w.c.s. - Development #41679

possibilité de retirer les métadonnées exif d'une image

14 avril 2020 14:49 - Frédéric Péters

Statut:	Fermé	**Début:**	14 avril 2020
Priorité:	Normal	**Echéance:**	
Assigné à:	Nicolas Roche	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

Cas très particulier #41064#note-13 où avoir l'image orientée selon ses métadonnées exif pose problème parce que l'agent doit donner une instruction d'orientation sur base de l'image "brute". Pour faire ça j'imagine qu'il y ait dans les données de traitement de la demande la photo tapée dans la demande, les métadonnées en moins.

Pour que ça ne soit pas très exposé vu le côté très spécial, j'aurais ce plan d'étendre evalutils.attachment()

- pour accepter un champ de type fichier dans 'content' (et en extraire les différentes données) (sans doute ici le plus pratique est de déléguer ce travail à FileField::convert_value_from_anything).
- pour avoir un argument strip_metadata=False supplémentaire, qui dégagerait les données exif.

L'idée étant comme ça d'avoir une action de stockage de donnée de traitement qui fasse utils.attachment(form_var_image_raw, strip_metadata=True).

Révisions associées

Révision 3b81d6f4 - 06 mai 2020 19:58 - Nicolas Roche

evalutils: strip exif medatada using utility function (#41679)

Historique

#1 - 14 avril 2020 16:14 - Nicolas Roche

- Assigné à mis à Nicolas Roche

(Je reformule pour voir si j'ai bien compris.)

Dans Gimp, quand on ouvre la photo d'identité donnée en exemple, il détecte les metadaonnées Exif et propose d'appliquer la rotation de correction.
https://formulaires.mesdemarches.lille.fr/backoffice/management/demander-mon-pass-lillemoi/5502/download?f=17

Plusieur paquets debian existent mais ne semblent pas correspondre à nos 3 attentes
(modifications des metadonnées, stretch+buster, python2+python3) :
python-exif
python-pexif
python-piexif
python3-exif
python3-exifread
python3-piexif

Autant utiliser PIL pour ré-enregistrer un fichier sans ses métadonnées Exif. Par exemple :

```
> from PIL import Image
> image = Image.open('image_file.jpeg')

> # next 3 lines strip exif
> data = list(image.getdata())
> image_without_exif = Image.new(image.mode, image.size)
> image_without_exif.putdata(data)

> image_without_exif.save('image_file_without_exif.jpeg')
```

Ainsi dans Gimp, l'image s'ouvre sans que la rotation soit appliquée.

L'idée étant d'intégrer un tel code de façon à ce que :

- dans un workflow, une action donnée de traitement telle que :
  Champ de type fichier / Valeur (expression python) : utils.attachment(form_var_image_raw, strip_metadata=True)
- qui déclenchera *wcs/qommon/evalutils.py::attachment()*
- qui pourrait utiliser *wcs/fields.py::FieField::convert_value_from_anything()* afin de factoriser le code
- du code similaire à celui évoqué ci-dessus serait alors ajouté à cette dernière fonction.

#2 - 14 avril 2020 16:31 - Frédéric Péters

> Autant utiliser PIL pour ré-enregistrer un fichier sans ses métadonnées Exif. Par exemple :

Oui (tu peux citer https://stackoverflow.com/a/23249933). (on utilise déjà pour la production de miniature l'info exif via pil/pillow).

> qui pourrait utiliser wcs/fields.py::FieField::convert_value_from_anything() afin de factoriser le code

Qui doit utiliser ce code afin de ne pas dupliquer le code.

#3 - 16 avril 2020 17:19 - Nicolas Roche

- Fichier 0001-evalutils-strip-exif-medatada-using-utility-function.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

(je suis pas très sur de mon coup pour la jonction avec FieField::convert_value_from_anything())

```
    if hasattr(content, 'base_filename'):
        content.strip_metadata = strip_metadata
        return content
```

#4 - 16 avril 2020 17:26 - Frédéric Péters

Malgré les échanges mon commentaire est mal passé, strip_metadata n'a rien à faire là. Les changements de ce ticket doivent rester (au maximum) confinés (ah ah) dans attachment; quand je parlais d'exploiter convert_from_anything je pensais à un truc de cet ordre :

```
def attachment(content, filename='', content_type='application/octet-stream', strip_metadata=False):
    if isinstance(content, (bytes, str)):
        content = {"content": force_bytes(content), "filename":...}
    attachment = FileField.convert_from_anything(content)
    if strip_metadata:
        # whatever
    ...
```

#5 - 17 avril 2020 11:57 - Nicolas Roche

- Fichier 0001-evalutils-strip-exif-medatada-using-utility-function.patch ajouté

(Ils ne sont pas évident à manipuler ces objets PicklableUpload.)

