ATOUTFOX NEWS

ATOUTFOX
COMMUNAUTÉ FRANCOPHONE DES PROFESSIONNELS FOXPRO

ATOUTFOX NEWS 2007—N°2 AOUT 2007

Réunion du Comité
(21 et 22 avril)

Etaient présents : Anatole Baudoin, Nathalie Chacon, Christophe Chenavier, Louis de Lacroix, Francis Faure, Patrick Gilles, François Lepage, Michel Lévy, Emmanuel Remy, Jean-Marc Testud

Le Comité s'est réuni les 21 & 22 Avril 2007 à Caumont sur Durance suite à l'annonce du 13 Mars 2007 par Microsoft

L'annonce de Microsoft :

- Le prochain service pack (SP2) est prévu pour l'été 2007, avec comme principal objectif la prise en charge de Windows Vista.
- Le support Microsoft pour VFP est confirmé jusqu'en janvier 2015 (ce qui implique des services packs si nécessaire pour les futurs OS ou bugs)
- Les travaux actuels sous nom de code « Sedna » (les CTPs sont actuellement disponibles publiquement) vont être publiés en licence « open source », à priori dans le support Codeplex (http://www.codeplex.com), Sedna assurera l'interopérabilité Visual FoxPro & DotNet
- Et enfin, il n'est pas planifié par Microsoft une version 10 de VFP...

Donc :

- La version actuellement supportée par Microsoft et qui aura la prise en charge de «Windows Vista» est très clairement: Visual FoxPro Version 9: le SP2 annoncé pour l'été 2007. Nous invitons les membres de l'association à:

1. Télécharger la CTP de SP2
2. L'installer sur une machine de tests sous Windows Vista
3. Procéder aux tests de leurs applications avec cette CTP
4. Remonter les éventuels dysfonctionnements sur Connect et/ou sur www.atoutfox.org

- Les prochaines Rencontres AtoutFox seront orientées vers la migration urgente en VFP9, son exploitation approfondie au travers des meilleures pratiques, l'utilisation de Sedna et la présentation de solutions viables de migration/portage qui s'offrent à nous.

- L'association vous informe qu'une pétition existe pour que Microsoft poursuive le développement de Visual FoxPro, confie ce développement à une société tierce, ou mette l'intégralité du produit en open source. Cette pétition est disponible sur : www.masfoxpro.com/. L'association ne donne aucune consigne à ce sujet.
Lors de cette réunion,

les solutions suivantes ont été répertoriées et rapidement visitées :

Autour de DotNet

- LINQ
- MS VFP 2 dotnet
- OLEDB
- Sedna
- CodeBase
- Etecnologia
- Olymars
- VfpConversion
- Vulcan
- Guineu

Autour du Monde Libre

- Dabo
- Harbour-projet

Autour du DBF

- Flagship
- Recital9
- Xbase++
- Xharbourg.com
- dBase

Au tour des autres solutions

Frontaux :

- Java
- Flash

Serveurs :

- IBM DB2
- MS SQL Server
- MySQL
- Postgres SQL
- Magic software
- Progress
- S7solutions
- Windev

Nous remercions tous les membres de nous faire connaître d'autres solutions non-répertoriées qu'ils auraient trouvées.

Dans l'esprit de notre communauté, l'aspect francophone (documentations, supports, ressources) de ces solutions doit rester un critère majeur.

La mise en place de groupes de travail est maintenant nécessaire pour étudier les solutions viables de migration qui seront présentées aux prochaines Rencontres AtoutFox 2007.

La réussite de la migration de chacun dépend de l'engagement de tous. Faites-nous connaître au plus vite le groupe auquel vous voulez participer ou que vous voulez animer en adressant un mail à comite@atoutfox.org

Lors de ces Rencontres, un débat sur la modification des objectifs et des statuts de l'association sera porté à l'ordre du jour de l'assemblée générale.

Quelques échos des premiers tests...

(les propos rapportés ici n'engagent que leurs auteurs, ne sont que des premières impressions, et ne constituent en aucune façon un jugement définitif)

Avec la sortie de MS Silverlight (concurent de Flash), qui utilise comme langage VB.NET (entr'autres, y a Python aussi), et qui dans sa syntaxe n'est pas si loin de Fox (documentation dans la langue de Molière)
d'autant plus que la prochaine version de Visual studio introduit le typage dynamique, je pense qu'il y en a qui vont économiser des licences TSE.

