Combo - Bug #10179

pagination dans les cellules listant les demandes de l'usager

03 mars 2016 15:20 - Frédéric Péters

Statut:	Fermé	**Début:**	03 mars 2016
Priorité:	Normal	**Echéance:**	
Assigné à:	Valentin Deniaud	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

(pour les usagers qui font quantité de demandes…) (un cas réel ce serait un avocat faisant des demandes d'acte de naissance).

Révisions associées

Révision 8196e9c1 - 22 novembre 2019 10:01 - Valentin Deniaud

wcs: add pagination for current forms/drafts cells (#10179)

Historique

#1 - 16 octobre 2019 17:42 - Valentin Deniaud

- Assigné à mis à Valentin Deniaud

#2 - 17 octobre 2019 11:40 - Valentin Deniaud

Premier truc qui apparaît bloquant :

```
215         for wcs_slug, wcs_site in wcs_sites.items():
```

ie la pagination doit être faite entre plusieurs wcs. Pour faire propre on veut avoir le total des formulaires, ce qui permet d'afficher les boutons « 1, 2, 3, 4, etc » et « last page ».

Donc on se dit :

1. Pour tous les sites, demander le total des formulaires (ajouter un endpoint/un paramètre à /api/user/forms
2. regarder quelle page est demandée, calculer, et faire une requête de la bonne taille au bon wcs, potentiellement à plusieurs pour remplir une page

Et en fait c'est 1. qui bloque : dans wcs la récupération des formulaires n'est pas une simple requête sql, où on aurait pu transformer un select en select count et garder des perfs correctes. Le select est suivi d'une cinquantaine de lignes de python pour déterminer la liste effective des formulaires à retourner, en fonction des permissions de l'utilisateur. Donc pour obtenir le total il faudrait select tout, filtrer tout, et faire len() du résultat, ce qui a des chances de nous faire retomber sur le problème qu'on souhaitait résoudre, rencontré dans #35023 (requête trop longue/trop coûteuse, quoique c'était peut-être juste la transmission des données par le réseau qui posait problème).

Peut-être que je passe à côté d'un truc, en attendant je continue sur le ticket en visant un semblant de pagination avec des boutons précédent/suivant/revenir au début, seuls possibles sans toucher à wcs.

#3 - 17 octobre 2019 12:05 - Frédéric Péters

Dans #35023 on parlait de plus de 4000 demandes en cours, on peut ignorer ça, rester à la limite qui a été posée de 100 demandes (par w.c.s.).

Évidemment c'était il y a trois ans et j'aurais du être plus complet sur ce que je pouvais avoir comme idée en tête à l'époque mais je dirais aujourd'hui qu'il s'agit ici davantage d'une question de présentation, qu'on peut très bien charger les 100 demandes dans l'html et avoir une pagination js.

#4 - 17 octobre 2019 12:22 - Valentin Deniaud

> une pagination js

Voilà qui est tout de suite moins sympa à coder. Dans #35403 d'il y a 2 mois tu semblais toujours avoir en tête de faire de la pagination côté serveur, j'ai quand même bien envie d'essayer de faire ça.

#5 - 17 octobre 2019 12:46 - Frédéric Péters

Attention, à prendre les choses par le côté serveur, tu vois les choses sur les API, ça peut amener à travailler dans w.c.s., pour remonter max de choses avant l'appel SQL, qu'il puisse arriver à être transformé en COUNT, ainsi il y aura moyen dans la cellule de définir de manière précise les appels à effectuer, puis il faudra caler ça sur des paramètres passés à la cellule, il n'y a actuellement pas de passage de paramètres, donc ajouter d'une manière ou d'une autre certains éléments de request.GET dans le contexte de rendu créé dans ajax_page_cell, puis faudra appeler ça et là ça voudra de toute façon dire javascript, limite davantage que dans l'option simple.

#6 - 17 octobre 2019 14:07 - Benjamin Dauvergne

Mes 2 cents : il faudrait paginer par curseur (i.e. sans pouvoir dire le nombre de pages) et pas par page, c'est toujours plus simple et plus stable, le curseur serait la date, le slug wcs et l'id de la dernière demande de la page courante, tout ça trié lexicographiquement.

Étant donné un curseur on demande à chaque w.c.s les 100 dernières demandes plus vieille que ce curseur et on les trie lexicographiquement par date, slug, id. Si tout le monde en renvoie moins de 100 on donne un nombre de demandes, sinon on dit "beaucoup trop". Pour pouvoir paginer dans les deux sens il faut gérer un curseur before et after.

PS: remarque à ne prendre en compte que si on décide vraiment de paginer la totalité des demandes; dans le cas de la récupération de maximum n*100 demandes sur les n wcs puis pagination dans la page on peut l'ignorer.

