w.c.s. - Development #17437

avoir dans utils un outil "compteur"

07 juillet 2017 11:48 - Thomas Noël

Statut:	Fermé	**Début:**	07 juillet 2017
Priorité:	Normal	**Echéance:**	
Assigné à:		**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Non	**Planning:**	Non

Description

Suite à #17420, quand wcs est utilisé pour produire des sorties (factures, documents officiels) qui doivent avoir un numéro d'ordre "sans trou".

L'idée serait de construire :

```
utils.get_new_counter_value('cheque-cantine')
```

ou plus simplement :

```
utils.counter('cheque-cantine')
```

qui renvoie un numéro qui s'incrémente à chaque appel. Ce numéro serait en général stocké dans un champ de traitement de la demande.

Coder cela en mode SQL seulement, avec utilisation d'une séquence, nommée selon le nom du compteur.

Historique

#1 - 21 septembre 2017 15:17 - Benjamin Dauvergne

Faut pas utiliser les séquences postgres, elles ne sont pas intégré aux transactions (en gros quand tu fais nextval('masequence') et que la transaction échoue le compteur n'est pas diminué de un, parce que sinon ça implique un lock sur le compteur et ça fait des contentions entre transactions qui font juste des insertions, mais ça veut dire que les primary key dans postgres n'ont rien à voir avec l'ordre de commit des lignes dans la table (on peut avoir t.id < t.id et t.time > t.time même si t.time à pour valeur par défaut now()).

Donc la bonne façon de faire c'est une table séquence à deux colonnes 'name' et 'value' et on fait des

```
update sequence set value = value + 1 where sequence = 'masequence' return value
```

et là on est sûr que le truc est incrémenté que si la transaction va jusqu'au bout (mais il y aura des contentions entre demandes de numéro, pas bien grave).

#5 - 22 mars 2018 20:13 - Thomas Noël

Note : on parle ici de utils. mais on est passé en Django, on pourrait donc partir sur un tag counter disponible dans dans {% load qommon %}

- {% counter "foo" %} : renvoie 1, 2, 3, ... incrémenté à chaque appel sur "foo"
- {% counter "foo" get %} : renvoie la valeur actuelle du compteur
- {% counter "foo" silent %} : incrémente mais ne dit rien
- {% counter "foo" as bar %} : incrémente et stocke la valeur dans {{bar}}
- {% counter "foo" set 10 %} : remet le compteur à 10 (pourrait être une variable)
- {% counter "foo" add 5 %} : ajoute 5 au compteur (peut être un nombre négatif ou une variable)

Et soyons fous, la possibilité d'avoir un compteur sur une variable, genre :

- {% counter form_slug %} : et hop, un compteur sur le formulaire, quel qu'il soit

Et si c'est bien fait on peut cumuler tout ça, genre :

- {% counter form_slug add 3 silent as counter %}On a encore vendu 3 pépitos, ça en fait {{ counter }} au total

#6 - 22 mars 2018 21:12 - Benjamin Dauvergne

Thomas Noël a écrit :

> Note : on parle ici de utils. mais on est passé en Django, on pourrait donc partir sur un tag counter disponible dans dans @{% load qommon ? [...]
> Et si c'est bien fait on peut cumuler tout ça, genre :
>
> - {% counter form_slug add 3 silent as counter %}On a encore vendu 3 pépitos, ça en fait {{ counter }} au total

Je ne suis pas fan des trucs qui ont un effet de bord au milieu d'un template, c'est mon coté Haskell mais sauf si c'est fait transactionellement et tout et tout je préfère encore une action de workflow explicite "Obtenir un nouveau numéro et affecter à une variable de traitement", là on multiplie les états cachés.

#7 - 22 mars 2018 22:37 - Thomas Noël

Benjamin Dauvergne a écrit :

> Je ne suis pas fan des trucs qui ont un effet de bord au milieu d'un template, c'est mon coté Haskell mais sauf si c'est fait transactionellement et tout et tout je préfère encore une action de workflow explicite "Obtenir un nouveau numéro et affecter à une variable de traitement", là on multiplie les états cachés.

Une action "Compteur" effectivement ça me semble aller dans le bon sens. Avec en paramétrage :

- le nom du compteur (qui peut être une variable via l'usage de template Django)
- éventuellement la valeur à ajouter (là aussi possibilité d'expression souple via template) ; par défaut c'est « 1 »
- la donnée de traitement cible, à choisir parmi les champs de type texte disponibles

Et ne rendre cette action visible dans la liste des choix d'action que s'il y a au moins une donné de traitement texte, et qu'on est en mode SQL, ça me semble une bonne idée.

A voir si ça répond à Meyzieu.

#8 - 23 mars 2018 09:38 - Aude MERITZA-BOZON

Bonjour,

> A voir si ça répond à Meyzieu.

cela peut répondre si, comme je l'ai compris, il s'agit de la mise à disposition d'une variable associée au formulaire (et non au téléservice usager) qui s'incrémente à chaque appel de l'action "compteur" dans le workflow associé. Sa valeur est alors stockée dans une donnée de traitement cible pour chaque téléservice usager auquel il s'applique.

