

**Cas et solutions BPM**

**17 juillet 2015**

**TABLE DES MATIÈRES**

[Cas et solutions 3](#_Toc424889142)

[1. Validation dans un champ 3](#_Toc424889143)

[2. Validation dans un message html 3](#_Toc424889144)

[3. Validation différente selon bouton appuyé 3](#_Toc424889145)

[4. Télécharger document en voûte 3](#_Toc424889146)

[5. Afficher un document de la voûte 4](#_Toc424889147)

[6. Afficher un document dans un tableau 4](#_Toc424889148)

[7. Utiliser la cache 4](#_Toc424889149)

[8. Validation à 2 niveau 4](#_Toc424889150)

[9. Appel SQL 4](#_Toc424889151)

[10. Appel SQL Multiple 5](#_Toc424889152)

[11. Dynamique de champ simple 5](#_Toc424889153)

[12. Dynamique de champ simple 5](#_Toc424889154)

[13. Appel Under Cover Agent 5](#_Toc424889155)

[14. Lire Fichier Texte Dans la voûte 5](#_Toc424889156)

# Cas et solutions

Voici une liste de cas et des solutions à chacun de ceux-ci. Les exemples se retrouvent dans le « ProcessApp » « Laboratoire Transfert Connaissance ». Les exemple porte le même nom que chaque sujet ici-bas.

## Validation dans un champ

Dans ce cas, une validation rend la valeur de plusieurs champs invalide avec un message d’erreur à côté du champ. Un exemple de validation pour ces cas est inclus :

* Chaîne de caractères : Champ requis
* Numérique : Champ requis et ne pouvant accepté que des caractères numérique.
* Date : Champ requis et ne pouvant être antérieur à la date du jour.
* Select : Champ requis.
* Tableau : Cellules requises

## Validation dans un message html

Dans ce cas, une validation rend la valeur de plusieurs champs invalide avec un message d’erreur html où le programmeur veut en utilisant le coachview « Dynamic\_Html ». Il est à noter que le message html peut avoir différents icônes. Pour modifier un icône, il faut ajouter une class CSS au coachview « Dynamic\_Html » (classe possible : message-error, message-success et message.warning). Un exemple de validation pour ces cas est inclus :

* Chaîne de caractères : Champ requis
* Numérique : Champ requis et ne pouvant accepté que des caractères numérique.
* Date : Champ requis et ne pouvant être antérieur à la date du jour.
* Select : Champ requis.
* Tableau : Cellules requises

## Validation différente selon bouton appuyé

Dans ce cas, une validation différente se fait selon le bouton appuyé par l’usagé. Pour réussir ceci, il faut utiliser le coachView « ButtonClickListener ». Voir documentation « Custom coach Views.docx » pour plus de détail.

## Télécharger document en voûte

Cet exemple permet de télécharger un document dans la voûte documentaire. Dans ce cas, il faut utiliser le coachView « FileUploader ». Il est très important de définir les paramètres avant l’ouverture du coach. Pour plus de détail, voir la documentation « Custom coach Views.docx ».

## Afficher un document de la voûte

Dans cet exemple, on demande à l’usager de saisir un « id » de document et on affiche celui-ci dans le viewer du BPM.

## Afficher un document dans un tableau

Cet exemple permet d’obtenir une liste des documents liés à un client saisi et l’afficher dans un écran. À l’aide du « coachView » « ImageForDocumentSelect » on ouvre une nouvelle fenêtre web pour afficher le document sélectionné. Pour plus de detail sur l’utilisation de celui-ci, voir la documentation « Custom coach Views.docx ». Afin de faire fonctionner le « ImageForDocumentSelect » il a été nécessaire de créer le service « Ajax » « Obtenir Document ». Celui-ci permet, à partir du « id » du document, de retourner le lien hypertext vers le document.

