DELPHI

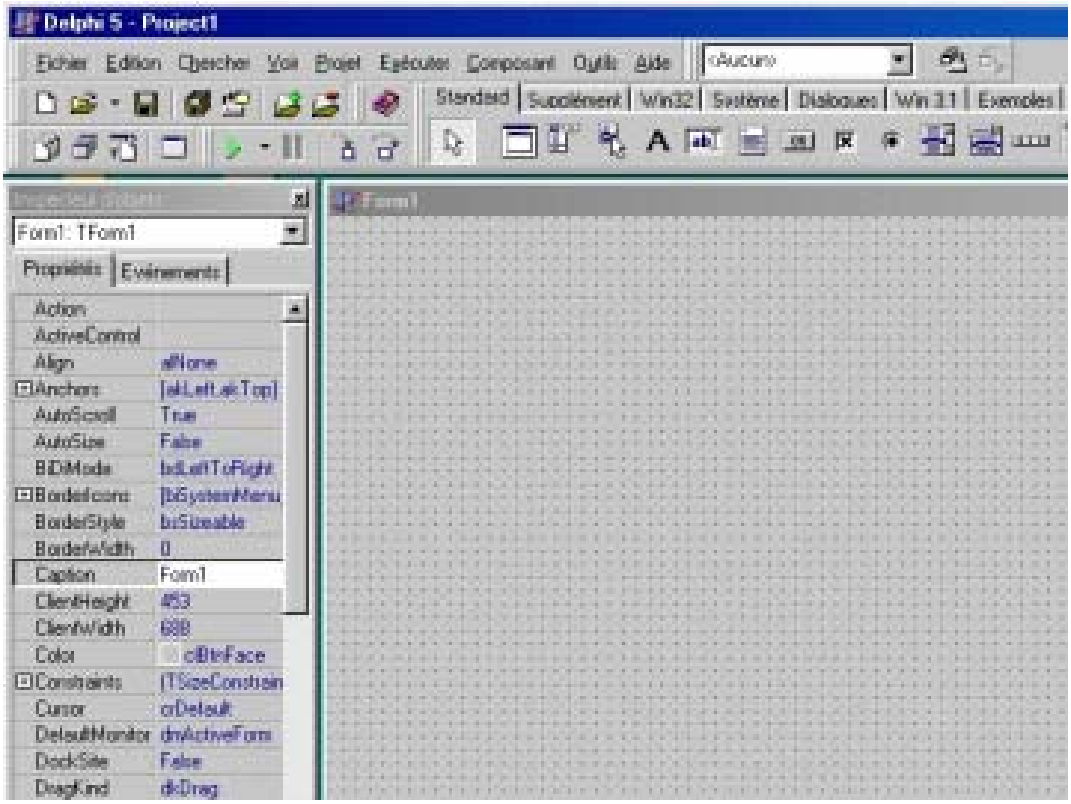
Introduction à DELPHI

Nous ne décrirons pas la procédure d'installation de DELPHI à partir du CD ,elle ne présente pas de difficulté, il suffit de suivre les indications apparaissant à l'écran. Les exemples suivants ont été écrits sous DELPHI 5 ,ils sont compatibles avec les autres versions du logiciel (au-delà de la version 3 sans doute). Actuellement BORLAND commercialise la version 7 et vient de sortir une version 8 (Février 2004) .La version 6 est téléchargeable gratuitement sur le web (plus de 100Mo) et fournie CD avec certains ouvrages.

On peut trouver sur le Web de nombreux cours sur Delphi par exemple :
http://delphi.developpez.com
http://delphipage.free.fr
parmi les cours j'ai particulièrement apprécié celui de J-C Armici www.unvrai.com/cours.php
,

L'Environnement DELPHI

La figure ci-dessous représente l'écran tel qu'il apparaît au lancement du programme .



La partie supérieure de l'écran donne accès au système de menus et à la barre d'outils.
Sur la première ligne :
Fichiers permet d'ouvrir un nouveau fichier, un nouveau projet, d'enregistrer votre travail et d'imprimer.
Edition donne accès aux fonctions copier coller classiques ainsi qu'a des outils de présentation.
Exécuter permet de lancer l'exécution d'un programme

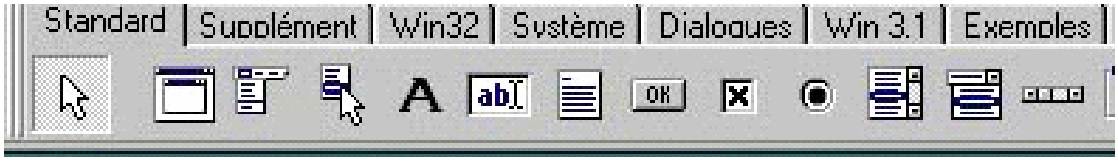
Les différents fichiers nécessaires à la mise au point d'un programme , la plupart d'entre eux crées automatiquement par DELPHI forment l'ensemble d'un **PROJET.**

La feuille de travail présente à l'écran est le rectangle situé en bas à droite(un projet peut comporter plusieurs feuilles de ce type , on parle de FICHES). La fiche présentée à l'écran porte le nom initial **Form1** .

