
`pagination`	bool / true	Active la pagination
`filtre_auto`	bool / true	Active les filtres par colonne
`longueur_defaut`	int / 10	Nombre de lignes par page par défaut
`show_nbr_lines`	bool / true	Si false : masque à la source le sélecteur « Afficher X lignes » (aucune génération, ni HTML ni CSS) — v3.8.0
`export`	bool / false	Active le bouton export (formats via export_formats)
`export_formats`	array / ['xlsx']	Formats proposés : 'xlsx' et/ou 'csv'. Un seul format = clic direct (comportement inchangé) ; deux formats = menu déroulant — v3.11.0
`action_custom`	bool / false	Bouton supplémentaire dans la colonne Actions, tire l'événement acgr:action-custom — v3.11.0, voir §12
`action_custom_icone`	string / ⚡	Glyphe du bouton d'action custom (comme ⧉ pour dupliquer) — v3.11.0
`action_custom_label`	string / null	Bulle d'aide du bouton d'action custom. null = libellé générique — v3.11.0
`epingle`	bool / true	Afficher la colonne étoile d'épinglage (ignoré si read_only:true)
`totaux`	bool / false	Affiche les totaux des colonnes numériques
`cookies`	bool / true	Mémorise les préférences utilisateur
`modal_titre`	string / id	Titre affiché dans la modale d'édition
`ordre_defaut`	array / [id,asc]	Tri par défaut : [champ, asc|desc]

Option	Type / Défaut	Description
`read_only`	bool / false	Mode lecture seule — aucune modification possible. Affiche un bandeau discret (icône œil, style neutre) — v3.8.0
`read_only_msg`	string / auto	Message affiché en mode lecture seule. Défaut depuis v3.8.0 : « Consultation — ce tableau n'est pas modifiable avec votre rôle. »
`symbole_monnaie`	string / €	Symbole monétaire global
`epingle_page_1`	bool (false)	true = n'envoie les identifiants épinglés au serveur qu'en page 1 (optimisation réseau ; le rendu, lui, a toujours ignoré les épinglées hors page 1). v3.12.0
`preset`	string|null (null)	Raccourci pré-remplissant plusieurs options d'un coup : mini | recherche | edition | edition_avancee | complet. null = comportement identique à son absence (le preset 'complet' est un tableau vide dans le code, jamais recopié à la main). Toute option explicitement fournie l'emporte toujours sur le preset. v3.12.0
`defaut_actif`	bool / false	false (défaut) = comportement d'origine inchangé : la clé defaut d'un filtre externe (§ filtres) reste posée en HTML (attribut selected) mais ignorée côté JS au premier chargement. true = le defaut s'applique réellement dès le premier charger(), persiste en localStorage, et resynchronise visuellement le select avec la valeur restaurée. À activer en même temps que le retrait de tout contournement dispatchEvent('change') manuel posé côté page pour compenser ce manque. v3.13.0
`on_avant_suppression`	callable / null	function(int $rowId, array $ligne): void|string — appelé juste avant le DELETE. Retourner une chaîne non vide bloque la suppression, remontée comme message d'erreur côté client. Rien ou null : suppression autorisée (comportement par défaut). Voir §14.

4. Options de configuration par colonne

4.1 Options générales

Option	Type / Défaut	Description
`champ`	string (requis)	Nom du champ SQL
`titre`	string / champ	Libellé affiché dans l'en-tête
`type`	string / text	Type : text | number | select | checkbox | date | datetime | textarea | email | url | tel | color (email/url/tel/color : v3.11.0)
`editable`	bool|string / true	Si false : colonne non éditable, non dupliquée. Si 'creation' : saisissable uniquement à la création, verrouillé et visible en modification — voir §14.8. v3.13.0
`pk`	bool / false	true : désigne explicitement la clé primaire, prioritaire sur l'heuristique editable:false. Exclue d'office de l'écriture PHP même sans editable:false — voir §14.7. v3.13.0
`formule`	string / null	Colonne calculée, ex. '({prix_ht} * {tva}) * {nbr_articles}' — {champ} référence toute autre colonne de la même ligne. Force editable:false automatiquement. Recalculée en direct côté navigateur à la saisie des champs sources, et côté serveur à chaque INSERT/UPDATE — voir §15. v3.14.0
`requis`	bool / false	Champ obligatoire à la saisie
`visible`	bool / true	Si false : colonne masquée (FK parent, champ caché)
`largeur`	string / null	Largeur CSS : 150px, 10%, etc.
`dupliquer`	bool / true	Si false : champ vidé lors d'une duplication
`monetaire`	bool / false	Formatage monétaire avec séparateurs
`symbole`	string / €	Symbole monétaire spécifique à cette colonne
`total`	bool / true	Si false : exclut cette colonne des totaux (utile pour un champ number non sommable, ex : une année). v3.7.1
`agregat`	string / somme	somme | moyenne | min | max | nombre — fonction appliquée aux totaux de cette colonne (total reste le simple interrupteur d'affichage). Préfixe visuel ⌀▼▲# par cellule pour tout agrégat non par défaut. v3.11.0
`pattern`	string / null	Regex (sans délimiteurs) qui remplace le pattern par défaut d'un type email/url/tel. Sans effet si widget est configuré sur la colonne. v3.11.0
`options`	array / []	Options pour type='select' : [{valeur, libelle}]
`sql_options`	string / null	SQL pour charger les options select dynamiquement
`sql_options_display`	string / null	v3.7.4 — SQL pour résoudre le libellé d'une valeur déjà enregistrée en lecture (liste options_display). Reste inclusif même si

