Combo - Development #15085

création d'une cellule permettant de faire une recherche

22 février 2017 16:12 - Thomas Noël

Statut:	Fermé	**Début:**	22 février 2017
Priorité:	Normal	**Echéance:**	
Assigné à:	Thomas Noël	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	

Description

Initialement, affiche un champ texte vide.
Lorsqu'on tape dans ce champ, lance une recherche et affiche la liste des résultats via un appel ajax.
Si un élément est sélectionné dans la liste, affiche cet élément (le champ input disparait, remplacé par un lien pour remettre à zéro la recherche)

En paramètre :

- l'URL où envoyer la requête de recherche
- l'URL où aller chercher un élément choisi (optionnel)
- type d'objet recherché = nom de variable (optionnel) qui sera utilisé dans l'URL "?varname=id" lorsque l'élément "id" est sélectionné

On doit pouvoir afficher plusieurs champs de recherche dans une même page et ils doivent savoir se compléter.

Les affichages de la liste des résultats comme d'un élément seul doivent être fonction du slug de la cellule et du type d'objet recherché (nom de variable).

Révisions associées

Révision 4c3cf5e1 - 03 mars 2017 11:09 - Thomas Noël

add generic search cell (#15085)

Historique

#1 - 22 février 2017 16:15 - Thomas Noël

Cas d'usage: page de travail RSU = trois systèmes de recherche sur une page : adult, spouse, child.

L'affichage spouse doit savoir qu'un adult a été trouvé, pour alors afficher des liens vers les formulaires en envoyant adult_id et spouse_id

Idem pour child.

La recherche d'un nouvel adulte met à zéro les autres.

#2 - 22 février 2017 16:15 - Thomas Noël

- Assigné à mis à Thomas Noël

#3 - 22 février 2017 17:29 - Thomas Noël

- Fichier 0001-add-generic-search-cell-15085.patch ajouté

- Fichier out-2.ogv ajouté

- Statut changé de Nouveau à En cours

- Patch proposed changé de Non à Oui

Voilà un code qui permet d'avoir quelque chose qui fonctionne comme indiqué.

J'ai laissé trainé dans le patch des templates "nanterre-rsu" pour donner une idée. En faisant deux cellules sur une page avec le slug "nanterre-rsu", l'une avec varname=adult et l'autre varname=child, et pour les deux search_url=https://zoo-nanterre.dev.entrouvert.org/rsu/search/?q=%(quoted_q)s et get_url=https://zoo-nanterre.dev.entrouvert.org/rsu/individu/%(quoted_id)s/

#4 - 23 février 2017 10:12 - Frédéric Péters

(relecture sur le fond)

> verbose_name = _('Backoffice Search')

Pas d'ambition pour le front ?

> ('auth', '0006_require_contenttypes_0002'),

Ça oblige django 1.8, on peut s'en passer.

> search_url = models.CharField(*, max_length=200,*
> *help_text*=('with %(q)s and %(quote_q)s'))
> get_url = models.CharField(*, max_length=200, blank=True,*
> *help_text*=('with %(id)s and %(quote_id)s'))

Les deux s'appellent "Search URL". Plutôt qu'un paramétrage dans l'UI, je préférerais que se définisse dans la config une liste de choix, dans l'idée que cette liste d'URL arrive derrière à être autoconfigurée.

Ça rendrait aussi l'ajout dans cette même cellule d'une recherche "interne" plus naturelle, il me semble.

> *varname = models.CharField(*('varname'), max_length=200, blank=True)

J'aimerais bien qu'on arrive à faire sans l'ajout d'un nouvel identifiant unique, ça pourrait pas être le slug ?

#5 - 23 février 2017 10:28 - Thomas Noël

Frédéric Péters a écrit :

> (relecture sur le fond)
>
> > verbose_name = _('Backoffice Search')
>
> Pas d'ambition pour le front ?

