

Utiliser ADOX

Version 1.99.37 – Juillet 2004 pour Developpez.com

COPYRIGHT ET DROIT DE REPRODUCTION

Ce support est libre de droit pour une utilisation dans un cadre privé ou non commercial. Vous ne devez pas le modifier sans l'autorisation écrite de son auteur. Pour un usage dans un but commercial, reportez-vous aux conditions d'utilisation à l'adresse :

www.beaussier.com/?pg=condition

Toute mise à disposition du support sur un autre site que Developpez.com est interdite.

Si vous souhaitez des améliorations, je suis évidemment ouvert à toute suggestion. Il en est de même si vous constatez une erreur (nul n'est parfait 😊). Pour cela, il suffit de m'écrire avec pour sujet « *Programmation Delphi / Utiliser ADOX* » dans la rubrique « *Contact* » de mon site principal :

www.beaussier.com

En revanche, je n'assure aucune aide, ni support sur des questions de programmation ou de compréhension de ce manuel. Je vous invite à vous reporter sur les excellents forums de Developpez.com (section *Delphi*).

Les marques et noms de société cités dans ce support sont déposées par leurs propriétaires respectifs. Delphi est la propriété exclusive de BORLAND. Windows et ACCESS sont la propriété exclusive de Microsoft Corporation.

Je ne suis lié avec aucun éditeur ou constructeur informatique.

Ce support a été réalisé avec la suite bureautique libre *Open Office* 1.1 (disponible gratuitement sur <http://fr.openoffice.org>) qui permet d'exporter nativement en PDF.

Avertissement complémentaire :

Les éléments (données ou formulaires) éventuellement inclus dans ce support vous sont fournis à titre d'exemple uniquement. Leur utilisation peut avoir, dans certains cas, des conséquences matériels et juridiques importantes qui peuvent varier selon le sujet dont ils traitent. Il est recommandé d'être assisté par une personne compétente en informatique ou de consulter un conseiller juridique ou financier avant de les utiliser ou de les adapter à votre activité.

Sommaire

- 1. INTRODUCTION.....4
 - 1.1. UN MOT SUR L'ADO..... 4
 - 1.2. ET SUR L'ADOX..... 4
 - 1.3. LIMITE DU SUPPORT..... 4
- 2. PRÉSENTATION..... 5
- 3. INSTALLATION (1ERE PARTIE)..... 6
- 4. BASE DE DONNÉES..... 8
- 5. APPLICATION.....10
 - 5.1. CRÉATION DE LA BASE..... 10
 - 5.2. TABLES..... 12
 - 5.3. INDEX..... 14
 - 5.4. RELATIONS..... 15
 - 5.5. EXÉCUTION..... 16
- 6. INSTALLATION (2E PARTIE).....17
- 7. COMPACTAGE..... 19
- 8. CONCLUSION..... 22

1. Introduction

Ce manuel est une suite à *Débuter avec l'ADO* (disponible sur ce même site). Il traite de l'accès vers une base de données MS-Access depuis Delphi à travers les composants ADOX.

Il est inspiré du support anglais fourni par Zarko Gajic sur le site About.Delphi.com.

1.1. Un mot sur l'ADO

L'ADO qui signifie en clair « ActiveX Data Objects », est un système d'accès aux données purement Microsoft dit de haut niveau. C'est-à-dire que la complexité de la connexion et des opérations de lecture écriture est masquée pour le développeur qui l'utilise.

Le système ADO a donc l'énorme avantage, si vous désirez vous connecter à des bases Microsoft comme Access ou SQL/Server, d'avoir ses bibliothèques en standard sur la plupart des versions récentes de Windows (98/2000/XP). Ce qui facilite le déploiement en clientèle de vos applications.



Note :

Pour des versions anciennes comme Windows 95 ou NT, vous aurez probablement besoin d'avoir recours au moteur ADO qui est distribué sous la forme du MDAC. Reportez-vous sur le site de Microsoft avec ce mot-clé.