Les fenêtres qui vont être placées à l'écran font partie de ce que l'on appelle des OBJETS
Il y a de nombreux objets définis dans DELPHI , l'utilisateur confirmé peut même en écrire d'autres , mais c'est une autre histoire ! Ces objets sont graphiques ou non .

Chaque objet possède un certain nombre de caractéristiques et peut effectuer diverses tâches qui caractérisent le **type** auquel il appartient. Par exemple ou pourrait définir l'objet CARRE de type carre (on écrit TCARRE) de propriétés longueur, largeur, couleur, visibilité (oui ou non) Tout objet est construit à partir d'objets plus simples dont il hérite des propriétés, par exemple l'objet PARQUET est un ensemble de rectangles dont le type est TPARQUET = TCARRE + nombre, position etc.. Dolphi possède d'origine un grand nombre d'objets prédéfinis, ils sont plus ou moins nombreux suivant la version commerciale du logiciel (standard ou pro) que vous avec acheté. Mais il est toujours possible d'en rajouter.

Pour accéder à ces objets regardons la seconde ligne de l'écran .En cliquant sur le premier bouton **Standard** nous voyons apparaître une première liste



Parmi tous ces objets nous retiendrons surtout

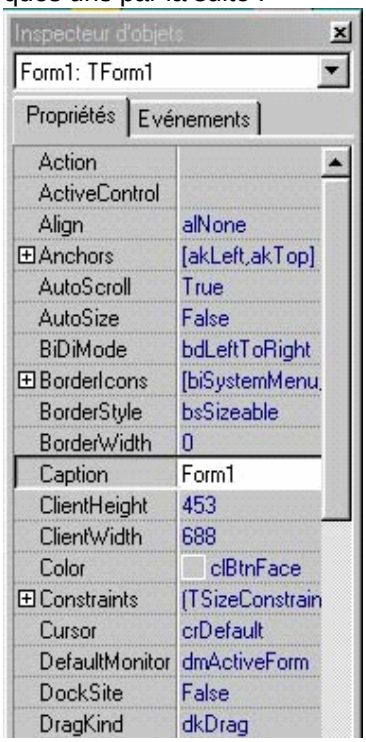
A Définit à l'écran une zone dans lequel on peut écrire une ligne de texte .Reconnu dans DELPHI sous le nom de **Label**

ab est une zone de saisie , c'est dans cette fenêtre que nous introduirons les données demandées par le programme nombres ou caractères. Reconnu dans DELPHI sous le nom de **Edit**

ok est un bouton de commande sur lequel il faudra cliquer avec la souris , sous DELPHI **Button**

Nous utiliserons surtout ces 3 outils . Les boutons supplémentaires de la seconde ligne:

Supplement Win32 Système etc donnent accès à d'autres objets , nous en rencontrerons quelques uns par la suite .



Chaque objet possède des **propriétés** qui lui sont propres , taille , fonte utilisée pour le texte, nature du texte , nom etc.. Ces propriétés sont définies grâce aux commandes apparaissant dans la partie gauche de l'écran .

Nous citerons

Caption (en français légende) c'est le texte qui apparaîtra dans l'objet considéré

Name c'est le nom de l'objet utilisé par le logiciel

Color la couleur du fond

Les propriétés proposées dans la colonne de gauche dépendent de l'objet qui a été sélectionné .

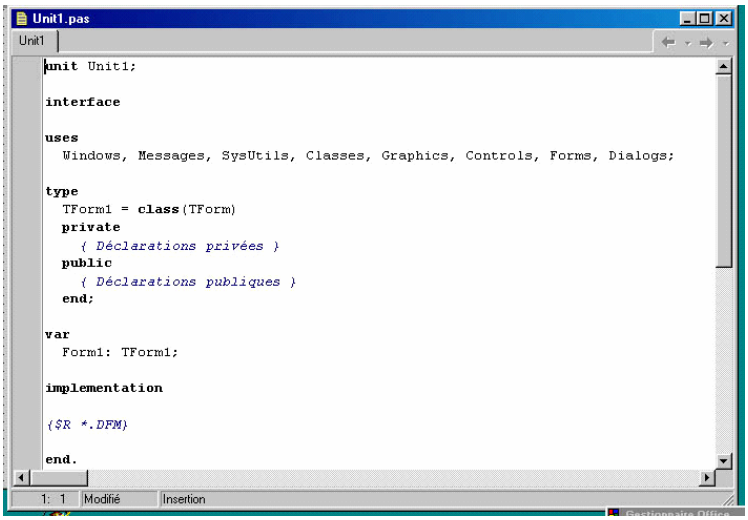
Le bouton **evenements** permet pour chaque objet de définir des circonstances le concernant, par exemple pour un bouton le déclenchement s'effectuera à l'enfoncement du bouton, son relâchement , par un simple clik dessus ou un double .

