Exercices sur les composants graphiques de Lazarus

Auteur: E. Thirion

Ce document est extrait du site cours.thirion.free.fr

Dernière mise à jour : 24/02/2019

Ce document fait partie d'un ensemble de cours du même auteur (programmation procédurale et objet, programmation web, bases de données) auxquels vous pouvez accédez en cliquant ici.

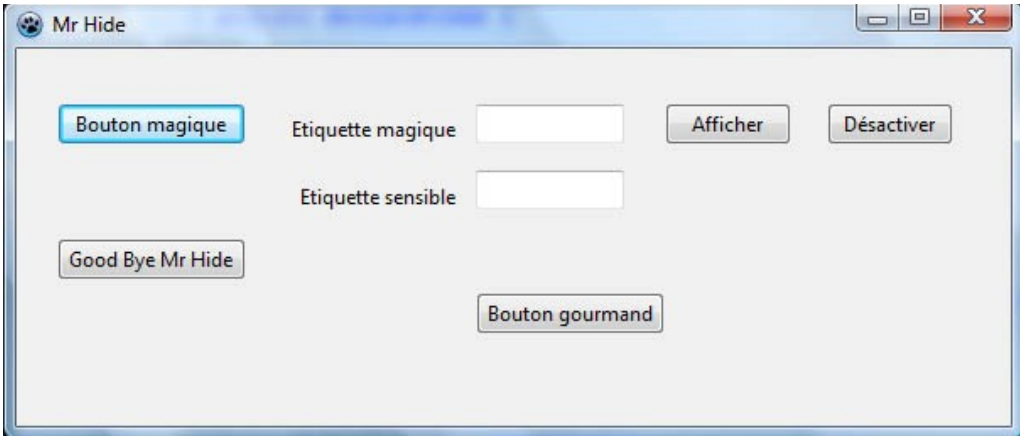
Exercice 1 : Mr Hide

Parties du cours utilisées: Héritage - composants abstraits, propriétés générales des composants héritant de **TControl** (attributs **enabled**, **width**, **height**, **hint**, **ShowHint** et méthodes **show**, **hide**) - attribut **Name** de la classe **TComponent**, methode **ClassName** de la classe **TObject**. Evènements **Click**, **DblClick**, **EditingDone**, **Create**, **Resize**, **CloseQuery**.

Projet à ouvrir : aucun

Librairie à importer : pour afficher ou lire du texte, vous aurez besoin des procédures **Lire** et **Afficher** de la librairie **entrees_sorties.pas.** Il faudra donc l'importer dans le projet.

Formulaire (à réaliser de A à Z)



Noms interne des composants (dans le sens de la lecture):

- Boutons : BT_Magique (bouton magique, BT_Afficher (bouton afficher), BT_Desact (bouton Désactiver), BT_Quit (bouton Good Bye Mr Hide), BT_Gourmand (bouton gourmand)
- Etiquettes : ET_Magique (étiquette magique), ET_Sensible (étiquette sensible)
- Zones de textes : ZT_Magique (zone de texte magique), ZT_Sensible (zone de texte sensible).

Question 1: Faire disparaitre et réapparaitre des composants

Ecrivez les gestionnaires d'évènement permettant de :

- faire disparaitre le bouton magique et l'étiquette magique lorsque l'on clique dessus.
- faire disparaitre la zone de texte magique lorsqu'on double-clique à l'intérieure de celle-ci.
- faire réapparaitre les trois composants précédents lorsque l'on clique sur le bouton Afficher.

Question 2 : Composants sensibles

Ecrivez les gestionnaires d'évènement permettant:

- d'afficher le message 'Aïe !' lorsque l'on clique dans la zone de texte sensible (à l'aide de **showmessage**).
- d'afficher 'Ouille ' dans la zone de texte sensible, lorsque l'on clique sur l'étiquette sensible.

Question 3 : Désactivation de composants

Ecrivez les gestionnaires d'évènement permettant de:

- désactiver le bouton magique et la zone de texte magique lorsque l'on clique sur le bouton **Désactiver**.
- désactiver le formulaire lorsque l'on double clique dessus.

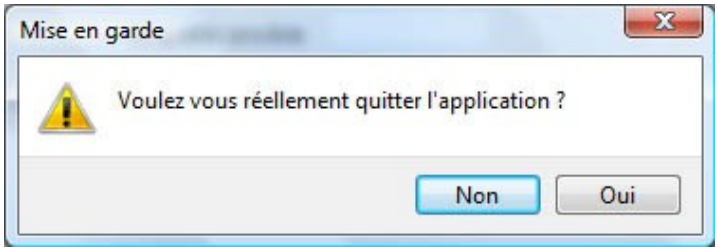
Question 4 : Modification de la dimension d'un composant

Ecrivez le gestionnaire d'évènement permettant d'agrandir le bouton gourmand lorsque l'on clique dessus (on l'agrandira de 5 pixels en largeur et en hauteur).

