Cette classe d'upload et redimensionnement est conçue pour offrir de multiples possibilités pour une configuration minimale. Version 3.1 compatible php >= 5.2, libre d'utilisation.

Exemples d'utilisation prêts à l'emploi dans le dossier Tests-Scripts-Upload. Placez ce dossier sur votre serveur d'évaluation. Les exemples devraient fonctionner sans configuration particulière tant que vous ne modifiez pas le contenu ou l'organisation de ce répertoire.

Auteur : Alain Bontemps http://www.abciweb.net/

Gestion automatique des erreurs :

- post_max_size
- upload_max_filesize
- max_file_uploads
- memory_limit (si redimensionnements, pour les images de type "jpg")
- Renomme les fichiers (en remplaçant ou en supprimant si besoin les caractères spéciaux) pour compatibilité maximum avec les serveurs web

Fonctions principales paramétrables en option :

- Contrôle des extensions
- Contrôle du poids des fichiers
- Contrôle du nombre de fichiers pour un téléchargement multiple
- Contrôle des images
- Contrôle anti écrasement simple si un fichier de même nom existe déjà sur le serveur
- Renomme un fichier, mode incrémental ou suffixe unique, si un fichier de même nom existe déjà sur le serveur
- Choix du nom du fichier de destination (compatible avec la fonctionnalité précédente)
- Redimensionnement multiple ou unique d'images en sauvegardant ou non l'image original
- Possibilité d'insérer des suffixes et/ou préfixes pour permettre plusieurs redimensionnements d'une même image dans un même répertoire
- Optimisation des images pour le web
- Retour des messages d'informations pour le visiteur (paramétrable)
- Retour d'un tableau de résultat (paramétrable) ex : pour enregistrement en bdd
- Upload simple ou multiple avec notation tableau du champ html "file" (compatible HTML5)

Exemples :

Pour télécharger des fichiers de types "pdf" ou "txt" dans un répertoire "Documents", si votre champ de formulaire de type file se nomme "doc" et une variable $_POST témoin d'envoi du formulaire se nomme "form1", il suffit d'écrire :

```
$up = new Telechargement('Documents','form1','doc');

$extensions = array('pdf','txt');

$up->Set_Extensions_accepte ($extensions);

$up->Upload('reload');
```

Ainsi, en considérant que vous avez enregistré la classe dans un fichier nommé "Classe_Upload.php", le code ci-dessous est déjà parfaitement fonctionnel :

```
<?php require('Classe_Upload.php');

$up = new Telechargement('Documents','form1','doc');

$extensions = array('pdf','txt');

$up->Set_Extensions_accepte ($extensions);

$up->Upload('reload');
?>
<form enctype = "multipart/form-data" action = "#" method = "post">
<input name = "doc" type = "file" />
<input type = "submit" name = "form1" value = "Envoyez"  />
</form>
```

c.f Tests-Scripts-Upload -> Exemples -> Basique.php

Pour télécharger des images de type gif, png, jpg, jpeg, avec vérification des images, dans le répertoire "Images" :

```
$up = new Telechargement('Images','form1','doc');

$up->Set_Controle_dimImg ();

$up->Upload('reload');
```

Pour redimensionner des images de type gif, png, jpg, jpeg, en maximum 600 de largeur et 500 pixels de hauteur et les télécharger dans le répertoire "Images" :

```
$up = new Telechargement('Images','form1','doc');

$up->Set_Redim ('600','500');

$up->Upload('reload');
```

Voilà pour les configurations de base.

SOMMAIRE

Les fonctions "**Set_**..." peuvent être appelées dans n'importe quel ordre après l'initialisation de la classe mais AVANT la fonction "Upload" et les fonctions **"Get_**..." APRES la fonction "Upload"

Certaines fonctions acceptent comme paramètre optionnel «tout caractère ou chaîne de caractères non nulle». Vous pouvez choisir n'importe quel caractère ou chaîne de caractère non nulle pour déclencher l'action associée à ce paramètre. Plus d'explications et exemples dans la note n°2 en fin du mode d'emploi (page 21).

Configuration html pour un upload multiple c.f. note n°3 en fin du mode d'emploi (page 22).

IMPORTANT et exemples divers :
En cas de dépassement de la directive post_max_size du serveur, les variables globales $_POST et $_FILES sont nulles. Aussi si vous souhaitez que la classe puisse gérer cette erreur, il faut impérativement que l'**instanciation de la classe** (pour l'interception de l'erreur) et la fonction de **récupération des messages** d'information (pour l'affichage du message correspondant) **ne soient pas** conditionnées à l'existence d'une variable $_POST ou $_FILES, c.f. exemples complets note n°4 en fin du mode d'emploi (page 22).

UTILISATION DE LA CLASSE

A/ INITIALISATION DE LA CLASSE

```
$up = new Telechargement('répertoire','input_verif','userfile','get_action_form',$tab_infos);
```

L'initialisation de la classe "Telechargement" requière trois paramètres indispensables, les quatrième et cinquième étant facultatifs :

1/ Le nom du répertoire de destination par rapport à la racine du serveur.

2/ Nom d'un input d'identification et de contrôle d'envoi du formulaire.

3/ Nom de l'input de type "file" de votre formulaire. Ce nom peut désigner un tableau pour l'envoi simultané de plusieurs fichiers depuis plusieurs champs du formulaire, ou pour un envoi multiple depuis un seul champ de formulaire (cette dernière fonctionnalité n'est supportée que par certains navigateurs modernes).

4/ Nom de la variable $_GET de contrôle d'envoi du formulaire (variable $_GET dans l'attribut "action" de la balise "form").

5/ Tableau de messages pour personnaliser ou traduire les messages d'informations de résultat.

Notes :

- Le répertoire de destination doit être indiqué par rapport à la racine www. Si vous testez en local avec plusieurs sites installés sur la racine du serveur d'évaluation (dans le répertoire www) vous devrez indiquer pour un chemin de répertoire valide :

nom_du_repertoire_du_site/nom_du_repertoire_du_dossier

et pour un fonctionnement sur le serveur distant, simplement :

nom_du_repertoire_du_dossier

L'avantage de cette configuration est que vous pouvez appeler ce script depuis n'importe quel endroit de votre site sans avoir à modifier le chemin des répertoires de destination.

- Le paramètre n°2 (variable post) identifie le formulaire. Par conséquent ce paramètre doit être différent pour chaque formulaire si vous avez plusieurs formulaires d'envoi de fichiers dans une même page.

