Gadjo - Development #5111

javascript non intrusif pour l'ouverture de formulaires en ajax

08 juillet 2014 09:29 - Frédéric Péters

Statut:	Fermé	**Début:**	08 juillet 2014
Priorité:	Normal	**Echéance:**	
Assigné à:	Frédéric Péters	**% réalisé:**	100%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:	0.2		
Patch proposed:	Oui	**Planning:**	

Description

Un peu comme ce que fait wcs/qommon/static/js/popup.js, plutôt que multiplier du code adhoc directement dans les templates.

Historique

#1 - 11 juillet 2014 13:15 - Frédéric Péters

- Fichier 0001-javascript-to-open-rel-popup-links-in-popups-5111.patch ajouté

- Fichier 0001-ui-add-popups-and-cancel-buttons-5111.patch ajouté

- Statut changé de Nouveau à En cours

- Assigné à mis à Frédéric Péters

- Patch proposed changé de Non à Oui

Voilà le premier patch pour gadjo, puis le second pour son utilisation dans passerelle.

La structure attendue est la suivante, le titre de la fenêtre sera pris du "#appbar h2", le contenu de la boîte sera le "#content form", les boutons ".buttons a, .buttons button" seront cachés et recréés en tant que vrai boutons de dialogue. Comme dans w.c.s., il n'y a pas de magie au submit pour garder le résultat à l'intérieur de la fenêtre.

```
<div id="content">
  <div id="appbar"><h2>Action</h2></div>
  <form>
    ...
    <div class="buttons">
       <button>Action!</button>
       <a class="cancel" href="...">Annuler</a>
    </div>
  </form>
</div>
```

#2 - 11 juillet 2014 14:16 - Frédéric Péters

La partie passerelle, je vais plutôt l'intégrer dans le patch qui est en cours dans #5124.

#3 - 17 juillet 2014 12:41 - Benjamin Dauvergne

J'ai commencé une implémentation concurrente dans le manager d'a2, mon souci a été de pouvoir implémenter des formulaires un peu complexes en utilisant une CBV classique et un mixin "AjaxFormViewMixin" au lieu de simplement faire un post sur la page du formulaire. La page du formulaire renvoie un JSON avec le contenu du formulaire ou le contenu de la redirection s'il y en a une, ça permet d'avoir une soumission en ajax, avec rafraîchissement en cas d'erreur et la redirection classique sur une réussite ou un rafraîchissement AJAX si l'URL dans le champs location est celle de la page en cours (le souci là c'est qu'en AJAX il n'est pas possible d'intercepter les redirections, sinon ton code suffirait pour faire tout ça).

Je pense qu'on pourrait se faire rejoindre les deux idées:

- utiliser un displayPopup() modifié qui accepte aussi du JSON en retour et avoir des conventions au niveau du template du formulaire pour pouvoir construire une vrai belle boite de dialogue
- utiliser ce mixin pour pouvoir implémenter les formulaires proprement et avoir quand même la gestion des erreurs sans sortir du popup.

#4 - 17 juillet 2014 12:52 - Frédéric Péters

Tu peux retourner un entête HTTP particulier précisant que le formulaire prend en charge le "post/retour json" ? Dans le chargement initial, cet entête serait alors lu, et le clic sur un bouton serait dès lors traité différemment.

Concertant l'html, le code du patch vise la balise form, on pourrait convenir de plutôt viser une classe. Pour les boutons, demander un , ça me semble rester le truc minimaliste qui convient (avec traitement particulier du class="cancel", pour juste fermer la popup sans

action).

#5 - 17 juillet 2014 13:23 - Frédéric Péters

> […] entête HTTP particulier […]

Ou autre chose, genre un attribut data-whatever posé sur le <form>.

Et si ça a une utilité, le js peut aussi ajouter un <input type="hidden" name="gadjo-ajax" value="1"/> dans le formulaire, pour aider à différencier le traitement côté serveur.