1.2. Et sur l'ADOX

L'ADOX est une partie supplémentaire de l'ADO (X étant l'abrégié de eXtensions). Il fournit aux développeurs un jeu d'outils pour accéder à la structure ou à la sécurité d'une base de données. Il n'est cependant pas fourni dans Delphi avec le jeu de composants ADO standard.

1.3. Limite du support

Pour une bonne compréhension de ce support (avec notamment les copies écran), je pars du principe que vous utilisez Delphi dans sa version 7 et MS-Access 2000 minimum.

La version de Windows n'a pas d'importance bien que mon ordinateur soit sous une antique version 98SE. 😞

En revanche, assurez-vous de disposer d'un paquetage MDAC sur votre ordinateur.

2. Présentation

Dans le support *Débuter avec l'ADO*, nous avons vu comment utiliser les composants du même nom pour utiliser une base de données MS-Access. Cependant, plusieurs questions pouvaient rester en suspens et parmi elles, une primordiale :

Comment créer cette base MDB si l'on a pas le logiciel de Microsoft ?

Si vous possédez en effet Delphi, vous n'avez en revanche pas forcément MS-Access pour fabriquer le fichier MDB à partir de zéro. Nous allons voir qu'avec l'ADOX, nous allons non seulement créer la base mais que nous allons y ajouter les tables, les index et l'intégrité référentielle qui vont avec.

Autre avantage, vous n'avez plus besoin de fournir la base vide (avec uniquement la structure) avec votre application. Cette dernière pourra la fabriquer à loisir autant de fois que nécessaire. Cela s'avère très pratique dans le cas où l'utilisateur a supprimé la base par accident.

Comment compacter une base MDB ?

Il arrive enfin qu'à force d'écriture dans les tables (ajout, modification ou suppression d'enregistrements), la base de données se fragmente et utilise de l'espace disque inutilement. Elle ralentit alors au niveau des temps de réponse. Il est alors nécessaire de faire une opération dite de compactage pour notre fichier MDB.

3. Installation (1ere partie)

Installer la bibliothèque d'Extension ADO est d'une simplicité enfantine. Néanmoins, j'aime bien détailler chaque étape.

Ouvrez Delphi et choisissez le menu *Projet / Importer une bibliothèque de type...*



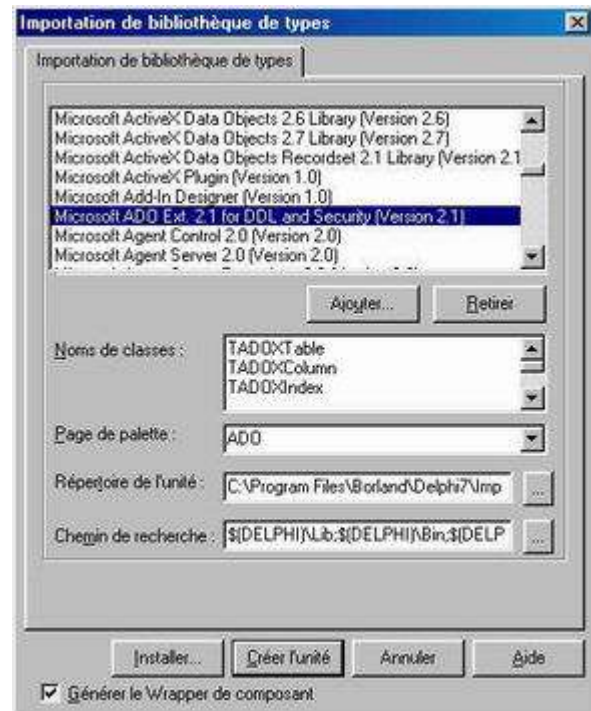
Dans la boîte de dialogue d'importation, sélectionnez la bibliothèque « *Microsoft ADO Ext. 2.x for DDL and Security* ».

