ac_fichier_upload

Classe JS autonome — Upload de fichiers avec prévisualisation

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

1. Présentation

ac_fichier_upload est une classe JavaScript autonome qui remplace un input file HTML par une drop zone ergonomique avec prévisualisation, validation et redimensionnement d'image côté client.

Caractéristiques principales

- Classe JS unique — CSS embarqué, zéro dépendance
- Drop zone : cliquer ou glisser-déposer
- Prévisualisation image (miniature) ou icône selon le type de fichier
- Validation type et taille côté client avec message d'erreur
- Redimensionnement image côté client via Canvas — ratio conservé
- Mode formulaire (multipart) ou mode AJAX (upload immédiat)
- Multi-fichiers configurable avec limite
- Icône SVG moderne par défaut, personnalisable
- Copyright discret cliquable vers artisan-code.fr

2. Installation et fichiers

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

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

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

3. Utilisation minimale

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

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

Instancier la classe :

```
const upload = new ac_fichier_upload('ma_photo');
```

⚠️ *L'input original est masqué. La classe gère tout visuellement.*

4. Options de configuration

Option	Type	Défaut	Description
mode	string	`'form'`	Mode d'envoi : 'form' ou 'ajax'
upload_url	string	null	URL du script PHP de réception (mode ajax uniquement)
multiple	bool	false	Autoriser la sélection de plusieurs fichiers
max_fichiers	int	5	Nombre maximum de fichiers (si multiple=true)
types	array	['image/*']	Types autorisés : image/*, .pdf, .docx, .xlsx...

Option	Type	Défaut	Description
taille_max	int	5	Taille maximale par fichier en Mo
redimensionner	bool	false	Redimensionner les images côté client avant envoi
largeur_max	int	1200	Largeur max en pixels après redimensionnement (ratio conservé)
largeur	string	`'100%'`	Largeur CSS du composant
hauteur	string	`'160px'`	Hauteur de la drop zone
icone	mixed	null	null = SVG upload par défaut, "" = rien, ou HTML personnalisé
callback_succes	function	null	fn(chemin, nom) — appelée après upload AJAX réussi
callback_erreur	function	null	fn(message) — appelée en cas d'erreur

5. Modes d'envoi

5.1 Mode formulaire (form)

Le fichier part avec le reste du formulaire au submit, via multipart/form-data. PHP le récupère dans $_FILES.

```
const upload = new ac_fichier_upload('photo', {
    mode      : 'form',
    types     : ['image/*'],
    taille_max: 2,
});
```

Réception PHP

```
$fichier = $_FILES['photo'];
move_uploaded_file($fichier['tmp_name'], 'uploads/' . basename($fichier['name']));
```

5.2 Mode AJAX

Le fichier est envoyé immédiatement à la sélection. PHP retourne le chemin du fichier uploadé. L'input hidden stocke ce chemin pour le submit du formulaire.

```
const upload = new ac_fichier_upload('photo', {
    mode          : 'ajax',
    upload_url    : 'upload.php',
    callback_succes: function(chemin, nom) {
        console.log('Uploadé :', chemin);
    }
});
```

Script PHP de réception (upload.php)

```
<?php
header('Content-Type: application/json');
$fichier = $_FILES['fichier'];
$nom     = uniqid('upload_') . '.' . pathinfo($fichier['name'], PATHINFO_EXTENSION);
move_uploaded_file($fichier['tmp_name'], 'uploads/' . $nom);
echo json_encode(['ok' => true, 'chemin' => 'uploads/' . $nom]);
```

⚠️ *Le script PHP doit retourner un JSON avec ok (bool) et chemin (string) en cas de succès, ou ok=false et message en cas d'erreur.*

6. Icône personnalisable

Valeur de icone	Résultat
null (défaut)	SVG upload vers le cloud — sobre, moderne, change de couleur au survol
`'' (chaîne vide)`	Aucune icône — texte seul
`'🖼️' (emoji)`	Emoji personnalisé
`'<i class="fas fa-camera"></i>'`	Icône Font Awesome (si chargée dans la page)
`'<svg ...></svg>'`	SVG personnalisé

7. Redimensionnement d'image

Quand redimensionner=true, les images plus larges que largeur_max sont réduites côté client via Canvas avant envoi. Les images déjà plus petites ne sont pas modifiées.

Comportement	Détail
Ratio conservé	La hauteur est calculée automatiquement
Qualité JPEG	88% — bon compromis qualité/poids
Images déjà petites	Non modifiées — pas d'agrandissement
Formats supportés	Tous les formats image supportés par Canvas (jpg, png, webp, gif)
Fonctionne en	Mode form ET mode ajax

Sélecteur de taille visuel

Quand redimensionner=true, un sélecteur de taille s'affiche automatiquement sous la drop zone. L'utilisateur choisit la largeur cible avant d'uploader.

Option	Largeur
Grande	600px
Moyenne	400px
Petite	200px
Custom	Saisie libre entre 50 et 3000px — champ activé quand sélectionné

> ⚠️ *La valeur de largeur_max passée au constructeur définit la sélection initiale. Si elle ne correspond pas à 600, 400 ou 200, l'option Custom est automatiquement sélectionnée avec cette valeur.*

Exemple

```
// Photo 4000x3000px → réduite à 800x600px automatiquement
new ac_fichier_upload('photo', {
    redimensionner : true,
    largeur_max    : 800,  // → Custom sélectionné avec 800px
});
```

8. Prévisualisations

Chaque fichier sélectionné génère une card avec :

- Image → miniature 120x80px
- PDF → icône 📄

- Word → icône 📝
- Excel → icône 📊
- ZIP → icône 🗄
- Vidéo → icône 🎬
- Audio → icône 🎵
- Autre → icône 📎

Chaque card affiche le nom du fichier (tronqué si nécessaire), la taille, et un bouton ✕ pour supprimer.

En mode AJAX, un spinner est affiché pendant l'upload. La card passe en vert après succès, en rouge en cas d'erreur.

9. Méthodes publiques

Méthode	Retour	Description
getFichiers()	array	Liste des fichiers validés : { nom, taille, type, chemin }. chemin est rempli uniquement en mode ajax après upload réussi.
clear()	void	Vide la drop zone, supprime toutes les cards, réinitialise
setFichier(chemin, nom)	void	Charge un fichier existant en mode édition. Affiche la prévisualisation sans re-uploader.

Exemple setFichier — mode édition

```
// Ouverture d'un enregistrement existant
upload.setFichier('uploads/photo_123.jpg', 'ma_photo.jpg');
```

10. Validation

La validation est effectuée côté client avant tout envoi. Elle ne remplace pas la validation côté serveur.

Validation	Comportement en cas d'échec
Type de fichier	Fichier ignoré, message d'erreur affiché dans la zone rouge
Taille maximale	Fichier ignoré, message d'erreur affiché
Nombre max	Fichiers supplémentaires ignorés, message d'erreur affiché

> ⚠️ *Toujours valider et sécuriser côté PHP : vérifier le type MIME réel (pas l'extension), la taille, et renommer le fichier avant stockage.*