Question 5 : Arrêt du programme

Ecrivez les gestionnaires d'évènement permettant de:

- quitter l'application lorsque l'on clique sur le bouton **Good Bye Mr Hide**
- d'afficher la boite de dialogue suivante pour confirmer cette action:



Indication: pour afficher cette boite de dialogue il vous faudra utiliser la fonction **Confirmation** de la librairie **entrees_sorties.pas**. Voir l'exemple du cours dans la partie évènement **CloseQuery** et également la documentation sur la fonction **Confirmation** dans la partie du cours concernant les boites de dialogue.

Question 6 : Utilisation de EditingDone

On souhaiterait que dès que l'utilisateur a 'fini de saisir' du texte dans la zone de texte magique (il appuie sur la touche **Entrée**, ou clique dans la zone de texte sensible), ce texte soit recopié dans la zone de texte sensible.

Ecrivez le gestionnaire de l'évènement **EditingDone** pour la zone de texte magique de manière à réaliser ceci.

Question 7 : Affichage des propriétés des composants, utilisation de l'héritage

Attibuts et méthodes héritées

Les classes des composants que nous utilisons dans cet exercice (**TEdit**: zone de texte, **TLabel**: etiquettes, **TButton**: bouton, **TForm**: formulaire) sont des sous-classes de **TControl**. Elles héritent donc toutes des attributs suivants:

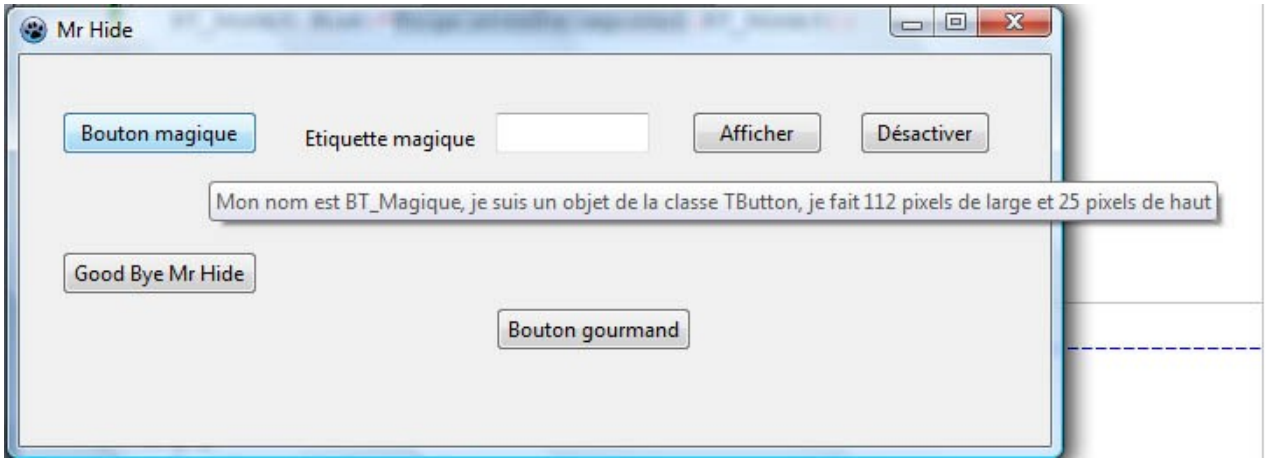
- **width** : largeur en pixel du composant
- **height**: hauteur en pixel du composant
- **hint** : texte s'affichant lorque lorsque l'on survole le composant si showhint vaut true.
- **showhint** : booléen indiquant s'il faut afficher le texte se trouvant dans hint lorsque l'on survole le composant.

Comme **TControl** est une sous classe de **TComponent**, ils héritent également de l'attribut **Name** contenant le nom du composant.

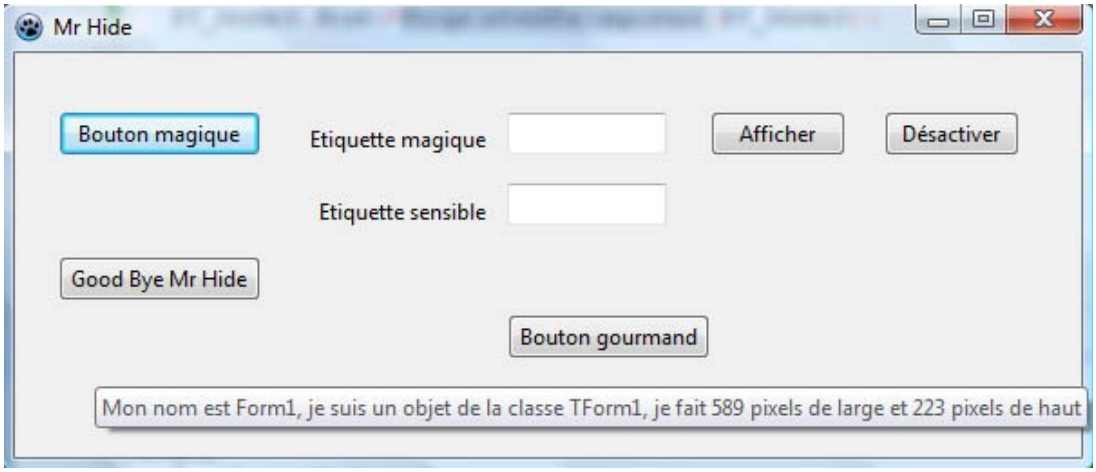
Et comme toute classe en pascal objet est une sous-classe de **TObject**, ils héritent également de la méthode **ClassName** qui retourne le nom de la classe à laquelle appartient un objet.

