

Paradox l'Euro et l'an 2000...

Tient donc, un sujet à la mode...

Comment donc Paradox va-t-il se comporter avec l'Euro et lors du passage à l'an 2000 ?

Un article qui vient en complément de ceux que vous avez sans aucun doute récemment lu dans les colonnes de Point DBF, plus exactement dans le numéro du mois de juin, où il était question des rustines de Microsoft...

Par Frédéric BROUARD

1 La grande peur de l'an 2000 et Paradox

Inutile d'avoir peur. Les dates stockées en interne dans les tables Paradox peuvent vous emmener jusqu'aux confins de l'an 9999 !

Vous avez donc de la marge pour prendre vos rendez-vous. Avant cette date, prière de vous munir d'un congélateur spécialement aménagé afin de vous conserver le mieux possible en attendant cet hypothétique rencontre...

Le seul problème, c'est que par défaut, si vous tapez l'année sur 2 chiffres seulement, Paradox y rajoute les fameux « 19 » devant, et vous vous retrouvez un millénaire en arrière (notez cependant que je commet une grave erreur, car le troisième millénaire ne commencera qu'à zéro heure le premier janvier 2001, le christ étant né l'année 1 et non l'année zéro...).

Afin de compenser ce comportement par défaut de Paradox, vous pouvez implanter la méthode suivante dans votre code, sur l'événement CanDepart :

```
var
    d    date
    j    smallint
    m    smallint
    a    smallint
endvar
; aucune valeur saisie : on rend la main
if self.isBlank()
then
    return
endif
; on tente de faire rentrer la saisie dans une variable de type date
TRY
    d = self.value
ONFAIL      ; en cas d'erreur, on purge le message d'erreur et on affiche
un message plus compréhensible
    errorClear()
    msgStop("ERREUR","Date invalide")
ENDTRY
; teste si l'année rentrée est sur deux position et si l'année de la date
; du jour est supérieur ou égale à 2000
```

```
if d.year() <100 and year(today())>=2000
then
; oui nous sommes en 2000 passé : on ajoute 2000 à l'année saisie
  j = d.day()
  m = d.month()
  y = d.year() + 2000
  d = date(m,j,a)
  dodefault
endif
```

Voir la fiche TESTDATE.FSL

En fait, dès que vous avez franchi le cap de l'an 2000, si vous saisissez une année sur deux positions, ce code y rajoute 2000...

2 Paradox et l'Euro

Deux écoles s'affrontent dans le domaine : les partisans du « y'a qu'à convertir toutes vos données monétaires au moment voulu... »

Cette technique peut s'avérer judicieuse dans le cas d'applications déjà développées.

Mais dans le cas de développement en cours, il parait plus attractif de dédoubler les champs monétaires de toutes les tables de l'application. Ainsi votre utilisateur aura la possibilité de choisir à tout moment d'afficher en Francs et en Euro voire même les deux à la fois, suivant la demande du client...

2.1 première école : le YAKA

Il y a cependant une condition : avoir fait reposer tous vos champs date sur le format de Windows par défaut ou encore un autre format unique. Si ce format est celui de Windows, il suffit de le modifier dans les paramètres régionaux. Dès lors, le format affiché sera celui défini dans le système. Sinon, si vous avez défini un format Paradox spécifique, il faudra le modifier...

En ce qui concerne la modification des données, un automate peut aller chercher dans toutes les tables de l'application (tables énumérées à l'aide de la méthode enumDatabaseTables) les champs de types monétaires (à l'aide de la méthode enumFieldStruct par exemple) puis de convertir toutes les valeurs situées dans ces colonnes.

2.2 seconde école : champs dédoublés

On peut aussi dédoubler dans la structure des tables, les champs des valeurs monétaires. La question se pose alors : Comment changer dynamiquement les UIObjects d'une fiche, afin qu'ils se repositionne sur le bon champ en fonction du choix Euro Franc ?

La réponse est simple : il suffit d'utiliser les propriétés des champs...

... et notamment les propriétés **FieldName** et **Format.NumberFormat**.

