

Prise en main de Delphi 2005

par [Sébastien Doeraene](#)

Date de publication : 15/01/2005

Dernière mise à jour : 27/01/2005

Découvrez Delphi 2005 et prenez en main cet outil fabuleux

Introduction

I - Bureau de Delphi

II - Options générales

II-A - Options d'environnement

II-B - Options de l'éditeur

II-C - Options du débogueur

III - Création d'un nouveau projet

IV - Outils de conception

IV-A - Inspecteur d'objets

IV-B - Palette des composants

IV-C - Interface de l'application

V - Editeur de code

V-A - Code repliable

V-B - Gestion des erreurs de syntaxe en live

V-C - Bulles d'infos

V-D - Complétion automatique

V-E - Suggestion des paramètres

V-F - Snippets de code

V-G - Codage sélectif des fichiers sources

V-H - Synchronisation des modifications

VI - Historique

VI-A - Ajout d'un champ numéro de téléphone

VI-B - L'onglet Historique

VI-C - Comparaison de fichiers

VI-D - Restauration

VII - Une application .NET

VII-A - Différences

VII-B - Réalisation

VIII - Installer d'anciens composants

VIII-A - SynEdit

VIII-B - JVCL

VIII-C - SJRUnits & SJRDComps

IX - Évolutions du langage

Remerciements

Introduction

La grande nouvelle de l'année pour les développeurs Delphi, c'est la sortie de Delphi 2005 ! Cependant, cette version étant le regroupement de 3 anciennes versions (à savoir Delphi 7 pour le Win32, Delphi 8 pour le .NET, et C# Builder pour le .NET en C#), elle déroute beaucoup lors de sa première utilisation.

Cet article a pour but de vous aider à prendre en main cette nouvelle prouesse de Borland.

I - Bureau de Delphi

Lors du premier lancement de Delphi 2005, on peut être renversé par l'apparence générale de son nouveau bureau.

En fait, peu de choses ont réellement changé :

- Les diverses fenêtres d'outils sont dockées
- La palette d'outils est devenue une fenêtre d'outils
- Un navigateur Web présent au milieu décrit Delphi 2005 et propose quelques autres informations

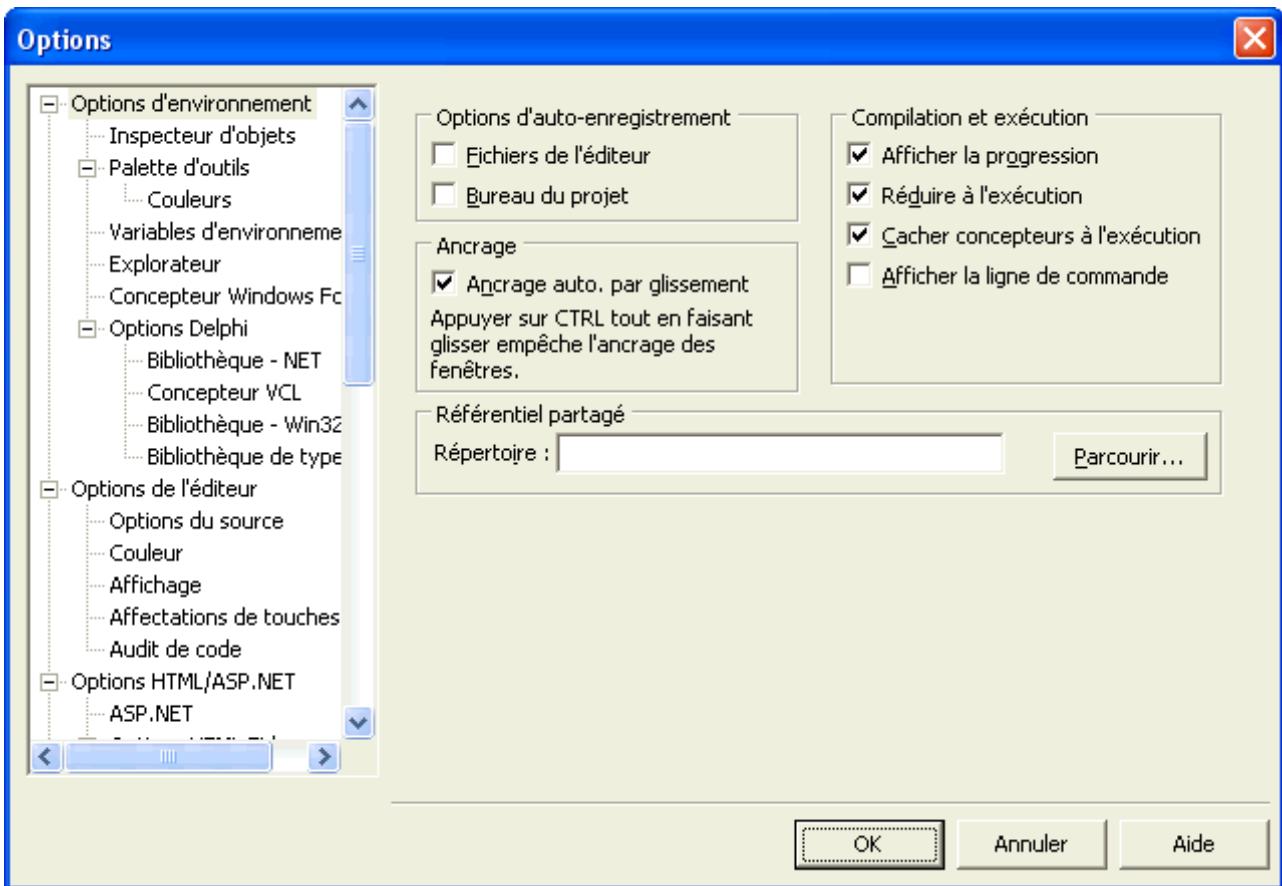
Vous retrouverez donc la structure de fiche et l'inspecteur d'objets sur la gauche ; le gestionnaire de projet et la palette des composants se trouvent sur la droite.