MySQL : cette base de données en contient 2 : MyISAM et InnoDB
MyISAM n'est pas vraiment en SGBDR, puisqu'il ne gère pas ce que l'on attend d'un tel produit (procédures stockées, triggers, ..)
InnoDB est la version "sérieuse" de MySQL, mais celle-ci a été rachetée par Oracle en septembre 2006.
C'est suite à cela que MySQL a décidé de réécrire un module complet, dont le nom est Falcon. Pour cela, ils ont embauché le "père" de Firebird (version n-2 du SGBDR de Borland : InterBase)
A ce jour, Falcon est en version Alpha, et est annoncée pour fin 2007.

je suis tout de même passé par le site IBM,
Et Oh Surprise.....La nouvelle version de DB2 Express-C, qui est leur version gratuite, possède comme limitation :
- Maxi 4Go de mémoire utilisée
- 2 processeurs (ou 2 core) Maxi d'utilisé.
Par contre, AUCUNE LIMITATION EN TAILLE DE BASE DE DONNEES, ET AUCUNE LIMITATION EN NOMBRE D'UTILISATEUR.

A part ca, j'ai testé Windev express que l'on doit forcer en mode compatibilité sous vista pour pas que cela plante (laisser tomber y a meme pas une instruction pour créer une fenêtre dynamiquement, punaise qu'est ce que j'ai cherché..., une version de mise à jour tous les mois, c'est plus des services packs, c'est un abonnement)
J'ai testé aussi 4D, cela me rappelle le feuilleton, ils n'en sont pas sortis.
J'ai chargé VB.net et la j'ai pu faire une fenêtre principale avec menu quitter au bout d'une demi heure sans rien n'y connaitre. Pour l'instant c'est VB qui gagne. Mais je vais tester avec Ironpython histoire de me garder une porte de sortie sur DABO qui reste pour moi le seul IDE dans la philosophie Fox.

VB.NET pour les développeurs FoxPro, 1ère partie

par Les Pinter

La version originale de cet article a été publiée dans
la newsletter d'Avril de VFUG
Traduction : Gregory Adam

Foxpro nous a gâté. Il est tellement facile d'écrire une application, qu'on panique rien qu'en voyant un autre langage. En tous cas, moi j'ai paniqué. Maintenant, cinq années plus tard, je sais faire en VB.Net à peu près tout ce que je savais faire en FoxPro. Et cela avec aussi peu de lignes de code. Ecrire le code n'est pas difficile, c'est l'apprentissage qui l'est.

Ceci est un article en deux parties. Je vais vous donner le top 10 de mes tuyaux, pour faire en VB.NET ce qui est facile à faire en FoxPro. Il y a des différences structurelles bien sur, mais une fois qu'on s'y habitue on écrit du code assez bien à la vitesse habituelle. Mais ce n'est pas votre FoxPro familier…

Le Framework de base…

En Foxpro, si on commence un projet on écrit :

```
MODIFY PROJECT FirstApp
```

En VB, dans l'IDE on fait New, Visual Basic Project. Jusqu'ici ce n'est pas tellement différent. Par contre, un projet Foxpro se trouve dans deux tables (données et champs mémo) avec les extensions .PJX et .PJT. Un projet en VB se trouve dans deux fichiers avec les extensions .vbproj et .vbproj.user. Ce sont des fichiers XML. C'est comme cela que Microsoft a traduit DBF. XML décrit les lignes et les colonnes avec quelques avantages. Mais un sérieux désavantage quand même : il n'y a pas d'index. Si on n'y met pas trop de données, ça peut aller. Mais pour beaucoup de données il faut SQL Server. Je me demande si Microsoft le savait.

La plupart des exécutables utilisent plus d'un projet. Vous-êtes vous déjà aperçu du fait que les programmes Basic utilisent beaucoup de petites DLL ?

Et bien, c'est comme cela qu'on implémente les Classes. Les DLL sont utilisables par différents programmes. Votre application VB comprendra différents projets. Eh bien, tous – sauf un – seront compiles en DLL.
Et la seule exception deviendra un EXE.