#7 - 17 octobre 2019 17:38 - Valentin Deniaud

Merci pour l'idée Benj. J'ai implémenté un truc incomplet par rapport à ce que tu as écrit, qui a l'air de marcher sur un wcs, pas encore testé sur plusieurs. Avec n le nombre de résultats à afficher par page, pour une page au pif, processus pour passer à la page suivante :

1. Faire le tour des wcs et demander n + 1 résultats plus vieux que le dernier résultat de la page courante
2. Agréger tous les résultats des wcs dans une liste, la trier sur la date
3. Tronquer à n résultats et les afficher

Le curseur pour passer à la page d'après est la date du dernier résultat après tronquage. Si on a pas eu n résultats, on sait qu'on est arrivé à la fin.

Ce qui me laisse avec les questions :

> le curseur serait la date, le slug wcs et l'id

Pourquoi sur le slug et l'id ? Si l'idée c'est de dédupliquer des demandes qui auraient le même timestamp, le problème se pose en amont dans wcs au moment de la récupération des données... Mais de manière générale ajouter un timestamp précis à la milliseconde ça pourrait permettre d'ignorer le pb.

> Si tout le monde en renvoie moins de 100 on donne un nombre de demandes, sinon on dit "beaucoup trop".

Pas compris ça.

#8 - 17 octobre 2019 17:46 - Benjamin Dauvergne

Valentin Deniaud a écrit :

> Ce qui me laisse avec les questions :
>
> > le curseur serait la date, le slug wcs et l'id
>
> Pourquoi sur le slug et l'id ? Si l'idée c'est de dédupliquer des demandes qui auraient le même timestamp, le problème se pose en amont dans wcs au moment de la récupération des données... Mais de manière générale ajouter un timestamp précis à la milliseconde ça pourrait permettre d'ignorer le pb.

Il faut un ordre absolu entre les demandes pour pas en rater d'une page à l'autre, à la milliseconde ça limite le problème mais ça ne le supprime pas, mais oui ça peut être "good enough", ta solution m'a l'air suffisante on verra à l'usage.

> > Si tout le monde en renvoie moins de 100 on donne un nombre de demandes, sinon on dit "beaucoup trop".
>
> Pas compris ça.

Je partais du postulat qu'on n'avait pas de requête pour faire un count(), mais si on en demande 100 à tout le monde et que tout le monde répond moins en fait on sait.

#9 - 17 octobre 2019 17:46 - Frédéric Péters

(sérieusement pour moi ce ticket ne vaut pas un patch de plus de vingt lignes) (je devrais être plus clair dès le début et dire qu'au-delà, ça sera très certainement rejeté).

#10 - 21 octobre 2019 15:34 - Valentin Deniaud

- Fichier 0001-Revert-wcs-limit-user-forms-cell-to-100-items-35407.patch ajouté

- Fichier 0002-wcs-add-cell-pagination-10179.patch ajouté

- Fichier 0001bis-api-add-after-before-to-api-users-.-forms-10179.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

J'explose le quota de lignes, donc je m'arrête au stade « truc qui a l'air de marcher », c'est à dire sans écrire de tests ni trop peaufiner. Le 0001bis est pour wcs, j'ouvrirai un ticket là bas si la solution a des chances de passer en prod.

Il manque aussi et surtout un commit pour ajouter un champ précis à la microseconde/modifier celui existant dans wcs, en l'état deux formulaires soumis à la même minute ne vont pas faire bon ménage.

#11 - 22 octobre 2019 09:26 - Frédéric Péters

Je ne vois pas trop comment ça marche; en local j'obtiens une cellule vide, il y a :

```
            wcs_sites = {'paginated': {'data': all_forms}}
```

mais j'imagine que WcsCurrentFormsCell::get_cell_extra_context devrait être modifié pour utiliser ça, ce qui n'est pas le cas.

Aussi, avant ça, pour l'affichage initial, quand il n'y a ni before ni after, toutes les demandes seront récupérées, ce qu'on voudrait justement éviter (ça va ramener les soucis qu'on avait avant le limit=100 de #35407).

De manière plus lointaine, quand j'écrivais ça :

> puis il faudra caler ça sur des paramètres passés à la cellule, il n'y a actuellement pas de passage de paramètres, donc ajouter d'une manière ou d'une autre certains éléments de request.GET dans le contexte de rendu créé dans ajax_page_cell

C'était parce que je ne suis pas vraiment pour un accès à request.GET dans le code des cellules. (mais en réfléchissant davantage peut-être j'arriverais à me convaincre que ce n'est pas un vrai problème).

~~

Mais même si je pense que la direction prise ici peut aboutir, je reste sur l'idée d'une version courte avec juste du javascript.

#12 - 22 octobre 2019 18:01 - Valentin Deniaud

> Je ne vois pas trop comment ça marche; en local j'obtiens une cellule vide

Au lieu d'avoir wcs_sites = {'slug1': {'data': data, 'extra_key': ...}, 'slug2': {'data': data, 'extra_key': ...}}, wcs_sites = {'paginated': {'data': all_forms}} écrabouille tout pour créer un wcs fictif au slug 'paginated' qui contient uniquement les données qu'on veut afficher. C'est sûrement pas terrible et ça n'a pas vocation à rester, mais pour mon PoC c'était le plus rapide. Je ne sais pas pourquoi ça ne marche pas chez toi mais des tests étant rouges pour l'instant ça ne m'étonne pas (je teste avec une cellule « Demandes de l'usager » qui inclut les demandes en cours et les demandes terminées).