#6 - 17 juillet 2014 15:27 - Benjamin Dauvergne

Frédéric Péters a écrit :

> Tu peux retourner un entête HTTP particulier précisant que le formulaire prend en charge le "post/retour json" ? Dans le chargement initial, cet entête serait alors lu, et le clic sur un bouton serait dès lors traité différemment.

Là je fais la différence simplement en faisant un if ("content" in data) car $.ajax traite automatiquement le JSON si le mime-type est bien renseigné. Ce que tu me proposes c'est de traiter le GET normalement en renvoyant du HTML mais sur le POST de renvoyer du JSON ?

#7 - 17 juillet 2014 15:42 - Frédéric Péters

Je ne suis vraiment pas sûr de comprendre ce que tu fais; ça intervient où ce "if ("content" in data)", ça fait référence à quoi ?

Pour moi, les moments importants :

1. GET... et on en extrait une partie, en visant une balise form, ou une classe particulière
2. clic sur un bouton, et là on veut savoir si le serveur a en place un traitement particulier pour l'"ajax".
3. dans le cas où il y a un traitement particulier, on récupère alors le json et on en fait le nécessaire (provoquer une redirection ou remplacer le contenu de la boite de dialogue)

#8 - 18 juillet 2014 10:36 - Thomas Noël

Fred, tu peux déjà committer les patches proposés ici.

On discutera d'une évolution plus tard ; même si je trouve que là, c'est simple, et moi j'aime les choses simples (et on restera le seul site du monde où la désactivation du js marche encore, youpi).

#9 - 18 juillet 2014 10:38 - Frédéric Péters

- Patch proposed changé de Oui à Non

C'est poussé ainsi. Je retire le "patch proposed", pour la discussion sur les évolutions possibles.

```
commit 78e215494d4a47bdb301f4244dc99e52801f2ee4
Author: Frédéric Péters <fpeters@entrouvert.com>
Date:   Fri Jul 11 13:10:51 2014 +0200

    javascript to open rel="popup" links in popups (#5111)
```

#10 - 18 juillet 2014 16:04 - Benjamin Dauvergne

Frédéric Péters a écrit :

> Je ne suis vraiment pas sûr de comprendre ce que tu fais; ça intervient où ce "if ("content" in data)", ça fait référence à quoi ?

C'est pas grave, je fais mon code dans authentic2 on pourra voir après ce qu'on reprend dans un ticket évolution comme tu le dis.

#11 - 18 juillet 2014 16:18 - Frédéric Péters

Je trouverais sympa d'arriver à avoir dans authentic l'utilisation de gadjo, plutôt que d'y avoir le développement de nouveau code qui serait ensuite à intégrer dans gadjo.

#12 - 18 juillet 2014 17:14 - Benjamin Dauvergne

Ces accusations font mal à mon petit cœur.

#13 - 18 juillet 2014 17:33 - Frédéric Péters

Oh, désolé, j'avais cru comprendre que tu allais faire **ton code** dans authentic et qu'on pourrait avoir **après** ce qu'on reprendrait dans gadjo.

#14 - 18 juillet 2014 17:41 - Benjamin Dauvergne

- Fichier 0001-gadjo.js-add-support-for-ajax-submission-of-form.patch ajouté

Pour l'instant je ne dépend pas de gadjo je ne sais pas encore si c'est une bonne idée pour a2, mais oui je contribue le js en question à gadjo, le voilà.

Ça utilise jQuery Form plugin[1] uniquement s'il est disponible pour gérer la soumission du formulaire en ajax. Au niveau de de la vue il faut un mixin comme celui-ci:

```
class AjaxFormMixin(object):
    def form_valid(self, form):
        if hasattr(form, 'save'):
            self.form_result = form.save()
        return super(AjaxFormViewMixin, self).form_valid(form)
```

```
    def dispatch(self, request, *args, **kwargs):
        response = super(AjaxFormViewMixin, self).dispatch(request, *args, **kwargs)
        if not request.is_ajax():
            return response
        data = {}
        if 'Location' in response:
            data['location'] = response['Location']
        if hasattr(response, 'render'):
            response.render()
            data['content'] = response.content
        return HttpResponse(json.dumps(data), content_type='application/json')
```

Mais si la vue ne le gère pas ou jQuery form plugin n'est pas chargé on revient sur le comportement classique.