Je sais même plus d'où ça vient. Mais oui, bien sûr.

> > ('auth', '0006_require_contenttypes_0002'),
>
> Ça oblige django 1.8, on peut s'en passer.

C'est bien possible, j'y suis pour rien :) (makemigrations et rien regardé).

> > search_url = models.CharField(*, max_length=200,*
> > *help_text*=('with %(q)s and %(quote_q)s'))
> > get_url = models.CharField(*, max_length=200, blank=True,*
> > *help_text*=('with %(id)s and %(quote_id)s'))
>
> Les deux s'appellent "Search URL".

Oups.

> Plutôt qu'un paramétrage dans l'UI, je préférerais que se définisse dans la config une liste de choix, dans l'idée que cette liste d'URL arrive derrière à être autoconfigurée.
> Ça rendrait aussi l'ajout dans cette même cellule d'une recherche "interne" plus naturelle, il me semble.

Ok.

> > *varname = models.CharField(*('varname'), max_length=200, blank=True)
>
> J'aimerais bien qu'on arrive à faire sans l'ajout d'un nouvel identifiant unique, ça pourrait pas être le slug ?

Yep, ça doit pouvoir, je vérifie ça.

Merci.

#6 - 23 février 2017 13:47 - Thomas Noël

- Fichier 0001-add-generic-search-cell-15085.patch ajouté

Voilà une version avec les remarques précédentes. (manquent encore les tests)

Concernant slug=varname, ça marche mais pour pouvoir jouer avec les variables dans le templates, je vais un varname=slug.replace('-','_') parce que j'aime pas trop les slug avec des _ ... mais ça peut être discuté.

J'ai aussi amélioré le javascript pour appeler la recherche 300ms après avoir lâché le clavier.

#7 - 23 février 2017 14:11 - Frédéric Péters

(détail, {{item.texti}}, i en trop)

> Concernant slug=varname, ça marche mais pour pouvoir jouer avec les variables dans le templates, je vais un varname=slug.replace('-','_') parce que j'aime pas trop les slug avec des _ ... mais ça peut être discuté.

Pas d'objection à ça.

> def modify_global_context(self, context, request):
> [...]
> result = requests.get(url, cache_duration=0).json()

J'hésite là-dessus, le modify_global_context il est appelé dans le rendu d'une page et ça va donc le ralentir; je me demande dans quelle mesure cela pourrait être évité. (autrement dit, cette cellule est bloquante, quelle part de js pour qu'elle ne le soit pas ?)

M'attardant là-dessus, je me rends compte que plus important encore, et passé à côté de ça ce matin, je suis gêné par le get_url, ça transforme une cellule de recherche qui affiche les résultats d'une recherche en une cellule qui doit aussi afficher le détail d'un élément seul. Peut-être que je serais plus à l'aise avec ça en l'ayant en deux temps, avec comme premier temps la cellule de recherche, qui correspondrait à :

```
    {% if item.url %}
    <a href="{{ item.url }}">{{ item.text }} ({{ item.id }})</a>
```

Et en deuxième temps la gestion de l'affichage d'un élément,

```
    {% elif cell.search_service.get_url %}
    <a href="?{{ cell.varname }}={{ item.id }}">{{ item.texti }} ({{ item.id }})</a>
```

#8 - 23 février 2017 17:49 - Thomas Noël

- Fichier 0001-add-generic-search-cell-15085.patch ajouté

Ouaip je pensais que c'était pas possible d'avoir de l'ajax autant, mais en fait si. Je continue cependant à faire en sorte que la cellule sache à la fois propose la recherche puis afficher un résultat sélectionné (quand c'est possible ie que le get_url existe).

Donc on a une méthode "render_item" qui se retrouve appelée (et ajaxisée) quand le render repère qu'un item a été choisi.

Ca simplifie le HTML, rien à dire c'est nettement mieux ainsi.