Si vous n'appréciez pas cette nouvelle disposition, vous pouvez choisir **Classic Undocked** dans la liste des bureaux.

II - Options générales

Nous allons tout d'abord visiter les options afin de retrouver les différents éléments qui nous permettront de nous y retrouver un peu plus.

Ouvrez donc le menu **Outils|Options**. La boîte de dialogue Options a bien changé depuis Delphi 7 :



À gauche, vous avez une liste de catégories d'options, et à droite, les options elles-mêmes.

Je vais vous présenter ici les quelques options qui seront généralement modifiées. Pour les autres, il y a un bouton **Aide** comme toujours.

II-A - Options d'environnement

Sélectionnez **Options d'environnement** dans la liste de gauche.

Vous trouverez ici les anciennes options d'environnement, comme les options d'auto-enregistrement et les modifications de L'EDI lors de l'exécution du programme.

Sélectionnez maintenant **Concepteur VCL** sous **Options d'environnement|Options Delphi**. Ici, vous trouverez les options concernant la conception de fiches VCL.

II-B - Options de l'éditeur

C'est sans doute la partie qui subira le plus de modifications, étant donné les changements de la syntaxe et de sa mise en surbrillance par défaut.

Sélectionnez **Options de l'éditeur**. Se trouvent là les options générales de l'éditeur et de l'édition.

Sous **Options de l'éditeur**, sélectionnez **Couleur**. Vous pourrez ici modifier les couleurs des syntaxes en visualisant un aperçu en HTML, en C#, En C/C++ et en Delphi.

Comme dit plus haut, beaucoup de couleurs ont changé.

Si vous n'aimez toujours pas les complétions automatiques et ce genre d'outils, sélectionnez **Audit de code** et modifiez les options en conséquence.

II-C - Options du débogueur

Ici, les options par défaut n'ont pas changé, mais vous pourrez retrouver des options que beaucoup de Delphistes modifient, notamment l'interruption des exceptions Delphi sous **Options du débogueur|Débogueurs Borland|Exceptions du langage**.

III - Création d'un nouveau projet

Nous allons créer un petit projet simple, histoire de nous familiariser à Delphi 2005 et surtout à sa présentation.

Dans la page de bienvenue du navigateur, vous trouverez un bouton **Nouveau** que vous pouvez utiliser pour créer un nouveau projet (ou n'importe quoi d'autre). Vous pouvez aussi utiliser le bouton **Nouveau** de la fenêtre "Gestionnaire de projets", le bouton **Nouveau** de la barre d'outil standard ou le menu **Fichier|Nouveau**.

