Authentic 2 - Development #36990

python3 : porter authentic2.compat.JSONField

16 octobre 2019 14:50 - Paul Marillonnet

Statut:	Fermé	**Début:**	16 octobre 2019
Priorité:	Normal	**Echéance:**	
Assigné à:		**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

SQLite et Django < 1.11 ne sont plus supportés.
Le patch simplifiera le passage à python3.

Historique

#1 - 16 octobre 2019 14:51 - Paul Marillonnet

- Fichier 0001-drop-JSONField-compatibility-with-multiple-db-vendor.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

#2 - 16 octobre 2019 14:59 - Benjamin Dauvergne

- Statut changé de Solution proposée à Rejeté

Je ne vois pas en quoi ça simplifie le support python3, et je suis attaché au support sqlite même partiel; le mieux c'est me poser la question avant de faire un patch.

#3 - 16 octobre 2019 15:16 - Paul Marillonnet

Benjamin Dauvergne a écrit :

> Je ne vois pas en quoi ça simplifie le support python3,

Solution de facilité, qui m'épargne d'implémenter des opérateurs de comparaisons entre champs.

> et je suis attaché au support sqlite même partiel; le mieux c'est me poser la question avant de faire un patch.

Je ne savais pas. Je vais laisser de côté cette solution de facilité et porter ce bout de code en python3.

#4 - 19 octobre 2019 19:57 - Benjamin Dauvergne

Paul Marillonnet a écrit :

> Benjamin Dauvergne a écrit :
>
> > Je ne vois pas en quoi ça simplifie le support python3,
>
> Solution de facilité, qui m'épargne d'implémenter des opérateurs de comparaisons entre champs.

Pour ?

#5 - 21 octobre 2019 18:18 - Paul Marillonnet

Benjamin Dauvergne a écrit :

> Pour ?

Le code de bisect.bisect qui semblerait-il a changé depuis :

```
  File "/home/paul/src/authentic/src/authentic2_auth_oidc/models.py", line 45, in <module>
    class OIDCProvider(models.Model):
  File "/tmp/tox-paul/authentic/py3-coverage-dj111-authentic-pg-/lib/python3.7/site-packages/django/db/models/
base.py", line 162, in __new__
    new_class.add_to_class(obj_name, obj)
  File "/tmp/tox-paul/authentic/py3-coverage-dj111-authentic-pg-/lib/python3.7/site-packages/django/db/models/
base.py", line 325, in add_to_class
    value.contribute_to_class(cls, name)
  File "/home/paul/src/authentic/src/authentic2/compat.py", line 90, in contribute_to_class
    cls._meta.add_field(self)
  File "/tmp/tox-paul/authentic/py3-coverage-dj111-authentic-pg-/lib/python3.7/site-packages/django/db/models/
options.py", line 277, in add_field
    self.local_fields.insert(bisect(self.local_fields, field), field)
TypeError: '<' not supported between instances of 'JSONField' and 'URLField'
ERROR: InvocationError for command /tmp/tox-paul/authentic/py3-coverage-dj111-authentic-pg-/bin/py.test --cov=
src --cov-branch --cov-append --cov-report xml --cov-report html tests/ --random-group (exited with code 1)
```

Je regarde dès que j'ai un peu de temps.

#6 - 21 octobre 2019 19:50 - Benjamin Dauvergne

On obtient cette trace en faisant quoi ?

#7 - 22 octobre 2019 13:30 - Paul Marillonnet

Benjamin Dauvergne a écrit :

> On obtient cette trace en faisant quoi ?

Directement à l'initialisation des tests[1], après avoir bien sûr adapté le fichier tox.ini :

```
diff --git a/tox.ini b/tox.ini
index 29f0921ed..34e58e60d 100644
--- a/tox.ini
+++ b/tox.ini
@@ -5,7 +5,7 @@

 [tox]
 toxworkdir = {env:TMPDIR:/tmp}/tox-{env:USER}/authentic/
-envlist = py27-coverage-dj111-authentic-pg-{oldldap,},py27-coverage-dj111-rbac-pg,pylint
+envlist = {py27,py3}-coverage-dj111-authentic-pg-{oldldap,},{py3,py27}-coverage-dj111-rbac-pg,pylint

 [testenv]
 whitelist_externals =
@@ -61,7 +61,8 @@ deps =
   ldaptools>=0.15
   oldldap: python-ldap<3
 commands =
-  ./getlasso.sh
+  py27: ./getlasso.sh
+  py3: ./getlasso3.sh
   rbac,authentic: py.test {env:FAST:} {env:REUSEDB:} {env:COVERAGE:} {env:JUNIT:} {posargs:{env:TESTS} --rand
om-group}

 [testenv:pylint]
```

La trace complète ici

#8 - 22 octobre 2019 14:30 - Benjamin Dauvergne

- Fichier JSONField_cmp.diff ajouté

Et avec ce patch ? Si c'est ok ouvre un ticket migration python3 pour ce patch (j'ai du mal à comprendre pourquoi ça n'arrive qu'en python3).