[1] http://malsup.com/jquery/form/

#15 - 18 juillet 2014 17:41 - Benjamin Dauvergne

- Patch proposed changé de Non à Oui

#16 - 18 juillet 2014 17:46 - Benjamin Dauvergne

La surcharge de form_valid() n'a pas de rapport avec la fonctionnalité évoquée ici, c'est un bout de code pour le manager de Django j'ai oublié de l'écarter.

#17 - 18 juillet 2014 18:01 - Frédéric Péters

Le window.refresh_page il arrive d'où ? (je n'en trouve pas mention dans http://malsup.github.io/jquery.form.js)

Comme "#content form" est utilisé plusieurs fois, on pourrait poser ce sélecteur dans une constante, je pense à ça parce que j'écrivais « le code du patch vise la balise form, on pourrait convenir de plutôt viser une classe », pas de commentaire à ce sujet ?

S'il y a un entête Location, appeler le render(), c'est parfois utile ?

#18 - 18 juillet 2014 18:42 - Benjamin Dauvergne

- Fichier 0001-gadjo.js-add-support-for-ajax-submission-of-form.patch ajouté

J'ai oublié une dépendance sur purl[1] aussi pour parser les URLs de redirection, je l'ajoute au test.

Frédéric Péters a écrit :

> Le window.refresh_page il arrive d'où ? (je n'en trouve pas mention dans http://malsup.github.io/jquery.form.js)

C'est un truc que j'utilise dans le manager d'authentic pour recharger la liste des rôles après la création d'un nouveau rôle. Je pourrai le remplacer par un événement lancer sur l'ancre du popup, je vais adapter mon code.

> Comme "#content form" est utilisé plusieurs fois, on pourrait poser ce sélecteur dans une constante, je pense à ça parce que j'écrivais « le code du patch vise la balise form, on pourrait convenir de plutôt viser une classe », pas de commentaire à ce sujet ?

Moi je viserais simplement "form" par défaut et sinon un sélecteur qu'on pourra poser sur l'ancre comme ça ... mais j'ai tendance à trop généraliser. J'aime bien fonctionner par convention mais il ne faut pas qu'elles soient trop compliquées à découvrir.

On est en train d'oublier l'autre sélecteur en dur: '#appbar h2'.

> S'il y a un entête Location, appeler le render(), c'est parfois utile ?

Pour nous je ne vois pas, mais dans l'absolu rien n'empêche de renvoyer un contenu en plus d'un entête Location; ce qu'on devrait en faire dans le cas d'un chargement ajax, je n'en sais rien.

J'ai par contre un doute sur ce bout de code:

```
    function ajaxform_submit (data, status, xhr, form) {
            ....
            var new_form = $(html).find('#content form');
            $(form).html(new_form.html());
            ....
    }
```

Dans le cas où le formulaire utilise des champs complexes utilisant du JS (select2 ?) ça ne marchera pas je pense, il faudrait remplacer les noeuds fils de $(form) par ceux de new_form sans les recréer mais $.fn.replaceWith ne fait pas ça. J'ai remplacé par un $(form).empty().append($(html).find(selector)).

[1] https://github.com/allmarkedup/purl

#19 - 18 juillet 2014 19:05 - Frédéric Péters

> Moi je viserais simplement "form" par défaut [...]

Restons donc à viser le <form>; le #appbar h2 je n'en parlais pas dans la mesure où il n'était pas dupliqué, et où dans gadjo, le titre, c'est là qu'il doit aller.

> J'ai oublié une dépendance sur purl [...]

La dépendance sur purl, elle n'est pas justement uniquement pour le refreshpage() ? Du coup si tu adaptes dans authentic pour "événement lancé sur l'ancre du popup", toute cette partie pourrait être réduite à juste le window.location = data.location ?