Option	Type / Défaut	Description
		sql_options est filtré. Si absent, retombe sur sql_options.
`champ_table`	string / champ	Nom réel en BDD si différent de champ
`defaut`	mixed / ''	Valeur par défaut à la création. Corrigé en v3.13.0 : l'option était documentée et lue côté JS depuis v3.6.7 (construireNouvelleLigne()), mais jamais transmise par buildConfigJS() — donc sans aucun effet jusqu'à cette version, quelle que soit la config.
`champ_libelle`	string / null	Colonne à fort volume (v3.9.0) : nom du champ (jointure SQL) fournissant le libellé en lecture, prioritaire sur options/options_display
`sql_options_ajax`	string / null	Colonne à fort volume (v3.9.0) : requête de recherche AJAX avec marqueur :terme, seuil 2 caractères, LIMIT 50 côté serveur
`placeholder`	string / null	Texte indicatif affiché dans le champ vide (attribut placeholder natif). Appliqué en édition (inline, modal, cellule) sur tous les types de saisie hors select et checkbox. v3.13.0
`aide`	string / null	Bulle d'aide (attribut title). Affichée sur le champ de saisie en édition ET sur l'en-tête de colonne (survol, hors édition) — décision produit du 17/08/2026 : pas de bulle sur chaque cellule en lecture, jugé trop bruyant sur une grille avec beaucoup de lignes. v3.13.0
`min`	number / null	Borne minimale native (attribut min), colonnes type:'number' uniquement. v3.13.0
`max`	number / null	Borne maximale native (attribut max), colonnes type:'number' uniquement. v3.13.0
`step`	number / null	Pas d'incrément natif (attribut step), colonnes type:'number' uniquement. v3.13.0
`maxlength`	int / null	Longueur maximale de saisie (attribut maxlength natif), colonnes type:'text' et type:'textarea'. v3.13.0
`decimales`	int / null	null (défaut) = comportement d'origine strictement inchangé (colonnes monetaire:true : 2 décimales fixes, comme avant cette version ; colonnes number non monétaires : aucun formatage, valeur brute). Posé : devient un PLAFOND de décimales à l'affichage, sans zéro forcé — 5 devient "5", 5,25 devient "5,25" avec decimales:2, jamais "5,00". S'applique à toute colonne type:'number', avec ou sans monetaire:true (découplé de monetaire depuis cette version). v3.13.0

4.2 Formatage conditionnel numérique

Pour les colonnes de type number, applique une couleur de texte selon la comparaison au seuil.

```
['champ'=>'tarif', 'type'=>'number',
  'format_conditionnel' => [
```

```
        'seuil'     => 60,
        'superieur' => '#dc2626',  // rouge si > 60
        'egal'      => '#f59e0b',  // orange si = 60
        'inferieur' => '#16a34a',  // vert si < 60
    ]
]
```

4.2bis Plusieurs paliers (v3.12.0)

> *Pour plus de deux couleurs (vert / orange / rouge par exemple), format_conditionnel accepte aussi un tableau de règles, évaluées dans l'ordre — la première qui correspond gagne. Les deux formes sont détectées automatiquement ; l'ancienne reste strictement inchangée, aucune migration nécessaire.*

['champ' => 'note', 'type' => 'number',
'format_conditionnel' => [
['operateur' => '>=', 'valeur' => 8.5, 'couleur' => '#16a34a', 'gras' => true],
['operateur' => '>=', 'valeur' => 7.6, 'couleur' => '#f59e0b'],
['operateur' => '<', 'valeur' => 7.6, 'couleur' => '#dc2626'],
]]

> *Clés de chaque règle : operateur (> >= < <= = ==), valeur (nombre, ou date ISO si la colonne est de type date/datetime), couleur, et gras (optionnel, bool).*
>
> *Les couleurs non définies sont simplement ignorées — on peut ne définir que superieur par exemple.*

4.3 Affichage vignette image

Pour les colonnes contenant une URL d'image, affiche une vignette cliquable avec lightbox.

```
['champ'=>'photo_url', 'type'=>'text',
  'url_image'    => true,
  'image_taille' => '48px',
  'image_round'  => 'rounded',  // round | rounded | square | none
]
```