Objectif de l'exercice

On souhaiterait que lorsque l'on survole un composant quelconque de l'application, un texte s'affiche indiquant le nom de ce composant, à quelle classe il appartient ainsi que ses dimensions. Voici par exemple, ce qui devrait s'afficher lorsque vous survolez le bouton magique:



ou lorsque vous survolez le formulaire:



On pourrait bien sur saisir ce texte à la main pour chaque composant en utilisant l'inspecteur d'objet, mais il y a plus rapide. Comme tous les composants peuvent être considérés comme des objets de la classe **TControl**, nous allons réaliser ceci en écrivant une procédure prenant en paramètre un objet de cette classe.

Question 7- A : La procédure ProprietesDuComposant

Voici l'entête de cette procédure:

```
Procedure ProprietesDuComposant (x : TControl);
```

Effets de cette procédure :

- elle affecte la valeur true à **ShowHint**.
- elle affecte à **Hint**, le message qui doit s'afficher. Il faudra donc construire ce message à partir des dimensions du composant, de son nom et du nom de sa classe.

Question 7- B : Initialisation des Hints dans FormCreate

Afin que tous les messages s'affichent dès le démarrage du programme appliquez la procédure **ProprietesDuComposant** à tous les composants de l'application dans la procédure évènementielle **FormCreate**.

Question 7-C : Prise en compte du redimensionnement

Lorsque le formulaire est redimensionné, le message affiché n'est plus correcte car il fait toujours référence à ses dimensions initiales. Ecrivez le gestionnaire d'évènement permettant de mettre le message à jour dès que le formulaire est redimensionné.

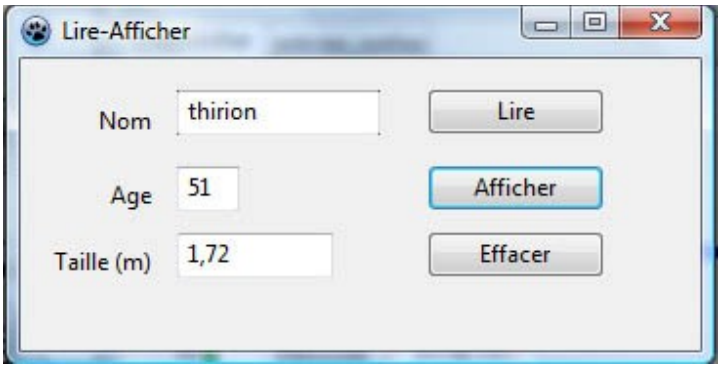
Le problème se pose également pour le bouton gourmand, puisqu'il grossit dès que l'on clique dessus ! Comment résoudre ce problème ?

Exercice 2 : Lire-Afficher

Parties du cours utilisées: composants standards, utilisation des zones de texte.

Projet à ouvrir : Exo-ProgObjet1/LireAfficher/ProjetLireAfficher.lpi

Formulaire



Objectif

Il s'agit de réaliser une nouvelle version du projet **LireAfficher**, que nous avions réalisé lors du premier cours de programmation (Premières Notions). Dans cette version, la librairie **entrees_sorties.pas** ne doit plus être utilisée. Interdiction donc d'utiliser les procédures **Afficher**, **AfficherEntier**, **AfficherNombre**, **Lire**, **LireEntier**, **LireNombre** et **Effacer**.

Indications utiles:

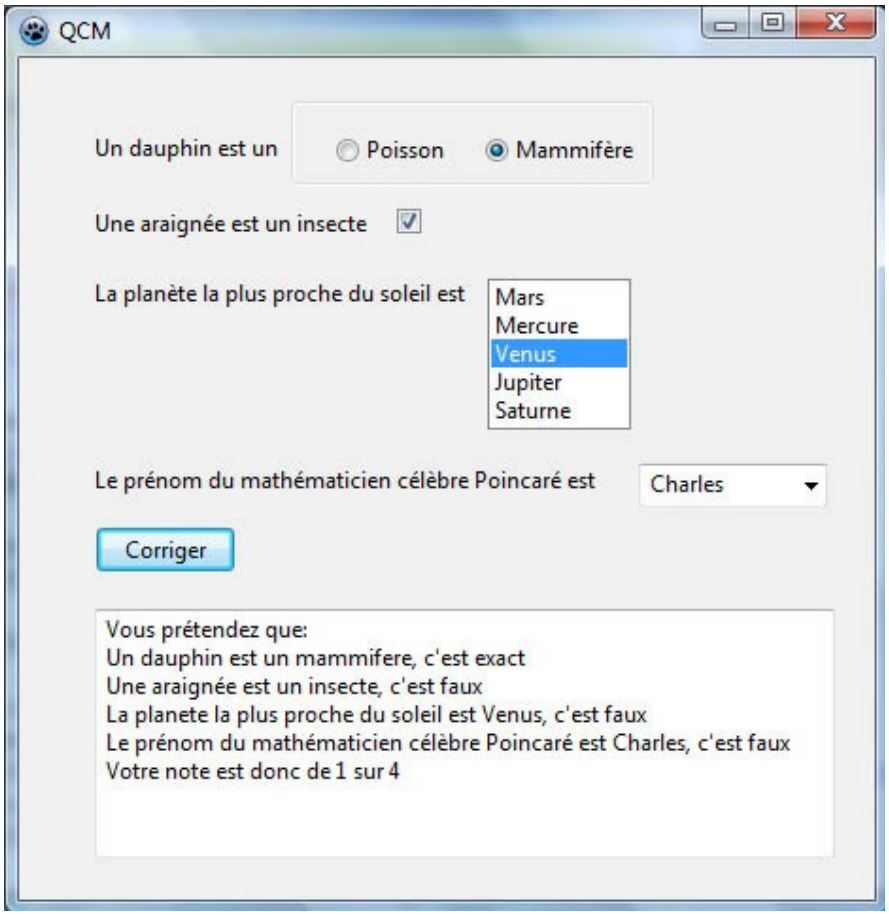
- **StrToInt (c)** convertit la chaine **c** en entier.
- **IntToStr (e)** convertit l'entier **e** en chaine de caractères.
- **StrToFloat (c)** convertit la chaine **c** en double.
- **FloatToStr (d)** convertit le double **d** en chaine de caractères.

Exercice 3 : QCM

Parties du cours utilisées: composants standards: boutons radio, group box, cases à cocher, zone de liste, combo box, mémo.

Projet à ouvrir : Il n'y en a pas !

Formulaire à réaliser



Objectif

Le formulaire à réaliser représente un QCM.

La réponse à la première question se fait via deux boutons radios contenus dans un group box. Pour la deuxième, une case à chocher. Pour la troisième, une zone de liste et pour la quatrième une combo box.

Lorsque l'utilisateur clique sur le bouton **Corriger**, un corrigé de ses réponses s'affiche dans un mémo.

On prendra garde à:

- Initialiser les choix dès le démarrage du programme.
- Afficher un corrigé très explicite (comme celui de l'exemple ci-dessus) précisant chaque réponse donnée par l'utilisateur et si elle est exacte ou fausse.

Les réponses exactes sont les suivantes:

- Le dauphin est un mammifère.
- L'araignée n'est pas un insecte.
- Planète la plus proche du soleil : mercure
- Henri Poincaré

Exercice 4 : Dictionnaire illustré

Parties du cours utilisées: le multi-fenetrage, les composants standards (menus, zones de liste, mémos) les boites de dialogues, quelques composants spéciaux (liste de fichiers, image), les évènements (resize, DblClick, EditingDone, CloseQuery).

Projet à ouvrir : Exo-ProgObjet1/Dictionnaire/Dictionnaire.lpi

Objectif

Il s'agit de réaliser un dictionnaire illustré interactif. Certains mots peuvent être illustrés par une image. Les mots représentant des concepts généraux (Insecte par exemple) peuvent être associés à des exemples (Mouche, Abeille, ...). Le dictionnaire peut être mis à jour par l'utilisateur: il peut ajouter un mot, modifier sa définition, ajouter ou supprimer des exemples, ajouter ou supprimer des images etc ...

Données du dictionnaire

Les données du dictionnaire sont contenus dans le répertoire **Fichiers** de l'application. Il contient des fichiers images (**.jpg** uniquement) et des fichiers texte d'extension **.def** ou .**exp**.

Chaque mot du dictionnaire est représenté par au plus trois fichiers:

- un fichier d'extension **.def** qui contient sa définition
- un fichier d'extension **.exp** qui contient des exemples
- un fichier d'extension **.jpg** qui contient une image.

Les noms de ces fichiers sont constitués du mot et de l'extension.