Tout est regroupé à un niveau supérieur : la Solution.

La Solution est dans deux fichiers avec les extensions .sln et .suo. Ces fichiers contiennent les noms des projets qui sont inclus dans la Solution. Le fichier .sln est en XML, l'autre a un format propriétaire. S'il y a plusieurs projets dans une Solution il n'y aura qu'un seul projet qui deviendra un EXE. Les autres (des librairies Classe) deviendront des DLL.

Références

Les références pointent vers les DLL dont l'EXE a besoin. Si on clique sur les Références du projet on verra 3 à 5 DLL qui ont été ajoutées automatiquement lors de la création du projet après avoir choisi le type du projet. Si on fait un clique droit, puis Add Reference et après select Project References, on verra une liste de toutes les DLL des autres projets compris dans la Solution. Si on veut inclure une autre DLL, on peut le faire en indiquant le chemin à cette DLL. Il se peut qu'on doive ajouter des références à des DLL qui n'ont pas été incluses automatiquement en créant la Solution.

Expliquons-nous. Quand on crée une librairie Classe, il n'y a que trois références qui sont ajoutées. C'est un strict minimum qui inclut les appels Système. Par contre, si on crée une Application Windows les classlib (assembly) Windows Forms et System.Data sont ajoutées parce que VS présume qu'on va utiliser des Forms et traiter des données.

FoxPro n'a qu'une seule librairie, le runtime. Celle-ci contient tout le langage. .NET a à peu près 400 DLL et on inclut celles dont on a besoin. Donc, si par exemple on avait créé une classe et on décide plus tard qu'on va utiliser des forms et traiter des données, on devra ajouter ces deux références.

_Screen et Form1.vb

Par défaut, Foxpro propose son _screen comme écran d'arrière-plan, auquel on peut ajouter un menu et dans lequel on peut montrer d'autres Forms/Windows. VB.NET ne le fait pas. On doit y pourvoir nous-mêmes et installer notre propre menu.

Quand on crée un projet Windows Application, un form Form1 est ajouté automatiquement. En Foxpro, l'ajout d'un form est une paire de fichiers (.SCX et .SCT).
En VB.NET, un form est une classe (programme). L'interface en mode design reflète le contenu du programme. Quand on manipule le form, par exemple quand on bouge des objets dessus, le code du programme est changé de suite. Foxpro fonctionne à peu près de la même manière. Mais en VB il est assez facile de planter le designer. Et quand on le fait on doit fermer et redémarrer Visual Studio. Quand on a un projet/Solution d'une taille comme en nous avons (plus de 2000 fichiers de code), on perd facilement 3 minutes avec un ordinateur muni d'un processeur puissant.

VB.NET pour les développeurs FoxPro

D'habitude, je change le nom Form1.vb en MainForm.vb. Ceci change le nom de la classe sur la première ligne du code qui définit le form. On fait Alt+A,E (Affichage-Code) pour voir le code et Alt+A,C (Affichage-Concepteur) pour voir le Designer. Changeons par exemple les dimensions du form en 1014 X 698, BorderStyle en Fixed3D, MaximizeBox en False, StartPosition en CenterScreen, KeyPreview en True et Text (Caption en Foxpro) en 'Mon Application', et on a presque fini.

Ajouter un menu et un menu click event handler

Ensuite, on ouvre la fenêtre Toolbox avec Ctrl, Alt, X et on sélectionne un MenuStrip de la boite 'Menus and Toolbars'. Nous avons alors maintenant ce que nous avons en VFP quand on clique sur l'icône de VFP dans le menu de démarrage. Entrons 'File' sur la première ligne et 'E&xit' sur la seconde. (En FoxPro : 'E<\xit')
Maintenant, en appuyant sur la touche F5 notre premier petit programme va se lancer. Ce programme ne fait pas grand-chose pour le moment, mais ayons un peu de patience.

DO Form FormName

En VB.Net tout form est une classe. Ajoutons une nouvelle classe form et un petit form apparaîtra dans le form designer. Ce form est une classe avec un nom qui est d'habitude le même que le nom du fichier dans lequel la classe est stockée.