- Le paramètre n°4 contribue à l'interception de l'erreur de dépassement du maximum du post autorisé par le serveur (directive «post_max_size»). A partir de la version 2.4 de cette classe, un système complémentaire prend le relais en cas d'absence de ce quatrième paramètre. Il est néanmoins

conseillé de continuer à renseigner ce quatrième paramètre pour avoir un code plus portable car le système complémentaire utilise la fonction "php error_get_last()" dont le comportement pourrait être modifié par une gestion personnalisée des erreurs php (avec l'utilisation de "set_error_handler()" quelque part avant dans un autre script php par exemple...).

- Le cinquième paramètre facultatif est destiné à recevoir un tableau de personnalisation ou de traduction des messages d'information de résultat. Ce tableau peut être alternativement renseigné (sous réserve) par la fonction "Set_Tab_messages()" (voir chapitre H). Pour plus d'infos sur les possibilités de personnalisation de ce tableau, voir le chapitre H.

B 1/ CONTROLE DES EXTENSIONS DE FICHIERS

Par sécurité, l'emploi de la fonction "Set_Extensions_accepte()", dans laquelle vous transmettez en paramètre un tableau d'extensions autorisées, est indispensable excepté si vous utilisez la fonction "Set_Controle_dimImg()" de contrôle des images ou la fonction "Set_Redim()" de redimensionnement des images (ces deux fonctions contrôlant déjà que le fichier est une image de type jpg ou jpeg ou png ou gif). Inutile de spécifier les variantes minuscules/majuscules.

Si vous souhaiter ne pas contrôler les extensions des fichiers (à éviter pour des raisons de sécurité), transmettez un tableau vide en paramètre.

Exemple pour limiter le téléchargement aux fichiers d'extension "jpg" ou "jpeg" :

```
$tab_extensions_autorisees = array('jpg','jpeg');

$up->Set_Extensions_accepte($tab_extensions_autorisees);
```

c.f Tests-Scripts-Upload -> Exemples -> Basique.php

B 2/ CONTROLE DU POIDS MAXIMUM DES FICHIERS

Vous pouvez utiliser un champ de formulaire de type = "hidden" name = "MAX_FILE_SIZE" … pour limiter la taille maximum des fichiers (à placer avant le champ de type file) mais cette valeur pourrait être contournée par une personne malveillante. En remplacement ou en complément, vous pouvez utiliser la fonction "Set_Max_poidsFicher()". Cette fonction accepte comme paramètre une valeur numérique représentant des octets ou une valeur numérique suivie d'une des unités suivante "K", "M", "G", "T" ou "Ko", "Mo", "Go", "To". Les espaces sont tolérés ainsi que les virgules ou les points comme séparateur décimal.
Un message vous avertira si vous dépassez la configuration «upload_max_filesize» du serveur.

```
$up->Set_Max_poidsFicher('1,5 Mo');
```

B 3/ CONTROLE DU NOMBRE MAXIMUM DE FICHIERS (pour un téléchargement multiple)

La fonction « Set_Max_nombreFichers() » permet de limiter le nombre de fichiers pour un téléchargement multiple, qu'il soit issu d'un seul champ avec l'attribut html multiple, ou constitué de plusieurs champs de type file de même nom. Un message est enregistré dans le tableau des messages en cas de dépassement. Cette fonction attend comme premier paramètre le nombre maximum autorisé de fichiers. Un message vous avertira si vous dépassez la configuration « max_file_uploads » du serveur. Si le second paramètre optionnel (constitué de n'importe qu'elle chaîne de caractères non nulle) est renseigné, l'ensemble du téléchargement sera annulé en cas de dépassement.
. Exemples :

```
// 3 fichiers au maximum seront téléchargés, les suivants seront ignorés. Message si dépassement.
$up->Set_Max_nombreFichers('3');

// 4 fichiers au maximum seront téléchargés. Si dépassement, message et aucun fichier ne sera téléchargé.
$up->Set_Max_nombreFichers('4','annulation upload');
```

C/ CONTROLE DES IMAGES

L'emploi de la fonction "Set_Controle_dimImg()" est facultatif. Elle vérifie que le fichier est effectivement une image de type gif, png, jpg ou jpeg et retourne ses dimensions dans le tableau des résultats de l'upload (accessible avec les fonctions "Get_Tab_upload()" et "Get_Tab_result()" cf. chapitre L). Si le fichier n'est pas une image, il ne sera pas téléchargé. Par sécurité, utilisez cette fonction pour un téléchargement d'images excepté si vous utilisez la fonction "Set_Redim()" qui effectue les mêmes contrôles.

```
$up->Set_Controle_dimImg();
```

D/ CONTROLE ANTI ECRASEMENT

L'emploi de la fonction "Set_Controle_fichier()" est facultatif. Si un fichier de même nom est déjà existant dans le répertoire de destination, le message "ce fichier existe déjà" sera créé dans le tableau des messages d'information et aucune autre action ne sera engagée. Si la vérification ne trouve pas de fichier de nom identique dans le répertoire de destination, le fichier sera téléchargé.

```
$up->Set_Controle_fichier();
```

note : voir notes importantes fin du chapitre E suivant.

E/ RENOMMAGE AUTOMATIQUE DES FICHIERS

L'emploi de la fonction "Set_Renomme_fichier()" est facultatif. Si un fichier de même nom existe déjà dans le répertoire de destination, le fichier sera renommé et téléchargé. Cette fonction peut prendre deux paramètres optionnels constitués de tout caractère ou chaîne de caractères non nulle.
- Le premier pour indiquer un renommage avec un suffixe incrémentiel plutôt qu'un renommage avec un suffixe unique (qui est le fonctionnement par défaut).
- Le second permet à la fonction, quand elle fonctionne en mode incrémentiel, d'être sensible à la casse pour la recherche des fichiers de même nom. N'utilisez ce second paramètre qu'en connaissance de cause et uniquement avec des serveurs unix.

```
$up->Set_Renomme_fichier(); // suffixe unique
// ou
$up->Set_Renomme_fichier('incr'); // suffixe incrémentiel

// ou éventuellement et en cas de besoin (uniquement compatible sur les serveurs unix)
$up->Set_Renomme_fichier('incr','casse_sensitive');
```

Cette fonction peut être combinée avec la fonction "Set_Nomme_fichier()" (voir chapitre suivant)

a/ Le suffixe unique est composé d'un tiret bas (underscore _) et du résultat de la fonction uniqid() de php.

b/ La fonction de suffixe incrémentiel cherche (en cas de fichier identique sur le serveur) un "même" nom de fichier comprenant un nombre situé juste avant l'extension et précédé d'un tiret bas. Si elle en trouve au moins un elle incrémente le plus grand nombre, sinon elle ajoute le suffixe _1.
Donc si un fichier "monimage.jpg" existe sur le serveur, un prochain fichier nommé "monimage.jpg" sera renommé successivement "monimage_1.jpg", puis "monimage_2.jpg", puis "monimage_3.jpg", etc.