Changez les noms de classes :

- **TTable** en **TADOXTable**
- **TColumn** en **TADOXColumn**
- **TIndex** en **TADOXIndex**
- **TKey** en **TADOXKey**
- **TGroup** en **TADOXGroup**
- **TUser** en **TADOXUser**
- **TCatalog** en **TADOXCatalog**

Modifiez la page de la palette pour mettre vos composants sous l'onglet *ADO*.

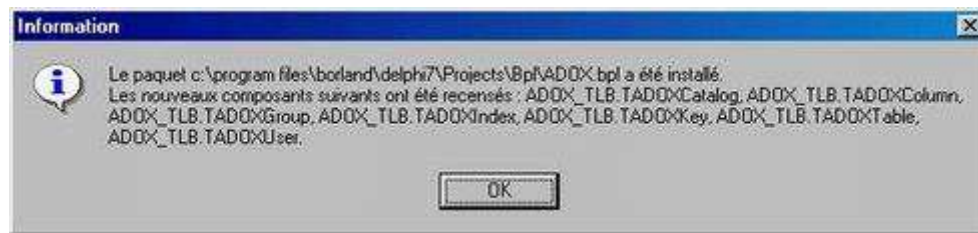
Appuyez sur le bouton « Installer ».



Important :

Il est impératif que vous changiez les noms de classes des composants avant de faire l'installation. Ceci afin de ne pas écraser ceux qui existent déjà comme le **TTable** du BDE.

Confirmez les étapes suivantes en appuyant sur « OK » une fois et « Oui » deux fois. A la fin du processus, la boîte ci-dessous doit apparaître :



Terminez la procédure par le menu *Fichier / Tout fermer* et confirmez par « Oui ».

Voilà, sept nouveaux composants ADOX sont désormais accessibles depuis la palette sous l'onglet ADO.



Ces composants vont nous servir pour la première partie de notre exercice : la création d'une base à partir de zéro.

4. Base de données

Pour notre exemple, nous allons créer une base de données nommée **Biblio.mdb**. Celle-ci sera composée de trois tables.

Une table principale...

Livre		
<u>Id</u>	Numéro automatique	Identifiant du livre
Titre	Chaîne (100)	Titre du livre
Auteur	Chaîne (100)	Nom de l'auteur
Editeur	Numérique	Clé étrangère
Prix	Monétaire	Prix du livre
Type	Numérique	Clé étrangère

Et deux tables filles...

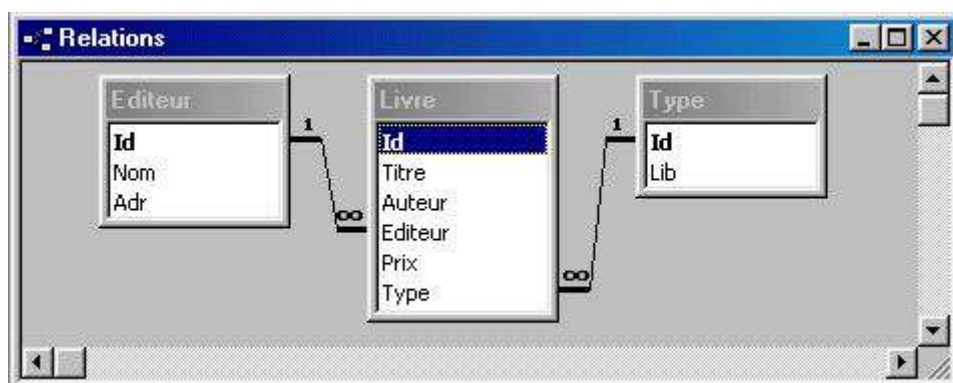
Editeur		
<u>Id</u>	Numéro automatique	Identifiant éditeur
Nom	Chaîne (100)	Nom de l'éditeur
Adr	Chaîne (200)	Adresse

Type		
<u>Id</u>	Numéro automatique	Identifiant unique
Lib	Chaîne (100)	Type de livre

Pour bien faire les choses, n'oublions pas les index sur les tables. La colonne `Livre.Id` étant déclarée en tant que clé primaire, elle sera ainsi automatiquement indexée¹. Même chose pour `Editeur.Id` et `Type.Id`.