#9 - 23 mars 2018 10:26 - Thomas Noël

Aude MERITZA-BOZON a écrit :

> Bonjour,
>
> > A voir si ça répond à Meyzieu.
>
> cela peut répondre si, comme je l'ai compris, il s'agit de la mise à disposition d'une variable associée au formulaire (et non au téléservice usager) qui s'incrémente à chaque appel de l'action "compteur" dans le workflow associé. Sa valeur est alors stockée dans une donnée de traitement cible pour chaque téléservice usager auquel il s'applique.

Le plan qui est imaginé ici est d'avoir un compteur lié à "un nom" (par exemple "inscription-base-nautique-2018"), qui est transversal à toute la plateforme. Ce nom serait à préciser dans l'action. Au passage dans l'action, quand l'inscription est validée, la demande concernée gagne un nouveau numéro d'inscription à la base nautique. On peut même imaginer que ça soit utilisé dans des workflows différents, ça restera un compteur unique.

Cela permet de générer des numéros d'inscription dans l'ordre, sans trou ; c'est je crois ce qui était attendu (dans la note numéro 4 du ticket #17420 https://dev.entrouvert.org/issues/17420#note-4)

J'espère avoir réussi à être clair...?

#10 - 23 mars 2018 14:52 - Aude MERITZA-BOZON

C'est tout à fait ça. :-)

#11 - 29 mars 2018 10:19 - Serghei Mihai

- Assigné à mis à Serghei Mihai

Je prends.

#12 - 29 mars 2018 12:28 - Benjamin Dauvergne

Pour inspiration seulement je transfère ma note privée du ticket Meyzieu:

Benjamin Dauvergne a écrit :

> Thomas Noël a écrit :
>
> > Frédéric Péters a écrit :
> >
> > > Je ne sais pas pourquoi tu veux attendre lundi, ce n'est pas avec trois minutes par personne sur un sujet que tout le monde découvrira qu'il y aura une réponse pertinente, permettant de fixer qui/quand/comment.
> >
> > J'ai posé une idée de syntaxe sur https://dev.entrouvert.org/issues/17437 ; Benjamin a donné la technique derrière (une simple table SQL clé/valeur "bête et méchante").
>
> Si personne a le temps on peut aussi poser un script dans teleservics.meyzieu.fr/scripts/counter.py

```
from wcs.sql import get_connection()
name = args[0]

conn = get_connection(new=True)
with conn.cursor() as cur:
    cur.execute('CREATE TABLE IF NOT EXISTS meyzieu_counter (name varchar primary key, value int)')
    conn.commit()
    while True:
        try:
             cur.execute('update meyzieu_counter set value = value + 1 where name = %s returning value', name)
             r = cur.fetchone()
             if r:
                 result = r[0]
             else:
                 cur.execute('insert into meyzieu_counter values (%s, 0)', name)
                 result = 0
        exception conn.Error:
             conn.rollback()
        break
    conn.commit()
```

#13 - 04 avril 2018 18:23 - Serghei Mihai

Je propose ça à Aude, car je ne suis qu'à un brouillon de patch.
Je viens de tester en local le code proposé par Benj avec quelques corrections mineures:

```
from wcs.sql import get_connection

name = args[0]

conn = get_connection(new=True)
with conn.cursor() as cur:
    cur.execute('CREATE TABLE IF NOT EXISTS meyzieu_counter (name varchar primary key, value int)')
    conn.commit()
    while True:
        try:
             cur.execute('update meyzieu_counter set value=value+1 where name=%s returning value', (name,))
             r = cur.fetchone()
             if r:
                 result = str(r[0])
             else:
                 cur.execute('insert into meyzieu_counter values (%s, 0)', (name,))
                 result = '0'
        except conn.Error:
            conn.rollback()
        break
    conn.commit()
```

et ça fonctionne.

#14 - 04 avril 2018 18:28 - Frédéric Péters

Ce ticket est pour avoir la fonctionnalité en standard dans w.c.s.

#16 - 15 novembre 2021 17:28 - Serghei Mihai

- Statut changé de Nouveau à Fermé

- Planning mis à Non

C'est possible aujourd'hui d'avoir le nombre de demandes avec une expression du genre : {{forms|objects:"slug"|count }}

#17 - 16 novembre 2021 08:25 - Frédéric Péters

- Statut changé de Fermé à Nouveau

- Assigné à Serghei Mihai supprimé

Ça ne répond pas (le compteur dont il serait question ici n'est pas lié au nombre de demandes pour une démarche particulière, il doit pouvoir s'appliquer sur plusieurs démarches, cf exemple "facture" qui pourraient être produites par différentes démarches, et doit également pouvoir être utilisé plusieurs fois sur la durée de vie d'une demande (genre si deux factures sont produites par la même demande).

#18 - 09 janvier 2024 11:09 - Frédéric Péters

- Statut changé de Nouveau à Fermé

Comme le besoin n'est jamais revenu, on peut effectivement fermer.