ASP.NET et Excel

Générer un flux ou un fichier Excel en ASP.NET

Ce script permet de pouvoir soit générer un flux Excel à la volée, soit de créer un fichier Excel en dynamique sur le serveur. Ensuite à partir de cet exemple, on peut prendre un fichier XLS préparamétrés et le modifier pour en faire un modèle.

Générer un Flux ou Fichier Excel en ASP.NET (sous Visual Studio.NET)

Le but de cet article est de montrer que .NET avec Visual Studio.NET permet de créer des solutions viables pour des projets d'envergure, liants des interfaces dites "Webisées" pour des applications professionnelles.

Excel fait parti de ces outils professionnels dont les entreprises ne peuvent plus se passer pour récupérer, manipuler et présenter des données (bilan d'activité, chiffre de ventes, ...).

Ainsi l'intégration des applications de Microsoft dans le développement de ces solutions devient alors tout à fait naturelle. Nous verrons donc avec des exemples simples comment lier Excel avec ASP.NET.

Dans cet article nous verrons 3 étapes :

- Création d'un Flux Excel en dynamique
- Création d'un fichier Excel en dynamique
- Utilisation d'un fichier Excel Modèle (Template)

Je m'appuierai sur un développement via Visual Studio.NET qui permet de séparer la partie programmation pure (xxx.asp.vb) de la partie graphique (xxx.aspx), ainsi le code fourni ne sera que du code xxx.asp.vb.
On pourra ainsi facilement transformer ces exemples de codes en fonctions (ou sub) que l'on placera dans le .vb.

Création d'un Flux Excel en dynamique

Création d'un Flux Excel en dynamique

Cette partie est simplement pour vous mettre en forme :)).
En effet, générer un flux Excel pour un navigateur (en particulier pour Internet Explorer) est très simple, car se gère par les "type mime".

Il suffit donc de créer un simple tableau et en disant au navigateur qu'il est de type Excel. Celui-ci chagera la page et appellera l'application associée au type mime Excel.

Exemple de Source

```
Dim lCnx As New accConnexionSQL(LaChaineDeConnexion)
lCnx.Open()
Try
    mTable = ChargeListe(lCnx)
    Dim MaLigne As DataRow
    Dim Temp As String
    Dim i As Integer = 0

    Temp = "<TABLE BORDER=1><TR>"
    Temp &= "<TD><B>Nom Prenom</B></TD>"
    Temp &= "<TD><B>Service</B></TD>"
    Temp &= "<TD><B>Localite</B></TD>"
    Temp &= "<TD><B>Telephone</B></TD>"
    Temp &= "<TD><B>Poste</B></TD>"
    Temp &= "<TD><B>Email</B></TD>"
    Temp &= "</TR>"
    For Each MaLigne In mTable.Rows
        Temp &= "<TR>"
        Temp &= "<TD>" & MaLigne("NomPrenom") & "</TD>"
        Temp &= "<TD>" & MaLigne("Service") & "</TD>"
        Temp &= "<TD>" & MaLigne("Localite") & "</TD>"
        Temp &= "<TD>" & MaLigne("Telephone") & "</TD>"
        Temp &= "<TD>" & MaLigne("Poste") & "</TD>"
        Temp &= "<TD>" & MaLigne("Email") & "</TD>"
        Temp &= "</TR>"
    Next
    Temp &= "</TABLE>"
    Temp = ZfStringUtil.SupprimeAccent(Temp)
    HttpContext.Current.Response.ContentType = "application/vnd.ms-excel"
    HttpContext.Current.Response.Write(Temp)
Finally
    lCnx.Close()
End Try
```

Cet exemple permet de créer en dynamique une liste de personnes issues une base de données (Annuaire d'une entreprise par exemple), et de fournir au visiteur un fichier excel qu'elle pourra enregistrer, imprimer ou modifier facilement suivant ses besoins.