¹ La documentation MS-Access précise qu'un index est automatiquement créé sur un champ déclaré en tant que clé primaire.

Voici la structure telle qu'elle apparaît dans l'éditeur graphique des relations (menu Outils) de MS-Access.



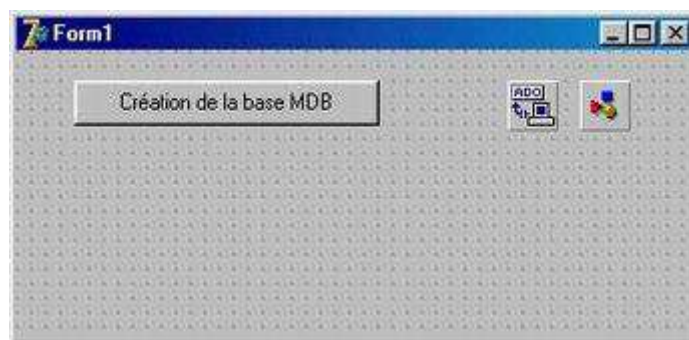
5. Application

Passons maintenant à la conception de notre programme Delphi. Enregistrez votre nouveau projet par exemple sous le nom « Biblio.dpr ».

5.1. Création de la base

Dans votre fenêtre **Form1**, vous allez placer trois composants :

- Un **TButton**
- Un **TADOConnection**
- Ainsi qu'un **TADOXCatalog**



Puis dans l'unité **Unit1.pas**, double cliquez sur le bouton « Création de la base MDB » pour y saisir le code ci-dessous :

```
procedure TForm1.Button1Click(Sender: TObject);
var
  cheminBD, chaineCnx : string;
begin
  cheminBD := 'C:\Bases\Biblio.mdb';
  chaineCnx:=
    'Provider=Microsoft.Jet.OLEDB.4.0;' +
    'User ID=Admin;' +
    'Data Source=' + cheminBD + ';' +
    'Mode=Share Deny None;Extended Properties="";' +
    'Jet OLEDB:Engine Type=5;';
  ADOXCatalog1.Create1(chaineCnx);
end;
```

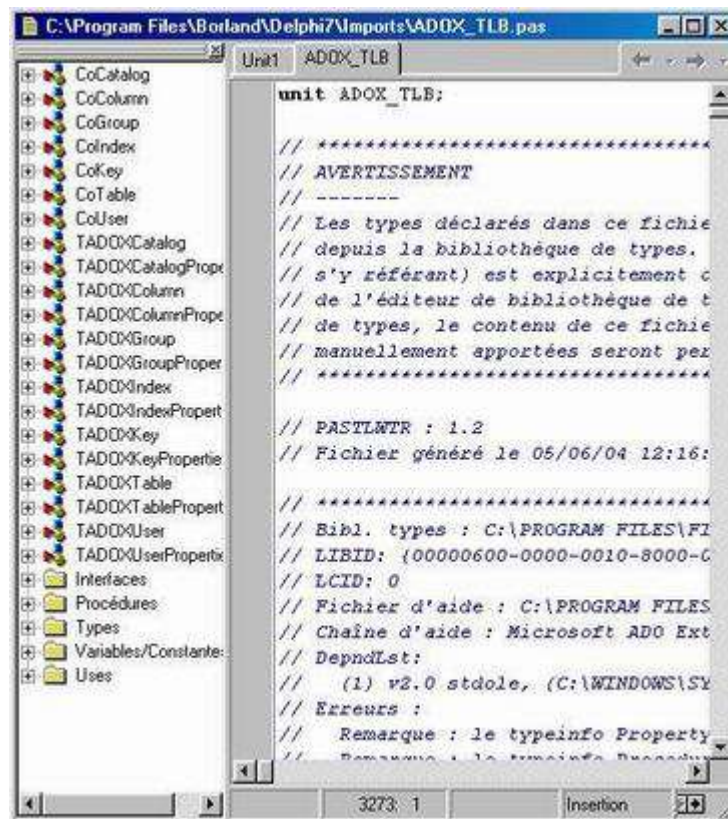
Difficile de faire plus simple. La fonction *Create1* va créer la nouvelle base pour le moment vide de tout contenu sur la base de la chaîne de connexion fournie.