En VB.Net « `Do form FormName` » se traduit en deux lignes :

```
Dim frm as new frmCustomers
frm.ShowDialog()
```

Remarquons que les forms de Foxpro sont en fait des classes. La structure des tables SCX et VCX et identique. En Foxpro, on pourrait stocker nos forms comme des classes et faire

```
Frm = CreateObject('frmCustomers')
Frm.Show()
```

Je ne connais personne qui fait comme ça en FoxPro, mais on pourrait !

Ouvrir une table

Dans les langages .Net, une table est stockée localement en XML dans une structure appelée Dataset.

Exemple en Foxpro

```
CREATE CURSOR SomeData ( Name Char(30), Phone Char(15) )
INSERT INTO SomeData VALUES ( 'Les Pinter - California', '(650) 344-3969')
INSERT INTO SomeData VALUES ( 'Les Pinter - Boston', '(508) 650-3610')
BROWSE
```

VB.NET pour les développeurs FoxPro

Traduisons en VB.Net

```
Public Class Form1
Private Sub Form1_Load(ByVal
Dim dt As New DataTable
      dt.Columns.Add(dc1)
      dt.Columns.Add(dc2)

      Dim dr As DataRow = dt.NewRow
      dr(0) = "Les Pinter - California"
      dr(1) = "(650) 344-3969"
      dt.Rows.Add(dr)

      Dim dr2 As DataRow = dt.NewRow
      dr2(0) = "Les Pinter - Boston"
      dr2(1) = "(508) 650-3610"
      dt.Rows.Add(dr2)

      ds.Tables.Add(dt)

      DataGridView1.DataSource = ds.Tables(0)
      DataGridView1.AutoResizeColumn(0)
      DataGridView1.AutoResizeColumn(1)

   End Sub

End Class
```

Patience… ça va s'améliorer….

Ouverture et lecture d'une table

Je vais me servir des tables SQL Server pour cet exemple

En Foxpro, c'est un simple USE et si on a une grille Grid1 sur le form, on écrira ceci :

```
USE CUSTOMER
THISFORM.Grid1.RecordSource = ALIAS()
THISFORM.Grid1.Autofit
```

Pour ouvrir et lire une table en .Net on ouvre d'abord une connexion, puis on crée un objet pour la lecture des données et après on fait la lecture.
Voici un petit exemple en utilisant SQL Server :

VB.NET pour les développeurs FoxPro

```
Dim dc As New SqlClient.SqlConnection("Server=localhost;Database=Northwind;UID=sa;PWD=;")
dc.Open()
Dim da As New SqlClient.SqlDataAdapter("SELECT * FROM CUSTOMERS", dc)
da.Fill(ds)
DataGridView1.DataSource = ds.Tables(0)

For I As Integer = 0 To ds.Tables(0).Columns.Count - 1
   DataGridView1.AutoResizeColumn(I)
Next
```

Notes

(1) On a besoin du préfixe SQLClient parce qu'il y a quatre types d'adaptateurs, chacun étant une classe différente. Vous pourriez mettre « Import SQLClient » tout au début. Cela évite d'écrire SQLClient. chaque fois, mais on perdra l'Intellisense.

(2) On doit mettre DataSource = ds.Tables(0), parce que les Datasets peuvent contenir plusieurs tables, un peu comme un DataEnvironment de Form en FoxPro. La première table étant `Tables(0)` et non `Tables(1)` comme en FoxPro.

(3) En Foxpro la méthode AutoFit le fait sur un grid. En VB.Net cela se fait colonne par colonne.

Mise-à-jour d'une table

En Foxpro on fait un BROWSE et les changements sont sauvés. Même chose pour un grid. En VB.Net il faut ajouter des commandes Update dans le code, sauf quand on utilise des 'wizards' qui créent le code pour nous.
Voila le code de notre grid, modifié en incluant la mise-à-jour :

```
Imports System.Data.SQLClient

Public Class Form1

   Dim dc As SqlConnection
   Dim da As SqlDataAdapter
   Dim ds As New DataSet

   Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
MyBase.Load

      dc = New SqlClient.SqlConnection("Server=localhost;Database=Northwind;UID=sa;PWD=;")
      dc.Open()
      da = New SqlClient.SqlDataAdapter("SELECT * FROM CUSTOMERS", dc)
      da.Fill(ds)

      DataGridView1.DataSource = ds.Tables(0)
      For I As Integer = 0 To ds.Tables(0).Columns.Count - 1
         DataGridView1.AutoResizeColumn(I)
      Next

   End Sub
```

```
   Private Sub cmdUpdate_Click(ByVal sender As System.Object, ByVal e As
          System.EventArgs) Handles cmdUpdate.Click

      Dim cmd As New SqlClient.SqlCommandBuilder(da)
      da.UpdateCommand = cmd.GetUpdateCommand
      da.Update(ds)
      Close()

   End Sub

End Class
```