Valeur image_round	Rendu
`round`	Cercle parfait — idéal pour les avatars
`rounded`	Coins arrondis 8px — usage général
`square`	Carré strict
`none`	Aucun arrondi (défaut)

- URL invalide ou vide → texte brut affiché
- URL valide mais fichier introuvable → placeholder gris avec ?
- URL valide et fichier accessible → vignette
- Clic sur la vignette → lightbox pleine taille (fermeture par clic, bouton ✕ ou Échap)

5. Modes d'édition

5.1 inline (défaut)

La ligne entière passe en mode édition. Tous les champs sont éditables en même temps. Boutons ✓ Valider et ✕ Annuler dans la colonne Actions.

5.2 modal

Un clic sur le crayon ouvre une modale avec tous les champs éditables dans une grille 2 colonnes. Boutons Enregistrer et Annuler. Supporte les widgets de la Classothèque (ac_carte, ac_telephone_multipays...). Depuis v3.9.0, un widget peut aussi s'activer en mode inline s'il déclare static MODES_SUPPORTES incluant 'inline' (ex : ac_big_select, widget de sélection à fort volume avec recherche AJAX) — repli sur modal uniquement pour les widgets qui ne la déclarent pas (comportement inchangé).

5.3 cell

Double-clic sur une cellule → seul l'input de cette cellule apparaît. Comportement selon le type :

Type	Validation
`text, number, date`	Entrée ou blur → validation et écriture BDD. Échap → annulation.
`select`	Choix d'une option → validation immédiate. Échap → annulation.
`checkbox`	Clic → validation immédiate. Échap → annulation.

> *En mode cell, les widgets Classothèque s'ouvrent en modal uniquement (ajout_mode:'modal' ou 'inline' configurable). (Aucun changement en mode cell avec v3.9.0 — seuls les modes modal et inline bénéficient de l'activation d'un widget en dehors de la modale.)*

6. Duplication de ligne

Option duplication:true — ajoute un bouton ⧉ dans la colonne Actions. La ligne dupliquée est insérée en haut de la grille en mode édition avec un bandeau d'avertissement.

6.1 Règles de duplication

- Champ id (PK) → nouvel id temporaire négatif
- Colonnes editable:false → vidées (sauf FK parent visible:false qui sont conservées)
- Colonnes dupliquer:false → vidées
- Toutes les autres colonnes → copiées depuis la ligne source

6.2 Mode d'insertion

- ajout_mode:'inline' → ligne insérée en haut avec bandeau jaune ⚠ 5 secondes
- ajout_mode:'modal' → modale ouverte avec bandeau ⚠ en haut

7. Import CSV

Option import_csv:true — ajoute un bouton ⬆️ CSV dans la toolbar. L'import se déroule en 3 étapes.

7.1 Étape 1 — Upload

Zone drag & drop ou sélecteur de fichier. Seuls les fichiers .csv sont acceptés. Le parsing est entièrement côté client (FileReader API).

7.2 Étape 2 — Mapping

Tableau d'association : colonne CSV → colonne grille. L'auto-mapping par nom est proposé (insensible à la casse). Chaque colonne CSV peut être ignorée via "— Ignorer —".

7.3 Étape 3 — Prévisualisation et import

Aperçu des 5 premières lignes. Barre de progression pendant l'import. Les lignes sont insérées une par une via l'endpoint AJAX existant (même validation que l'ajout manuel). Rapport final : X lignes importées, détail des erreurs avec numéro de ligne CSV.

> *Les colonnes editable:false sont exclues du mapping. Les champs requis sont validés avant l'appel AJAX.*

8. Suppression des lignes épinglées

Quand au moins une ligne est épinglée (★) et que delete:true, un bouton 🗑️ Supprimer (N) apparaît dans la toolbar. Il permet de supprimer en masse les lignes épinglées.

8.1 Flux

- Clic sur le bouton → AcNotifModale.confirmer() affiche la confirmation directement (repli sur confirm() natif du navigateur si AcNotifModale n'est pas chargée sur la page) — pas d'événement DOM
- Si confirmé : suppressions en série via l'endpoint AJAX existant
- Suppressions en série via l'endpoint AJAX existant
- Événement acgr:apres-suppression-multiple émis avec le rapport
- Les lignes supprimées avec succès sont automatiquement désépinglées
- Les lignes en erreur (FK) restent épinglées — le bouton reste visible

8.2 Événements

AcNotifModale.confirmer()	(callback JS, pas un événement)	Appelé directement en interne avant toute suppression groupée (repli sur confirm() natif si AcNotifModale n'est pas chargée) — la page hôte ne l'écoute jamais, ce n'est pas un événement DOM.
`acgr:avant-suppression-multiple`	{ ids, nb, grille_id, confirmer }	Émis avant la suppression. Appeler confirmer() pour procéder.
`acgr:apres-suppression-multiple`	{ ids, nb_ok, erreurs, grille_id }	Émis après la suppression. erreurs : [{id, message}].

