Combo - Development #33393

get_template_url ne fonctionne plus comme attendu

23 mai 2019 19:12 - Benjamin Dauvergne

Statut:	Fermé	**Début:**	23 mai 2019
Priorité:	Bas	**Echéance:**	
Assigné à:		**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

C'est certainement un bug dans le code appelant suite à passage en Django 1.11 je peux en convenir, dans combo-plugin-nanterre on fait ça :

```
    context = RequestContext(request, {'request': request,
                                       'rsu_id': rsu_id,
                                       'qf_id': qf_id})
    url = get_templated_url(url, context=context)
```

et get_templated_url() fait ça :

```
    template_vars = Context(use_l10n=False)
    if context:
        template_vars.update(context)
        template_vars['user_email'] = ''
        template_vars['user_nameid'] = ''
        user = getattr(context.get('request'), 'user', None)
        if user and user.is_authenticated():
            template_vars['user_email'] = quote(user.email)
            user_nameid = user.get_name_id()
            if user_nameid:
                template_vars['user_nameid'] = quote(user_nameid)
    template_vars.update(settings.TEMPLATE_VARS)
```

sauf que RequestContext ne se comporte visiblement plus comme un dico :

```
n [14]: c = Context(use_l10n=False)

In [15]: req = HttpRequest()

In [16]: c.update(RequestContext(req, {'rsu_id': 'coin'}))
Out[16]: {}

In [17]: c
Out[17]: [{'False': False, 'None': None, 'True': True}, {}]

In [18]: c.update({'rsu_id': 'coin'})
Out[18]: {'rsu_id': 'coin'}

In [19]: c
Out[19]: [{'False': False, 'None': None, 'True': True}, {}, {'rsu_id': 'coin'}]
```

Révisions associées

Révision 6b7d5938 - 27 mai 2019 10:00 - Frédéric Péters

utils: flatten context passed to get_templated_url (#33393)

Historique

#2 - 23 mai 2019 19:30 - Benjamin Dauvergne

M'est avis qu'on devrait faire usage de django.template.context.make_context comme les implémentations de render dans les backends de template :

```
def make_context(context, request=None, **kwargs):
    """
    Create a suitable Context from a plain dict and optionally an HttpRequest.
    """
    if context is not None and not isinstance(context, dict):
        raise TypeError('context must be a dict rather than %s.' % context.__class__.__name__)
    if request is None:
        context = Context(context, **kwargs)
    else:
        # The following pattern is required to ensure values from
        # context override those from template context processors.
        original_context = context
        context = RequestContext(request, **kwargs)
        if original_context:
            context.push(original_context)
    return context
```

Ça veut dire passer explicitement request à get_templated_url, et juste un dico dans context.

#3 - 23 mai 2019 19:39 - Benjamin Dauvergne

- Priorité changé de Normal à Bas

#4 - 23 mai 2019 21:40 - Thomas Noël

Ouaip. J'avoue que ces contextes qui "ressemblent" à des dict mais en fait non, ça me perd souvent ; avec l'impression brumeuse que Django a plusieurs fois changé d'idée sur le sujet.

Sinon comme résolution, on pourrait aussi dans get_template_url, si context dispose d'une méthode flatten(), faire un context = context.flatten() afin d'en faire un vrai dico. Comme ça, on se fiche que context soit un dico ou un "vrai" context Django, ça marche...

Genre :

```
--- a/combo/utils/urls.py
+++ b/combo/utils/urls.py
@@ -36,6 +36,9 @@ def get_templated_url(url, context=None):
         return url
     template_vars = Context(use_l10n=False)
     if context:
+        if hasattr(context, 'flatten'):
+            # it's a django Context, dictionnarize it:
+            context = context.flatten()
         template_vars.update(context)
         template_vars['user_email'] = ''
         template_vars['user_nameid'] = ''
```

Ceci dit, j'ai ajouté le test ci-dessous dans tests/test_templated_url.py et il marche très bien avec le code actuel... je rate quoi ?

```
+    # requestcontext
+    context = RequestContext(request, {'request': request, 'foo': 'bar'})
+    assert get_templated_url('{{ foo }}', context=ctx) == 'bar'
+    assert get_templated_url('[foo]', context=ctx) == 'bar'
```

#5 - 23 mai 2019 22:08 - Benjamin Dauvergne

En django 1.11 ?