Et ça pourrait aussi être l'approche pour les widgets js, en faisant un $(xxx).trigger('dialog-done', [dialog]), qui laisserait libre court à du code ailleurs pour faire son boulot. (d'ailleurs ça me fait pense que je préfère cette construction simple, plutôt que de passer $.Event('popup-success')).

Aussi on peut prendre ce qu'il y aurait dans des balises <script> du formulaire, et en faire un eval() après le $(form).html(new_form.html());.

> Pour nous je ne vois pas, mais dans l'absolu rien n'empêche de renvoyer un contenu en plus d'un entête Location; ce qu'on devrait en faire dans le cas d'un chargement ajax, je n'en sais rien.

C'est bien parce qu'on ne l'utilise pas que je me disais qu'on pouvait zapper ça.

#20 - 18 juillet 2014 19:09 - Frédéric Péters

> $(xxx).trigger('dialog-done', [dialog])

Sur les événements tant que j'y pense, ça pourrait être propre de les mettre dans un namespace, "gadjo:dialog-done".

#21 - 18 juillet 2014 19:49 - Benjamin Dauvergne

Frédéric Péters a écrit :

> > J'ai oublié une dépendance sur purl [...]
>
> La dépendance sur purl, elle n'est pas justement uniquement pour le refreshpage() ? Du coup si tu adaptes dans authentic pour "événement lancé sur l'ancre du popup", toute cette partie pourrait être réduite à juste le window.location = data.location ?

Purl permet d'appeler window.location.reload() quand window.location et data.location sont identique au hashtag près, ça me parait un comportement assez utile. Là sur le manager d'authentic ce comportement est constant, je crée un objet, je met à jour le listing.

Et donc le souci ici est de pouvoir empêcher ce comportement par défaut si finalement on sait gérer le refresh via ajax via un event.preventDefault() et de détecter cet appel au niveau du code de displayPopup, ce n'est possible que si on construit l'objet Event nous même. Pour le reste on peut très

bien appeler l'événement dialog-done peu m'en chaut.

> Et ça pourrait aussi être l'approche pour les widgets js, en faisant un $(xxx).trigger('dialog-done', [dialog]), qui laisserait libre court à du code ailleurs pour faire son boulot. (d'ailleurs ça me fait pense que je préfère cette construction simple, plutôt que de passer $.Event('popup-success')).

Je ne sais pas de quoi tu parles quand tu dis ...l'approche pour les widgets js...

> Aussi on peut prendre ce qu'il y aurait dans des balises <script> du formulaire, et en faire un eval() après le $(form).html(new_form.html());.

$.fn.html s'en charge déjà.

#22 - 18 juillet 2014 20:18 - Frédéric Péters

> La dépendance sur purl, elle n'est pas justement uniquement pour le refreshpage() ? Du coup si tu adaptes dans authentic pour "événement lancé sur l'ancre du popup", toute cette partie pourrait être réduite à juste le window.location = data.location ?

> Purl permet d'appeler window.location.reload() quand window.location et data.location sont identique au hashtag près, ça me parait un comportement assez utile. Là sur le manager d'authentic ce comportement est constant, je crée un objet, je met à jour le listing.

Je reconnais encore me débrouiller assez mal à imaginer le code dont tu parles. Il y a quoi de particulier dans ce que tu fais qui empêche la simple comparaison entre les deux location ? (ou même pas, et simplement faire window.location = data.location; window.location.refresh(), et dans tous les cas la page sera rechargée).

Ou quel est le truc évident à côté duquel je passe ?

> Et donc le souci ici est de pouvoir empêcher ce comportement par défaut si finalement on sait gérer le refresh via ajax via un event.preventDefault() et de détecter cet appel au niveau du code de displayPopup, ce n'est possible que si on construit l'objet Event nous même.

Je m'essaie cette fois à deviner; le bout qu'il me manque c'est qu'en fait tu as du code, ailleurs, pour attraper le 'popup-success' et réagir en ayant une action différente, non pas recharger toute la page, mais charger uniquement le contenu de la balise reprenant un tableau ?