A noter :

Dans la chaîne de connexion, le paramètre *Engine Type* est à **5** pour créer une base MS-Access au format 2000. Pour une base en version 97, mettez le type à 4.

Le fait d'exécuter le programme la première fois va importer automatiquement toutes les fonctions de la bibliothèque ADOX dans votre projet.



Vous pouvez constater maintenant la présence du fichier **Biblio.mdb** sur votre disque. La base est pour le moment vide.



A noter :

La fonction *Create1* ne vérifie pas l'existence du fichier. Si vous tentez d'exécuter une seconde fois ce code, vous aurez une erreur d'exception OLE.

Effacez le fichier **Biblio.mdb** et continuons l'exercice.

5.2. Tables

Passons à l'étape de création des tables.



A noter :

Pour cette étape et les suivantes, nous pourrions passer par l'ADOX et les composants **TADOXTable**, **TADOXIndex** et **TADOXKey**. Mais il est plus simple d'exécuter directement une requête SQL sur la base avec un **TADOCommand**.

Revenez sur votre **Form1** pour y ajouter deux nouveaux composants :

- Un second **TButton**
- Un **TADOCommand**



Puis dans la fiche, double cliquez sur le bouton « Création tables et relations » pour accéder à l'unité **Unit1.pas** et y saisir le code ci-dessous. Ceci afin de permettre de nous connecter dans un premier temps à la base et y attacher le composant **TADOCommand** :

```
procedure TForm1.Button2Click(Sender: TObject);
var
  cheminBD, chaineCnx, req : string;
begin
  cheminBD := 'C:\Bases\Biblio.mdb';

  chaineCnx:=
    'Provider=Microsoft.Jet.OLEDB.4.0;' +
    'User ID=Admin;' +
    'Data Source=' + cheminBD + ';' +
    'Mode=Share Deny None;Extended Properties="";' +
    'Jet OLEDB:Engine Type=5;';

  ADOConnection1.ConnectionString := chaineCnx;
  ADOConnection1.LoginPrompt := False;
  ADOCommand1.Connection := ADOConnection1;

  ...
```

Nous allons maintenant créer les deux tables détails *Editeur* et *Type* en passant leur structure par une requête *CREATE TABLE*.



A noter :

Je vous invite à vous reporter à des supports d'apprentissage du SQL si vous ne maîtrisez pas les concepts de champs et les définitions de type que j'énonce après.

D'abord la table *Type*...

```
...  
  
req:= 'CREATE TABLE Type (' +  
      ' Id INTEGER, ' +  
      ' Lib VARCHAR(100) ' +  
      ')';  
ADOCmd1.CommandText := req;  
ADOCmd1.Execute;  
  
...
```

Puis la table *Editeur*...

```
...  
  
req:= 'CREATE TABLE Editeur (' +  
      ' Id INTEGER, ' +  
      ' Nom VARCHAR(100) ' +  
      ' Adr VARCHAR(200) ' +  
      ')';  
ADOCmd1.CommandText := req;  
ADOCmd1.Execute;  
  
...
```

5.3. Index

Passons aux déclarations des index où nous reprenons le même principe de requête SQL avec notre composant **TADOCommand** :

```
...  
  
req:= 'CREATE INDEX CleType ' +  
      'ON Type (Id) WITH PRIMARY';  
ADOCommand1.CommandText := req;  
ADOCommand1.Execute;  
  
req:= 'CREATE INDEX CleEditeur ' +  
      'ON Editeur (Id) WITH PRIMARY ';  
ADOCommand1.CommandText := req;  
ADOCommand1.Execute;  
  
...
```

Un code tout simple qu'il est inutile de commenter.