Tenez compte de ce fonctionnement si vous voulez utiliser conjointement cette fonction en mode incrémentiel et la fonction Set_Redim() avec un suffixe. En effet si avec Set_Redim() vous ajoutez un suffixe composé d'un tiret bas suivi d'un nombre vous pourrez obtenir des résultats inattendus (bien que cela ne nuise pas à la protection des fichiers). Pour éviter cet effet de bord indésirable il suffit que le nombre (si vous souhaitez que cela soit un nombre) composant le suffixe donné en paramètre à Set_Redim() ne soit pas précédé d'un tiret bas (utilisez un tiret normal ou tout autre caractère à l'exception d'un tiret bas).

Si vous avez vraiment beaucoup (plusieurs dizaines de milliers) de fichiers dans votre répertoire, utilisez plutôt la fonction "Set_Renomme_fichier()" dans sa configuration avec un suffixe unique (configuration par défaut) car le mode incrémentiel sera d'autant plus gourmand en ressource mémoire et d'autant moins rapide que le nombre de fichiers ayant un nom identique sera important (mais suffisamment économe et rapide pour un usage courant).

Notes importantes concernant les fonctions de contrôle "Set_Controle_fichier()" et de renommage automatique "Set_Renomme_fichier()" (chapitres D et E) :

Important : En cas d'emploi d'une des deux fonctions "Set_Controle_fichier()" ou "Set_Renomme_fichier()" conjointement avec la fonction de redimensionnement des images "Set_Redim()" détaillée au chapitre "G", la vérification d'un fichier déjà existant se fait uniquement dans le répertoire déclaré dans l'initialisation de la classe afin de préserver la cohérence des données entre les différents répertoires de destination.

Dans le cas d'une utilisation avec Set_Redim configurée avec un suffixe et/ou un préfixe :

- Si le fichier original doit être sauvegardé, la vérification se fait sur le nom du fichier original.

- Si le fichier original ne doit pas être sauvegardé la vérification se fait sur le nom composé du fichier + suffixe et/ou préfixe contenu dans le ***premier appel*** à la fonction Set_Redim ***dont*** le paramètre du répertoire de destination n'est pas renseigné (ce qui indique implicitement le même nom de répertoire que celui indiqué dans l'initialisation de la classe).

```
$up->Set_Redim ('1000','800',array('-max));

$up->Set_Redim ('200','150', array('-min'));
```

Dans cet exemple, pour un fichier nommé "monimage.jpg" le nom du fichier pris en compte pour la vérification sera "monimage-max.jpg". Si l'ordre d'utilisation des fonctions était inversé, le contrôle se ferait sur "monimage-min.jpg"

- Si vous n'employez aucune des deux fonctions "Set_Controle_fichier()" ou "Set_Renomme_fichier()", les fichiers du serveur ayant un nom identique aux fichiers téléchargés seront écrasés. Si vous employez les deux, la fonction "Set_Controle_fichier()" sera la seule prise en compte.

F/ IMPOSER UN NOM DE DESTINATION

L'emploi de la fonction "Set_Nomme_fichier()" est facultatif. Elle sert à imposer un nom pour le fichier en téléchargement et peut prendre trois paramètres, les deux derniers constitués de tout caractère ou chaîne de caractères non nulle, sont optionnels :

1/ Le nom de destination avec ou sans extension

2/ Permet d'indiquer que le nom de destination peut prendre l'extension du fichier téléchargé (évidemment si vous renseignez ce paramètre, indiquez comme premier paramètre un nom sans extension)

3/ Permet d'indiquer à la fonction, dans le cas où le nom indiqué en premier paramètre est non valide, que le message d'erreur doit être reporté dans le tableau des messages d'information et non pas comme une erreur webmestre (qui est le comportement par défaut).

```
$up->Set_Nomme_fichier('ma_photo.jpg');
//ou
$up->Set_Nomme_fichier('ma_photo','ext_fichier_téléchargé');
//ou
$up->Set_Nomme_fichier($nom_fichier,'ext_fichier_téléchargé','erreur_report_mes_info');
```

Vous pouvez aussi utiliser cette fonction conjointement avec la fonction "Set_Renomme_fichier()".
Par exemple si vous avez trois champs de téléchargement ou un champ de téléchargement multiple :

```
$up->Set_Renomme_fichier('incr');
$up->Set_Nomme_fichier('ma_photo.jpg');
```

ce code permettra de nommer le premier fichier "ma_photo.jpg", le second "ma_photo_1.jpg", le troisième "ma_photo_2.jpg", le quatrième "ma_photo_3.jpg" etc.

G/ REDIMENSIONNER, OPTIMISER DES IMAGES

Formats d'images acceptés pour le redimensionnement : gif, png, jpg, jpeg.

L'emploi de la fonction "Set_Redim()" est facultatif et permet un ou plusieurs redimensionnements d'images avec la possibilité de sauvegarder l'image originale. Elle accepte 5 paramètres tous optionnels :

1/ Largeur maximale de l'image redimensionnée en pixels

2/ Hauteur maximale de l'image redimensionnée en pixels

3/ Répertoire de destination. Si non renseigné, prend la valeur du répertoire indiqué dans l'initialisation de la classe.
Ce paramètre peut être alternativement renseigné par un tableau pour définir un suffixe et/ou un préfixe qui s'ajouteront au nom du fichier ce qui permet plusieurs redimensionnements d'une même image dans un même répertoire. Des exemples de cette configuration tableau sont donnés au second paragraphe de ce chapitre.

4/ Qualité du redimensionnement, en pourcentage de 1 à 100. Plus la qualité est importante, plus les fichiers seront lourds. La qualité par défaut est de 88 soit le maximum "raisonnable". Au delà de ce seuil les fichiers deviennent très lourds sans gain significatif de qualité. Ce paramètre a peu d'influence sur les fichiers de type .png et n'est pas pris en compte pour les fichiers de types .gif.

5/ Paramètre quelconque constitué de tout caractère ou chaîne de caractères non nulle pour permettre un redimensionnement supérieur au dimensions de l'image originale (agrandissement).