9. Événements JavaScript

Tous les événements sont dispatchés sur le conteneur de la grille avec bubbles:true.

Événement	detail	Description
`acgr:apres-insert`	{ donnees, grille_id }	Après un INSERT réussi.
`acgr:apres-update`	{ donnees, grille_id }	Après un UPDATE réussi.
`acgr:avant-suppression`	{ id, grille_id, confirmer }	Avant une suppression unitaire. Appeler confirmer().
`acgr:apres-suppression`	{ id, grille_id }	Après une suppression unitaire réussie.
`acgr:avant-suppression-multiple`	{ ids, nb, grille_id, confirmer }	Avant suppression multiple. Appeler confirmer().
`acgr:apres-suppression-multiple`	{ ids, nb_ok, erreurs, grille_id }	Après suppression multiple.
`acgr:erreur`	{ message, code }	En cas d'erreur AJAX ou de validation.
`acgr:charger-sous-grille`	{ parentId, container, masterId }	Déclenché pour initialiser une sous-grille.
`acgr:log`	{ action, table, id, grille_id, donnees }	v3.7.0 — Dispatché sur document (pas sur le conteneur) après chaque écriture BDD confirmée. action : CREATE | UPDATE | DELETE. Permet aux projets hôtes de logger sans modifier la classe. donnees ajouté en v3.13.0 (cohérence avec le 4e paramètre du hook PHP on_log) : les champs de la ligne concernée (null pour une ligne annulée via acgr:before_delete, non applicable ici puisque acgr:log n'est dispatché qu'après confirmation serveur).
`acgr:row-click`	{ id, grille_id, selected, row }	Clic sur une ligne (hors boutons/inputs). row = objet complet de la ligne, ajouté en v3.12.0 (auparavant seul id était transmis).
`acgr:apres-chargement`	{ total, page, grille_id }	v3.12.0 — Après chaque chargement réussi (initial, pagination, tri, recherche, filtre). Utile pour un chrono de performance ou un spinner externe.
`acgr:before_delete`	{ id, ligne, table, grille_id }	v3.13.0 — Dispatché sur le conteneur, AVANT tout appel serveur, sur suppression simple ET groupée (lignes épinglées). Annulable : appeler event.preventDefault() dans un listener bloque la suppression côté client, sans aller-retour réseau. Pendant JS du hook PHP

Événement	detail	Description
		on_avant_suppression, mais évalué plus tôt (avant même la requête AJAX) et sans accès BDD. Sur suppression groupée, une ligne annulée est simplement sautée (ni comptée en succès, ni en erreur) et le lot continue.

10. Thèmes CSS

La méthode genereCSS($theme) accepte les thèmes suivants :

Thème	Description
`sobre`	Blanc/bleu marine — usage général professionnel (défaut)
`techno`	Fond sombre, texte vert — style terminal
`nature`	Tons verts naturels
`corporate`	Noir/blanc, typographie condensée
`pastel`	Tons doux, cookies désactivés

> *Depuis v3.6.7/3.7.0, le localStorage est automatiquement ignoré si la configuration (notamment `longueur_defaut`) a changé depuis la dernière session — la configuration est prioritaire. Chaque thème surcharge les variables CSS custom --acgr-* définies dans la base. Il est possible de créer des thèmes personnalisés en surchargeant ces variables dans la page appelante.*

11. Exemple complet — Praticiens

```
$grille = new ac_datagrid([
    'id'             => 'praticiens',
    'table'          => 'acgr_praticiens',
    'bdd'            => $dbCfg,
    'edit_mode'      => 'cell',
    'ajout_mode'     => 'modal',
    'modal_titre'    => 'Praticien',
    'delete'         => true,
    'duplication'    => true,
    'import_csv'     => true,
    'serverside'     => true,
    'export'         => true,
    'colonnes'       => [
        ['champ'=>'id',    'titre'=>'ID',    'type'=>'number', 'editable'=>false,
'largeur'=>'60px'],
        ['champ'=>'nom',   'titre'=>'Nom',   'type'=>'text',   'requis'=>true,
'largeur'=>'150px'],
        ['champ'=>'prenom','titre'=>'Prénom','type'=>'text',   'requis'=>true,
'largeur'=>'120px'],
        ['champ'=>'tarif', 'titre'=>'Tarif', 'type'=>'number', 'monetaire'=>true,

'format_conditionnel'=>['seuil'=>60,'superieur'=>'#dc2626','inferieur'=>'#16a34a']],
        ['champ'=>'photo', 'titre'=>'Photo', 'type'=>'text',
         'url_image'=>true, 'image_taille'=>'40px', 'image_round'=>'round'],
    ],
]);
```

12. Nouveautés v3.11.0

Cinq chantiers livrés le 4 août 2026, tous testés unitairement (batterie tests/run.php, 12 tests) et rétrocompatibles par défaut — aucune grille existante en prod n'est affectée sans configuration explicite.