Premier PROJET

Il est très simple mais illustre bien le mécanisme de DELPHI .
Nous nous proposons d'écrire un programme de multiplication par 5 .L'écran sous Windows sera constitué de deux zones de texte et d'un bouton.
La première est la zone de saisie dans laquelle nous allons entrer un nombre entier
Dans la seconde apparaît le résultat obtenu en cliquant sur le bouton .
Ces trois objets seront placés sur un fond vert.

Si l'écran d'entrée n'est pas celui décrit plus haut y revenir par **Fichier** ⇒ Nouveau une fenêtre s'ouvre, sélectionner ⇒**application**

Avant de faire quoique ce soit regardons la fiche principale dont le nom initial est **Form1**.On peut apercevoir sous elle une seconde fenêtre que l'on peut faire passer au premier plan en cliquant dessus (éventuellement déplacer Form1 pour l'apercevoir) Cette seconde fenêtre a pour titre Unit1.pas elle présente une version initiale du logiciel associé à Form1.



D'abord le titre Unit1,
le mot interface puis **uses** suivi des utilitaires standard qui sont chargés à la compilation , comme dans un programme en Pascal 5 et au-delà.
Ensuite **Type** , zone ou seront présentés les objets , variables , procédures et fonctions nécessaires au programme en précisant à chaque fois leur type. Pour le moment il n'apparaît que TForm1=class(Tform) ce qui veut dire que la fiche Tform1 est du type TForm .Derrière **Private** seront définis les éléments qui ne sont reconnus que dans la fiche Form1,derrière **Public** ceux qui seront accessibles dans la totalité du programme même s'il est formé de plusieurs fiches.
Enfin après i**mplantation** le détail du programme écrit en Pascal Objet. Jusqu'au **end** final.

Sauvetage du travail

Avant de commencer nous allons définir les noms que nous allons affecter à notre projet, il est possible de le faire bien plus tard mais c'est une bonne habitude de le faire au début
Dans la barre du haut cliquer sur **Fichier** puis **Enregistrer sous :**
Une fenêtre s'ouvre et vous propose un nom , souvent unit1.Vous pouvez le modifier à volonté par exemple ***essai1***, au cours du travail Delphi va créer une série de fichiers portant ce nom; ***essai1.pas*** (c'est le texte du programme en Pascal Objet), ***essai1.dpr*** , ***essai1.dfm*** .
Valider par enregistrer .
Une seconde fenêtre s'ouvre alors (la première fois seulement , par la suite il faudra l'ouvrir par **Fichier** - **Enregistrer projet sous**
Le nom proposé est projet1 , il vaut mieux choisir un nom proche du précédent (***ce ne peut pas être même***) pour faciliter sa recherche ultérieure . Par exemple **pr_essai1**
Par la suite les fichiers de travail seront automatiquement créés avec ces noms en cliquant sur **Fichiers -Tout Enregistrer**

Passons maintenant à la construction du projet.

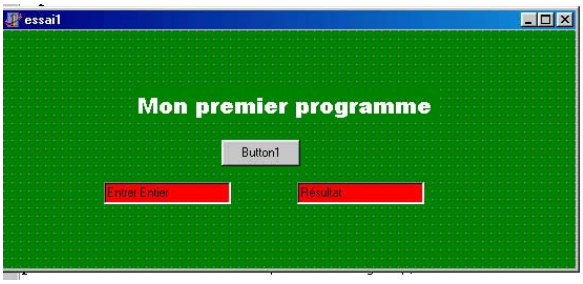
1° Définir le fond d'écran

Nous allons d'abord définir les propriétés de la fenêtre principale dont le titre initial est Form1
Dans la fenêtre de gauche colonne **propriétés** cliquer sur **caption** et dans la case de droite correspondante introduire le nom **essai1**

Puis plus bas sélectionner **color** , en cliquant à droite apparaît une table des couleurs; choisir vert foncé **clGreen .** Cette couleur est celle qui apparaîtra à l'écran au lancement du programme , elle sera éventuellement modifiée par le programme en modifiant la propriété **caption** de la fiche principale qui s'appelle toujours Form1 car nous n'avons pas changé son nom

Il est prudent de sauver le travail à chaque étape par **Enregistrer tout**

2° Insertion des objets
Le but est d'obtenir l'écran ci-dessous .



En haut de l'écran nous placerons **un titre** *Mon Premier Programme* , c'est un texte inclus dans une zone d'affichage définie à l'origine par DELPHI sous le nom **Label .** Pour cela :

Cliquons sur le bouton **A** de la liste des objets La lettre A s'entoure d'un carré gris pour indiquer que la commande est sélectionnée.