Choisissez dans le sous-groupe **Projets Delphi**, l'élément **Application Fiches VCL** et validez.

Et là : horreur ! On voit une fiche Delphi dessinée dans la fenêtre ancrée, comme avec ce poisson Editeur VBA d'Office !

En réalité, on ne peut pas y faire grand-chose. Soit vous utilisez le bureau *undocked* et vous retrouverez votre bonne vieille fenêtre flottante, soit vous utilisez la nouvelle configuration et vous devrez vous y faire.

C'est à chacun de voir.

IV - Outils de conception

Maintenant que le coup de gueule sur la présentation de la fiche en conception est passé, on va pouvoir découvrir un peu plus en profondeur les outils de conception.

IV-A - Inspecteur d'objets

Mises à part les améliorations graphiques (qui au passage sont présentes dans tout Delphi 2005), l'inspecteur d'objets n'a pas changé.

Les propriétés sont triées par défaut par catégorie. Si vous préférez les trier par nom, utilisez le sous-menu **Trier|Par Nom** du menu contextuel de l'inspecteur.

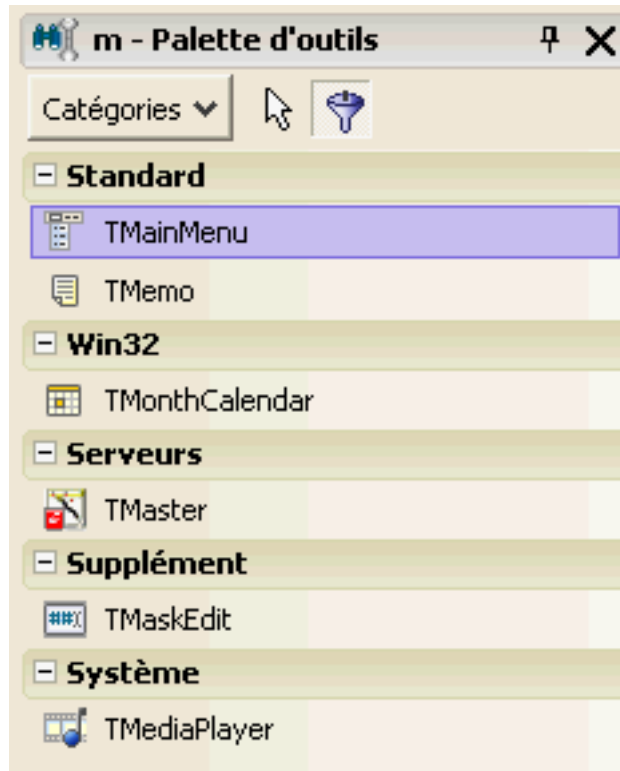
IV-B - Palette des composants

La palette des composants, qui pour rappel est devenue une fenêtre placée en bas à gauche (par défaut) de l'EDI, a été sensiblement améliorée.

Elle liste tous les composants installés triés par palette, puis par ordre alphabétique. Jusque là rien d'impressionnant.

En revanche, si vous sélectionnez la palette, puis entrez quelques touches au clavier, le titre de la fenêtre fonctionnera comme une zone de texte et un tri filtré sera effectué automatiquement.

Par exemple, si vous entrez la lettre 'm', la palette ressemblera à ceci :



Pour supprimer le filtre, utilisez la touche Retour arrière pour retirer les lettres une à une.

IV-C - Interface de l'application

Voici l'interface de notre application :



Je ne détaillerai pas la procédure de conception de cette interface. Ca n'en vaut pas la peine.

Toutefois vous pourrez remarquer dans le bas de la fenêtre du milieu (celle qui contient la fiche) trois onglets : **Code**, **Conception** et **Historique**.



L'onglet **Code** vous permet de visualiser le code en rapport avec la fiche que vous éditez. L'onglet **Conception** vous permet de modifier l'apparence de la fiche. Et l'onglet **Historique** vous permet de gérer l'historique du fichier.

V - Editeur de code

Une fois l'interface réalisée, passons au code. Cliquez donc sur l'onglet **Code** dont nous avons parlé ci-dessus.