12.1 Export CSV

Nouvelle clé export_formats (défaut ['xlsx']). Un seul format déclaré = bouton unique, clic direct, comportement identique à l'octet près aux versions antérieures. Deux formats déclarés = le bouton devient un menu déroulant.

```
'export' => true,
'export_formats' => ['xlsx', 'csv'],
```

Le CSV réutilise csv_separateur (même clé que pour l'import), applique un échappement RFC4180 et ajoute un BOM UTF-8 en tête de fichier, indispensable pour qu'Excel FR affiche correctement les caractères accentués.

12.2 Action custom

Bouton optionnel dans la colonne Actions (action_custom, défaut false). Au clic, la grille tire l'événement acgr:action-custom et ne fait rien d'autre — à charge de la page appelante de le traiter. Fonctionne même en read_only, contrairement aux autres boutons d'action.

```
'action_custom' => true,
'action_custom_icone' => '👁',
'action_custom_label' => 'Voir détail',
```

Le detail de l'événement contient : id (clé primaire), ligne (objet complet de la ligne), table (nom de la table), grille_id. Bouton désactivé pendant l'édition d'une autre ligne, absent sur une ligne en cours de création.

12.3 Agrégats multiples

Nouvelle clé par colonne agregat (défaut somme) : somme | moyenne | min | max | nombre. La clé total garde son rôle inchangé de simple interrupteur d'affichage de la ligne de totaux ; agregat détermine la fonction appliquée.

```
['champ' => 'note', 'type' => 'number', 'agregat' => 'moyenne'],
```

Calculé en SQL (SUM/AVG/MIN/MAX/COUNT) en mode serverside, en JavaScript en mode clientside. Un préfixe visuel par cellule (⌀ pour moyenne, ▼ pour min, ▲ pour max, # pour nombre) apparaît uniquement pour les agrégats non par défaut — zéro changement visuel pour les colonnes qui restent en somme.

12.4 Hooks serveur PHP

Deux clés de config acceptent un callable PHP exécuté côté serveur. Toutes deux opt-in (défaut null), aucun changement de comportement si absentes.

- on_log : function(string $action, int $id, string $table) — appelé après chaque écriture confirmée (CREATE/UPDATE/DELETE). Équivalent serveur de l'événement JS acgr:log.
- on_avant_sauvegarde : function(array $donnees, bool $estNouveau, ?int $rowId): array — appelé juste avant l'INSERT/UPDATE, reçoit tous les champs y compris invisibles/FK, doit retourner le tableau (modifié ou non).

```
'on_avant_sauvegarde' => function($donnees, $estNouveau, $rowId) {
$donnees['id_asso'] = id_asso_courant(); // forçage serveur, non falsifiable
```

```
if (!$estNouveau && $donnees['montant'] > 10000) {
throw new RuntimeException('Montant trop élevé.');
}
return $donnees;
},
```

Pour bloquer la sauvegarde : throw new RuntimeException('message'), remonté tel quel au client via le même mécanisme que le token CSRF. Barrière de sécurité : toute clé ajoutée par le hook hors du schéma de colonnes déclaré est silencieusement ignorée — impossible d'injecter une clé SQL arbitraire.

12.5 Types email, url, tel, color

Quatre nouveaux types de colonne, chacun avec un input HTML natif adapté (clavier mobile, sélecteur couleur) et, pour email/url/tel, un pattern de validation appliqué à la sauvegarde — inline, modal et mode cellule.

- email — pattern par défaut : ^[^\s@]+@[^\s@]+\.[^\s@]+$
- url — pattern par défaut : ^https?://.+\..+
- tel — pattern par défaut : ^[+]?[0-9 .\-()]{6,20}$ (générique, aucun pays présumé)
- color — sélecteur natif, aucun pattern (toujours valide par construction)

La clé pattern (par colonne) permet de remplacer le pattern par défaut d'un type par une regex personnalisée. Le contrôle de format ne s'applique que si le champ est non vide, et jamais si widget est configuré sur la colonne — le widget (ex. ac_telephone_multipays) reste seul responsable de sa propre validation.

13. Nouveautés v3.12.0

Fusion architecturale du noyau JavaScript avec ac_datagrid_pro — voir la note en tête de document (section 1). Cinq nouveautés du noyau commun, toutes rétrocompatibles par défaut et couvertes par la batterie de tests (17 tests).

13.1 Presets

Nouvelle clé preset : raccourci qui pré-remplit plusieurs options d'un coup.