Amenez la souris à l'endroit ou vous voulez placer le texte et tirez vers le bas à droite en maintenant le bouton gauche enfoncé. Une zone rectangulaire est ainsi définie limitée par 8 plots noirs et un titre provisoire Label1 L'objet Label1 étant sélectionnée la fenêtre de l'inspecteur d'objet à gauche concerne cet objet. Dans la colonne **propriétés** à gauche introduisons alors:

Caption = ***Mon premier programme*** le mot label1 est remplacé par ce nouveau texte
(cliquer sur le bouton caption puis sur la fenêtre à sa droite et y introduire le texte)

Les caractères sont trop petits changeons la fonte

Font = ***Arial Black Normal Taille 16 couleur Blanche***
(cliquer sur Font , puis sur les trois points dans la case de droite .Une fenêtre de dialogue s'ouvre permettant de choisir la fonte et ses caractéristiques)

Pour le moment ce texte est sur fond vert comme la première fiche , pour supprimer tout cadre définissons (en bas de la liste des **propriétés**)

Transparent = ***True*** (Vrai) Le texte est alors blanc sur fond vert .

En cliquant sur le titre il est possible de le déplacer à volonté avec la souris.

Il faut noter que le nom sous lequel la zone de titre est reconnu par le programme est toujours Label1 car la propriété **Name** n'a pas été modifiée .

Name = Label1 Il est bien sûr possible de le modifier mais laissons le par commodité.
Passons maintenant aux deux fenêtres de saisie et de résultats.

Fenêtre de saisie (Edit)

Activez la bouton de saisie **ab** et comme plus haut avec la souris placer un rectangle de saisie sous le titre précédent.A cette fenêtre ainsi ouverte est attribués par défaut le nom Edit1
Il est possible de modifier sa taille en agissant sur les plots qui l'entourent. .Cliquez sur la fenêtre de gauche Edit1 et dans l'inspecteur d'octets modifier

Color = ***clRed*** (fenêtre rouge)
(cliquer sur le mot color puis sur la case de droite, une fenêtre s'ouvre présentant les couleurs au choix)

Font = ***Arial Black Normal 12 Blanc***
(comme plus haut)

Text = ***Entrer entier***

Fenêtre de résultat.

Nous utiliserons une fenêtre Edit ,(il est possible aussi de placer le résultat dans une fenêtre d'affichage Label). Activons donc le bouton de saisie **ab** comme plus haut et mettons le en place sur la fiche. Puis dans la fenêtre de l'inspecteur d'objets:

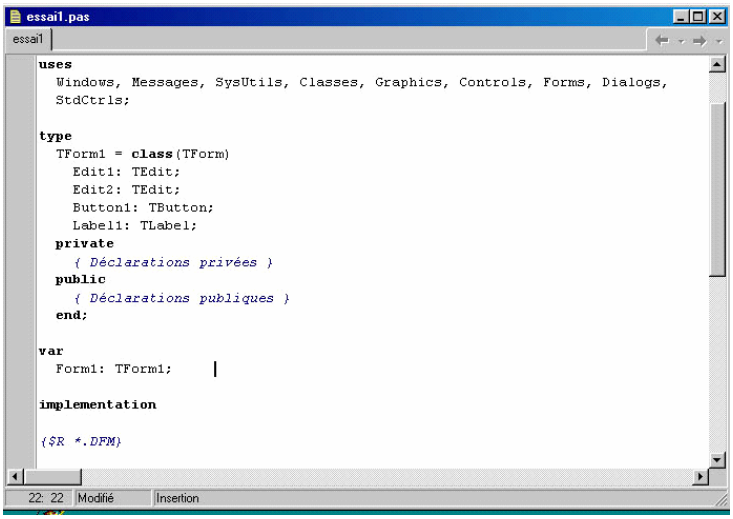
Font = Arial Black Normal 12 Jaune
Color = clRed (rouge)
Text= Résultat;

Comme plus haut il vaut mieux ne pas modifier le nom donné automatiquement par DELPHI , soit Edit2 (mais il est possible de le faire)

Enfin pour placer le bouton activer **OK** et avec la souris comme plus haut , le placer au centre de l'écran
Les propriétés accessibles sont :

Caption = calculer
Font = Arail black Normal 12 Jaune

Pour cet objet la couleur de fond n'est pas modifiable, il existe d'autres boutons qui le permettent, nous les rencontrerons plus loin.



Pour améliorer la présentation il est possible de modifier la taille de la fiche principale de façon que les objets précédents soient centrés. Il suffit d'utiliser la souris en déplaçant les coins comme dans Windows. Ce cadrage se conservera dans le programme final en sélectionnant la propriété **Align AlNone**

Il est prudent à ce niveau de sauver le travail par **Fichier ⇒ Tout enregistrer**

Revenons sur le fond d'écran en cliquant en dehors des objets précédents.. Cliquer sur la fenêtre sous jacente. pour l'activer.
Le texteà l'écran est maintenant est le programme en PASCAL OBJET associé au projet dans son état actuel . Dans le paragraphe Type les objets que nous avons implantés sont apparus /

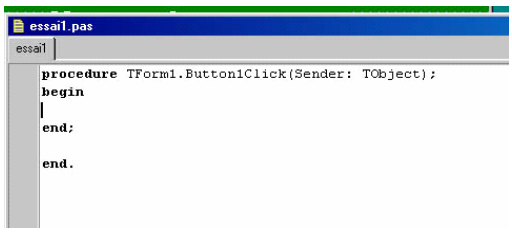
```
TForm1 = class(TForm)
  Label1: TLabel;          Label1 est un objet de type TLabel
  Edit1: TEdit;
  Edit2: TEdit;
  Button1: TButton;
.
```

Ecriture du programme .