Exemples de configurations :

1 / Exemples avec le troisième paramètre non renseigné ou renseigné par une chaîne de caractère représentant le répertoire de destination.
c.f Tests-Scripts-Upload -> Exemples -> Instances_multiples_exemple_general.php
c.f Tests-Scripts-Upload -> Exemples -> Formulaire_inscription_complet_affichage_vignette.php
...et d'autres exemples

a / Fichier redimensionné en 900 pixels de largeur maximum, 800 pixels de hauteur maximum et téléchargé dans le répertoire indiqué dans l'initialisation de la classe.

```
$up = new Telechargement('PHOTO','input_verif','userfile','get_action_form');

$up->Set_Redim('900','800');

$up->Upload('reload');
```

(fichier redimensionné et téléchargé dans le répertoire "PHOTO")

b / Sauvegarde de l'image originale :

Si le répertoire indiqué en troisième paramètre de la fonction "Set_Redim()" est différent de celui indiqué dans l'initialisation de la classe, l'image originale (sans traitement) sera téléchargée dans le répertoire indiqué dans l'initialisation de la classe et un second fichier redimensionné sera téléchargé dans ce répertoire.

```
$up = new Telechargement('PHOTO','input_verif','userfile','get_action_form');

$up->Set_Redim('900','800', 'PHOTO_GF');

$up->Upload('reload');
```

(fichier original téléchargé dans le répertoire "PHOTO" + fichier redimensionné en max 900 x 800 téléchargé dans le répertoire "PHOTO_GF")

c / A noter que l'on peut utiliser plusieurs fois la fonction "Set_Redim()"

```
$up = new Telechargement('PHOTO','input_verif','userfile','get_action_form');

$up->Set_Redim('900','800', 'PHOTO_GF');

$up->Set_Redim('150','150', 'PHOTO_PF');

$up->Upload('reload');
```

(fichier original téléchargé dans le répertoire "PHOTO" + fichier redimensionné en max 900 x 800 téléchargé dans le répertoire "PHOTO_GF" + fichier redimensionné en max 150 x 150 téléchargé dans le répertoire "PHOTO_PF")

ou encore

```
$up = new Telechargement('PHOTO_GF','input_verif','userfile','get_action_form');

$up->Set_Redim('900','800');

$up->Set_Redim('150','150', 'PHOTO_PF');
```

(fichier redimensionné en max 900 x 800 téléchargé dans le répertoire "PHOTO_GF" + fichier redimensionné en max 150 x 150 téléchargé dans le répertoire "PHOTO_PF")

d / Vous pouvez omettre de renseigner un des deux paramètres "largeur max" ou "hauteur max" ex:

```
$up->Set_Redim('','800');
```

(hauteur max 800 pixels. La largeur sera calculée en proportion des dimensions de l'image mais sans largeur maximale prédéfinie)

Si vous n'indiquer aucun des deux paramètres "largeur max" et "hauteur max" la photo ne sera pas redimensionnée mais simplement optimisée (compressée) pour le web avec la qualité par défaut de 88% :

```
$up->Set_Redim();
```

e / Si vous souhaitez un autre indice de qualité pour avoir par exemple des fichiers moins lourds, renseignez le quatrième paramètre :

```
$up = new Telechargement('PHOTO','input_verif','userfile','get_action_form');

$up->Set_Redim('','','','75');

$up->Upload('reload');
```

(optimisation avec une qualité de 75%, pas de redimensionnement et fichier téléchargé dans le répertoire "PHOTO")

f / Un cinquième paramètre optionnel (constitué de n'importe qu'elle chaîne de caractères non nulle) permet d'autoriser un agrandissement de l'image si le fichier original a des dimensions plus petites que la largeur max et la hauteur max spécifiées dans la fonction :

```
$up = new Telechargement('PHOTO','input_verif','userfile','get_action_form');

$up->Set_Redim('150','150','PHOTO_PF','85','A+');

$up->Upload('reload');
```

(fichier original téléchargé dans le répertoire "PHOTO" + fichier redimensionné avec une hauteur et une largeur max de 150 pixels et téléchargé dans le répertoire "PHOTO_PF" avec une qualité de 85%.
J'ai indiqué "A+" en cinquième paramètre (j'aurais pu indiquer "toto", peu importe) pour indiquer à la fonction de redimensionnement qu'elle pouvait agrandir des images de plus petites dimensions que 150 x 150 pixels jusqu'à 150 x 150 pixels.
N'utilisez cette option qu'en connaissance de cause : les agrandissements peuvent donner des résultats moyens (ou des images dégradées suivant le coefficient de redimensionnement).

2 / Exemples avec le troisième paramètre de la fonction Set_Redim() renseigné par un tableau

Le tableau attend 3 paramètres tous optionnels :
- Le premier étant le suffixe à ajouter à la fin du nom du fichier
- Le second étant le répertoire de destination
- Le troisième étant le préfixe à ajouter au début du nom du fichier

Le suffixe est demandé en premier et le préfixe en dernier non pas pour tromper l'ennemi mais parce qu'on emploie peu souvent les préfixes ;)

On peut éventuellement ajouter un préfixe et un suffixe.

Si le répertoire de destination n'est pas indiqué, ce paramètre prend la valeur du répertoire indiqué dans l'initialisation de la classe.

Note : Si vous souhaitez utiliser un suffixe composé d'un nombre, évitez de faire précéder ce nombre par un underscore (tiret bas _) si vous voulez pouvoir utiliser conjointement la fonction "Set_Renomme_fichier()" en mode incrémentiel sans effet de bord indésirable (c.f notes chapitre E).

a / Plusieurs redimensionnements d'un même fichier dans un même répertoire :

```
$up = new Telechargement('PHOTO','input_verif','userfile');

$up->Set_Redim ('1000','800',array('_max'));

$up->Set_Redim ('200','150', array('_min'));
```

Un fichier nommé par exemple "image.jpg" sera redimensionné en deux exemplaires.
Un premier fichier redimensionné en maximum 1000x800 pixels puis renommé "image_max.jpg" et téléchargé dans le dossier "PHOTO".
Un second fichier redimensionné en maximum 200x150 pixels puis renommé "image_min.jpg" et téléchargé dans le dossier "PHOTO".
c.f Tests-Scripts-Upload -> Exemples ->Redimensionnements_avec_suffixe_ou_prefixe.php

b / Sauvegarde de l'image originale :