- mini — grille minimale lecture seule, clientside, sans recherche ni filtres
- recherche — mini + recherche, filtres, cookies
- edition — CRUD en modale, serverside
- edition_avancee — edition + cellule par cellule, ajout inline, épingles
- complet (défaut) — comportement standard de la classe, aucune restriction — tableau vide dans le code, garantit qu'il ne diverge jamais des défauts

13.2 Formatage conditionnel multi-règles

format_conditionnel accepte désormais, en plus de la forme historique à seuil unique, un tableau de règles ordonnées pour gérer plus de deux couleurs (voir section 4.2bis).

13.3 epingle_page_1

N'envoie les identifiants épinglés au serveur qu'en page 1 — optimisation réseau pure, le rendu a toujours ignoré les épinglées hors page 1. Défaut false : comportement inchangé.

13.4 Événement acgr:apres-chargement

Émis après chaque chargement réussi (clientside et serverside), avec {total, page, grille_id}. Absent jusqu'ici : aucun événement n'existait après un chargement, seules les écritures (CREATE/UPDATE/DELETE) en avaient un (acgr:log).

13.5 e.detail.row sur acgr:row-click

L'événement acgr:row-click transmet désormais l'objet complet de la ligne cliquée (e.detail.row), en plus de son identifiant (e.detail.id).

14. Nouveautés v3.13.0

Huit chantiers livrés entre le 17 et le 18 août 2026, tous rétrocompatibles par défaut — aucune grille existante en prod n'est affectée sans configuration explicite. Certains produits en prod restent volontairement sur une version antérieure ; ces nouveautés ne sont déployées que progressivement, produit par produit. Vérification exhaustive menée avant déploiement : aucune des nouvelles clés (placeholder, aide, min, max, step, maxlength, decimales, defaut_actif, on_avant_suppression, pk) n'était déjà utilisée par accident dans une config existante, à une exception près documentée en 14.6.

14.1 Correctifs

Deux bugs préexistants, sans aucun impact sur les grilles en prod au moment du correctif (vérifié par grep exhaustif avant livraison) :

- 'defaut' de colonne — l'option était déjà lue côté JS depuis v3.6.7 (construireNouvelleLigne()) et documentée comme fonctionnelle, mais jamais transmise par buildConfigJS() : sans aucun effet réel jusqu'à cette version. Corrigé, livré sans drapeau de compatibilité.
- Collision du registre window.acGrilleInstances entre sous-grilles — cfg.acgr_id est partagé par toutes les instances d'une même sous-grille (une par parent ouvert) : la dernière sous-grille ouverte écrasait silencieusement les précédentes dans le registre. Corrigé par indexation sur cfg.id (unique par instance DOM), cfg.acgr_id conservé en alias de rétrocompatibilité — aucun code appelant existant à modifier.

14.2 Confort de saisie — placeholder, aide, min/max/step, maxlength

Quatre nouvelles options de colonne, purement additives (absentes par défaut, zéro impact sur les grilles existantes) :

- placeholder (string) — texte indicatif dans le champ vide, en édition (inline, modal, cellule).
- aide (string) — bulle d'aide (title), sur le champ de saisie en édition ET sur l'en-tête de colonne. Décision produit : pas de bulle sur chaque cellule en lecture, jugé trop bruyant sur une grille avec beaucoup de lignes.
- min / max / step (number) — bornes et pas natifs HTML5, colonnes type:'number' uniquement.
- maxlength (int) — longueur maximale native, colonnes type:'text' et type:'textarea'.

```
['champ' => 'quantite', 'type' => 'number',
  'min' => 0, 'max' => 100, 'step' => 0.5,
  'placeholder' => 'ex: 10', 'aide' => 'Entre 0 et 100'],
```

14.3 decimales — plafond de décimales à l'affichage

Nouvelle option de colonne decimales (int, défaut null). null = comportement d'origine strictement inchangé : une colonne monetaire:true continue d'afficher 2 décimales fixes (5 → "5,00"), une colonne number non monétaire reste sans formatage (valeur brute). decimales posé devient un PLAFOND, sans zéro forcé : 5 reste "5", 5,25 reste "5,25", 5,255 arrondit à "5,26" avec decimales:2 — jamais de "5,00" artificiel. S'applique à toute colonne type:'number', avec ou sans monetaire:true : depuis cette version, le plafond de décimales est découplé du formatage monétaire (auparavant, seules les colonnes monetaire:true bénéficiaient d'un quelconque formatage numérique à l'affichage).

```
['champ' => 'quantite', 'type' => 'number', 'decimales' => 1], // sans symbole
['champ' => 'prix', 'type' => 'number', 'monetaire' => true, 'decimales' => 1], //
avec €
```

14.4 defaut_actif — filtres externes

Nouvelle option de grille defaut_actif (bool, défaut false). La clé defaut d'un filtre externe posait déjà l'attribut selected en HTML (genereHTML()), mais filtresActifs (état JS pilotant les requêtes) l'ignorait toujours au premier chargement — un contournement manuel (dispatchEvent(new

Event('change'))) était nécessaire côté page pour compenser. defaut_actif:true active réellement ce defaut au premier charger(), le fait persister en localStorage, et resynchronise visuellement le select avec la valeur restaurée (cookie ou config) au chargement suivant.