5.4. Relations

Terminons enfin par la table *Livre*...

```
...  
  
req := 'CREATE TABLE Livre (' +  
      'Id INTEGER, ' +  
      'Titre VARCHAR(100), ' +  
      'Auteur VARCHAR(100), ' +  
      'Editeur INTEGER CONSTRAINT EditeurLivre ' +  
      'REFERENCES Editeur (Id), ' +  
      'Prix CURRENCY, ' +  
      'Type INTEGER CONSTRAINT TypeLivre ' +  
      'REFERENCES Type (Id) ' +  
      ')';  
ADOCmd1.CommandText := req;  
ADOCmd1.Execute;  
  
...
```

Sont créées lors de la création de la table, les contraintes d'intégrité sur les tables *Editeur* et *Type*. Dans notre cas, nous ne pourrions pas supprimer un éditeur ou un type si un enregistrement de la table *Livre* contient le tuple supprimé.

Il reste encore à poser un index pour l'identifiant de la table *Livre*.

```
...  
  
req:= 'CREATE INDEX CleLivre ' +  
      'ON Livre (Id) WITH PRIMARY';  
ADOCmd1.CommandText := req;  
ADOCmd1.Execute;  
  
end;
```

5.5. Exécution

C'est tout, votre application est prête à être exécutée. Cliquez successivement sur le bouton « Création d'une base MDB » puis « Création tables et index ». Fermez l'application.

Votre base MDB est installée sur votre disque. Si vous avez MS-Access, vous pouvez vérifier la présence des tables, des index et des relations qui ont été créés dans le fichier.

6. Installation (2e partie)

Passons maintenant à la seconde partie de notre manuel. Il s'agit de pouvoir gérer la fonction de compactage de base de données MS-Access.

Ouvrez Delphi et choisissez le menu *Projet / Importer une bibliothèque de type...*



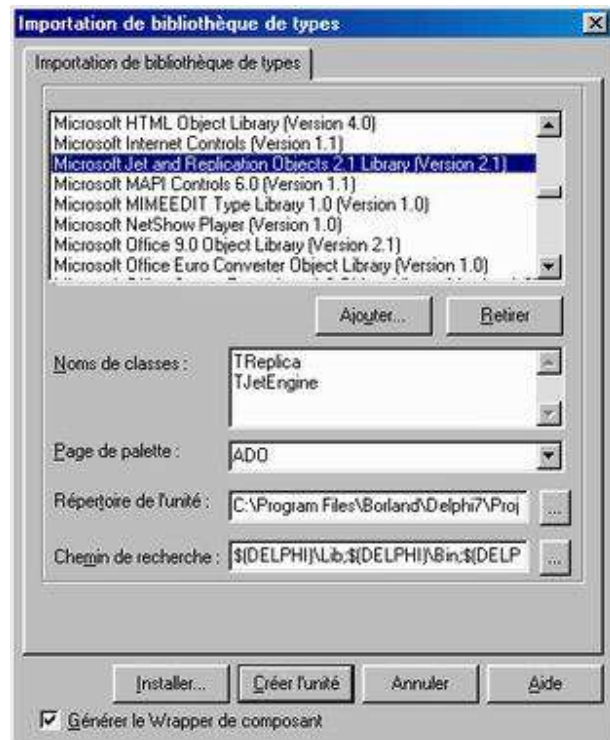
Dans la boîte de dialogue d'importation, sélectionnez la bibliothèque « *Microsoft Jet and Replication Objects 2.x Library* ».