Illustration par des exemples :

On présuppose qu'est stocké dans une variable d'environnement de nom « PDXW_Monaie » la valeur E ou F, suivant que l'on travaille en Euro où en Franc. Pour nous y aider, nous avons créé le script de test : basculeEF.SSL. Afin d'illustrer notre exemple, nous avons créé la table COMPTE.DB dont la structure est la suivante :

Nom champ	Type	Clef
NO_LIGNE_CPT	+	*
DEBIT_FRANC_CPT	\$	
DEBIT_EURO_CPT	\$	
CREDIT_FRANC_CPT	\$	
CREDIT_EURO_CPT	\$	

2.2.1 champs dans une vue champ

Attention : lors de la conception, le champ doit avoir quelques propriétés par défaut d'établis :

- Tous les champs doivent avoir la propriété Conception, Ajuster la taille à faux.
- Les zones d'édition des champs doivent avoir la propriété Conception, Attacher horizontalement à vrai.

On utilise l'événement Open de chacun des champs concernés :

```

errorTrapOnWarnings (Yes)
TRY
    if ReadEnvironmentString("PDXW_Monaie") = "F"
    then ; on travaille en Francs :
        Self.FieldName = "DEBIT_FRANC_CPT"
        Self.ZedDebit.Format.NumberFormat = "W. $, E$WNWDWLWPW, S+W-W"
    else
        Self.FieldName = "DEBIT_EURO_CPT"
        Self.ZedDebit.Format.NumberFormat = "W.2, E(E)N D, L1PB, S+4-5"
    endif
ONFAIL
    ErrorShow()
    errorTrapOnWarnings (No)
ENDTRY
errorTrapOnWarnings (No)
LblDebit.text = "Débit"

```

On réassigne la propriété FieldName du champ au champ de la table représentant la bonne monnaie.

On redéfinit le format d'affichage de la zone d'édition du champ en fonction de la monnaie choisie à l'aide de la propriété Format.NumberFormat.

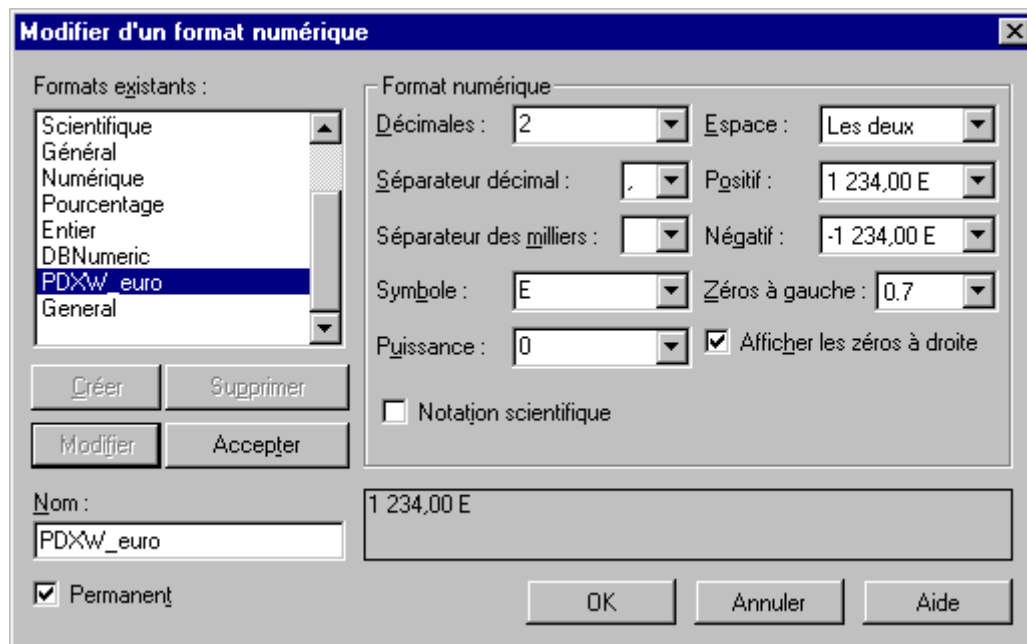
Enfin, on réassigne la propriété text du libellé du champ (nom ZedDebit pour Zone d'édition du champ débit) au texte de présentation désiré.