Un passage à true sur une grille existante doit impérativement s'accompagner du retrait du contournement dispatchEvent manuel côté page — sinon double appel réseau au chargement (la classe applique le filtre nativement, puis le contournement force un second charger()).

```
'defaut_actif' => true,
'filtres' => [
  ['id' => 'filtre_statut', 'label' => 'Statut', 'champ_sql' => 'statut',
   'sql_options' => "SELECT 0 AS id,'En cours' AS libelle UNION ALL SELECT
1,'Terminés'",
   'defaut' => 0],
],
```

14.5 Hooks serveur et événements JS — suppression

Nouveau hook PHP on_avant_suppression, symétrique de on_avant_sauvegarde : function(int $rowId, array $ligne): void|string, appelé juste avant le DELETE. Retourner une chaîne non vide bloque la suppression, remontée comme message d'erreur côté client (même mécanique que RuntimeException dans on_avant_sauvegarde). Rien ou null : suppression autorisée.

```
'on_avant_suppression' => function(int $rowId, array $ligne) {
  if ($ligne['statut'] === 'facture') return 'Impossible de supprimer une ligne déjà
facturée.';
  return null;
},
```

Pendant côté client : nouvel événement acgr:before_delete, dispatché sur le conteneur AVANT tout appel serveur (suppression simple ET groupée). Annulable via event.preventDefault() — bloque la suppression sans aller-retour réseau. Sur suppression groupée, une ligne annulée est simplement sautée (ni succès, ni erreur), le lot continue.

```
wrap.addEventListener('acgr:before_delete', function(e) {
  if (e.detail.ligne.statut === 'facture') e.preventDefault();
});
```

Enfin, l'événement JS acgr:log (v3.7.0) gagne un detail.donnees, cohérent avec le nouveau 4e paramètre du hook PHP on_log — function(string $action, int $id, string $table, array $donnees). Rétrocompatible : PHP ignore les arguments en trop passés à une closure qui en déclare moins.

14.6 Ajustement associé — clé de config erronée corrigée

À l'occasion de cette vérification, une clé de configuration erronée a été repérée sur un produit en prod (grilles Témoignages et Événements) : 'obligatoire' => true sur 5 colonnes, alors que la clé reconnue par la classe est requis. Sans lien avec la 3.13.0 (l'erreur existait déjà en v3.12.0 et avant), mais corrigée dans le même mouvement — ces 5 champs sont désormais réellement obligatoires à la saisie, comme demandé à l'origine.

14.7 pk — clé primaire explicite

Nouvelle option de colonne pk (bool, défaut false). Prioritaire sur l'heuristique existante (première colonne editable:false et visible !== false) : si une colonne porte pk => true, elle est retenue directement par getPK() (JS) et getPrimaryKey() (PHP), sans balayer les autres colonnes. Zéro usage en prod avant cette version — l'heuristique reste le comportement par défaut, strictement inchangée, tant que pk n'est posé nulle part.

Protection supplémentaire côté écriture PHP : une colonne pk => true est exclue d'office de l'ensemble des champs écrits (INSERT et UPDATE), même si elle n'est pas explicitement editable:false — barrière contre un POST forgé qui tenterait de modifier la clé.

```
['champ' => 'uuid', 'titre' => 'ID', 'pk' => true, 'editable' => false],
```

14.8 editable:'creation' — verrouillage après création

La clé editable accepte désormais, en plus d’un booléen, la valeur 'creation' : le champ est saisissable uniquement à la création d’une ligne, puis verrouillé sur toute ligne existante. Décision produit du 18/08/2026 : le champ reste visible en modification (pas retiré du formulaire), mais verrouillé — et le libellé est résolu pour un select, pas la valeur brute.

- Ligne nouvelle (inline, cellule, modale, duplication) → traité comme editable:true, saisissable et pré-rempli lors d’une duplication
- Ligne existante, inline et cellule → non éditable (le double-clic reste sans effet en mode cell)
- Ligne existante, modale → champ visible mais verrouillé : input disabled affichant la valeur résolue (libellé select le cas échéant), sans attribut data-champ — donc jamais soumis au serveur
- requis sur un champ 'creation' → validé uniquement à la création, côté client ET serveur ; sinon toute modification d’une ligne existante échouerait puisque le champ verrouillé n’est jamais soumis

```
['champ' => 'numero_dossier', 'titre' => 'N° dossier', 'editable' => 'creation',
'requis' => true],
```

15. Nouveautés v3.14.0

Une nouveauté livrée le 18 août 2026, clôturant une série d'améliorations avant le gel de la classe jusqu'à fin septembre 2026 : les colonnes calculées.