Les tests sont un peu délicats à lire, parce que je "coupe" le mode ajax et je lance le rendu après avoir fait le modify_global_context (qui permet à toutes les cellules de connaitre le contexte des autres recherches, quand il y a une dépendance entre elles), bref je simule le travail de rendu du combo.

#9 - 26 février 2017 09:52 - Frédéric Péters

> Je continue cependant à faire en sorte que la cellule sache à la fois propose la recherche puis afficher un résultat sélectionné (quand c'est possible ie que le get_url existe).

Je continue à trouver ça intéressant à discuter; avec comme idée alternative de permettre ...?adult_id=[adult] dans l'URL paramétrée d'une cellule JSON (en étendant #15152 pour ne pas passer juste "user" mais tout "context").

#10 - 26 février 2017 19:00 - Thomas Noël

Voici donc une version simple, qui peut délèguer l'affichage du résultat à une cellule Json (suite à son amélioration #15142 + #15154 + #15152).

Pour cela, quand la page est appelée avec ?<slug_cellule_recherche>=foobar alors une variable slug_cellule_recherche est posée dans le contexte

général de la page, et est donc utilisable dans l'URL de la cellule JSON via [slug_cellule_recherche]

#11 - 26 février 2017 19:01 - Thomas Noël

- Fichier 0001-add-generic-search-cell-15085.patch ajouté

#12 - 27 février 2017 16:42 - Thomas Noël

- Fichier 0001-add-generic-search-cell-15085.patch ajouté

Version qui impose plus le format de la réponse, selon discussion jabber avec Frédéric (cf README et search-cell-results.html)

#13 - 27 février 2017 16:59 - Frédéric Péters

```
  combo_search_input_{{ cell.pk }}.on('keyup', function() {
```

Faire .on('keyup change', ..., pour gérer aussi le changement de contenu qui interviendrait autrement que par le clavier. (genre clic du milieu) En fait vu le setTimeout on ne peut sans doute pas mêler les deux événements dans la même fonction. Juste un .on('change', combo_search_update_... en plus, du coup, a priori.

À part ça, peut-être avoir un test qui ajoute cette cellule dans tests/manager.py ?

#14 - 27 février 2017 18:00 - Thomas Noël

- Fichier 0001-add-generic-search-cell-15085.patch ajouté

J'ai pas trouvé mieux que keyup.

Pour le test dans manager.py je sais pas si je t'ai bien compris, tu diras...

#15 - 27 février 2017 18:04 - Frédéric Péters

> J'ai pas trouvé mieux que keyup.

Pour moi il faut keyup **et** change.

> Pour le test dans manager.py je sais pas si je t'ai bien compris, tu diras...

Yep, c'est ça.

#16 - 27 février 2017 18:39 - Thomas Noël

Frédéric Péters a écrit :

> > J'ai pas trouvé mieux que keyup.
>
> Pour moi il faut keyup **et** change.

Mais j'arrive pas à comprendre, pour "change" (il ne fait vraiment rien par rapport au clic du milieu, par exemple)

#17 - 27 février 2017 21:42 - Frédéric Péters

"change" sera appelé à coup sûr si le contenu du champ change, en cas de modification au clavier, ça sera lors du focus out, mais il y a d'autres interactions imaginables que le clavier. Exemple : copicollage par clic du milieu, clic en-dehors de l'<input>, un événement "change" sera lancé, aucun événement keyup.

#18 - 27 février 2017 22:10 - Thomas Noël

- Fichier 0001-add-generic-search-cell-15085.patch ajouté

Me documentant sur l'affaire, voici une nouvelle version, avec ces deux modifications :

```
-  combo_search_input_{{ cell.pk }}.on('keyup', function() {
+  combo_search_input_{{ cell.pk }}.on('change paste keyup', function() {
     clearTimeout(combo_search_timeout_{{ cell.pk }});
     combo_search_timeout_{{ cell.pk }} = setTimeout(combo_search_update_{{ cell.pk }}, 300);
-    return false;
   });
```

…et sur mon navigateur tout marche très bien, même un clic-milieu (qui provoque un "paste").