Par exemple, pour le mot **Abeille** il y a deux fichiers **Abeille.def** et **Abeille.jpg** . Le fichier **Abeille.def** contient le texte suivant:

Insecte social, vivant dans une ruche, produisant le miel.

et le fichier **Abeille.jpg** contient l'image d'une abeille:

Pour le mot **Insecte**, il y a deux fichiers: **Insecte.def** et **Insecte.exp**. **Insecte.def** contient la définition d'un insecte:

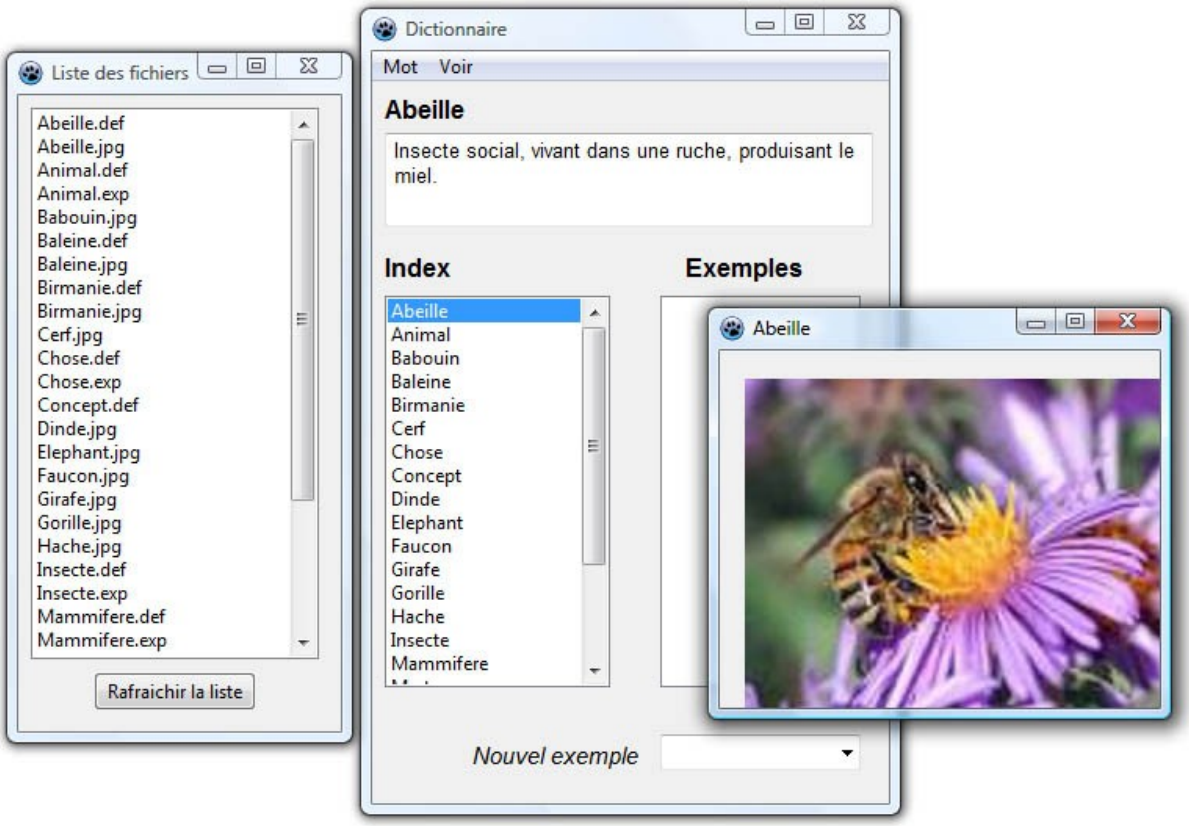
Animal invertébré à six pattes respirant par les trachées et subissant des métamorphoses.

et **Insecte.exp** contient des exemples d'insectes:

Abeille *Mouche*

Les fenêtres de l'application

L'application finale devra comporter trois fenêtres: une fenêtre principale et deux fenêtres secondaires :



La fenêtre principale (au centre) permet d'afficher la définition d'un mot ou les exemples associés. Elle permet également d'ouvrir ou de fermer les deux fenêtres secondaires : la fenêtre de visualisation des fichiers (à gauche) et la fenêtre de visualisation des image (à droite).