Si le répertoire de destination (second élément du tableau) est ***renseigné et non nul pour chaque*** utilisation de la fonction "Set_Redim()", alors l'image originale sera sauvegardée avec son nom original et sans traitement dans le répertoire indiqué dans l'initialisation de la classe.

```
$up = new Telechargement('PHOTO','input_verif','userfile');

$up->Set_Redim ('1000','800',array('_max', 'PHOTO_GF'));

$up->Set_Redim ('200','150', array('_min', 'PHOTO_PF'));
```

Un fichier nommé "image.jpg" sera redimensionné en max 1000x800px, renommé "image_max.jpg" et téléchargé dans le dossier "PHOTO_GF". Un second exemplaire sera redimensionné en max 200x150px, renommée "image_min.jpg" et téléchargé dans le dossier "PHOTO_PF". Et la photo originale sans traitement sera téléchargée dans le répertoire "PHOTO"

ou encore

L'image originale sera également sauvegardée si le répertoire indiqué en second paramètre dans le tableau est identique au répertoire indiqué dans l'initialisation de la classe, ***pour au moins une*** utilisation de la fonction Set_Redim().

```
$up = new Telechargement('PHOTO','input_verif','userfile');

$up->Set_Redim ('1000','800',array('_max', 'PHOTO'));

$up->Set_Redim ('200','150', array('_min'));
```

Un fichier nommé "image.jpg" sera redimensionné en max 1000x800, renommé "image_max.jpg" et téléchargé dans le dossier "PHOTO". Un second exemplaire sera redimensionné en max 200x150, renommée "image_min.jpg" et téléchargé dans le dossier "PHOTO" **et** la photo originale "image.jpg" sera téléchargée (avec son nom original et sans traitement) dans le répertoire "PHOTO"

Par contre dans l'exemple suivant l'image originale ne sera pas sauvegardée

```
$up = new Telechargement('PHOTO','input_verif','userfile');

$up->Set_Redim ('1000','800',array('-1000x800', 'PHOTO_GF'));

$up->Set_Redim ('200','150', array('-min'));
```

Un fichier nommé "image.jpg" sera redimensionné en max 1000x800px, renommé "image-1000x800.jpg" et téléchargé dans le dossier "PHOTO_GF". Un second exemplaire sera redimensionné en max 200x150px, renommée "image-min.jpg" et téléchargé dans le dossier "PHOTO".
(tous les répertoires de destination n'étant pas renseignés dans les tableaux, et aucun de ceux renseignés n'ayant le même nom que le répertoire indiqué dans l'initialisation de la classe, l'image originale n'est pas sauvegardée)

c / IMPORTANT Lorsque le troisième paramètre de la fonction Set_Redim() est renseigné par un tableau, l'ordre des différents appels à cette fonction influera sur le nom du fichier qui sera contrôlé par les fonctions Set_Controle_fichier() et Set_Renomme_fichier(). c.f notes fin du chapitre E

d / Les fonctionnalités décrites dans les sous paragraphes **d/, e/, f**/ du paragraphe **1/** de ce chapitre sont également valables.

3 / Vous pouvez utilisez simultanément la fonction Set_Redim() dans ses deux types de configurations, avec ou sans tableau comme troisième paramètre. Faites cependant attention de ne pas provoquer de conflit sinon certains redimensionnements pourraient être écrasés.

```
$up = new Telechargement('PHOTO','input_verif','userfile');

$up->Set_Redim ('1200','1000');

$up->Set_Redim ('800','600',array('-max'));

$up->Set_Redim ('200','100',array('-min', 'PHOTO_PF'));
```

Un fichier nommé "image.jpg" sera redimensionné en max 1200 x 1000px et téléchargé dans le dossier "PHOTO". Un second exemplaire sera redimensionné en max 800 x 600px, renommé "image-max.jpg" et téléchargé dans le dossier "PHOTO". Un troisième exemplaire sera redimensionné en max 200x100px, renommé "image-min.jpg" et téléchargé dans le dossier "PHOTO_PF".

Un conflit surviendrait si dans l'exemple ci-dessus la deuxième utilisation de Set_Redim() était configurée comme suit :

```
$up->Set_Redim ('800','600',array('-max','PHOTO'));
```

En indiquant un nom de répertoire identique au répertoire indiqué dans l'initialisation de la classe dans le second paramètre du tableau, vous spécifiez de sauvegarder l'image originale. L'image originale étant toujours sauvegardée en dernier, elle écrasera le premier fichier de même nom déjà redimensionné en maximum 1200 x 1000px et téléchargé dans le répertoire "PHOTO".

4 / Vous pouvez aussi utiliser la fonction Set_Redim conjointement avec la fonction Set_Nomme_fichier décrite au chapitre F.

5 / Important : Une fonction interne vérifie que les images en téléchargement peuvent être redimensionnées sans dépasser la mémoire disponible allouée par le serveur pour vos scripts. Cette fonction ne teste que les fichiers de types "jpg" ou "jpeg". A noter aussi qu'elle utilise la fonction "memory_get_usage" qui doit être activée sur votre serveur sinon la vérification ne se fait pas.
Les fichiers de type "png" ou "gif" ne sont pas testés et renverront souvent une page vide sans message d'avertissement (le retour affiché dans ce cas dépend de la configuration de votre serveur) s'ils possèdent une trop grande résolution pour pouvoir être redimensionnés . Pour éviter cette éventualité dans le cas de fichiers à redimensionner, limitez si possible le format des images aux formats jpg et jpeg en utilisant la fonction "Set_Extensions_accepte" évoquée au chapitre B /.

H/ TRADUCTION OU PERSONNALISATION DES MESSAGES D'INFORMATIONS ET DE RESULTATS

Les fonctions ci-dessous permettent de personnaliser les tableaux de messages et de résultats obtenus respectivement par l'emploi des fonctions "Get_Tab_message()" (cf chapitre K), "Get_Tab_upload()" et "Get_Tab_result()" (cf chapitre L).

L'emploi de la fonction "Set_Tab_messages" est facultatif et permet de personnaliser et/ou de traduire le tableau des messages d'information. Elle accepte donc comme paramètre un tableau qui doit correspondre au tableau "private $tab_mes" de la classe de téléchargement. Alternativement pour visualiser ce tableau sans avoir à regarder le code de la classe vous pouvez envoyer la fonction sans paramètre ce qui affichera un message d'erreur vous indiquant le tableau de correspondance demandé.

```
$up->Set_Tab_messages(); // uniquement pour visualiser le tableau à personnaliser ou à traduire en phase de développement

$up->Set_Tab_messages($tableau_personnalisé_ou_traduit); // paramétrage normal
```

Attention : il est néanmoins conseillé de renseigner ce tableau en cinquième paramètre dans l'initialisation de la classe sinon en cas d'erreur de dépassement de la directive post_max_size du serveur, le message retourné sera celui contenu dans le tableau interne de la classe (index 0 du tableau) et non celui du tableau fourni en paramètre.