Observez maintenant le tout nouvel éditeur tout beau tout neuf de Borland.

V-A - Code repliable

Hormis la beauté graphique de l'éditeur, on remarque de petits [-] dans la gouttière. Ces boutons permettent de replier les différentes parties du code.

Vous pouvez ainsi replier :

- L'unité entière
- La partie **interface**
- Chaque définition de classe
- La partie **implementation**
- Chaque implémentation de routine/méthode

Outre ces parties repliables par défaut, vous pouvez en définir de personnalisées grâce à la *directive de compilation* **{\$REGION}**.

Exemple de code repliable personnalisé

```
implementation
{
{$REGION 'Ressources'}
{$R *.dfm}
{$ENDREGION}
...
}
```

Une fois ces deux lignes placées autour de la liaison du .dfm, un nouveau petit [-] s'ajoute devant la ligne **{\$REGION 'Ressources'}**. Si vous cliquez dessus, le code situé jusqu'au **{\$ENDREGION}** se replie.

Je trouve cette fonctionnalité très pratique, si on n'en abuse pas.

V-B - Gestion des erreurs de syntaxe en live

Double-cliquez sur le bouton OK pour générer un événement **OnClick**. Commencez à entrer le code suivant :

```
if E
```

Delphi souligne alors en rouge ondulé le E, parce qu'il n'est pas reconnu comme identificateur déclaré. De même, il souligne le **end** car il s'attend à trouver le **then** correspondant au **if**.

De même, vous pourrez voir apparaître une nouvelle catégorie dans la fenêtre **Structure**, intitulée **Erreurs** et comportant les deux erreurs indiquées. En double-cliquant sur une erreur, le curseur vient se placer à l'endroit où cette erreur a été décelée.

Qui aurait pensé que Borland copierait sur le souligneur d'erreurs de grammaire de Word ? lol.

V-C - Bulles d'infos

Déplacez le curseur de la souris jusque sur le mot **TFormMain** dans la déclaration de la variable **FormMain**. Une bulle d'aide très instructive apparaît :

```
in: TFormMain;
```

```
tati
```

TFormMain Type - [EssaisMain.pas \(10,3\)](#)

TFormMain - [EssaisMain.TFormMain](#)

Cette bulle montre diverses infos variant selon la nature de l'identificateur pointé.

V-D - Complétion automatique

Continuez à entrer le code jusqu'au point :

```
if EditFirstName.
```

Comme dans les dernières versions de Delphi, l'éditeur propose alors une fenêtre qui vous permet de choisir aisément parmi les propriétés/méthodes de l'objet **EditFirstName**.

Cependant, une bulle d'infos apparaît pour chaque élément que vous choisissez.

Choisissez ici la propriété **Text**.

Terminez le code ci-dessous :

```
if EditFirstName.Text = '' then
begin
end;
```

V-E - Suggestion des paramètres

Pour ceux qui auraient pris du retard dans les versions de Delphi, nous allons voir les suggestions de paramètres pour les routines/méthodes.

Dans le code dont on a parlé ci-dessus, ajoutez :

```
if EditFirstName.Text = '' then
begin
  MessageDlg(
end;
```

Delphi souligne le **end** en rouge mais, plus important, il propose une bulle d'aide indiquant les divers paramètres attendus par la fonction **MessageDlg**.

Terminez le code ci-dessous :

```
if EditFirstName.Text = '' then
begin
```

```

MessageDlg('Vous devez donner un prénom', mtError, [mbOK], 0);
exit;
end;

```

V-F - Snippets de code

Les **Snippets de code** sont des portions de code que Delphi enregistre et qui sont réutilisables à plusieurs endroits.

Nous allons transformer le bloc **begin..end** avec message d'erreur ci-dessus pour en faire un snippet de code.

Modifiez d'abord temporairement le message d'erreur pour être un peu plus générique :

```

begin
  MessageDlg('Écrivez ici le message d''erreur', mtError, [mbOK], 0);
  exit;
end;

```

Assurez-vous que vous pouvez voir la palette de composants. Puis sélectionnez les quatre lignes ci-dessus.

Maintenez la touche **Alt** enfoncée et glissez la sélection sur la palette des composants.