Il se réduit à écrire les routines associées à chaque objet. Ici seul le Bouton commande une action.;
Cliquons 2 fois sur le bouton , une fenêtre présente la forme initiale du programme qui lui est associé
A nous d'écrire le texte entre les mots Begin et End;

Les procédures et fonctions que vous introduisez doivent être écrites en PASCAL, plus exactement en PASCAL OBJET , cependant (comme pour le C et C++) le PASCAL classique est presque toujours compatible , avec évidemment les limites liées au mode d'affichage. Les instructions

de commande d'écran et de saisie **ClrScr , Writeln, Write, Readln, Read** sont évidemment interdites. Il en est de même bien sûr des fonctions graphiques valables sous DOS. De nombreuses instructions supplémentaires existent , nous en rencontrerons quelques unes dans la suite.



Dans le cas présent l'opération à réaliser est une multiplication par 5 , nous introduirons donc la fonction Cinqfois définie par les lignes ci-dessous inclues derrière la directive de compilation ($R*DFM)

```
Function Cinqfois(a:integer):Integer ;
Begin
Cinqfois:=a*5;
End;
```

C'est du PASCAL le plus classique

Le nombre à multiplier par 5 est écrit dans la fenêtre de gauche qui sera accessible comme la propriété **text** de l'objet Edit1 .Le résultat obtenu en injectant cette valeur dans la fonction CinqFois est affiché dans la fenêtre de droite , c'est-à-dire la propriété **text** de Edit2.

Attention le contenu des fenêtres à l'écran est de type chaîne (String) et des conversions Nombre⇔chaîne seront nécessaires .

Lorsque l'on clique sur le bouton les opérations suivantes doivent donc être effectuées :

- Lire la données c'est-à-dire la valeur text de Edit1
- C'est un texte qu'il faut transformer en un entier (commande strTOint)
- Appliquer la valeur de cet entier à la fonction Cinqfois
- Transformer le nombre résultat en une chaîne de caractères pour l'affichage
- Envoyer le résultat dans la propriété Text de Edit2

Complétons donc la routine associée à Button1 par (en caractéres gras ci-dessous)

```
procedure TForm1.Button1Click(Sender: TObject);
        var  a ,b :integer;
begin
        a:= StrToInt(Edit1.text);
        b:=Cinqfois(a);
        Edit2.Text :=InfToStr(b);
end;
```

*Remarquez le nom de la procédure associée au bouton : **TForm1** pour indiquer qu'il s'agit d'une procédure incluse dans Form1, **Button1** le nom du bouton , **Click** car l'évènement déclenchant est un click sur le bouton.(en ouvrant la colonne de droite de l'inspecteur d'objets associé à Button1 ; **Evénements** on verrait que l'événement déclenchant est **On Click= Button1Click**) Dans la parenthèse **Sender** (expéditeur) indique que cette procédure hérite des propriétés de la classe TObject Si les noms Tform1 et Button1 avaient été changés ce sont les nouveaux qui seraient apparus dans le nom de la procédure.*