## Utiliser la cache

Dans cet exemple, on utilise la « Cache ». La « Cache » permet de mettre des données en mémoire qui seront accessibles par toutes les sessions. Même si cet exemple ne représente pas vraiment un cas typique, il fait appel à toutes les fonctionnalités offertes. On met en « Cache » une liste et on l’extrait ensuite pour l’afficher à l’écran. Voici une brève description des fonctionnalités pouvant être utilisées pour interagir avec la « Cache »

1. Cache Accessor PUT IS : Ce service permet de mettre en « Cache » un objet. Il faut lui spécifier le nom de la « Cache », la clé utilisé pour l’extraire plus tard ainsi que l’objet à insérer (Exemple : une liste de valeur).
2. Cache Accessor EXISTS IS : Ce service permet de vérifier si un objet est présent dans la « Cache » indexé avec une clé donnée.
3. Cache Accessor GET IS : Ce service permet d’obtenir un objet présent dans la « Cache » sous une clé donnée.
4. Cache Accessor CLEAR IS : Ce service permet de vider complètement la « Cache ». Malheureusement, il ne semble pas être possible de supprimer une iteration spécifique de la Cache… snif

## Validation à 2 niveau

Dans cet exemple, on utilise l’objet « Validation » pour gérer deux niveau de validation. La validation s’adressant à des champs spécifique et la validation dite « Business » s’adressant à plusieurs champs en même temps. Ce deuxième type de validation pourrait aussi être utilisé si la validation demande une vérification dans la base de données. Dans le cas présent, on empêche la saisi d’un intervalle de date dans l’année courante si le nom saisi n’est pas « BIDON ». Pour plus de détail sur l’utilisation de l’objet « Validation » veuillez vous référer au document « Custom coach Views.docx ».

## Appel SQL

Dans cet exemple, on utilise les services « Get SQL Sequence With Error Handling » et « SQL Execute Statement With Error Handling » pour montrer comment interragir avec la base de données. Les opération SQL standard tel que « Insert », « Select », « Update » et « Delete » sont utilisées pour montrer le fonctionnement.

## Appel SQL Multiple

Dans cet exemple, on utilise le service « SQL Execute Multiple Statement With Error Handling » pour démontrer la notion de requête multiple dans une même transaction SQL. Il est à noter que lorsqu’on utilise l’exécution multiple SQL il faut utiliser un « datasource » de type « XA » afin que la gestion la transaction et la gestion d’erreur fonctionne correctement.

## Dynamique de champ simple

Dans cette exemple, on démontre comment une dynamique de champ peut être effectué. Afin d’accomplir ceci, nous utilisons le coachview « Data Change Boundary Trigger ». Celui-ci permet de sortir du coach si une valeur spécifique change. Le code de la dynamique codé dans diagramme est ensuite exécuté sur le serveur pour ensuite renvoyé les changements à l’écran. Il est à noter que même si les applications UN1C utilisent presqu'exlusivement cette solution, celle-ci n’est pas très performance et fait plus travailler le serveur. Dans un monde idéal, on ne devrait utiliser cette solution que si la dynamique demande un appel à la base de données ou un service web.

## Dynamique de champ simple

Dans cette exemple, on démontre comment une dynamique de champ peut être effectué du côté client. Afin d’accomplir ceci, il faut créer un « coachview » contenans tous les contrôle de l’écran. On y attache toutes les variable nécessaire au fonctionnement de l’écran. La dynamique est codé ensuite directement dans le dans les « Event Handlers » du « coachview ». Cette solution est beaucoup plus performante et simplifie le diagramme BPM. Dommage que j’aie découvert les possibilités de ce type de dynamique qu’à la fin de VISA.

## Appel Under Cover Agent

Dans cet exemple, on démontre comment utiliser les « Under Cover Agent » (UCA pour les intimes lol). Deux cas, qui sont les plus courants, y sont exposé. Lancer un processus et relancer un processus en attente.

## Lire Fichier Texte Dans la voûte

Dans cet exemple, on va extraire un document texte de la voûte et le transformer en une liste de « String » où chaque « String » est une ligne du fichier.