Sous la catégorie **Snippets de code** dans la palette des composants, vous pouvez voir une nouvelle entrée commençant par **begin MessageDlg**. C'est votre snippet de code.

Les snippets de code étant communs à tous vos projets, vous pourrez réutiliser ce code facilement dans toutes vos applications.

Remplacez l'ancien message d'erreur, puis continuez le code ci-dessous :

```

if EditFirstName.Text = '' then
begin
  MessageDlg('Vous devez donner un prénom', mtError, [mbOK], 0);
  exit;
end;
if EditSurname.Text = '' then

```

Nous allons maintenant utiliser le snippet de code précédemment enregistré pour réaliser le bloc du **then**.

Si ce n'est pas déjà fait, déployez la catégorie **Snippets de code** de la palette des composants. Puis glissez le snippet que vous avez enregistré précédemment après le **then** dans le code.

Le code du snippet est ajouté automatiquement :

```

if EditSurname.Text = '' then
begin
  MessageDlg('Écrivez ici le message d''erreur', mtError, [mbOK], 0);
  exit;
end;

```

Modifiez alors le message d'erreur.

V-G - Codage sélectif des fichiers sources

Le langage Delphi a été étendu pour inclure les caractères Unicode alphanumériques et alphabétiques dans les identificateurs.

Vous pouvez maintenant choisir comment Delphi codera vos fichiers source. Les options incluent la norme ANSI, binaire, l'UTF8, etc.

Pour spécifier dans l'éditeur le type de codage du fichier source courant, faites un clic droit et choisissez le codage que vous voulez employer.

Pouvoir choisir le codage des fichiers source est particulièrement utile quand vous écrivez des fichiers source à destination d'autres pays.

Par exemple, les fichiers source codés en utilisant UTF-8 maintiendront correctement l'identité des différents caractères même lorsqu'ils sont ouvert avec une localisation différente. Par comparaison, les caractères spéciaux dans un fichier source peuvent changer si le fichier source est codé dans la norme ANSI puis ouvert avec un codepage différent de la norme ANSI.

Nous allons donc continuer notre code et utiliser une variable nommée **Données**.

```

procedure TFormMain.ButtonOKClick(Sender: TObject);
var Données : string;
begin
  if EditFirstName.Text = '' then
  begin
    MessageDlg('Vous devez donner un prénom', mtError, [mbOK], 0);
    exit;
  end;
  if EditSurname.Text = '' then
  begin
    MessageDlg('Vous devez donner un nom de famille', mtError, [mbOK], 0);
    exit;
  end;
  Données := EditFirstName.Text+';'+EditSurname.Text;
  if EditNickName.Text <> '' then
    Données := Données+';'+EditNickName.Text;
  MessageDlg('Données correctes :'+#10+Données, mtInformation, [mbOK], 0);
  Close;
end;

```

Les caractères Unicode ne sont pas autorisés pour les identificateurs dans les sections publiées des classes, ni dans les types utilisés par les membres publiés.

Le SDK .NET déconseille l'utilisation de caractères de soulignement au début des identificateurs, car ce symbole est réservé au système.

V-H - Synchronisation des modifications

Ajoutez un événement **OnClick** au bouton Annuler et complétez son code comme suit :

```

procedure TFormMain.ButtonCancelClick(Sender: TObject);
begin
  Close;
end;

```

Après avoir rédigé ces quelques lignes de code, vous vous dites que **Données** n'est pas un nom de variable très explicite et vous voudriez le changer. Avec les anciennes versions de Delphi, vous deviez modifier une à une toutes les occurrences de **Données** dans la procédure. Avec Delphi 2005, il y a une nouvelle méthode plus simple.

Sélectionnez l'intégralité du code de la procédure, puis cliquez sur le petit bouton qui est apparu dans la gouttière et intitulé **Mode Synchronisation des modifications**.

Le texte sélectionné prend un fond bleu clair. Sélectionnez alors un des mots **Données**, les autres s'encadrent. Saisissez un nouveau nom pour cette variable, par exemple **StrInfos** : toutes les autres instances du mot **Données** sont modifiées en même temps.

VI - Historique

VI-A - Ajout d'un champ numéro de téléphone

Enregistrez ce projet et exécutez-le (toujours F9). Vous avez réalisé votre première application avec Delphi 2005 !