Attention : Avec cette génération, un problème avec les caractères spéciaux (accents, ç, ...) s'est présenté, ce qui explique l'utilisation de la fonction de suppression des accents (dans le module de manipulation des chaînes).

Passons maintenant à un peu plus dur, la création d'un fichier excel sur le serveur.

Création d'un fichier Excel en dynamique

Dans cette partie nous allons créer un document Excel sur le serveur lui même. Pour ceci, il faut que sur le serveur Excel soit installé.

Il faut ensuite référencer l'objet COM Excel dans Visual Studio.NET (cf le référenceme nt dans mes autres articles de ASP.NET).

Exemple de Source

```
Dim xlapp As Excel.Application
Dim xlbook As Excel.Workbook
Dim xlsheet As Excel.Worksheet
Dim Repertoire As String = Server.MapPath(".")

xlapp = New Excel.Application()
xlapp.Visible = False
xlbook = xlapp.Workbooks.Add

xlsheet = xlapp.Sheets(1)
xlsheet.Name = "Ma Feuille Excel"
xlsheet.Range("A1").Value = "Bienvenue"
xlsheet.Range("A2").Value = "Sur"
xlsheet.Range("A3").Value = "ASP-PHP"
xlsheet.Range("A4").Value = "DotNet"

xlsheet.SaveAs(Repertoire & "\Export_Excel\" & LeNomFichier & ".xls")
```

Il faut veiller à ce que le User qui exécute le Framework .net (ASPNET en général) ait les droits d'utilisation d'Excel et de création de documents sur le serveur (dans le répertoire spécifié).

Passons enfin au stade final de cet article, la création d'un fichier excel sur le serveur en partant d'un modèle de base (un Template).

Utilisation d'un fichier Excel Modèle (Template)

Deux solutions sont possibles pour atteindre ce but :

- Ouvrir le fichier en créant et instanciant un objet Excel
- Utiliser la connexion ADO sur le fichier Excel

Le choix se fera suivant le but choisi et les contraintes de la machine ou sera l'application .NET.

La méthode la moins risquée (à mon sens) pour l'installation est celle de la connexion ADO, présentée ci-dessous.

Cette exemple va ouvrir le fichier avec une connexion "Excel over ADODB.Connection", qui permettra de faire du pseudo SQL, avec un Refresh pour valider les Insert.

Exemple de Source

```
Dim oWorkSheets As String
Dim oExcel As New ADODB.Connection()
Dim oRS As New ADODB.Recordset()

ODBCConnString = "DSN=Soubory Excel;DBQ=" + txtSelectKit.Text + ";"_
     "DefaultDir=" + UCase(txtSelectKit.Text.Substring(0, txtSelectKit.Text.LastIndexOf("\"))) + ";"_
     "DriverId=790;MaxBufferSize=2048;PageTimeout=5;"

oExcel.ConnectionString = ODBCConnString
oExcel.Open()
oRS = oExcel.OpenSchema(ADODB.SchemaEnum.adSchemaTables)


While Not oRS.EOF
   oWorkSheets = oRS.Fields("TABLE_NAME").Value
   oWorkSheets = oWorkSheets.Substring(0, oWorkSheets.Length - 1)
   If oWorkSheets.IndexOf("$") = -1 Then
      chklstKit.Items.Insert(chklstKit.Items.Count - 2, oWorkSheets)
   End If
   chklstKit.Refresh()
   oRS.MoveNext()
End While
```

Cette partie est a bien méditer afin de ne pas avoir de problème de sécurité avec l'ouverture du fichier ou avec le lancement de EXCEL (même en tache de fond).

Conclusion

Cet article m'a été inspiré dans le cadre de développement professionnel, et donc s'applique totalement dans les développements complets que prône Microsoft (Cf les publicités) pour la plate-forme .NET.
Il permet aussi de réassocier le monde des applications fixes (Excel) avec les développements Web.

En vous souhaitant de bons projets de développement.

Romelard Fabrice (alias F___)