1ère Partie : Réalisation de l'interface graphique

Question 1 : Réalisation de la fenêtre principale

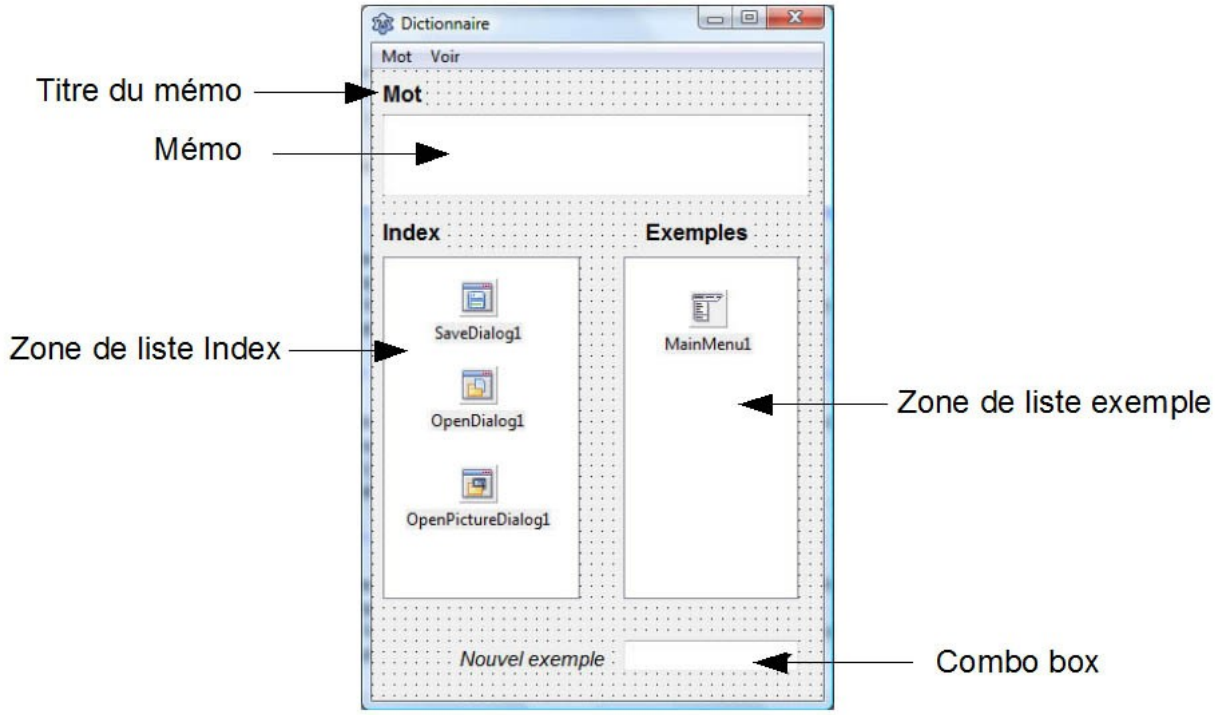
Pour l'instant l'application ne contient qu'une seule fenêtre nommée **Form1**, que nous allons utiliser pour réaliser la fenêtre principale. Elle est associée à l'unité **Unite_Principale.pas**.

Pour commencer renommez cette fenêtre **Frm_Principal** et donnez lui le titre **Dictionnaire**.

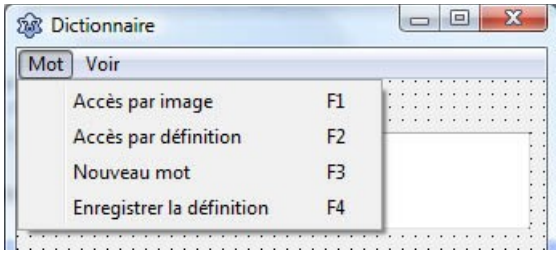
Ajoutez y ensuite les composants suivants:

- Une barre de menu avec deux entrées : **Mot** et **Voir**
- Un mémo qui servira à afficher, créer ou modifier la définition d'un mot.
- Une étiquette située juste au-dessus du mémo. Nous la nommerons par la suite **titre du mémo**.
- Un zone de liste surmontée du titre **Index** . Nous la nommerons par la suite **zone de liste Index**. Elle servira à sélectionner un mot du dictionnaire.
- Un zone de liste surmontée du titre **Exemples**. Nous la nommerons par la suite **zone de liste Exemples**. Elle servira à afficher les exemples d'un mot du dictionnaire et à les modifier. On donnera la valeur **True** à l'attribut **Sorted** de cette zone de liste.
- Un combobox précédé à gauche de l'étiquette *Nouvelle Exemple*. Il nous servira à rajouter un nouvel exemple à un mot. On donnera la valeur **True** aux attributs **Autocomplete**, **AutoDropDown** et **Sorted** de ce combobox.
- Une boite de dialogue d'ouverture de fichier. Elle nous servira à ouvrir des fichiers de définition. Définissez donc son attribut **Filter** afin de filtrer uniquement les fichier d'extensions .**def**
- Une boite de dialogue d'enregistrement de fichier. Elle nous servira à enregistrer un fichier de définition. Donc même chose pour l'attribut **Filter**.
- Une boite de dialogue d'ouverture de fichier image.

