Authentic 2 - Development #24401

crash de /api/user/ sur un profil contenant un champ date

10 juin 2018 10:20 - Frédéric Péters

Statut:	Fermé	**Début:**	10 juin 2018
Priorité:	Normal	**Echéance:**	
Assigné à:	Benjamin Dauvergne	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

```
Environment:

Request Method: GET
Request URL: https://authentic.fred.local.0d.be/api/user/

Django Version: 1.8.18
Python Version: 2.7.13
Installed Applications:
''
Installed Middleware:
''

Traceback:
File "/home/fred/src/eo/venv/local/lib/python2.7/site-packages/django/core/handlers/base.py" in ge
t_response
  132.                     response = wrapped_callback(request, *callback_args, **callback_kwargs)
File "/home/fred/src/eo/venv/local/lib/python2.7/site-packages/django/views/decorators/vary.py" in
 inner_func
  21.             response = func(*args, **kwargs)
File "/home/fred/src/eo/venv/local/lib/python2.7/site-packages/django/views/decorators/cache.py" i
n _cache_controlled
  43.             response = viewfunc(request, *args, **kw)
File "/home/fred/src/eo/authentic/src/authentic2/decorators.py" in f
  297.         json_str = json_dumps(result)
File "/usr/lib/python2.7/json/__init__.py" in dumps
  244.         return _default_encoder.encode(obj)
File "/usr/lib/python2.7/json/encoder.py" in encode
  207.         chunks = self.iterencode(o, _one_shot=True)
File "/usr/lib/python2.7/json/encoder.py" in iterencode
  270.         return _iterencode(o, 0)
File "/usr/lib/python2.7/json/encoder.py" in default
  184.         raise TypeError(repr(o) + " is not JSON serializable")

Exception Type: TypeError at /api/user/
Exception Value: datetime.date(2018, 3, 20) is not JSON serializable
```

Révisions associées

Révision ba721dbd - 05 septembre 2019 12:52 - Benjamin Dauvergne

api: factorize making a DRF field for an attribute (#24401)

Révision cb021541 - 05 septembre 2019 12:52 - Benjamin Dauvergne

custom_user: user DRF field to serializer custom attributes to JSON (#24401)

Historique

#1 - 14 juin 2018 14:45 - Paul Marillonnet

- Assigné à mis à Paul Marillonnet

Je prends.

#2 - 14 juin 2018 14:51 - Paul Marillonnet

Frédéric, si tu as quelques infos en plus, pour reproduire le bogue par exemple, je suis preneur stp.

Edit: par exemple, sais-tu à quoi correspond ce champ date ? Le BaseUserSerializer des vues d'API d'authentic prends en charge User.date_joined et User.last_login, mais qui sont tous deux des datetime.datetime, pas des datetime.date

#3 - 14 juin 2018 14:55 - Frédéric Péters

Avoir un champ date dans le profil. Avoir une valeur dedans. Aller avec son navigateur à l'adresse /api/user/.

#4 - 14 juin 2018 15:09 - Paul Marillonnet

Frédéric Péters a écrit :

> Avoir un champ date dans le profil.

Pas compris. Ce champ date ne correspond à aucun attribut dans le modèle utilisateur d'authentic.

Tu comprendras peut-être ce que je ne comprends pas si je copie-colle ma tentative de reproduction du bogue :

```
In [1]: import datetime

In [2]: from django.contrib.auth import get_user_model

In [3]: User = get_user_model()

In [4]: User.objects.all()
Out[4]: [<User: u' (e3b055)'>, <User: u'admin (844b60)'>, <User: u'paul (ee1dc9)'>, <User: u'root (a25fcb)'>]

In [5]: root = User.objects.last()

In [6]: root.date = datetime.date(2018,3,20)

In [7]: root.save()

In [8]:
Do you really want to exit ([y]/n)?
[2018-06-14 Thu 15:06:51] - - - WARNING py.warnings.adapt_datetime_with_timezone_support: /home/paul/eodevel/v
irtualenvs/main2/local/lib/python2.7/site-packages/django/db/backends/sqlite3/base.py:57: RuntimeWarning: SQLi
te received a naive datetime (2018-06-14 15:06:51.994598) while time zone support is active.
  RuntimeWarning)

(main2) paul@eosandbox:~/eodevel/authentic$ ./authentic2-ctl runserver 192.168.56.101:8080
Performing system checks...

System check identified no issues (1 silenced).
June 14, 2018 - 15:06:55
Django version 1.8.19, using settings 'authentic2.settings'
Starting development server at http://192.168.56.101:8080/
Quit the server with CONTROL-C.
[14/Jun/2018 15:06:56] "GET /api/user/ HTTP/1.1" 200 243
```

#5 - 14 juin 2018 15:25 - Benjamin Dauvergne

Les attributs ne sont pas stockés dans le modèle User mais dans le modèle AttributeValue, il y a un descripteur qui produit des accésseurs via user.attributes.<nom_de_l'attribut>.

#6 - 14 juin 2018 15:25 - Benjamin Dauvergne

Un descripteur c'est ça : https://docs.python.org/2/howto/descriptor.html

#7 - 14 juin 2018 15:32 - Frédéric Péters

Si tu utilises Publik, tu peux activer le champ "date de naissance" depuis la page de définition du profil dans Hobo. Si c'est un authentic pur, tu peux aller dans /admin/ créer un objet de type Attribute.