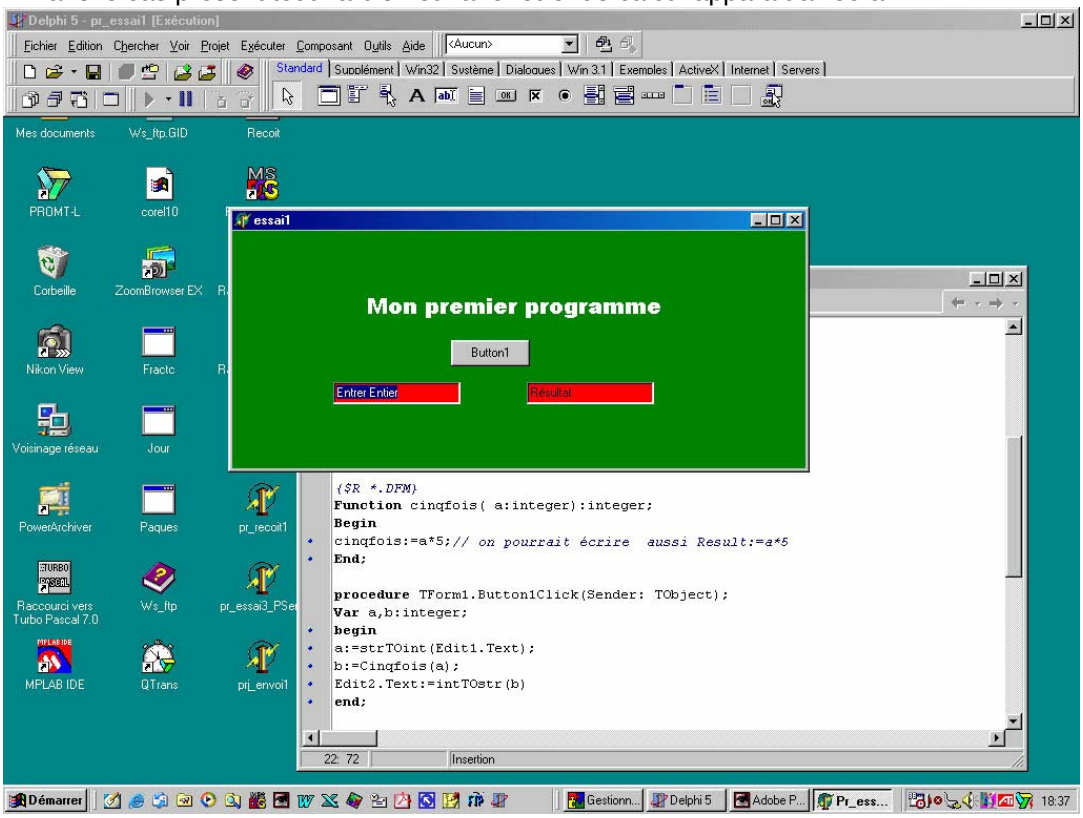
Après la définition de deux variables locales a et b , on trouve l'instruction StrToInt (Edit1.text),c'est une instruction du PASCAL OBJET qui transforme une chaîne de caractères STRING en entier (String TO Int) .A la ligne suivante l'entier **a** obtenu est introduit dans la fonction CinqFois .La troisième ligne donne à la propriété Text de Edit2 la valeur (ce doit être une chaîne de caractères) obtenue en convertissant l'entier b en une chaîne (instruction Int TO String

.. Il n'était pas nécessaire de définir une fonction Cinqfois, la multiplication par 5 aurait pu être effectuée directement dans la routine associée au bouton mais nous avons choisi de définir une fonction car cette méthode est plus générale.

Sauver de nouveau le travail par **ToutEnregistrer** .

Pour vérifier qu'aucune erreur n'a été commise nous allons lancer l'exécution par la commande Executer dans le bandeau en haut de l'écran. Toute erreur sera signalée avec des indications (parfois succinctes) pour sa localisation et correction.

Dans le cas présent tout va bien et la fenêtre de calcul apparaît à l'écran.



Il suffit avec la souris d'effacer dans la fenêtre de gauche le texte Entrer Nombre et le remplacer par 45 (par exemple) puis de cliquer sur le bouton. Le résultat 225 apparaît dans la fenêtre de droite.

Pour revenir à la fiche cliquer sur le X de fermeture.

Le projet fonctionne mais il reste à effectuer la compilation pour créer le fichier exécutable .exe .Pour cela **Projet ? Compiler Projet** (ou Ctrl F9)

L'explorateur Windows permet de visualiser les différents fichiers crées :

Essai1.dfm
Essai1.dpr
Essai1.pas
Essai1.dcu
Pr_Essai1.res
Pr _Essai1. cfg
Pr_Essai1.dof
Pr_Essai1.exe // C'est l'exécutable.On peut ne garder que lui mais aucune modification ne sera plus possible.

Remarques sur ce premier programme

Ce programme nous a permis d'utiliser les objets les plus importants , Zone de Texte ((Label) de saisie (Edit) et boutons (button) .Nous avons vu comment en cliquant dessus on fait apparaître le logiciel PASCAL qui leur est associé. Le programme le plus important est celui qui est associé au bouton , on y trouve les fonctions et procédures nécessaires pour exécuter les tâches souhaitées , il peut lire ou modifier les propriétés des autres objets, caption, text, mais aussi les tailles couleurs ou visibilité. etc..

Nous avons noté que le contenu des fenêtres à l'écran est de type texte ce qui oblige à des conversions .Dans la liste des instructions du PASCAL OBJET cette conversion est assurée par les instructions suivantes.

StrToInt	de chaîne de caractère en entier
IntToStr	d'entier en chaîne de caractère
FloatToStr	Un nombre en virgule flottante est transcodé en chaîne de caractère
FloatTOsrtF	effectue la même action mais autorise un formatage précis du résultat
StrToFloat	l'inverse

Quelques objets importants

Outre les objets de base rencontrés dans l'exemple précédent il en existe quelques autres qui sont forts utiles .

Le Timer

Fait partie des objets Système..

Ce n'est pas un objet graphique, il apparaît sur la fiche comme une petite horloge mais ne figurera pas dans la présentation finale à l'écran . Cet objet est un compteur interne qui délivre à des intervalles réguliers un signal déclenchant une action choisie par le programmeur.

Dans l'Inspecteur d'Objets il n'y a que 4 propriétés:

Enabled // Si True (vrai) le compteur interne est lancé sinon il est inhibé.
Intervalle **//**C'est en millisecondes le temps qui s'écoule entre deux événements déclenchants
Name //comme d'habitude le Nom par défaut Timern (n=1 2 3…)
Tag // est simplement un entier associé au Timer à la disposition du programmeur.