Changez les noms de classes :

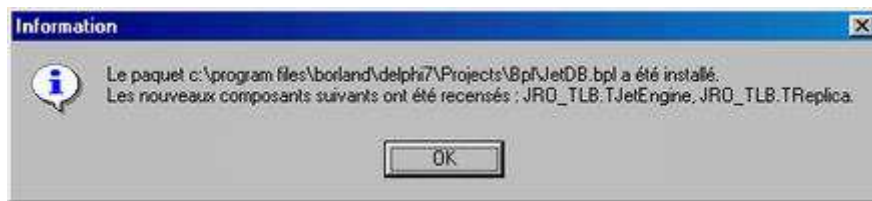
- **TReplica** en **TADOXReplica**
- **TJetEngine** en **TADOXJetEngine**

Modifiez la page de la palette pour mettre vos composants sous l'onglet *ADO*.

Appuyez sur le bouton « Installer ».



Confirmez les étapes suivantes en appuyant sur « OK » une fois et « Oui » deux fois. A la fin du processus, la boîte ci-dessous doit apparaître :



Terminez la procédure par le menu *Fichier / Tout fermer* et confirmez par « Oui ».

Voilà, deux nouveaux composants ADOX sont désormais accessibles depuis la palette sous l'onglet ADO.



A noter :

Si vous avez réalisé la 1ere partie du support, vous avez bien sûr les sept autres composants ADOX sur la palette.

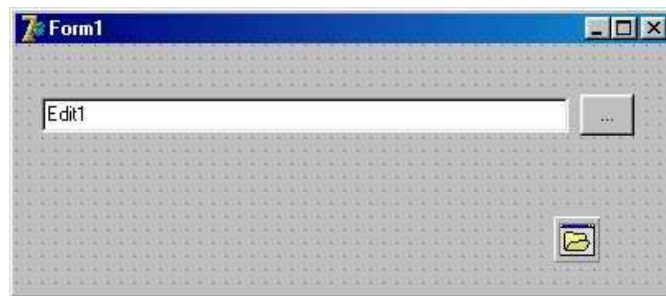
Ces composants vont donc nous servir pour la seconde partie de notre exercice : le compactage d'une base MDB.

7. Compactage

Passons maintenant à la conception de notre programme Delphi. Enregistrez votre nouveau projet par exemple sous le nom « Compact.dpr ».

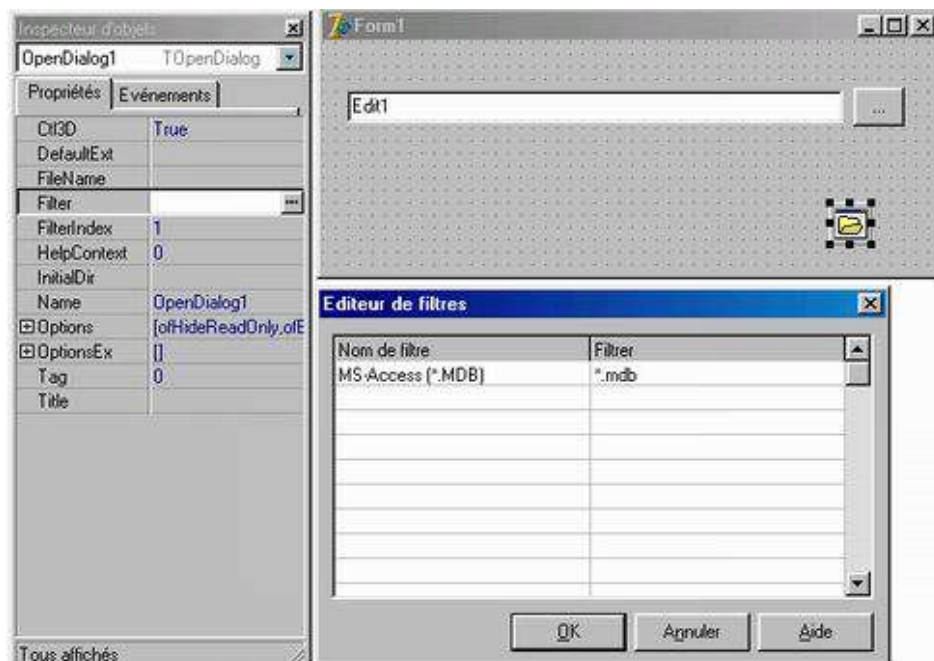
Sur notre formulaire, placez les trois composants suivants :