#19 - 27 février 2017 22:15 - Frédéric Péters

Ce que j'exprimais mal plus haut, c'est que sur événement change, la recherche doit se faire immédiatement, pas après 300ms, que "change" et "keyup" ne peuvent pas partager la même fonction de callback.

#20 - 27 février 2017 22:24 - Frédéric Péters

Oh et dans le sujet des événements js, je verrais bien le <input> dans un <form>, avec aussi un <button>, et sur le <form> un .on('submit', function() { tuer le timeout; faire la recherche }). Pour les aveugles.

#21 - 27 février 2017 23:04 - Thomas Noël

- Fichier 0001-add-generic-search-cell-15085.patch ajouté

Voici donc le patch avec un search-cell.html que je crois contenir ce que tu proposes.

#22 - 27 février 2017 23:29 - Frédéric Péters

Yep, ça me semble bien ça. Je le teste en local demain.

#23 - 28 février 2017 17:06 - Thomas Noël

- Fichier 0001-add-generic-search-cell-15085.patch ajouté

Cette modification au niveau de modify_global_context:

```
-        if self.varname:
-            context[self.varname] = request.GET.get(self.varname) or ''
+        if self.varname and self.varname in request.GET:
+            context[self.varname] = request.GET.get(self.varname)
```

Cela évite de polluer le contexte avec des variables vides.

Par ricochet, l'utilisation de /get/[varname]/ dans les URLs d'une JsonCell va planter (UnknownTemplateVariabel). C'est heureux cela évitera à la cellule de tenter la requête /get// qui peut donner un tout autre résultat que celui espéré, et dans le meilleur des cas un 404, en tout cas des choses que c'est pas bien.

Autre modif que tu avais demandée Frédéric : si q_<slug> est présent dans la query_string, alors ça initialise la recherche avec ce q. (ajouts dans models.py et search-cell.html)

#24 - 28 février 2017 20:50 - Frédéric Péters

> Autre modif que tu avais demandée Frédéric : si q_<slug> est présent dans la query_string, alors ça initialise la recherche avec ce q. (ajouts dans models.py et search-cell.html)

Chouette, ça fait du coup un appel genre /ajax/search/1/?q_test=chat&q=chat; tu penses quoi en présence d'un slug de modifier name="q", request.GET.get('q'), etc. pour faire q → q_<slug> ? (si ça amène pas de complications, ça m'irait bien, sinon je vivrai sans)

Côté style, pour ne pas devoir surcharger côté publik-base-theme, ça me dirait bien d'avoir ça dans le markup :

```
--- a/combo/apps/search/templates/combo/search-cell-results.html
+++ b/combo/apps/search/templates/combo/search-cell-results.html
@@ -1,3 +1,4 @@
+<div class="links-list">
 <ul>
   {% for item in results.data %}
   <li><a href="{{ item.url }}">{{ item.text }}</a>{% if item.description %}
@@ -5,3 +6,4 @@
   {% endif %}</li>
   {% endfor %}
 </ul>
+</div>
```

#25 - 01 mars 2017 00:26 - Thomas Noël

- Fichier 0001-add-generic-search-cell-15085.patch ajouté

Frédéric Péters a écrit :

> Autre modif que tu avais demandée Frédéric : si q_<slug> est présent dans la query_string, alors ça initialise la recherche avec ce q. (ajouts dans models.py et search-cell.html)

> Chouette, ça fait du coup un appel genre /ajax/search/1/?q_test=chat&q=chat; tu penses quoi en présence d'un slug de modifier name="q", request.GET.get('q'), etc. pour faire q → q_<slug> ? (si ça amène pas de complications, ça m'irait bien, sinon je vivrai sans)