Le seul événement proposé est **OnTimer** c'est l'impulsion déclenchante citée plus haut.

A titre d'exemple le programme suivant met en œuvre 2 boutons permettant de lancer un compteur à une vitesse définie dans la fenêtre Edit1. L'intervalle est introduit dans Edit1 et en pressant le bouton les nombres successifs défilent dans Label1 .

Les objets sont sélectionnés et placés sur la fiche comme plus haut;.On définira :

Timer1 **Enabled** False (il n'est pas lancé au départ)
Intervalle 100 (Dix chiffres par seconde par défaut)
Label1 (ou seront affichés les nombres) **color** clYellow (jaune)
Label1.Caption ' ' // effacement
AutoSize False // pour que la fenêtre reste de largeur constante
Label2 (le commentaire intervalle) caption intervalle
Font Arial Black Noir 12

Button1
Caption Lancer
Button2
Caption stop
Edit1 (ou sera introduite la valeur de la période)
Text : 100; // valeur par défaut 100mS . Si on avait placé au début un texte, par exemple Intervalle il y aurait un risque d'erreur si l'utilisateur appuyait sur le bouton avant d'y entrer un entier.

Un entier a est déclaré dans la zone Private .
Le programme associé à Button1 Entre Begin et end

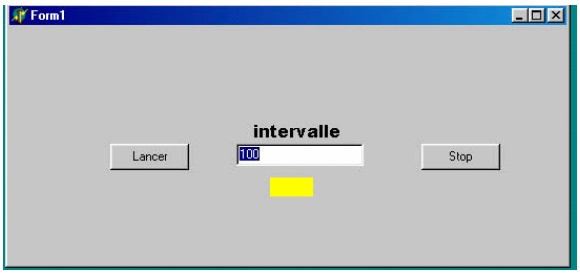
```
a:=0                  // initialisation de a
 Timer1.Enabled:=True; // le timer est lancé)
Timer1.Interval:= strTOint(Edit1.Text) ; // lecture de l'intervalle
```

Le programme associé à Button2 , entre Begin et End;

```
Timer1.Enabled:=False          // stop timer
Label1.Caption:='  '           // effacement du contenu de Label1.
```

Le programme associé au timer

```
a:=a+1;           // à chaque événement a est  incrémenté
Label1.Caption:=intTOstr(a);     // affichage du nombre
```



Ce projet est sauvé sous les noms de Horloge et pr_Horloge .

Le programme:

```
unit Horloge;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls;

type
  TForm1 = class(TForm)
    Timer1: TTimer;
    Edit1: TEdit;
    Button1: TButton;
    Button2: TButton;
    Label1: TLabel;
    Label2: TLabel;
    procedure Timer1Timer(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Déclarations privées }
    a:integer;
  public
    { Déclarations publiques }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

procedure TForm1.Timer1Timer(Sender: TObject);
```

```
begin
a:=a+1;
Label1.Caption:= intTOstr(a)
end;

procedure TForm1.Button1Click(Sender: TObject);
var i:integer;
begin
a:=0;
i:=strTOint(Edit1.Text);
Timer1.interval:=i;
Timer1.enabled:=True;
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
Timer1.enabled:=False;
Label1.caption:=' ';
end;

end.
```

Le Timer peut être utilisé pour réaliser un retard unique par exemple :

```
procedure TForm1.Button1Click(Sender: TObject);
var i:integer;
begin
edit1.color:=clTeal;      // lorsque l'on presse le bouton Edit1 devient verte  et le
                          //Timer :est activé
Timer1.Enabled:=true;

end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
edit1.color:=clred;               // à l'evênement Timer suivant  EDIT1 passe au
                                  //rouge et le Timer est stoppé .
Timer1.Enabled:=False;
end;
```

La ListBox 

C'est une extension de Edit à plusieurs lignes. Elle est disponible parmi les objets standards

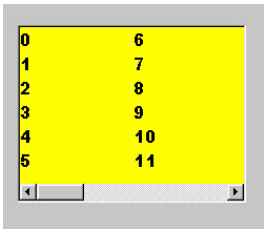
Ses propriétés:

Color la couleur de fond
Font la fonte des caractères comme pour Edit ou Label
Clear pour l'effacer
Column le nombre de colonnes
Mais surtout la procedure **Items.Add** qui ajoute une ligne après la dernière ligne présente Ce doit être une chaîne de caractères :

```
Liste1.Items.Add (' ceci est une nouvelle ligne ');
```

Par exemple avec une ListBox et un bouton En choisissant column=2 l'affichage des nombres de 0 à 20 s'obtient avec :

```
Procedure TForm1.Button1Click(Sender: TObject);
Var i:integer;
Begin
        ListBox1(Clear);                    // effacement préalable
        For I;=0 to 20 do
                ListBox1.Items.Add(intTOstr(i));
End;
```



Compte tenu de la taille de la fenêtre tracée au départ et de la fonte choisie, les nombres de 0 à 20 ne tiennent pas sur 2 colonnes , apparaît alors un curseur permettant un balayage horizontal, il n'y a toujours que 2 colonnes dans la fenêtre.