Mais comment a-t-il fait pour aller pêcher de tels formats pour les valeur numériques (monétaires) ?

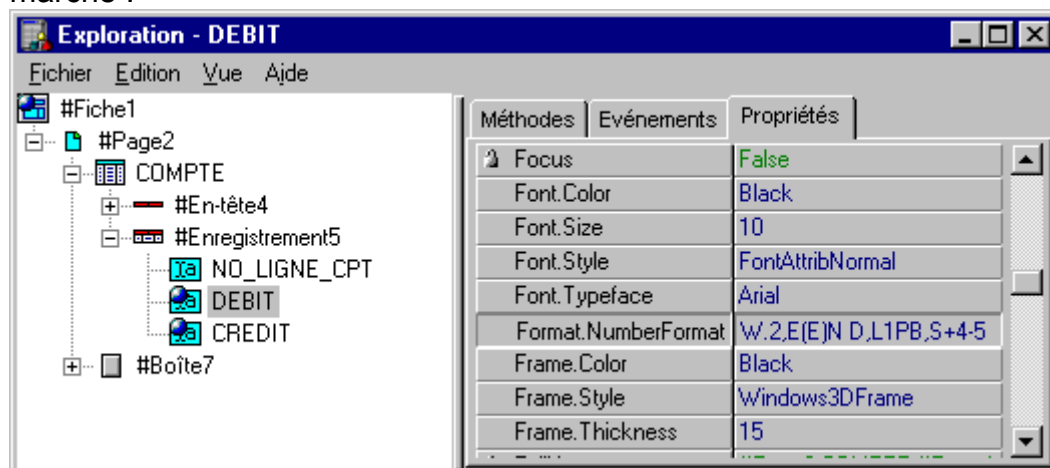
C'est vrai que Paradox ne fait pas dans la dentelle quand il s'agit de créer un format quelconque. Allez-donc vous amuser à suivre l'aide en ligne de la méthode format...

C'est parfaitement inutile, car il y a un moyen simple et efficace...

Construisez un nouveau format dans la boîte de dialogue des propriétés du champ



...puis récupérez ce format dans l'inspecteur d'objet, onglet, propriété, ligne Format.NumberFormat. C'est aussi simple que cela. Pas besoin de comprendre, ça marche !



Voir la fiche CptLigne.fsl

2.2.2 champs dans une vue cadre de table

Les choses se compliquent un peu, car si vous portez le code tel quel dans la méthode open, seule le premier champ dans le cadre de table sera concerné par cette modification. Si aucune valeur n'est présente dans la table cela n'est pas grave. Mais si plusieurs valeurs sont visibles dans le cadre de table, alors, c'est la catastrophe...

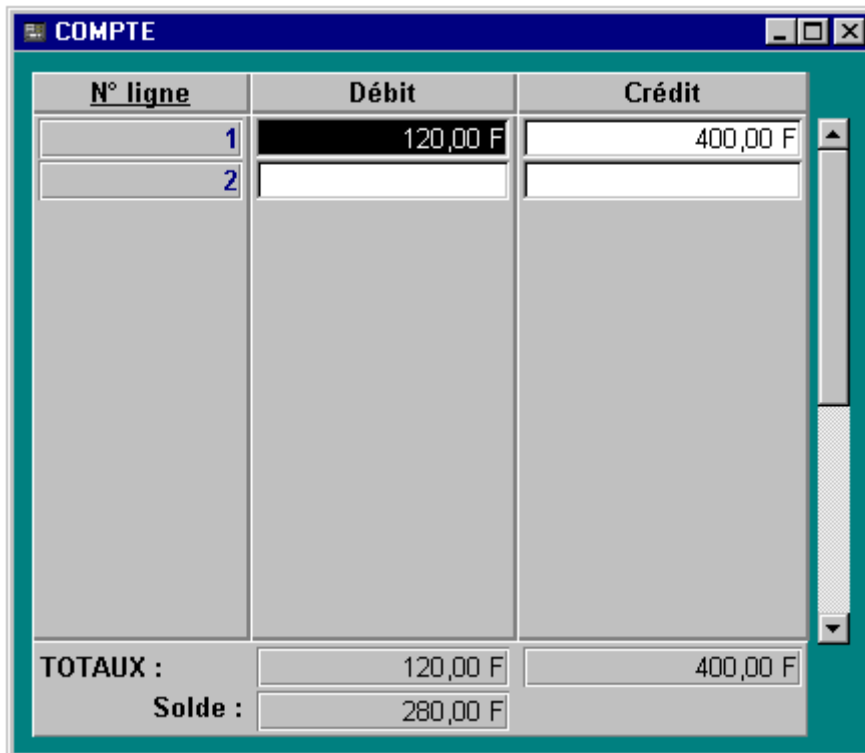
Il faut comprendre que Paradox construit autant d'UIObject, par réplication de l'UIObject Champ situé en premier dans la table, mais de nom différent. Comment faire alors pour répliquer la méthode Open autant de fois que sont créé les champs « fantôme » ?