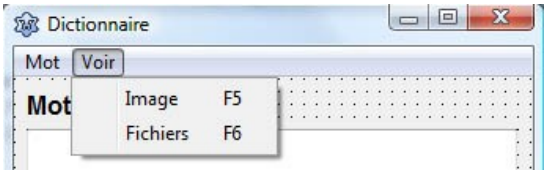
Disposez ces différents composants de la manière suivante:



Voici les entrées du menu **Mot** avec leurs raccourcis clavier:

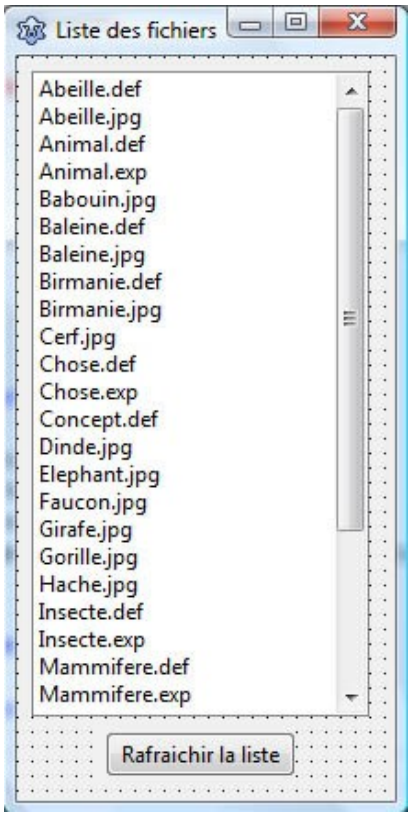


et voici celles du menu **Voir**:



Question 2 : Réalisation de la fenêtre de visualisation des fichiers

Créez une nouvelle fenêtre nommée la **Frm_Fichiers** en lui ajoutant une liste de fichiers et un bouton pour rafraichir cette liste. Sauvegardez le projet et nommez l'unité associée **Unite_Fichiers.pas**.

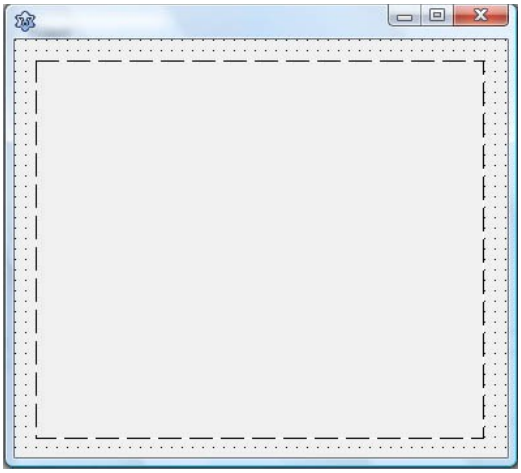


La liste de fichier devra nous permettre de voir les noms des fichiers contenus dans le répertoire **Fichiers**. Il vous faudra donc définir l'attribut **Directory** de ce composant en conséquence.

Lorsque l'utilisateur clique sur le bouton **Rafraichir la liste**, la liste des fichiers affichées est mise à jour. Ecrivez le gestionnaire d'évènement réalisant ceci et vérifiez qu'il fonctionne correctement en supprimant ou en ajoutant des fichiers dans le répertoire **Fichiers** sans quitter l'application.

Question 3 : Réalisation de la fenêtre de visualisation des images

Créez une nouvelle fenêtre. Nommez la **Frm_Img** et placez y un composant **Image**. Sauvegardez le projet en nommant cette unité **Unite_Image.pas**.



Question 4 : Affichage des fenêtre secondaires depuis la fenêtre principale

L'affichage d'une des deux fenêtres secondaires se fait via le menu **Voir**. L'entrée **Image** affiche la fenêtre de visualisation des images et l'entrée **Fichiers** affiche la fenêtre de visualisation des fichiers.

Ecrivez les gestionnaires d'évènement permettant ceci. Si votre fenêtre de visualisation des fichiers est correcte, elle devrait déjà vous afficher les fichiers du répertoire **Fichiers**. Vérifiez également que les raccourcis clavier fonctionnent.

2ème Partie : Gestion des définitions et des images via les boites de dialogues

Question 1: La procédure ChargerMot

On vous demande ici d'écrire la procédure **ChargerMot** dont l'entête est la suivante:

```
procedure ChargerMot (mot:string; Exemple: boolean);
```

On ne tiendra pas compte du paramètre **Exemple** pour l'instant, car dans cette partie de l'exercice on ne tiendra pas compte des fichiers exemples.

La procédure **ChargerMot** effectue les actions suivantes:

- le titre du mémo devient le mot.
- si le fichier image associé à ce mot existe, il est chargé dans le composant image de la fenêtre. Sinon le contenu du composant image est effacé.
- si le fichier de définition de ce mot existe, il est chargé dans le mémo. Sinon, le contenu du mémo est effacé.

Rappel: un fichier de nom **F** existe si **FileExists**(F) = **True**.

Question 2: Accès par image

L'entrée **Accès par image** du menu **Mot** permet de visualiser l'image associée à un mot ainsi que sa définition, si le fichier de définition de ce mot existe. Nous allons donc utiliser pour cela la procédure **ChargerMot** précédente.