> Pour le reste on peut très bien appeler l'événement dialog-done peu m'en chaut.

Pour dialog-done je ne parlais pas de renommer cet événement, mais un autre sujet (paragraphe suivant).

> > Et ça pourrait aussi être l'approche pour les widgets js, en faisant un $(xxx).trigger('dialog-done', [dialog]), qui laisserait libre court à du code ailleurs pour faire son boulot. (d'ailleurs ça me fait pense que je préfère cette construction simple, plutôt que de passer $.Event('popup-success')).
>
> Je ne sais pas de quoi tu parles quand tu dis ...l'approche pour les widgets js...

Ça faisait référence au problème que je comprenais dans cette phrase, « Dans le cas où le formulaire utilise des champs complexes utilisant du JS (select2 ?) ça ne marchera pas je pense ».

L'événement "dialog-done" viendrait une fois la boîte de dialogue remplie, dans son chargement initial ou quand son contenu a été remplacé dans ajaxform_submit(). Il permettrait, dans du code ailleurs, d'avoir une function qui se chargerait des widgets évolués, genre :

```
$('body').on('dialog-done', function(event, dialog) { $(dialog).find('select.select2').select2(); }).
```

(une modif par rapport au .trigger() que je donnais, c'est de le faire sur le <body>).

#23 - 19 juillet 2014 00:47 - Benjamin Dauvergne

- Fichier 0001-gadjo.js-add-support-for-ajax-submission-of-form.patch ajouté

Frédéric Péters a écrit :

> Je reconnais encore me débrouiller assez mal à imaginer le code dont tu parles. Il y a quoi de particulier dans ce que tu fais qui empêche la simple comparaison entre les deux location ? (ou même pas, et simplement faire window.location = data.location; window.location.refresh(), et dans tous les cas la page sera rechargée).

Oui, tu as raison c'est suffisant, je voulais aussi gérer le cas ou les URLs diffèrent seulement par le fragment, mais je n'ai pas besoin de purl,

window.location.href.split('#')[0] suffit.

> Et donc le souci ici est de pouvoir empêcher ce comportement par défaut si finalement on sait gérer le refresh via ajax via un event.preventDefault() et de détecter cet appel au niveau du code de displayPopup, ce n'est possible que si on construit l'objet Event nous même.

> Je m'essaie cette fois à deviner; le bout qu'il me manque c'est qu'en fait tu as du code, ailleurs, pour attraper le 'popup-success' et réagir en ayant une action différente, non pas recharger toute la page, mais charger uniquement le contenu de la balise reprenant un tableau ?

Tout à fait.

> Pour le reste on peut très bien appeler l'événement dialog-done peu m'en chaut.

> Pour dialog-done je ne parlais pas de renommer cet événement, mais un autre sujet (paragraphe suivant).
>
> > Et ça pourrait aussi être l'approche pour les widgets js, en faisant un $(xxx).trigger('dialog-done', [dialog]), qui laisserait libre court à du code ailleurs pour faire son boulot. (d'ailleurs ça me fait pense que je préfère cette construction simple, plutôt que de passer $.Event('popup-success')).
>
> Je ne sais pas de quoi tu parles quand tu dis ...l'approche pour les widgets js...

> Ça faisait référence au problème que je comprenais dans cette phrase, « Dans le cas où le formulaire utilise des champs complexes utilisant du JS (select2 ?) ça ne marchera pas je pense ».
>
> L'événement "dialog-done" viendrait une fois la boîte de dialogue remplie, dans son chargement initial ou quand son contenu a été remplacé dans ajaxform_submit(). Il permettrait, dans du code ailleurs, d'avoir une function qui se chargerait des widgets évolués, genre :
>
> [...]

On a pas besoin de toute ça si on charge bien le contenu complet de la page avec $(html).

> (une modif par rapport au .trigger() que je donnais, c'est de le faire sur le <body>).