Si column est mis à 0 il y a une seule colonne avec curseur vertical, pour column=1 le curseur est horizontal .

Bouton BitMap BitBtn

C'est un objet de **supplément** Il joue le même rôle que l'objet bouton mais peut contenir une image . Les propriétés essentielles sont:
Caption //Le texte apparaissant sur le bouton
Height et **Width** //Hauteur et largeur du bouton en pixels
Left et **Top** //position du bouton sur la fiche l'origine étant en haut à gauche .
Mais surtout
Glyph //Une fenêtre s'ouvre demandant la position de l'image à utiliser. Attention cette image doit avoir une taille compatible avec les paramètres précédents, si elle est plus grande elle sera tronquée.Seuls les fichiers .bmp sont admis.
Visible // vrai ou faux visible ou non /

Image

Objet de supplément comme le précédent.

Permet d'insérer une image qui peut être une image de fond ou localisée. Lorsqu'on l'amène sur la fiche elle se présente sous forme d'un rectangle de dimensions réglables En cliquant dessus l'Editeur d'image s'ouvre et permet d'aller chercher dans la mémoire de l'ordinateur l'image souhaitée. Ses dimensions doivent être compatibles avec celles du rectangle de départ dont la taille et la position sont déterminès par les mêmes paramètres que pour le bouton bitmap précédent. Comme dans ce cas elle peut être rendue visible ou non .

Il y a un grand nombre d'autres objets que vous découvrirez en consultant l'aide de Delphi, par exemple la **StringGrid** qui est une grille de nxp cases pouvant contenir chacune une chaîne .Sa manipulation est assez complexe , voir l'aide .

Notes sur les instructions du Pascal Objet.

Il ne s'agit pas ici de faire un cours de Pascal Objet , consulter à ce propos le document fourni avec le CD du programme ou encore des cours complets disponibles sur le WEB .Je vous conseille en particulier le cours de J-C Armici www.unvrai.com/cours.php très bien fait et accompagné de nombreux exercices avec leur correction.

Procedures

Vous serez nécessairement conduits à écrire des procédures ou des fonctions adaptées à votre problème Elles seront placées après l'entête **implementation** ..Dans le premier exemple nous avons ainsi écrit la fonction CinqFois .Nous aurions pu définir une procédure réalisant le même travail :

```
Procedure  Cinqfois(a:integer; var b:integer);
    Begin
        b:=a*5;
```

End;

Mais s'il vous était venu à l'esprit d'utiliser la même procédure pour afficher le résultat vous auriez écrit :

```
Procedure  Cinqfois(a:integer; var b:integer) ;
   Begin
      b:=a*5;
      Edit1.Text:=intTOstr(b);
   End;
```

Et le compilateur aurait affiché une erreur , en effet la procédure Cinqfois est complètement indépendante de la fiche Form1 et de ses composants, elle ignore l'existence de Edit1 .

Pour qu'une procédure puisse utiliser les objets de Form1 il faut :

D'abord faire une annonce anticipée (prototype de la procédure) au début du programme dans les zones private ou public.

Puis dans l'écriture proprement dite, faire précéder le titre du terme Form1 pour indiquer qu'elle appartient à cette fiche.

Il faut noter que dans l'appel de la procédure le préfixe Form1 n'est pas mentionné.

Exemple:

Soit une procédure permettant de calculer la moyenne quadratique de deux nombres a et b

$m=\sqrt{(a^2+b^2)/2}$

Les deux nombres a et b sont entrés dans deux fenêtres de saisie edit1 et edit2 , un bouton déclenche le calcul , le résultat est visualisé dans edit3. .Le transfert du résultat dans edit3 est inclus à l'intérieur de la procédure Moyenne .

Le programme est le suivant :

```
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs,
  StdCtrls;

type
  TForm1 = class(TForm)
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Button1: TButton;
  Procedure Moyenne(a,b:integer); // annonce anticipée
    procedure Button1Click(Sender: TObject);
  private
    { Déclarations privées }
  public
    { Déclarations publiques }
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}
Procedure Tform1.Moyenne(a,b:integer);  // description de la procédure avec entête
                                        //TForm1
var m:real;             // la variable m est interne (locale ) à la procédure Moyenne
                        //elle n'est pas visible de l'extérieur.
Begin
   m:=sqrt((a*a+b*b)/2);
   edit3.text:=floatTostr(m);           // affichage du résultat
```

End;

```
procedure TForm1.Button1Click(Sender: TObject);
var a,b:integer;
m:real;
begin
 a:=strTOint(edit1.text);
 b:=strTOint(edit2.text);
Moyenne(a,b);        // appel de la procédure sans  l'entête TForm1
end;

end.
```