Il suffit de compter...

Exemple :

Dans la fiche COMPTE.FSL, nous voyons un cadre de table avec la table compte qui contient deux champs Débit et Crédit.

Les champs à modifier sont les champs affichant les lignes de crédit et débit de la table, et les trois champs calculés qui font la somme des débits, des crédits et le solde.



N° ligne	Débit	Crédit
1	120,00 F	400,00 F
2		
TOTAUX :		400,00 F
Solde :		280,00 F

Dans la case var de la fiche, créer autant de compteur qu'il y a de type de champs concernés dans le cadre de table. Ici 2 suffisent

```
Var
    nCreateF1,nCreateF2 smallint
endVar
```

Dans l'événement Open de la fiche, assignez ces compteurs à la valeur 0
Dans l'événement Open du champ Débit, placez le code suivant :

```
var
    nCreate smallint
endVar
; création d'un compteur local
method open(var eventInfo Event)
; le compteur local est vide, on lui donne la valeur 1
if not nCreate.isAssigned()
then
    nCreate = 1
endif
```

```

; le compteur local est-il différent du compteur niveau fiche
(nous verrons que la réponse est toujours OUI
if nCreate <> nCreateF1
then ; alors on change
; changement EURO / FRANC
errorTrapOnWarnings(Yes)
TRY
    if ReadEnvironmentString("PDXW_Monaie") = "F"
    then
        Self.FieldName = "DEBIT_FRANC_CPT"
        Self.Format.NumberFormat = "W. $,E$WNWDWLWPW,S+W-W"
    else
        Self.FieldName = "DEBIT_EURO_CPT"
        Self.Format.NumberFormat = "W.2,E(E)N D,L1PB,S+4-5"
    endif
ONFAIL
    ErrorShow()
    errorTrapOnWarnings(No)
ENDTRY
errorTrapOnWarnings(No)
LblDebit.text = "Débit"
nCreateF1 = nCreate ; le compteur niveau fiche reçoit
; la valeur du compteur local
nCreate = nCreate + 1 ; le compteur local est incrémenté
endif
endMethod

```

Bien entendu il en est de même pour l'autre champ, en se reposant sur le compteur nCreateF2 de niveau fiche.

2.2.3 champs calculés

Rien de plus simple...

Modifiez les formules de calcul de la manière suivante

Calcul du total débit

```

iif(ReadEnvironmentString("PDXW_Monaie")="F",Somme([COMPTE.DEBIT_FRANC_CPT]
),Somme([COMPTE.DEBIT_EURO_CPT]))

```

Calcul du solde :

```

iif(ReadEnvironmentString("PDXW_Monaie")="F",Somme([COMPTE.CREDIT_FRANC_CPT
])-Somme([COMPTE.DEBIT_FRANC_CPT]),Somme([COMPTE.CREDIT_EURO_CPT])-
Somme([COMPTE.DEBIT_EURO_CPT]))

```

CONCLUSION

Ni l'Euro, ni l'an 2000, ne saurait être un frein à la modification, ni au développement à l'aide de Paradox. Reste l'aspect financier de l'opération : faire croire que c'est

difficile, délicat, et donc assommer le client avec un devis bien sentis... comme presque tout le monde !