Je pense que tous les événements "bubble" donc pas la peine de lancer le trigger sur le <body> (testé à l'instant c'est le cas).

J'attache la dernière version avec la dépendance sur purl supprimée. Par contre j'ai ajouté la possibilité de surcharger le sélecteur pour le titre avoir #appbar h2 toujours en dur ça me chagrine.

#24 - 19 juillet 2014 09:43 - Benjamin Dauvergne

- Fichier 0001-gadjo.js-add-support-for-ajax-submission-of-form.patch ajouté

Encore un changement pour mieux supporter le rechargement des pages, déplacer le handler de click des a[rel=popup] sur document.

```
diff --git a/gadjo/static/js/gadjo.js b/gadjo/static/js/gadjo.js
index b6706bd..c34462d 100644
--- a/gadjo/static/js/gadjo.js
+++ b/gadjo/static/js/gadjo.js
@@ -79,5 +79,5 @@ function displayPopup(event)
 }

 $(function() {
-    $('a[rel=popup]').click(displayPopup);
+    $(document).on('click.gadjo', 'a[rel=popup]', displayPopup);
 });
```

#25 - 19 juillet 2014 09:52 - Frédéric Péters

> On a pas besoin de toute ça si on charge bien le contenu complet de la page avec $(html).

Mais le code qui s'occupe de transformer les widgets, il n'est peut-être pas dans la page mais dans un onload() dans un fichier js; mais ne faisons rien là-dessus pour le moment, si le besoin se révèle à un moment, il sera temps de le placer. (c'est que dans le js de plone ça m'aurait été utile quelques fois qu'ils aient prévus ce genre de mécanisme)

> J'attache la dernière version avec la dépendance sur purl supprimée. Par contre j'ai ajouté la possibilité de surcharger le sélecteur pour le titre

> avoir #appbar h2 toujours en dur ça me chagrine.

Ok, mais l'autre sélecteur, pour le contenu de la boîte, il peut selon moi arriver que ce soit pas un <form>, ce serait bien alors qu'on ait d'abord content = $html.find(selector); et puis déterminer la variable form selon que content point sur un <form> ou pas.

#26 - 19 juillet 2014 11:52 - Benjamin Dauvergne

- Fichier 0001-gadjo.js-add-support-for-ajax-submission-of-form.patch ajouté

Frédéric Péters a écrit :

> > J'attache la dernière version avec la dépendance sur purl supprimée. Par contre j'ai ajouté la possibilité de surcharger le sélecteur pour le titre avoir #appbar h2 toujours en dur ça me chagrine.
>
> Ok, mais l'autre sélecteur, pour le contenu de la boîte, il peut selon moi arriver que ce soit pas un <form>, ce serait bien alors qu'on ait d'abord content = $html.find(selector); et puis déterminer la variable form selon que content point sur un <form> ou pas.

Voili voilou.

```
diff --git a/gadjo/static/js/gadjo.js b/gadjo/static/js/gadjo.js
index c34462d..17fc84e 100644
--- a/gadjo/static/js/gadjo.js
+++ b/gadjo/static/js/gadjo.js
@@ -34,7 +34,12 @@ function displayPopup(event)
                 var html = html;
             }
             var $html = $(html);
-            var $form = $html.find(selector);
+            var $content = $html.find(selector)
+            if ($content.is('form') {
+              var $form = $content;
+            } else {
+              var $form = $content.find('form');
+            }
             var title = $html.find(title_selector).text();
             var buttons = Array();
```

#27 - 21 juillet 2014 11:04 - Benjamin Dauvergne

- Fichier 0001-gadjo.js-add-support-for-ajax-submission-of-form.patch ajouté

Avec une typo corrigée:

```
diff --git a/gadjo/static/js/gadjo.js b/gadjo/static/js/gadjo.js
index 17fc84e..357fd29 100644
--- a/gadjo/static/js/gadjo.js
+++ b/gadjo/static/js/gadjo.js
@@ -34,8 +34,8 @@ function displayPopup(event)
                 var html = html;
             }
             var $html = $(html);
-            var $content = $html.find(selector)
-            if ($content.is('form') {
+            var $content = $html.find(selector);
+            if ($content.is('form')) {
               var $form = $content;
             } else {
               var $form = $content.find('form');
```

