w.c.s. - Development #22106

Création paresseuse des variables de substitution

25 février 2018 19:29 - Frédéric Péters

Statut:	Fermé	**Début:**	25 février 2018
Priorité:	Bas	**Echéance:**	
Assigné à:	Frédéric Péters	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Non	**Planning:**	

Description

Aujourd'hui on calcule nécessairement tout et ça prend du temps, on doit pouvoir mettre en place de quoi calculer une variable uniquement quand elle est référencée.

Demandes liées:

Lié à w.c.s. - Development #22738: Visibilité dynamique de champs dans une page	**Rejeté**	**22 mars 2018**
Lié à w.c.s. - Development #436: Affichage conditionnel sur une même page	**Fermé**	**16 mai 2011**

Révisions associées

Révision fda6dd4a - 17 août 2018 13:44 - Frédéric Péters

general: add lazy evaluation to substitution subvariables (#22106)

Historique

#1 - 22 mars 2018 13:32 - Frédéric Péters

- Lié à Development #22738: Visibilité dynamique de champs dans une page ajouté

#2 - 21 mai 2018 21:59 - Frédéric Péters

- Statut changé de Nouveau à En cours

- Assigné à mis à Frédéric Péters

Je viens de pousser https://git.entrouvert.org/wcs.git/log/?h=wip/22106-lazy-variables qui assure ça; pour le moment la vue /inspect dépend encore d'une création statique/~exhaustive des variables possibles; ça reste encore à changer dans la branche.

A priori à venir aussi, il deviendra possible de dégager le code de cache de la création des variables de substitution.

#3 - 22 mai 2018 08:47 - Frédéric Péters

À faire aussi, mettre en place des dictionnaires qui fournissent les clés sur un *getattr*, parce qu'en Python le mélange d'accès via . et [] n'est pas clair. (cela étant, en attendant, l'accès exclusif par _ fonctionne, je ne sais pas si on veut à la fois mettre en place ce ticket et se mettre à suggérer de ne plus utiliser les _).

#4 - 08 août 2018 10:02 - Frédéric Péters

- Lié à Development #436: Affichage conditionnel sur une même page ajouté

#5 - 08 août 2018 10:11 - Frédéric Péters

- Statut changé de En cours à Solution proposée

La branche contient maintenant tests et cie. Aussi pour limiter les risques on passe par le mode lazy uniquement pour les conditions django; et ça se contrôle via site-options.cfg dans lazy-variables-modes, qui est une liste des endroits où on veut appliquer ce nouveau mode (et par défaut c'est juste django-condition).

#6 - 16 août 2018 17:47 - Thomas Noël

Si je comprends l'idée, c'est de produire pour les sources capable de faire du lazy deux méthodes

- get_static_substitution_variables : toutes les variables, exhaustivement, donc "comme d'hab"
- get_substitution_variables : une source capable de réagir en lazy via l'usage d'un dico CompatibilityNamesDict

C'est cela ?

Sinon, j'ai l'impression que le mode python-conditions ne sera jamais vraiment fonctionnel, à cause de l'habituel et souvent nécessaire usage de la forme vars().get('form_var_chose'). Me trompe-je ? Si oui, pourquoi ne pas activer le lazy en Django, toujours (et donc jamais pour Python) ?

#7 - 16 août 2018 17:54 - Frédéric Péters

> Sinon, j'ai l'impression que le mode python-conditions ne sera jamais vraiment fonctionnel, à cause de l'habituel et souvent nécessaire usage de la forme vars().get('form_var_chose'). Me trompe-je ?

D'une part il y a espoir de lentement faire évoluer les documentations et usages vers des formes plus réelles form.foo.bar, plutôt que ces underscores.

D'autre part, pour gérer le vars(), je pensais l'avoir fait mais c'est le locals() que j'ai touché,

```
         return {'datetime': datetime,
                 'Decimal': Decimal,
+                'locals': compat_locals,
                 'random': random.SystemRandom(),
                 're': re,
                 'date': utils.date,
```

> Si oui, pourquoi ne pas activer le lazy en Django, toujours (et donc jamais pour Python) ?

Je pense qu'on peut y arriver pour le Python, un jour; et en même temps que côté django, frilosité au début, faire ça uniquement sur les conditions, pas sur les templates.

#8 - 17 août 2018 09:14 - Thomas Noël

Frédéric Péters a écrit :

> > Sinon, j'ai l'impression que le mode python-conditions ne sera jamais vraiment fonctionnel, à cause de l'habituel et souvent nécessaire usage de la forme vars().get('form_var_chose'). Me trompe-je ?
>
> D'une part il y a espoir de lentement faire évoluer les documentations et usages vers des formes plus réelles form.foo.bar, plutôt que ces underscores.

Yes.

> D'autre part, pour gérer le vars(), je pensais l'avoir fait mais c'est le locals() que j'ai touché,

Ah oui en plus je l'avais vu... on pourrait faire pareil pour vars ? (c'est la pratique que j'ai répandue, parce que je pensais le nom vars un peu plus "compréhensible" que locals, mouaich mouaich)

> [...]
>
> > Si oui, pourquoi ne pas activer le lazy en Django, toujours (et donc jamais pour Python) ?
>
> Je pense qu'on peut y arriver pour le Python, un jour; et en même temps que côté django, frilosité au début, faire ça uniquement sur les conditions, pas sur les templates.

Effectivement.

(Je reprends la lecture de la branche)

#9 - 17 août 2018 09:39 - Frédéric Péters

> Ah oui en plus je l'avais vu... on pourrait faire pareil pour vars ? (c'est la pratique que j'ai répandue, parce que je pensais le nom vars un peu plus "compréhensible" que locals, mouaich mouaich)

Yes, ajoutée à la branche.

#10 - 17 août 2018 12:31 - Thomas Noël

- Statut changé de Solution proposée à Solution validée

Rien d'autre à ajouter... c'est un ack.

#11 - 17 août 2018 13:44 - Frédéric Péters

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

```
commit fda6dd4a354eb5a90ce1edeacca0b03f290ea59a
Author: Frédéric Péters <fpeters@entrouvert.com>
Date:   Mon May 21 10:35:37 2018 +0200

    general: add lazy evaluation to substitution subvariables (#22106)
```

#12 - 23 décembre 2018 14:42 - Frédéric Péters

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