Autres fonctions de personnalisation des messages :

1/ Par défaut la classe renvoie dans le tableau des messages d'information, un renseignement complet sur le résultat du téléchargement, à savoir :
- nom du fichier original + éventuel nouveau nom en cas de renommage + éventuelles nouvelles dimensions de l'image en cas de redimensionnement + dossier de destination. Ex :
"mon chat.jpg" renommé mon-chat.jpg redimensionné en 400x300 téléchargé dans le répertoire photo (et si échec renvoie un message correspondant à l'erreur)

Ces informations détaillées peuvent être utiles dans un espace administrateur mais pour permettre un renseignement plus sommaire destiné aux visiteurs vous pouvez utiliser la fonction "Set_Message_court()". Elle accepte comme paramètre le message que vous voulez indiquer et qui sera positionné à la suite du nom du fichier.
En l'absence de paramètre renseigné aucun message ne confirmera le téléchargement du fichier et seuls les messages d'erreurs seront éventuellement affichés.

Exemple avec un fichier téléchargé nommé "mon-chat.jpg" :

```
$up->Set_Message_court ('téléchargement ok');   // renverra : "mon-chat.jpg"  téléchargement ok
//ou
$up->Set_Message_court ();   // ne renverra rien excepté s'il se produit une erreur lors du téléchargement du fichier.
```

2/ Par défaut la classe ne renvoie aucun message si un champ de téléchargement n'est pas renseigné lors de l'envoi du formulaire (champ d'upload vide). Vous pouvez utiliser la fonction "Set_Message_champVide()" pour envoyer un message si un champ de téléchargement n'a pas été utilisé lors de l'envoi du formulaire. Elle accepte comme paramètre le message que vous voulez indiquer, ou un tableau de message si vous avez plusieurs champs d'upload de même nom (si vous utilisez la notation tableau pour **plusieurs** champs html de formulaire de même nom). Dans le cas d'un tableau, certains éléments du tableau peuvent être renseignés et d'autres pas.

```
// Avec un seul champ
$up->Set_Message_champVide('champ d\'upload non renseigné');

// Avec par exemple 3 champs
$up->Set_Message_champVide(array('champ 1 non renseigné', '' ,'champ 2 non renseigné'));
// Message si un des champs 1 ou 3 sont non renseignés
```

Le message renvoyé pourra être erroné pour les champs qui suivent un champ possédant l'attribut html "multiple" (qui permet de télécharger plusieurs fichiers depuis un même champ d'upload). Si l'on sélectionne deux fichiers dans un champ possédant l'attribut "multiple", le second fichier téléchargé dans ce champ sera confondu avec le premier fichier du champ suivant (car c'est un comptage numérique). Il devient alors impossible d'avoir un message d'erreur cohérent pour les champs qui suivent un champ multiple. Mais dans tous les cas il est possible d'envoyer un message si aucun fichier n'a été téléchargé lors de l'envoi du formulaire. Pour une gestion plus affinée des champs qui suivent un champ multiple pour pouvez utiliser plusieurs instanciation de la classe.

A noter que les deux fonctions ci-dessus s'appliquent sur le tableau des messages d'information obtenu par la fonction "Get_Tab_message()" (cf chapitre K) mais n'ont aucune incidence sur le détail du tableau des résultats obtenu par les fonctions "Get_Tab_upload()" et "Get_Tab_result()" (cf chapitre L).

La fonction "Set_Separateur_dimImg()" permet de définir le caractère qui sépare les largeurs et les hauteurs retournées dans les tableaux de messages et de résultats lors de l'emploi de la fonction "Set_Redim", et dans le tableau de résultats lors de l'emploi de la fonction "Set_Controle_dimImg()". Si la fonction n'est pas utilisée (ou si aucun caractère n'est indiqué comme paramètre), c'est l'élément n°21 du tableau de messages de la classe qui sera utilisé (actuellement défini par : "x")

```
$up->Set_Separateur_dimImg ('-');// renverra, pour une largeur de 800px et une hauteur de 600px, les dimensions au format : 800-600 (par défaut 800x600)
```

I/ ENVOI DU TELECHARGEMENT

L'emploi de la fonction "Upload ()" est bien entendu INDISPENSABLE et peut prendre un paramètre facultatif (constitué de tout caractère ou chaîne de caractères non nulle) pour provoquer un rechargement de la page suite au téléchargement :

```
$up->Upload('reload');
//ou
$up->Upload();
```

Cette fonction doit être appelée APRES les fonctions de configuration **Set_**... décrites dans les chapitres précédents et AVANT les fonctions **Get_**... décrites dans les chapitres suivants

Note : Le rechargement de la page permet d'éviter un multi post en cas de rafraîchissement de la page de la part du visiteur. N'utilisez pas ce paramètre si vous avez un traitement PHP à faire à la suite du téléchargement ou si vous utilisez plusieurs fois la classe pour un même formulaire.
c.f Tests-Scripts-Upload -> Exemples

J/ RAFRAICHISSEMENT DE LA PAGE

A utiliser facultativement APRES la fonction "Upload()", la fonction "Get_Reload_page()" permet un rechargement de la page à l'endroit du code qui vous convient. Cette fonction n'est évidemment utile que si l'on utilise la fonction "Upload()" sans l'option "reload".

```
$up->Get_Reload_page();
```

K/ RECUPERATION DES MESSAGES D'INFORMATION

A utiliser facultativement APRES la fonction "Upload()" : la fonction "Get_Tab_message()" retourne le tableau des messages d'information.
A noter que cette fonction ne peut être appelée qu'une fois, après quoi elle retourne un tableau vide.

```
$tab_messages = $up->Get_Tab_message();
```

Si vous n'utilisez pas cette fonction les éventuelles erreurs de téléchargement ne pourront pas être récupérées et affichées. En effet les autres fonctions Get_Tab_result() et Get_Tab_upload() décrites plus loin ne récupèrent que les fichiers téléchargés (résultats positifs).

c.f Tests-Scripts-Upload -> Exemples

L/ RECUPERATION DU TABLEAU DES RESULTATS

1 / Get_Tab_result()

A utiliser facultativement APRES la fonction "Upload()", la fonction "Get_Tab_result()" retourne le tableau des fichiers transférés en fonction des champs de téléchargement, des répertoires de destination et des redimensionnements.