#28 - 21 juillet 2014 13:56 - Frédéric Péters

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

J'ai poussé le commit, puis un autre qui ajoute des commentaires.

```
commit de23f42b790f349907ba75eba8ab5b592ba27a25
Author: Frédéric Péters <fpeters@entrouvert.com>
Date:   Mon Jul 21 13:54:20 2014 +0200

    js: add comments (#5111)
```

```
commit 58763ed2250bda6bb99c926008b06a484cbab1f4
Author: Benjamin Dauvergne <bdauvergne@entrouvert.com>
Date:   Fri Jul 18 17:35:45 2014 +0200

    js: add support for ajax submission of form (#5111)
```

#29 - 21 juillet 2014 14:58 - Benjamin Dauvergne

- Fichier 0001-js-add-new-event-when-dialog-has-finished-loading.patch ajouté

Je propose un évènement de plus 'gadjo:dialog-loaded' une fois que le contenu du dialogue est chargé ou rechargé.

Je l'utilise ainsi dans a2:

```
$(function () {
    $(document).on('gadjo:dialog-loaded', function (e, form) {
      $('.messages', form).delay(3000*(1+$('.messages li', form).length)).fadeOut('slow');
    });
})
```

C'est une copie du code que tu as écrit pour faire disparaître progressivement les messages venant de django.contrib.messages dans Docbow, le formulaire d'édition utilisateur[1] a une commande 'Réinitialiser le mot de passe' et le retour se fait via un tel message.

[1] Voir le mockup http://perso.entrouvert.org/~fred/portail-admin/mockups/authentic-utilisateurs-bis.html

#30 - 21 juillet 2014 15:01 - Benjamin Dauvergne

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

#31 - 21 juillet 2014 15:37 - Frédéric Péters

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

Très bien, poussé.

```
commit b2fff62ed941b7d292fa1b70d10f5fa64955f5d9
Author: Benjamin Dauvergne <bdauvergne@entrouvert.com>
Date:   Mon Jul 21 14:24:26 2014 +0200

    js: add new event when dialog has finished loading (#5111)
```

#32 - 28 juillet 2014 13:38 - Thomas Noël

- Version cible mis à 0.2

#33 - 28 juillet 2014 14:15 - Thomas Noël

- % réalisé changé de 0 à 100

#34 - 28 juillet 2014 14:16 - Thomas Noël

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

#35 - 28 juillet 2014 14:16 - Thomas Noël

- Statut changé de Solution déployée à Fermé

Fichiers

0001-javascript-to-open-rel-popup-links-in-popups-5111.patch	2,72 ko	11 juillet 2014	Frédéric Péters
0001-ui-add-popups-and-cancel-buttons-5111.patch	9,03 ko	11 juillet 2014	Frédéric Péters
0001-gadjo.js-add-support-for-ajax-submission-of-form.patch	3,25 ko	18 juillet 2014	Benjamin Dauvergne
0001-gadjo.js-add-support-for-ajax-submission-of-form.patch	3,39 ko	18 juillet 2014	Benjamin Dauvergne
0001-gadjo.js-add-support-for-ajax-submission-of-form.patch	4,22 ko	18 juillet 2014	Benjamin Dauvergne
0001-gadjo.js-add-support-for-ajax-submission-of-form.patch	4,4 ko	19 juillet 2014	Benjamin Dauvergne
0001-gadjo.js-add-support-for-ajax-submission-of-form.patch	4,56 ko	19 juillet 2014	Benjamin Dauvergne
0001-gadjo.js-add-support-for-ajax-submission-of-form.patch	4,56 ko	21 juillet 2014	Benjamin Dauvergne
0001-js-add-new-event-when-dialog-has-finished-loading.patch	1,56 ko	21 juillet 2014	Benjamin Dauvergne