w.c.s. - Bug #50181

rendu décimaux dans les gabarits

15 janvier 2021 15:25 - Frédéric Péters

Statut:	Fermé	**Début:**	15 janvier 2021
Priorité:	Normal	**Echéance:**	
Assigné à:	Frédéric Péters	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

{{ "1000"|decimal }} → "1E+3".

```
>>> import decimal
>>> decimal.Decimal('1000').quantize(decimal.Decimal('1.000000')).normalize()
Decimal('1E+3')
```

Révisions associées

Révision c1ec3ef6 - 17 janvier 2021 13:49 - Frédéric Péters

misc: use django localize method to format decimals in json exports (#50181)

Révision cc1a6a11 - 17 janvier 2021 13:49 - Frédéric Péters

misc: use django rendition of variables as complex data string values (#50181)

Révision d9aa4149 - 17 janvier 2021 13:51 - Frédéric Péters

misc: use django localize method to format decimals in json exports (#50181)

Révision c546d5a1 - 17 janvier 2021 13:51 - Frédéric Péters

misc: use django rendition of variables as complex data string values (#50181)

Historique

#2 - 15 janvier 2021 15:27 - Frédéric Péters

(y compris avec complex-data désactivé).

#3 - 15 janvier 2021 18:02 - Frédéric Péters

- Fichier 0001-misc-add-custom-handling-for-decimals-as-complex-dat.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

#50180 parle de données de traitement mais a priori cette situation va se trouver gérée par #50184 (parce que les champs texte ont allow_complex à False et le code qui était à moitié désactivé le sera désormais totalement).

Reste ici alors les appels webservices, pour cette situation dans la sérialisation json utilisée, plutôt que str() sur les décimaux j'appelle le localize() de django qui met ça en forme correctement. Dans une première version du patch j'avais modifié le traitement pour envoyer des entiers/flottants natifs json mais finalement non, quand arrivera un appel webservice qui demandera ça je préférerai l'ajout de filtres type |integer.

Aussi, pour les situations où la chaine "complexe" se trouverait quand même produite et juste nettoyée des marqueurs, je fais pareil j'appelle localize() au moment de la production du rendu texte.

#4 - 16 janvier 2021 17:34 - Benjamin Dauvergne

Est-ce que ça n'aurait pas été plus simple d'utiliser orig_render_value_in_context pour obtenir la valeur de type str plutôt que de reproduire le comportement de Django de manière ad-hoc ?

#5 - 16 janvier 2021 17:41 - Frédéric Péters

Je dirais que non, juste pour répondre ça je m'interroge déjà sur le fait d'appeler depuis cache_complex_data la méthode render_value_in_context qui elle-même peut faire un cache_complex_data et de devoir alors réfléchir pour être sûr que non ça ne va pas boucler etc. et tu notes

orig_render_value_in_context, pas render_value_in_context, et problème différent donc, peut-être importer le module monkeypatch dans wcs/publisher.py mais alors devoir s'interroger sur le risque qu'il y ait à ce qu'il soit chargé deux fois, etc. et donc vraiment pour moi c'était plus simple ainsi.

#6 - 16 janvier 2021 22:24 - Benjamin Dauvergne

Pas clair de ma part, je pensais à ça pour laisser à Django le soin de transformer les valeurs en chaînes dans tous les cas, ça ne boucle pas.

```
diff --git a/wcs/monkeypatch.py b/wcs/monkeypatch.py
index 6797e415..fa59040c 100644
--- a/wcs/monkeypatch.py
+++ b/wcs/monkeypatch.py
@@ -117,9 +117,10 @@ if not hasattr(django.template.base, '_monkeypatched'):
     orig_render_value_in_context = django.template.base.render_value_in_context

     def new_render_value_in_context(value, context):
+        rendered_value = orig_render_value_in_context(value, context)
         if context.get('allow_complex') and not isinstance(value, str):
-            return get_publisher().cache_complex_data(value)
-        return orig_render_value_in_context(value, context)
+            return get_publisher().cache_complex_data(value, rendered_value)
+        return rendered_value

     django.template.base.render_value_in_context = new_render_value_in_context
     django.template.defaulttags.render_value_in_context = new_render_value_in_context
diff --git a/wcs/publisher.py b/wcs/publisher.py
index e070cef6..ccec5060 100644
--- a/wcs/publisher.py
+++ b/wcs/publisher.py
@@ -394,7 +394,7 @@ class WcsPublisher(StubWcsPublisher):
         finally:
             self.complex_data_cache = None

-    def cache_complex_data(self, value):
+    def cache_complex_data(self, value, rendered_value):
         # Keep a temporary cache of assocations between a complex data value
         # (value) and a string reprensentation (str(value) or the dedicated
         # str_value parameter).
@@ -406,8 +406,7 @@ class WcsPublisher(StubWcsPublisher):
             # it doesn't do anything unless initialized.
             return value

-        str_value = str(value)
-        str_value += chr(0xE000 + len(self.complex_data_cache))
+        str_value = rendered_value + chr(0xE000 + len(self.complex_data_cache))
         self.complex_data_cache[str_value] = value
         return str_value
```

#7 - 17 janvier 2021 07:55 - Frédéric Péters

Ok je vois mieux, oui ça fait le job et n'est pas plus compliqué, j'ai mis à jour la branche avec ça (et le commentaire de la méthode mis à jour).

#8 - 17 janvier 2021 11:48 - Benjamin Dauvergne

- Statut changé de Solution proposée à Solution validée

Ok, dernière remarque pour la forme, j'aurai mis la modification à JSONEncoder dans un commit à part, genre represent Decimal in JSON as with Django templates ou alors en commentaire.

#9 - 17 janvier 2021 13:49 - Frédéric Péters

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

Voilà, en deux,

```
commit cc1a6a112b0328263d752d926e216ccd85d75d74
Author: Frédéric Péters <fpeters@entrouvert.com>
Date:   Sun Jan 17 11:56:16 2021 +0100

    misc: use django rendition of variables as complex data string values (#50181)

commit c1ec3ef67a83920759d302ef9b3b7d962c181c4d
Author: Frédéric Péters <fpeters@entrouvert.com>
Date:   Sun Jan 17 11:55:22 2021 +0100
```

```
misc: use django localize method to format decimals in json exports (#50181)
```

#10 - 18 janvier 2021 09:16 - Frédéric Péters

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

Fichiers

0001-misc-add-custom-handling-for-decimals-as-complex-dat.patch	4,77 ko	15 janvier 2021	Frédéric Péters