#8 - 14 juin 2018 16:30 - Paul Marillonnet

Frédéric Péters a écrit :

> Si tu utilises Publik, tu peux activer le champ "date de naissance" depuis la page de définition du profil dans Hobo. Si c'est un authentic pur, tu peux aller dans /admin/ créer un objet de type Attribute.

Ok, je vais faire comme ça, merci.

Pour info, j'avais essayé, en vain, de créer un nouvel attribut de type date depuis cette page de définition du profil dans Hobo. On dirait que les nouveaux attributs de profil usager ne peuvent être que de type string. Est-ce que ça pourrait faire l'objet d'un ticket d'évolution ?
J'ai l'impression qu'on ne peut pas supprimer de tels attributs une fois ceux-ci créés, on peut seulement les désactiver, je me plante ? Est-ce que ça pourrait être une autre demande d'évolution ?

#9 - 14 juin 2018 16:32 - Paul Marillonnet

Bogue reproduit, je m'y colle, merci pour votre aide Frédéric et Benjamin.

#10 - 14 juin 2018 16:38 - Frédéric Péters

> Pour info, j'avais essayé, en vain, de créer un nouvel attribut de type date depuis cette page de définition du profil dans Hobo. On dirait que les nouveaux attributs de profil usager ne peuvent être que de type string. Est-ce que ça pourrait faire l'objet d'un ticket d'évolution ?

Been there, done that, #23306. Tu devrais mettre à jour.

#11 - 20 juin 2018 11:21 - Paul Marillonnet

Mis à jour, merci.
#23306 n'apporte pas la définition d'attributs de type date, je vais donc passer par l'interface /admin/ pour reproduire le problème.

#12 - 02 août 2018 14:08 - Paul Marillonnet

- Fichier 0001-user-api-fix-attribute-value-serialization-24401.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

#13 - 08 août 2018 11:51 - Frédéric Péters

Là tu zappes tout à fait le code de désérialisation des attributs.

Ce qu'il faut à mon sens c'est plutôt modifier le décorateur json, qu'il utilise django.core.serializers.json.DjangoJSONEncoder, qui sait comment gérer les dates.

#14 - 08 août 2018 12:22 - Benjamin Dauvergne

Il faudrait soir utiliser directement le serializer DRF dans User.to_json soit ajouter une méthode to_json() à Attribute qui utilise le paramétrage rest_framework_field_class des types d'attribut (voir authentic2/attribute_kinds.py).

#15 - 08 août 2018 12:28 - Paul Marillonnet

Je loupe visiblement quelque chose.

Pour moi c'est justement l'appel à to_python qui zappe le code précédemment exécuté pour la sérialisation des attributs.

models.Attribute.set_value place la valeur sérialisée dans Attribute.content, et c'est normal qu'on accède à ce champ lors de la méthode custom_user.models.User.to_json, non ?

Pourquoi faire appel à de la désérialisation (to_python) à ce moment là ?
to_python va executer models.AttributeValue.attribute.get_kind()['deserialize'], qui est censé être l'opération complémentaire de la sérialisation précédemment effectuée.
C'est le serpent qui se mord la queue, non? :)

#16 - 30 août 2018 10:41 - Paul Marillonnet

Frédéric Péters a écrit :

> Là tu zappes tout à fait le code de désérialisation des attributs.
>
> Ce qu'il faut à mon sens c'est plutôt modifier le décorateur json, qu'il utilise django.core.serializers.json.DjangoJSONEncoder, qui sait comment gérer les dates.

Ok, en relisant le code je comprends mon erreur.
Les attributs sont sérialisés à des fins n'ayant rien à voir avec l'API.
Il faut les désérialiser dans un premier temps, puis un nouveau coup de moulinette (décorateur json) pour que le tout sorte en JSON.

#17 - 19 janvier 2019 10:19 - Frédéric Péters

- Statut changé de Solution proposée à En cours

- Patch proposed changé de Oui à Non

#18 - 07 août 2019 10:19 - Benjamin Dauvergne

- Assigné à changé de Paul Marillonnet à Benjamin Dauvergne

#19 - 07 août 2019 10:41 - Benjamin Dauvergne

- Fichier 0002-custom_user-user-DRF-field-to-serializer-custom-attr.patch ajouté

- Fichier 0001-api-factorize-making-a-DRF-field-for-an-attribute-24.patch ajouté

- Tracker changé de Bug à Development

- Statut changé de En cours à Solution proposée

- Patch proposed changé de Non à Oui

En utilisant un DRF field pour sérialiser les attributs étendus.

#20 - 19 août 2019 17:15 - Emmanuel Cazenave

- Statut changé de Solution proposée à Solution validée

#21 - 09 septembre 2019 13:03 - Benjamin Dauvergne

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

#22 - 11 septembre 2019 16:15 - Frédéric Péters

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

Fichiers

0001-user-api-fix-attribute-value-serialization-24401.patch	915 octets	02 août 2018	Paul Marillonnet
0002-custom_user-user-DRF-field-to-serializer-custom-attr.patch	4,86 ko	07 août 2019	Benjamin Dauvergne
0001-api-factorize-making-a-DRF-field-for-an-attribute-24.patch	3,79 ko	07 août 2019	Benjamin Dauvergne