> quand il n'y a ni before ni after, toutes les demandes seront récupérées

Bien vu, effectivement il y a une indentation en trop.

> je ne suis pas vraiment pour un accès à request.GET dans le code des cellules

Dac, si c'est ça le problème je vois moins comment m'en sortir.

> je reste sur l'idée d'une version courte avec juste du javascript

Cette solution ne m'attire pas parce qu'avoir une limite arbitraire en dur dans le code ne me semble pas très joli, mais surtout parce que l'approche actuelle me permet de me familiariser avec le fonctionnement de combo/wcs, alors que clairement je n'apprendrais rien d'utile en pondant 20 lignes de js.

#13 - 22 octobre 2019 18:12 - Frédéric Péters

> Dac, si c'est ça le problème je vois moins comment m'en sortir.

Mon idée était que la cellule déclare les paramètres qui l'intéressent (dans un attribut de la classe) et que cette info soit utilisée dans ajax_page_cell

pour alimenter le contexte.

> Cette solution ne m'attire pas parce qu'avoir une limite arbitraire en dur dans le code ne me semble pas très joli, mais surtout parce que l'approche actuelle me permet de me familiariser avec le fonctionnement de combo/wcs, alors que clairement je n'apprendrais rien d'utile en pondant 20 lignes de js.

En rester aux solutions simples est un apprentissage utile. Avoir quelque chose d'isolé et compréhensible facilitera la maintenance, pourra s'appliquer à d'autres listes affichées par Combo, qu'elles viennent d'autres endpoints de w.c.s. ou d'autres applications (comme la liste des documents récents tirée de Fargo).

#14 - 23 octobre 2019 15:59 - Valentin Deniaud

- Fichier 0001-wcs-add-pagination-for-current-form-cell-10179.patch ajouté

Voilà pour le javascript, effectivement ça devrait s'appliquer sans modif à fargo.

#15 - 05 novembre 2019 08:30 - Frédéric Péters

Il manque dans le patch attaché le fichier combo/pagination.html, et tu peux envoyer ça vers une branche ? (j'étais perturbé par le jenkins rouge mais il ne correspond pas à ce dernier code).

#16 - 05 novembre 2019 10:09 - Valentin Deniaud

- Fichier 0001-wcs-add-pagination-for-current-form-cell-10179.patch ajouté

#17 - 20 novembre 2019 15:18 - Frédéric Péters

- Fichier 0001-wcs-add-pagination-for-current-forms-drafts-cells-10.patch ajouté

Désolé j'ai pris du temps avant d'arriver ici; remarque principale il faut pouvoir fonctionner avec plusieurs cellules avec de la pagination sur la même page, ce qui entraine de charger une seule fois le code de pagination, ce qui fait qu'il peut plutôt aller dans le combo.public.js existant, ce qui fait aussi qu'on ne peut pas utiliser d'id, etc. (en remarque mineure, previous/next n'étaient pas traduits)

Mais donc j'étais parti pour faire ce tas de remarques et pour ne pas lancer dans des mauvaises pistes j'ai voulu baliser un peu mieux le terrain et plouf le js était déplacé/adapté.

Pour le rendu, à accompagner d'une modif comme celle-ci dans publik-base-theme (_cells.scss) :

```
+div.gru-content div.cell div.cell-items-pagination {
+       padding: 1rem;
+       .cell-items-pagination-next {
+               float: right;
+               margin-right: 0;
+       }
+}
```

#18 - 21 novembre 2019 11:00 - Valentin Deniaud

- Statut changé de Solution proposée à Solution validée

Nickel, juste

```
-    if (items.length < paginate_by) {
+    if (items.length <= paginate_by) {
       return;
     }
```

Sinon ça affiche les boutons (grisés) dans le cas limite où il n'y a qu'une seule page pleine.

Qui qui merge ?

#19 - 22 novembre 2019 10:02 - Frédéric Péters

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

Je viens de faire la correction pointée et de pousser :

```
commit 8196e9c1ebbd703ba8eb2464116d07366b95221a
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Wed Oct 23 15:56:15 2019 +0200

    wcs: add pagination for current forms/drafts cells (#10179)
```

#20 - 22 novembre 2019 11:15 - Frédéric Péters

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

Fichiers

0001-Revert-wcs-limit-user-forms-cell-to-100-items-35407.patch	998 octets	21 octobre 2019	Valentin Deniaud
0002-wcs-add-cell-pagination-10179.patch	5,76 ko	21 octobre 2019	Valentin Deniaud
0001bis-api-add-after-before-to-api-users-.-forms-10179.patch	1,66 ko	21 octobre 2019	Valentin Deniaud
0001-wcs-add-pagination-for-current-form-cell-10179.patch	4,89 ko	23 octobre 2019	Valentin Deniaud
0001-wcs-add-pagination-for-current-form-cell-10179.patch	5,51 ko	05 novembre 2019	Valentin Deniaud
0001-wcs-add-pagination-for-current-forms-drafts-cells-10.patch	4,66 ko	20 novembre 2019	Frédéric Péters