Maintenant vous aimeriez ajouter un champ **Numéro de téléphone**.

Récupérez donc la fiche et modifiez-la comme suit :

Modifiez le code de l'événement **OnClick** du bouton OK comme suit :

```

procedure TFormMain.ButtonOKClick(Sender: TObject);
var StrInfos : string;
begin
  if EditFirstName.Text = '' then
  begin
    MessageDlg('Vous devez donner un prénom', mtError, [mbOK], 0);
    exit;
  end;
  if EditSurname.Text = '' then
  begin
    MessageDlg('Vous devez donner un nom de famille', mtError, [mbOK], 0);
    exit;
  end;
  StrInfos := EditFirstName.Text+' '+EditSurname.Text;
  if EditNickName.Text <> '' then
    StrInfos := StrInfos+' '+EditNickName.Text;
  if EditPhoneNumber.Text <> '' then
  begin
    if EditNickName.Text = '' then StrInfos := StrInfos+' ';
    StrInfos := StrInfos+' '+EditPhoneNumber.Text;
  end;
  MessageDlg('Données correctes :'+#10+StrInfos, mtInformation, [mbOK], 0);

```

```
Close;  
end;
```

Enregistrez votre projet et exécutez-le. Le nouveau champ est opérationnel.

VI-B - L'onglet Historique

Tout compte fait, vous n'aimez pas ce champ et voulez le supprimer. Avec les anciennes versions de Delphi, vous deviez repérer tous les endroits où intervenait ce champ et les supprimer. Avec cette nouvelle version, l'historique va beaucoup nous aider.

Cliquez donc sur l'onglet **Historique**. Trois sous-onglets sont disponibles, en bas à gauche : **Contenu**, **Info** et **Diff** (pour Différences).

Dans le sous-onglet **Contenu**, vous pouvez consulter le contenu de chaque sauvegarde historique de votre fichier.

Dans le sous-onglet **Info**, vous pouvez consulter quelques infos sur chaque sauvegarde mais aussi éditer un commentaire pour chaque. Ce peut être fort utile pour repérer les sauvegardes stables par exemple.

Dans le sous-onglet **Diff**, vous pouvez consulter les différences entre deux sauvegardes ou entre le fichier actuel et une sauvegarde.

Par défaut, Delphi gère un historique des 10 derniers enregistrements. Cette limite peut être augmentée jusqu'à 90 sauvegardes.

Pour modifier la limite des sauvegardes de l'historique, utilisez le menu **Outils|Options**, puis **Options de l'éditeur**. Modifiez alors la valeur de la zone **Limite sauv. fichiers**.

VI-C - Comparaison de fichiers

Étudions les différences entre le fichier actuel et la dernière sauvegarde. Sélectionnez pour ce faire **Fichier** à gauche et ~1~ (le numéro le plus haut qui existe). Vous pouvez alors constater dans la fenêtre de code du bas que certaines lignes sont surlignées en bleu clair avec un + devant (parfois il y a un -) :

```

17   ButtonOK: TBitBtn;
18   ButtonCancel: TBitBtn;
19   LabelPhoneNumber: TLabel;
20   EditPhoneNumber: TEdit;
21   procedure ButtonCancelClick(Sender: TObject);
22   procedure ButtonOKClick(Sender: TObject);
23   private
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52   if EditNickName.Text <> '' then
53     StrInfos := StrInfos+';'+EditNickName.Text;
54   if EditPhoneNumber.Text <> '' then
55   begin
56     if EditNickName.Text = '' then StrInfos := StrInfos+';';
57     StrInfos := StrInfos+';'+EditPhoneNumber.Text;
58   end;
59   MessageDlg('Données correctes :'+#10+StrInfos, mtInformation, [ml
60   Close;
61 end;

```

VI-D - Restauration

Nous allons maintenant utiliser la fonctionnalité la plus intéressante de l'historique, à savoir la **restauration**.

Retournez sous l'onglet **Contenu** ou **Info**. Sélectionnez la sauvegarde que nous venons de comparer avec le fichier actuel et cliquez sur le bouton **Restaurer la révision précédente**. Retournez dans le code et en conception : les modifications apportées pour le numéro de téléphone ont disparu.