#9 - 22 octobre 2019 15:53 - Paul Marillonnet

Benjamin Dauvergne a écrit :

> Et avec ce patch ? Si c'est ok ouvre un ticket migration python3 pour ce patch (j'ai du mal à comprendre pourquoi ça n'arrive qu'en python3).

Toujours pas.

#10 - 22 octobre 2019 17:50 - Benjamin Dauvergne

Paul Marillonnet a écrit :

> Benjamin Dauvergne a écrit :
>
> > Et avec ce patch ? Si c'est ok ouvre un ticket migration python3 pour ce patch (j'ai du mal à comprendre pourquoi ça n'arrive qu'en python3).
>
> Toujours pas.

En fait il faudrait juste déléguer *eq* et *lt* à .__real_field__()

#11 - 23 octobre 2019 11:08 - Paul Marillonnet

Après quelques modifs pour que ça tourne, par dessus ton diff :

```
diff --git a/src/authentic2/compat.py b/src/authentic2/compat.py
index f7e48e49c..520de2815 100644
--- a/src/authentic2/compat.py
+++ b/src/authentic2/compat.py
@@ -51,16 +51,16 @@ class JSONField(object):
         self.__kwargs = kwargs
         if django.VERSION >= (1, 11):
             try:
-                from django.contrib.postgres.fields import JSONField
-                self.__dj11_field = JSONField(*args, **kwargs)
+                from django.contrib.postgres.fields import JSONField as Dj11Field
+                self.__dj11_field = Dj11Field(*args, **kwargs)
             except ImportError:
                 pass
         try:
-            from jsonfield.fields import JSONField
-            self.__jsonfield_field = JSONField(*args, **kwargs)
+            from jsonfield.fields import JSONField as JsonfieldField
+            self.__jsonfield_field = JsonfieldField(*args, **kwargs)
         except ImportError:
             pass
-        self.creation_counter = Field.creation_counter
+        super(JSONField, self).__setattr__('creation_counter', Field.creation_counter)
         Field.creation_counter += 1

     def __real_field__(self):
```

on tombe sur une trace qui laisse entendre qu'il va falloir un peu plus de boulot.
(i.e. le authentic2.compat.JSONField en opérande de comparaison à droite.)

Je n'ai pas d'idée pour l'instant, je creuse l'affaire dès que j'ai un peu de temps.

#12 - 24 octobre 2019 16:03 - Paul Marillonnet

- Sujet changé de Retirer la compatibilité JSONField pour SQLite et Django < 1.11 à python3 : porter authentic2.compat.JSONField

#13 - 24 octobre 2019 16:03 - Paul Marillonnet

- Statut changé de Rejeté à En cours

#14 - 25 octobre 2019 16:42 - Paul Marillonnet

Après réflexion je pense que le plus simple est d'altérer le comportement du builtin isinstance au moment de la définition des modèles qui ont un champ de type authentic2.compat.JSONField, et ceci afin de leurrer la méthode __lt__ de la classe native Field de django à l'origine de l'erreur lors de la tentative de comparaison à droite.

Peut-être un gestionnaire de contexte qui ferait un truc du genre :

```
from django.utils.six.moves import builtins as __builtin__

@contextmanager
def fake_isinstance_contextmanager():
    isinstance_orig = __builtin__.isinstance

    def fake_isinstance(obj, clz):
```

```
        if isinstance_orig(obj, compat.JSONField) and issubclass(clz, models.Field):
            return True
        else:
            return isinstance_orig(obj, clz)
```

```
    try:
        __builtin__.isinstance = fake_isinstance
        yield
    finally:
        __builtin__.isinstance = isinstance_orig
```

```
with fake_isinstance_contextmanager():
    class OIDCProvider(models.Model):
        # […]
```

#15 - 29 octobre 2019 15:43 - Paul Marillonnet

Benj, en réunion dev :

> je pense qu'on va dumper, j'abandonne

Je pense qu'on peut en profiter pour délaisser la dépendance à django-jsonfield et se baser sur le champ natif postgresql de django.

#16 - 29 octobre 2019 21:15 - Benjamin Dauvergne

Tu peux reprendre ton premier patch et y ajouter de virer tout ce qui concerne sqlite dans les tests.