15.1 formule — colonnes calculées

Une colonne peut désormais être calculée à partir d'autres colonnes de la même ligne, avec une syntaxe type tableur :

```
['champ' => 'prix_ttc', 'titre' => 'Prix TTC', 'type' => 'number', 'monetaire' =>
true,
 'formule' => '({prix_ht} * {tva}) * {nbr_articles}'],
```

Comportement :

- editable:false est forcé automatiquement dès qu'une colonne porte formule — aucune double déclaration nécessaire, dans ac_datagrid.php comme dans ac_datagrid_pro.
- Recalcul en direct côté navigateur à chaque saisie d'un champ source : fonctionne en édition inline, en modale, et immédiatement à l'ouverture d'une ligne dupliquée (les sources copiées déclenchent un calcul initial, pas seulement les saisies suivantes).
- Champ verrouillé (input disabled), mais data-champ conservé contrairement à editable:'creation' — la valeur est bien soumise au serveur.
- Si un champ source manque ou n'est pas numérique, la colonne calculée reste vide — jamais 0 par défaut.
- Autorité du calcul serveur : pour ac_datagrid.php, la valeur n'est jamais lue depuis $_POST — elle est recalculée côté PHP à chaque INSERT/UPDATE à partir des valeurs déjà castées des autres colonnes, protection contre toute requête forgée. Pour ac_datagrid_pro avec un backend externe (Laravel, Node…), c'est la valeur calculée côté navigateur qui est envoyée telle quelle — la classe n'a pas de code serveur à patcher dans ce cas ; un développeur souhaitant une double vérification doit l'implémenter dans son propre backend.
- Sécurité : après substitution des {champ} par leur valeur numérique, l'expression est validée par une whitelist stricte (chiffres, points, opérateurs arithmétiques uniquement) avant toute évaluation, côté PHP (evaluerFormule()) comme côté JS (_evaluerFormule()) — aucune exécution de code arbitraire possible.
- Pas de chaînage dans cette version : une colonne formule ne peut pas référencer une autre colonne formule. Prévu pour la 4.0.0.

16. Index des clés — recherche FR/EN

Toutes les clés de configuration sont en français. Si vous cherchez une option en pensant en anglais, voici la correspondance.

Options globales

- preset — preset
- edit_mode — editMode
- ajout_mode — insertMode, addMode
- modal_titre — modalTitle
- ajout — add, insert, canAdd
- label_ajouter — addButtonLabel
- read_only — readOnly
- read_only_msg — readOnlyMessage
- show_action_buttons — showActions, actionsColumn
- duplication — duplicate
- import_csv — importCsv, csvImport
- csv_separateur — csvSeparator, delimiter
- recherche — search
- longueur_defaut — perPage, pageSize, pageLength
- show_nbr_lines — showPageSizeSelector, rowsPerPageSelector
- ordre_defaut — defaultSort, initialSort
- filtre_auto — autoFilter, columnFilters, filterRow
- totaux — totals, showTotals
- symbole_monnaie — currencySymbol
- serverside — serverSide
- export_filename — exportFilename
- export_formats — exportFormats
- action_custom — customAction
- action_custom_icone — customActionIcon
- action_custom_label — customActionLabel, customActionTooltip
- epingle — pinned, pin
- epingle_page_1 — pinnedOnFirstPage
- cookies — localStorage, persistence, remember preferences
- labels — labels, i18n, translations
- defaut_actif — activateDefaultFilter, applyFilterDefault
- on_avant_suppression — beforeDelete (hook PHP), preDelete

Options de colonne

- champ — field
- titre — title, label, header
- requis — required
- visible — visible, hidden
- largeur — width
- parent_label — parentLabel
- widget_options — widgetOptions
- widget_ancre — widgetAnchor, widgetContainer

- monetaire — monetary, currency
- symbole — symbol
- dupliquer — duplicate (colonne)
- format_conditionnel — conditionalFormat, conditionalFormatting
- url_image — imageUrl
- image_taille — imageSize
- image_round — imageRound, roundedImage, avatar
- agregat — aggregate, aggregation
- pattern — pattern, regex, validation
- options_display — optionsDisplay, displayOptions
- champ_libelle — labelField, displayField
- recherche_ajax — ajaxSearch, remoteSearch, searchAsYouType
- placeholder — placeholder
- aide — title, tooltip, helpText
- min — min
- max — max
- step — step
- maxlength — maxLength, maxlen
- decimales — decimals, decimalPlaces
- pk — primaryKey, isPrimaryKey

— Fin de la documentation —