Vous voyez que les déclarations ont été mises au début pour que ces objets soient visibles dans les méthodes `Form1_Load()` et `cmdUpdate_Click()`.

La mauvaise nouvelle, absolument évidente, est qu'il faut beaucoup plus de code pour faire quoi que ce soit en VB.Net. La bonne nouvelle est que c'est toujours le même code, et donc qu'on peut ré-utiliser ce code. VB.Net (contrairement à VB6) est complètement orienté objet. Ce qui nous force à mettre notre code dans des classes ré-utilisables, par exemple : la construction d'un Dataset pour l'affichage dans une grille. Il suffit de l'écrire une fois et d'appeler notre méthode `GetDataset()` avec nos paramètres. Nous ne le faisons pas en FoxPro parce que nous n'y sommes pas obligés, mais nous pourrions le faire.

Etats/Reports

Une version réduite de Crystal Reports est incluse dans .Net. Quand on crée un état, il se rappelle du nom de notre connexion SQL Server. Si vous mettez votre exécutable sur un autre PC, Crystal Reports va chercher cet SQL Server, et vous n'y pouvez rien !

La solution est de créer un Typed Dataset pour chaque table et supprimer le DataAdapter que VS ajoute pour vous. Ensuite on va créer l'état en utilisant ce Typed Dataset. Dans notre programme on aura :

```
Dim dsCusts as New dsCustomers
ds = GetDataset("SELECT * FROM CUSTOMERS")
dsCusts.Merge(ds,"Customers")

Dim rpt as new rptCustomerReport
rpt.SetDataSource(dsCusts.Tables(1))
frm.CrystalReportViewer.ReportSource = rpt
```

Debugging

Quand on fait un clique-droit sur notre Solution, la quatrième ligne sera 'Configuration Manager'. C'est là qu'on indique le mode de notre Solution : Debug ou Release. Dans FoxPro on peut faire le debugging dans les deux. La différence, c'est que l'information pour le debugging est supprimée en mode Release. En VB.Net il est impossible de définir des points d'arrêt ou d'utiliser le DEBUGGER.BREAK (SET STEP ON en Foxpro) si on n'a pas pris la configuration Debug. Et même si on le fait le debugger de FoxPro, simple et « propre », va nous manquer. VB.Net est bien, mais Foxpro est tout simplement comme il le faut.

Conclusion

J'espère que cette petite introduction ne vous a pas trop effrayé. Je vous ai dit les choses comme elles le sont. J'aime bien VB.Net. Mais lors d'un moment de mélancolie je redémarre Foxpro et je me rappelle du bon vieux temps passé.

Dans la deuxième partie, je parlerai de la POO et comment écrire des fonctions et des méthodes. On verra aussi comment construire des user-controls, un sujet que j'aimerais présenter aux rencontres INETA dans les Etats-Unis.

Allez, au revoir

Les Pinter lespinter@earthlink.net

ATOUTFOX

Chez François LEPAGE

NOUIS 37250 SORIGNY

Directeur de la Publication :

François LEPAGE

Téléphone : +33 (0) 682 90 83 60
Télécopie : +33 (0) 247 34 87 89

Messagerie : fox@ater.net

Pour un développement durable ...

Communauté Francophone des Professionnels FoxPro

Association Loi de 1901, créée en Mai 2004.

L'association est à but non lucratif et à vocation non commerciale.

Elle a pour objectifs de :

* Promouvoir la plate-forme de développement FoxPro et Visual FoxPro, les outils associés et les applications qui en sont issues.
* Etre une force de proposition auprès de l'éditeur du produit et de ses représentants.
* Faciliter le partage d'expériences et de ressources entre les utilisateurs professionnels de FoxPro.