- TEdit pour la saisie du chemin à la base
- TOpenDialog pour l'accès à la boîte de sélection de fichier
- TButton pour l'ouverture du TOpenDialog



Mettez la propriété *ReadOnly* du champ *Edit* à *True* pour obliger l'utilisateur à passer par la boîte de sélection de fichiers. Modifiez pour le bouton la propriété *Caption* pour avoir ... (trois petits-points) en libellé.

Créez un filtre pour la boîte de dialogue d'ouverture de fichier afin que l'utilisateur ne puisse sélectionner que des .MDB.

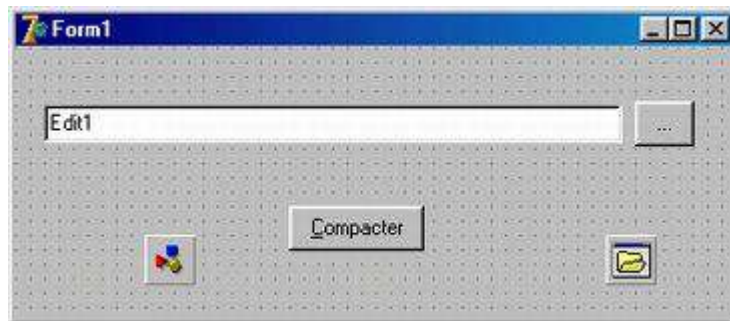


Enfin, saisissez le code ci-dessous pour le bouton afin d'ouvrir la boîte et récupérer le fichier sélectionné.

```
procedure TForm1.Button1Click(Sender: TObject);  
begin  
    if OpenDialog1.Execute then  
        Edit1.Text := OpenDialog1.FileName;  
end;
```

Revenons sur notre formulaire, pour y placer les deux composants suivants :

- TADOXJETEngine pour l'accès à la fonction de compactage
- TButtonpour l'exécution



Modifiez pour le bouton la propriété *Caption* pour avoir &Compacter comme libellé.

Terminons par le code de l'évènement *OnClick* du bouton *Compacter* :

```
procedure TForm1.Button2Click(Sender: TObject);  
var  
    sDNS :      string;  
    sBaseSrc :  string;  
    sBaseDst :  string;  
  
begin  
    sDNS := 'Provider=Microsoft.Jet.OLEDB.4.0;Data Source=';  
    sBaseSrc := sDNS + Edit1.Text;  
    sBaseDst := sDNS + 'BaseTmp.mdb';  
  
    if FileExists(Edit1.Text) then  
        begin  
            JetEngine1.CompactDatabase(sBaseSrc, sBaseDst);  
            DeleteFile(Edit1.Text);  
            RenameFile('BaseTmp.mdb', Edit1.Text);  
            ShowMessage('Base ' + Edit1.Text + ' compactée');  
        end  
    else  
        ShowMessage('Base ' + Edit1.Text + ' introuvable');  
end;
```

Il suffit d'appeler la procédure *CompactDatabase* de notre composant **JetEngine** pour effectuer l'opération de compactage de notre base.

On notera que l'on passe par une seconde base pour effectuer l'opération. Il suffit ensuite de détruire la base d'origine et de renommer le fichier temporaire.

8. Conclusion

Voilà ! Vous avez désormais toutes les billes pour gérer efficacement votre base de données MDB sans recourir au logiciel de l'éditeur.

Vous pouvez poursuivre votre apprentissage de l'ADO avec une application beaucoup plus complète. Je vous invite à télécharger la suite intitulée « *La connexion ADO Première application* ». Pour plus d'information, reportez-vous à l'adresse suivante : www.beaussier.com/?pg=doc