#17 - 15 novembre 2019 17:44 - Paul Marillonnet

- Fichier 0001-Revert-add-compatibility-layer-for-support-of-Django.patch ajouté

- Fichier 0002-auth_oidc-drop-now-redundant-django-jsonfield-depend.patch ajouté

- Fichier 0003-tests-drop-partial-sqlite-support-36990.patch ajouté

- Statut changé de En cours à Solution proposée

Sans doute comme ça. Dans le premier patch je passe par un revert tout bête qui réintroduit la dépendance django-jsonfield, pour la virer juste après, dans le second patch. Est-ce que c'est grave docteur ? :)

Autre question que je me pose : aurait-on intérêt à tester que le patch modifiant la migration ne va pas casser les champs jusque-là stockés sous forme de jsonfield.JSONField, et qui à l'insu de leur plein gré deviennent des django.contrib.postgres.fields.JSONField ?

#18 - 21 novembre 2019 17:14 - Paul Marillonnet

- Fichier 0001-Revert-add-compatibility-layer-for-support-of-Django.patch ajouté

- Fichier 0002-auth_oidc-drop-now-redundant-django-jsonfield-depend.patch ajouté

- Fichier 0003-tests-drop-partial-sqlite-support.patch ajouté

Paul Marillonnet a écrit :

> Autre question que je me pose : aurait-on intérêt à tester que le patch modifiant la migration ne va pas casser les champs jusque-là stockés sous forme de jsonfield.JSONField, et qui à l'insu de leur plein gré deviennent des django.contrib.postgres.fields.JSONField ?

En ayant relu le code de django-jsonfield et du code de contrib de django, je suis rassuré.
On est sur du pg>9.4 sur Debian Stretch, donc la situation où des données jusque là stockées en type pg 'TEXT' ou 'JSON' seraient interprétées comme du 'JSONB' suite à la migration ne se posera pas (et quand bien même elle se poserait, il y a l'intelligence dans psycopg2 pour faire face à ce genre de situation[1]).

Je remets donc sur la table les trois patches (quasiment) inchangés.

[1] https://github.com/psycopg/psycopg2/blob/master/lib/_json.py#L158

#19 - 21 novembre 2019 17:19 - Benjamin Dauvergne

- Statut changé de Solution proposée à Solution validée

#20 - 21 novembre 2019 17:46 - Paul Marillonnet

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

```
commit 2cc2cf4200b69ddc36b6262456b7afbadfa0d963
```

```
Author: Paul Marillonnet <pmarillonnet@entrouvert.com>
Date:   Fri Nov 15 17:16:50 2019 +0100

    tests: drop partial sqlite support

commit 6d064a2c223e900d670b398578e971f73aea6a8b
Author: Paul Marillonnet <pmarillonnet@entrouvert.com>
Date:   Fri Nov 15 17:01:36 2019 +0100

    auth_oidc: drop now-redundant django-jsonfield dependency

commit cef04caf83376793c29cb58d4bc0fc690b251762
Author: Paul Marillonnet <pmarillonnet@entrouvert.com>
Date:   Tue Oct 29 11:18:16 2019 +0100

    Revert "add compatibility layer for support of Django native JSONField (fixes #29193)"

    This reverts commit d730dba525215d615fa2b974d44183d16909f4d2.
```

#21 - 21 novembre 2019 17:49 - Paul Marillonnet

Et j'avais les bons messages de commit dans une branche locale, que j'ai écrasée par erreur sans m'en rendre compte. Mes excuses pour ces messages de commit qui ne font pas apparaître le numéro de ticket. :/

#22 - 15 février 2020 20:18 - Frédéric Péters

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

Fichiers

0001-drop-JSONField-compatibility-with-multiple-db-vendor.patch	7,3 ko	16 octobre 2019	Paul Marillonnet
JSONField_cmp.diff	1,46 ko	22 octobre 2019	Benjamin Dauvergne
0002-auth_oidc-drop-now-redundant-django-jsonfield-depend.patch	2,65 ko	15 novembre 2019	Paul Marillonnet
0001-Revert-add-compatibility-layer-for-support-of-Django.patch	10,2 ko	15 novembre 2019	Paul Marillonnet
0003-tests-drop-partial-sqlite-support-36990.patch	5,41 ko	15 novembre 2019	Paul Marillonnet
0002-auth_oidc-drop-now-redundant-django-jsonfield-depend.patch	3,03 ko	21 novembre 2019	Paul Marillonnet
0001-Revert-add-compatibility-layer-for-support-of-Django.patch	10,1 ko	21 novembre 2019	Paul Marillonnet
0003-tests-drop-partial-sqlite-support.patch	5,4 ko	21 novembre 2019	Paul Marillonnet