VII - Une application .NET

Maintenant que vous savez vous servir des nouveaux outils de Delphi 2005, nous allons passer à ce que tout le monde attend : réaliser une application .NET.

Fermez le projet en cours. Puis ouvrez la boîte de dialogue **Nouveaux éléments**. Sélectionnez **Application fiches VCL** sous **Projets Delphi pour .NET**.

VII-A - Différences

On peut remarquer deux différences majeures par rapport à une application Win32 :

La première chose qui frappe est le gestionnaire de projets, qui contient beaucoup plus de dossiers et fichiers que pour un programme Win32. Vous n'aurez cependant pas à vous en soucier.

La deuxième est le source du programme (extension .bdsproj) qui a l'air bien plus compliqué que les sources des programmes VCL. N'ayez crainte, Delphi se charge, comme d'habitude, de ce fichier lui-même.

Une troisième minuscule différence : on utilise des fichier .nfm (**.NET Form**) au lieu de .dfm (**Delphi Form**).

Donc, pour la majorité de vos programmes, vous ne remarquerez même pas que vous faites du .NET !

VII-B - Réalisation

Réalisez la même application que tout à l'heure. Vous remarquerez que rien ne change dans votre démarche et qu'aucun problème ne se présente.

Vous pouvez enregistrer et compiler votre projet. Vous avez réalisé une application .NET aussi facilement qu'une application Win32 grâce à Delphi 2005 !

VIII - Installer d'anciens composants

Comme vous vous êtes beaucoup habitué à la **JVCL**, les composants **SynEdit** ou encore (sait-on jamais) les composants **SJRDComps**, vous aimeriez pouvoir les réinstaller sous Delphi 2005.

Nous allons voir que ce n'est pas bien compliqué, même si quelques démarches sont à effectuer.

VIII-A - SynEdit

Je commence par **SynEdit** car [Nono40](#) vous offre un excellent tutoriel pour l'installer sous Delphi 2005 :

[Installer SynEdit dans Delphi 2005](#)

Comme on ne peut rien ajouter à ce brillant tutoriel, passons tout de suite à la JVCL.

VIII-B - JVCL

Je base cette partie sur la version 1.94 de la JCL et la version 3.0.0.1 de la JVCL. Cette dernière version peut être téléchargée depuis [la page JCL de SourceForge](#) et [celle de la JVCL](#).

Pour pouvoir installer les productions Jedi dans Delphi 2005, vous devez posséder une édition complète (donc pas d'évaluation) de Delphi 2005.

Jedi propose un installateur très bien fait pour l'installation de ses librairies.

Après avoir décompressé la JCL et la JVCL dans deux répertoires distincts, ouvrez celui de la JCL, lancez **Install.bat**, et suivez les indications.

Vous ne devriez pas rencontrer de problème lors de l'installation de la JCL. Une fois ceci fait, lancez le batch **Install.bat** situé dans le répertoire de la JVCL pour installer la JVCL.

VIII-C - SJRDUnits & SJRDComps

Ouvrez le package **SJRDUUnits.dpk** et convertissez-le au format Win32. Supprimez toute la partie **requires** du package. Enregistrez.

Lors de la compilation, ajoutez tous les package que Delphi vous propose d'ajouter. Une fois compilé, installez le package.

Ouvrez ensuite le package **SJRDCompsR.dpk**, que vous convertissez également au format Win32. De même, supprimez toute la partie **requires**. Puis, ajoutez à cette partie le package **SJRDUUnits.dcp**. Faites de même pour la compilation que pour SJRDUUnits.

Finalement, répétez la marche à suivre pour **SJRDCompsD.dpk**. Un fois installé, vous pourrez utiliser les composants **SJRDComponents**.

IX - Évolutions du langage

Lors du passage d'anciennes applications vers Delphi 2005, vous remarquerez probablement beaucoup d'erreurs. La plupart sont dues aux différentes évolutions du langage dans Delphi 2005.

De plus, l'usage de certains types de variables comme les pointeurs est proscrit par .NET à cause du nouveau système de gestion de la mémoire.

[Le langage Delphi pour .NET](#) par [Laurent Dardenne](#)

Remerciements

Merci à **Anomaly** pour la correction orthographique et à **Laurent Dardenne** pour ses précieuses informations.