A noter que cette fonction ne peut être appelée qu'une fois, après quoi elle retourne un tableau vide.

```
$tab_transfert = $up->Get_Tab_result();
```

Rappel pour les débutants : on peut visualiser clairement l'organisation et le contenu d'un tableau en faisant :

```
echo '<pre>';
print_r($tab_transfert);
echo '</pre>';
```

c.f Tests-Scripts-Upload -> Exemples- > Redimensionnements_avec_suffixe_ou_prefixe.php

2 / Get_Tab_upload()

La fonction Get_Tab_upload() est préservée pour assurer la compatibilité avec les anciens scripts et exemples.
c.f Tests-Scripts-Upload -> Exemples -> Formulaire_inscription_complet_affichage_vignette.php

Accessoirement elle permet aussi un accès plus simple aux données puisque le tableau des résultats ne comporte que trois dimensions.

Mais elle n'est pas compatible pour récupérer tous les résultats du téléchargement quand vous configurez la fonction Set_Redim() avec un tableau pour renseigner le troisième paramètre (pour définir des suffixes ou des préfixes). Utilisez la fonction Get_Tab_result() pour récupérer le tableau des résultats dans ce cas (ou dans tous les cas pour un code plus générique).

Elle s'utilise dans les même conditions que Get_Tab_result() et ne peut être appelée qu'une fois après quoi elle retourne un tableau vide.

```
$tab_transfert = $up->Get_Tab_upload();
```

N'utilisez pas les deux fonctions Get_Tab_result() et Get_Tab_upload() conjointement (le second appel serait vide).

NOTES :

1 / Si besoin vous pouvez utiliser les fonctions «Return_Octets()», «Nettoie_Nom_fichier()», «Dim_Prop_max()», ou «Return_Config_serveur()».

Return_Octets($nombre_format) retourne le nombre d'octets en fonction d'un nombre formaté avec les unités K, M, G, T ou Ko, Mo, Go ou To (nombre décimaux et variantes en majuscules/minuscules acceptés)

```
echo $up->Return_Octets('1 ko'); // →  1024
```

Nettoie_Nom_fichier($chaine) teste la chaîne de caractère envoyée en paramètre et remplace ou supprime les caractères non compatibles avec un nom de fichier (caractères spéciaux ou accentués). Renvoie le résultat ou false si le résultat est une chaîne vide.

```
echo $up->Nettoie_Nom_fichier('toto =véner'); // → totovener
```

Dim_Prop_max($largeur_ini, $hauteur_ini, $largeur_max, $hauteur_max,$agrandissement) renvoie les dimensions proportionnelles maximales d'une image en fonction de ses dimensions initiales $largeur_ini, $hauteur_ini et des dimensions limites maximales autorisées $largeur_max et $hauteur_max. Un cinquième paramètre constitué de toute chaîne de caractère non nulle permet d'indiquer à la fonction que les dimensions finales pourront être supérieures aux dimensions initiales. La fonction retourne un tableau contenant la largeur et la hauteur calculée.

```
$dimensions_vignette = $up->Dim_Prop_max(800,600,150,100);
// →  Array ([0] => 133 [1] => 100);

$dimensions_vignette = $up->Dim_Prop_max(135,90,140,100,'A+');
// →  Array ([0] => 140 [1] => 93);

$dimensions_vignette = $up->Dim_Prop_max(135,90,140,100);
// →  Array ([0] => 135 [1] => 90);
```

Un des deux paramètre $largeur_max ou $hauteur_max peut être non renseigné. Dans ce cas cette dimension sera déterminée en fonction du ratio de l'image et de la dimension maximale renseignée. Cas particuliers : la fonction retourne false si un des deux premiers paramètres n'est pas renseigné ou égal à 0 et retourne les dimensions initiales si le troisième **et** le quatrième paramètre ne sont pas renseignés ou égal à 0. Exemple d'utilisation
c.f Tests-Scripts-Upload -> Exemples -> Formulaire_inscription_complet_affichage_vignette.php

Config_serveur() retourne les configurations 'post_max_size', 'upload_max_filesize', 'max_file_uploads' et 'memory_limit' du serveur. Renvoie une chaîne de caractère si aucun paramètre n'est passé ou un tableau si vous renseigner la fonction avec un paramètre non nul.

```
echo $up->Return_Config_serveur();
// ou
print_r($up->Return_Config_serveur('tableau'));
```

2 / Certaines fonctions acceptent «une chaîne de caractère non nulle» comme paramètre optionnel. Ces chaînes de caractère de votre choix (permettant de décrire l'utilité de ces paramètres) sont traduites en instructions booléennes (true/false) dans la classe pour déclencher ou non l'action. Ils peuvent donc être constitués de valeurs booléennes ou de leur équivalent textuels ou numériques. Exemple avec les deux derniers arguments de la fonction Set_Nomme_fichier() (cf chapitre F/) :

```
$nom_fichier = 'login';

$up->Set_Nomme_fichier($nom_fichier,'ext fichier telecharge','si erreur message information');
```

(Le nom imposé 'login' prendra l'extension du fichier téléchargé pour composer le nom du fichier, et si erreur dans $nom_fichier elle sera transmise dans les messages d'information)
Ainsi les déclarations suivantes sont équivalentes:

```
$nom_fichier = 'login';

$up->Set_Nomme_fichier($nom_fichier,'ext fichier telecharge','si erreur message information');
// =
$up->Set_Nomme_fichier($nom_fichier,1,1);
// =
$up->Set_Nomme_fichier($nom_fichier,true,true);
```

Dans cette autre configuration, toutes les lignes sont également équivalentes :

```
$nom_fichier = 'login.jpg';

$up->Set_Nomme_fichier($nom_fichier,'','si erreur message information');
// =
$up->Set_Nomme_fichier($nom_fichier,'','toto');
// =
$up->Set_Nomme_fichier($nom_fichier,' ','titi');
// =
$up->Set_Nomme_fichier($nom_fichier,0,1);
// =
$up->Set_Nomme_fichier($nom_fichier,'0',1);
// =
$up->Set_Nomme_fichier($nom_fichier,false,true);
//=
$up->Set_Nomme_fichier($nom_fichier,null,true);
```

à noter que pour éviter les étourderies, les espaces vides ' ' et la chaîne « '0' » sont traités comme étant équivalents à false, ou à 0, ou à '', ou encore à null.