#6 - 25 avril 2020 19:52 - Frédéric Péters

```
+    from django.utils.six import BytesIO
+    try:
+        from PIL import Image
+    except ImportError:
+        Image = None
```

Les imports dans les fonctions sont uniquement faits quand c'est nécessaire pour éviter des imports cycliques; ce n'est à coup sûr pas le cas pour ceux-ci.

```
+            data = list(image.getdata())
...
+            image_without_exif.putdata(data)
```

Ça sonne bizarre de faire cette transformation en liste, à tester rapidement chez moi, image_without_exif.putdata(image.getdata()) marche sans peine.

Plus haut, le premier .save() :

```
+    UploadStorage().save(upload)
```

est-il nécessaire ? Il me semble que c'est lui qui amène le del upload.qfilename qui est une ligne de code qui ne devrait pas exister (faisant référence à un détail d'implémentation de l'objet).

Alternativement, pourquoi réutiliser upload et faire ça plutôt que créer un nouvel objet ?

De là aussi, en terme de structure, pourquoi pas :

```
upload = FileField.convert_value_from_anything(content)
if strip_metadata and Image:
    ...
    upload = ...
if filename:
    upload.base_filename = filename
if content_type:
    upload.content_type = content_type
UploadStorage().save(upload)
return upload
```

#7 - 26 avril 2020 21:03 - Nicolas Roche

- Fichier 0001-evalutils-strip-exif-medatada-using-utility-function.patch ajouté

> Plus haut, le premier .save() ... est-il nécessaire ?

Oui, sinon je n'ai pas accès au contenu des champs de type fichier après leur passage dans convert_value_from_anything.

```
> /home/nroche/src/wcs/wcs/qommon/upload_storage.py(70)get_content()
> if hasattr(self, 'qfilename'):
(Pdb) hasattr(self, 'qfilename')
(faux)
```

> Alternativement, pourquoi réutiliser upload et faire ça plutôt que créer un nouvel objet ?

Je n'y avais pas pensé, ça me permet de ne pas rentrer dans le détail d'implémentation de l'objet.

#8 - 05 mai 2020 14:34 - Frédéric Péters

Je trouve un peu bizarre ce content_type qui a désormais une chaine vide comme valeur par défaut; si ça revient au même d'avoir application/octet-stream ou None, je préférerais None à la chaine vide.

#9 - 05 mai 2020 16:14 - Nicolas Roche

- Fichier 0001-evalutils-strip-exif-medatada-using-utility-function.patch ajouté

Oui, du coup j'étais tenté pour remplacer aussi filename='',
mais finalement (bien que les tests passent) je préfère ne pas y toucher.

#10 - 05 mai 2020 16:21 - Frédéric Péters

- Statut changé de Solution proposée à Solution validée

Ok, mais le truc important sur lequel je vais insister : des éléments étranges comme ça (mais pourquoi donc modifier content_type?), réalisés sans explication, compliquent vraiment la relecture. (là ça oblige par exemple à aller voir partout comment content_type est utilisé).

#11 - 06 mai 2020 19:59 - Thomas Noël

- Statut changé de Solution validée à Résolu (à déployer)

```
commit 3b81d6f4fb8f13a71519613c8e8ad4a353b9ae32
Author: Nicolas ROCHE <nroche@entrouvert.com>
Date:   Fri Apr 17 00:57:08 2020 +0200

    evalutils: strip exif medatada using utility function (#41679)
```

#12 - 06 mai 2020 22:16 - Frédéric Péters

- Statut changé de Résolu (à déployer) à Solution déployée

Fichiers

0001-evalutils-strip-exif-medatada-using-utility-function.patch	8,67 ko	16 avril 2020	Nicolas Roche
0001-evalutils-strip-exif-medatada-using-utility-function.patch	5,51 ko	17 avril 2020	Nicolas Roche
0001-evalutils-strip-exif-medatada-using-utility-function.patch	6,32 ko	26 avril 2020	Nicolas Roche
0001-evalutils-strip-exif-medatada-using-utility-function.patch	6,32 ko	05 mai 2020	Nicolas Roche