#6 - 23 mai 2019 22:17 - Thomas Noël

Benjamin Dauvergne a écrit :

> En django 1.11 ?

Oui, j'ai revérifié plusieurs fois aussi, j'arrive pas à bien piger ... tox -e django111

#7 - 23 mai 2019 23:01 - Benjamin Dauvergne

Sur nanterre-test-web1 ça marche pas (shell dans tenant agents) :

```
In [8]: get_templated_url('[foo]', RequestContext(HttpRequest(), {'foo': 'zoo'}))
---------------------------------------------------------------------------
TemplateError                             Traceback (most recent call last)
<ipython-input-8-820f9a5e7469> in <module>()
----> 1 get_templated_url('[foo]', RequestContext(HttpRequest(), {'foo': 'zoo'}))
```

```
/usr/lib/python2.7/dist-packages/combo/utils/urls.pyc in get_templated_url(url, context)
     62             raise TemplateError('unknown variable %s', varname)
     63         return force_text(template_vars[varname])
---> 64     return re.sub(r'(\[.+?\])', repl, url)

/usr/lib/python2.7/re.pyc in sub(pattern, repl, string, count, flags)
    153     a callable, it's passed the match object and must return
    154     a replacement string to be used."""
--> 155     return _compile(pattern, flags).sub(repl, string, count)
    156
    157 def subn(pattern, repl, string, count=0, flags=0):

/usr/lib/python2.7/dist-packages/combo/utils/urls.pyc in repl(matchobj)
     60             return '['
     61         if varname not in template_vars:
---> 62             raise TemplateError('unknown variable %s', varname)
     63         return force_text(template_vars[varname])
     64     return re.sub(r'(\[.+?\])', repl, url)

TemplateError: unknown variable foo

In [9]: django.version
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-9-271baf40d83f> in <module>()
----> 1 django.version

NameError: name 'django' is not defined

In [10]: import django

In [11]: django.VERSION
Out[11]: (1, 11, 20, u'final', 0)
```

#8 - 24 mai 2019 08:31 - Frédéric Péters

- Fichier 0001-utils-flatten-context-passed-to-get_templated_url-33.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

```
+    # requestcontext
+    context = RequestContext(request, {'request': request, 'foo': 'bar'})
+    assert get_templated_url('{{ foo }}', context=ctx) == 'bar'
+    assert get_templated_url('[foo]', context=ctx) == 'bar'
```

La variable s'appelle context mais c'est ctx qui est passé à get_templated_url. Cela étant même en changeant ça, ça passe.

Anyway, patch avec le flatten et le test de Benj, et hop.

#9 - 24 mai 2019 09:01 - Paul Marillonnet

Frédéric Péters a écrit :

> Anyway, patch avec le flatten et le test de Benj, et hop.

Et un seul 'n' à dictionarize :)

#10 - 24 mai 2019 09:59 - Thomas Noël

- Statut changé de Solution proposée à Solution validée

Je confirme que le test fait bien foirer en django111 (quand on l'écrit correctement... misère). Et que le patch sur utils/urls.py corrige.

#11 - 27 mai 2019 10:01 - Frédéric Péters

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

```
commit 6b7d5938fcc1446e0ace7f81ed0c71e839c391ce
Author: Frédéric Péters <fpeters@entrouvert.com>
Date:   Fri May 24 08:27:31 2019 +0200
```

```
    utils: flatten context passed to get_templated_url (#33393)
```

#12 - 27 mai 2019 15:15 - Frédéric Péters

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

Fichiers

0001-utils-flatten-context-passed-to-get_templated_url-33.patch	2,13 ko	24 mai 2019	Frédéric Péters