3 / Pour autoriser l'upload multiple avec plusieurs champs de formulaire ces champs doivent simplement être écrit avec la notation tableau []. En reprenant mes premiers exemples :

```
<input name = "doc[]" type = "file" />
<input name = "doc[]" type = "file" />
<input name = "doc[]" type = "file" />
…
```

Pour autoriser l'upload multiple avec un seul champ de formulaire le champ doit être écrit avec la notation tableau [] et posséder l'attribut multiple :

```
<input name = "doc[]" type = "file" multiple = "multiple" />
```

Cette fonctionnalité est supportée uniquement par les navigateurs modernes. Elle permet à l'utilisateur de sélectionner plusieurs fichiers à l'aide des touches Maj et Ctrl dans son explorateur de fichiers.

Dans les deux cas, aucune modification n'est nécessaire dans l'initialisation de la classe qui gère indifféremment les champs uniques ou avec la notation tableau.

Suivant les navigateurs (respect des dernières normes html5) le glisser/déposer est possible ou non (indépendamment du paramétrage de la classe).

c.f Tests-Scripts-Upload -> Exemples -> Upload_multiple_multiple.php

4 / IMPORTANT et exemples divers :

En cas de dépassement de la directive post_max_size du serveur, les variables globales $_POST et $_FILES sont nulles. Aussi si vous souhaitez que la classe puisse gérer cette erreur, il faut impérativement que l'**instanciation de la classe** (pour l'interception de l'erreur) et la fonction de **récupération des messages d'information** (pour l'affichage du message correspondant) **ne soient pas** conditionnées à l'existence d'une variable $_POST ou $_FILES.

a / Exemple 1: un formulaire qui inclus un champ de type text : name='pseudo' et un champ de type file : name= 'photo'. En partant du principe que le nom du pseudo doit être obligatoirement renseigné pour permettre l'enregistrement d'un fichier téléchargé. Si pas de fichier on met un avatar par défaut.

c.f Tests-Scripts-Upload -> Exemples -> Formulaire_inscription_simple.php

L'initialisation de la classe et la récupération des messages visiteurs étant exclus de toute condition $_POST, la classe pourra gérer l'erreur serveur de dépassement «post_max_size».
Notez que si besoin, on aurait pu mettre les fonctions de configuration «Set_Extensions_accepte()», «Set_Redim()» et «Set_Message_court()» à l'intérieur de la condition «if (isset($_POST['...']))». Cela dit, mettre les fonctions de configuration indépendamment d'une condition post permet de faire afficher les messages d'erreur de configuration (pour les fonctions qui en retournent) sans avoir à faire un test d'upload.

b / Exemple 2 plus complet

Dans le code précédent, si un problème survient durant le téléchargement d'une photo (erreur d'extension, fichier trop lourd etc.), le pseudo sera enregistré avec l'avatar par défaut, tout comme si l'utilisateur n'avait pas fourni de photo. On pourrait conditionner l'enregistrement du pseudo à l'absence de problème lors de l'upload dans le cas où une photo est proposée en téléchargement, et si erreur d'upload, différer l'enregistrement du pseudo et afficher le message d'erreur correspondant.

Une solution est d'utiliser la fonction «Set_Message_court()» sans paramètre car dans ce cas la fonction Get_Tab_message() retourne uniquement les messages d'erreur d'upload qu'il suffira de compter pour déclencher ou non l'action d'enregistrement.

- On utilisera une deuxième fois Get_Tab_message() en dehors de la condition post pour pouvoir gérer l'éventuelle erreur de dépassement «post_max_size».

- Et on se servira du tableau des résultats pour afficher un message et une vignette de la photo dans le html en cas d'upload réussi, sinon l'avatar par défaut.

J'en profite pour ajouter un second champ texte pour donner un exemple plus générique et gérer la persistance des champs textes dans le formulaire si un problème survient lors de l'upload ou si un champ texte n'est pas correctement renseigné.

c.f Tests-Scripts-Upload -> Exemples -> Formulaire_inscription_complet_affichage_vignette.php

Si l'on souhaite ensuite que l'upload d'une image valide soit indispensable pour l'enregistrement des données, on pourrait bien sûr modifier les conditions d'enregistrement en testant la variable $nom_final_fichier et en ajoutant un message d'erreur correspondant, mais le plus souple sera d'ajouter la fonction de configuration Set_Message_champVide() :

```
// cf note !******! dans le code ci-dessus
$up->Set_Message_champVide('Champ de téléchargement vide. Un fichier est requis !');
```

En effet si le champ d'upload est vide, cette fonction ajoutera une ligne d'erreur dans Get_Tab_message(). Et puisque dans notre contexte Get_Tab_message() ne retourne que les messages d'erreur d'upload (du fait de la fonction Set_Message_court() appelée sans paramètre) et que l'enregistrement est conditionné à l'absence de messages d'erreur d'upload, le script fonctionnera comme attendu sans aucune autre modification ;)

Note * : il faudrait compléter la configuration de la classe avec Set_Renomme_fichier() ou Set_Controle_fichier() si vous souhaitez éviter l'écrasement de fichiers déjà existants sur le serveur.

c / Exemple 3, autre type de configuration, on souhaite maintenant que le dossier de destination des fichiers téléchargés soit choisi dans le formulaire. Toujours dans l'optique de ne pas conditionner l'instanciation de la classe à une variable post on peut faire le code suivant.

c.f Tests-Scripts-Upload -> Exemples -> Dossier_defini_dans_le_formulaire.php

5 / La valeur par défaut de qualité de redimensionnement des images (exprimée en pourcentage) est définie par la variable interne de la classe : "private $qualite"

6 / Les messages d'information et les résultats sont enregistrés dans une variable de session dont l'index est défini par la variable interne de la classe : "private $index_ses"

DEBUG :

- Si, après l'envoi d'un fichier, aucun fichier n'est téléchargé et aucun message ne s'affiche (dans le cas où vous faites afficher les messages d'information), un ou plusieurs paramètres passés dans la déclaration de la classe sont erronés ou les variables de session ne fonctionnent pas sur votre serveur.

- Si le message "Le poids total maximum du formulaire autorisé par le serveur est dépassé" s'affiche même pour un fichier de faible poids, le nom de l'input d'identification du formulaire passé en deuxième paramètre lors de l'initialisation de la classe est erroné.

Forums de discussion :
http://www.developpez.net/forums/d1093577/php/langage/contribuez/classe-dupload-redimensionnement/

http://forum.phpfrance.com/vos-contributions/upload-fichiers-verification-renommage-redimensionnements-t254129.html#p330901