Comme moi j'ai déjà une belle surcharge de ?rsu_adult=...&rsu_spoose=...&rsu_child=... j'avais même pas remarqué :)

Bon c'est pas over joli mais voici ce que j'ai ajouté :

```
     def get_cell_extra_context(self, context):
         extra_context = super(SearchCell, self).get_cell_extra_context(context)
-        extra_context['initial_q'] = None
-        if self.varname and context.get('request'):
-            q_varname = 'q_%s' % self.varname
-            if q_varname in context['request'].GET:
-                extra_context['initial_q'] = context['request'].GET[q_varname]
+        # if there is a q_<slug> in query_string, send it to the template (to be
+        # used as an initial query) and remove it from query_string
+        initial_q = None
+        initial_query_string = None
+        if context.get('request'):
+            request_get = context['request'].GET.copy()
+            if self.varname and context.get('request'):
+                q_varname = 'q_%s' % self.varname
+                if q_varname in request_get:
+                    initial_q = request_get[q_varname]
+                    del request_get[q_varname]
+            initial_query_string = request_get.urlencode()
+        extra_context.update({
+            'initial_q': initial_q,
+            'initial_query_string': initial_query_string
+        })
         return extra_context
```

et utilisation de {{ initial_query_string }} dans le template :

```
 <form id="combo-search-form-{{ cell.pk }}" class="combo-search-form">
   <input type="text" name="q" autocomplete="off" id="combo-search-input-{{ cell.pk }}" class="combo-search-in
put"
-    data-autocomplete-json="{% url 'combo-search-ajax-results' cell_pk=cell.pk %}?{{ request.META.QUERY_STRIN
G }}" />
-    {# QUERY_STRING pass some context in ajax call #}
+    data-autocomplete-json="{% url 'combo-search-ajax-results' cell_pk=cell.pk %}{% if initial_query_string %
}?{{ initial_query_string }}{% endif %}" />{# initial_query_string pass some context to ajax call #}
   <button class="submit-button">{% trans "Search" %}</button>
 </form>
```

> Côté style, pour ne pas devoir surcharger côté publik-base-theme, ça me dirait bien d'avoir ça dans le markup :

Merci ; c'est fait aussi.

#26 - 03 mars 2017 10:46 - Frédéric Péters

Ack pour moi; et je verrai côté style gadjo.

#27 - 03 mars 2017 11:11 - Thomas Noël

- Statut changé de En cours à Résolu (à déployer)

```
commit 4c3cf5e198651a73d7d71313ccaa4dc05767be63
Author: Thomas NOEL <tnoel@entrouvert.com>
Date:   Wed Feb 22 16:15:37 2017 +0100

    add generic search cell (#15085)
```

#28 - 23 décembre 2018 15:21 - Frédéric Péters

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

Fichiers

0001-add-generic-search-cell-15085.patch	15,1 ko	22 février 2017	Thomas Noël
out-2.ogv	1,65 Mo	22 février 2017	Thomas Noël
0001-add-generic-search-cell-15085.patch	16,4 ko	23 février 2017	Thomas Noël
0001-add-generic-search-cell-15085.patch	22,6 ko	23 février 2017	Thomas Noël
0001-add-generic-search-cell-15085.patch	16,6 ko	26 février 2017	Thomas Noël
0001-add-generic-search-cell-15085.patch	17 ko	27 février 2017	Thomas Noël
0001-add-generic-search-cell-15085.patch	19,1 ko	27 février 2017	Thomas Noël
0001-add-generic-search-cell-15085.patch	19,1 ko	27 février 2017	Thomas Noël
0001-add-generic-search-cell-15085.patch	19,6 ko	27 février 2017	Thomas Noël
0001-add-generic-search-cell-15085.patch	20,4 ko	28 février 2017	Thomas Noël
0001-add-generic-search-cell-15085.patch	21 ko	28 février 2017	Thomas Noël