L'utilisateur sélectionne l'image à l'aide de la boite de dialogue d'ouverture de fichier image. Le nom de fichier que vous allez récupérer dans cette boite de dialogue est le nom complet du fichier image comprenant donc son chemin et son extension (.**jpg** nécessairement).

Pour pouvoir utiliser la procédure **ChargerMot**, il vous faudrait en principe retrouver le mot à partir du nom de fichier, ce qui est assez fastidieux. Pour vous épargner ce travail, j'ai écrit pour vous la fonction suivante:

```
function MotViaNomFichier (nf: string): string;
```

Cette fonction retourne le mot à partir du nom de fichier complet.

Pour être cohérent avec la suite des opérations, il faudra appeler ici la fonction **ChargerMot** avec **Exemple** = **false**.

Question 3: Accès par définition

C'est en gros la même question que la précédente, mais cette fois-ci pour l'entrée **Accès par définition**. On utilisera ici la boite d'ouverture de fichier standard.

Question 4: Créer un nouveau mot

Ecrivez le gestionnaire d'évènement de l'entrée **Nouveau Mot** du menu **Mot** afin qu'un clic sur cette entrée efface:

- le titre du mémo
- l'image contenue dans le composant image
- la définition contenue dans le mémo.
- le contenu de la zone de liste Exemple

Question 5: Enregistrer la définition

Pour ajouter un nouveau mot dans le dictionnaire, l'utilisateur procède de la manière suivante: si le mémo contient déjà du texte, il va tout d'abord cliquer sur l'entrée **Nouveau Mot** pour effacer son contenu. Ensuite, il saisi la définition de ce mot dans le mémo, puis clique sur l'entrée **Enregistrer la définition** (menu **Mot**).

Ecrivez le gestionnaire d'évènement de cette entrée afin qu'elle ait les effets suivants:

- Elle affiche la boite de dialogue d'enregistrement de fichiers.

- Dès que l'utilisateur a sélectionné ou saisi un nom de fichier, le programme y enregistre le contenu du memo.

- Le titre du mémo devient le mot correspondant au nom du fichier. Par exemple, si le nom de fichier est **parapluie.def,** le titre du mémo devient **parapluie**. Pour réaliser ceci on pourra encore une fois utiiser la fonction **MotViaNomFichier**.

3ème Partie : Gestion des définitions et des images via la zone de liste Index

Question 1 : Initialisation de la liste de l'index

La zone de liste Index permet à l'utilisateur d'accèder à un mot quelconque en cliquant dessus. Elle doit donc contenir tous les mots du dictionnaire. On utilisera pour cela le fichier **Chose.exp**, qui devra contenir en permanence la liste de tous les mots du dictionnaire. Commencez donc par charger ce fichier dans la zone de liste au démarrage de l'application.

Question 2 : Accès via l'index

Votre zone de liste Index étant à présent initialisée, vous allez pouvoir réaliser l'accès à un mot via cette liste. Ecrivez le gestionnaire d'évènement permettant d'afficher la définition d'un mot et son image lorsque l'utilisateur clique sur une entrée. On utilisera pour cela la fonction **ChargerMot** (avec **Exemple = false**).

Question 3 : Mise à jour de l'index et du fichier Chose.exp

Dès qu'un fichier du dictionnaire est ajouté ou supprimé, la liste des mots doit être modifiée en conséquence.

Question 3 - A Ecriture de la procédure RegenererIndex

Pour réaliser ceci nous allons tout d'abord écrire une procédure nommée **RegenererIndex** qui effecturera les actions suivantes:

- Elle réactualise la liste des fichiers contenus dans la fenêtre de visualisation des fichiers.
- Elle remplit la zone de liste Index à partir de la liste des fichiers. Par exemple, si la liste des fichiers contient **Chat.def** ou **Chat.jpg**, le mot **Chat** sera rajouté dans l'index. Il faudra donc faire attention à ne pas ajouter un mot plusieurs fois lorsqu'il est associé à plusieurs fichiers. Pour supprimer l'extension, vous pouvez utiliser la fonction **NomFichierSansExtension**.
- Elle enregistre le contenu de la zone de liste Index dans le fichier **Chose.exp**

Question 3 - B Mise à jour par double-clic sur la zone de liste Index

Ecrivez le gestionaire d'évènement permettant de réactualiser la zone de liste Index lorsque l'utilisateur double-clique sur une de ses entrées.

Question 3 - C Mise à jour lors de l'adjonction d'un nouveau mot

Lorsque l'utilisateur enregistre la définition d'un mot, il ajoute ce mot au dictionnaire et un nouveau fichier d'extension **.def** est créé. Il faut donc mettre à jour l'index. Modifiez le gestionnaire d'évènement de l'entrée **Enregistrer la definition** du menu **Mot** pour tenir compte de ceci.

4ème Partie : Gestion des exemples

Pour l'instant, nous n'avons pas tenu compte des fichiers exemple (extension **.exp**) associés aux mots du dictionnaire, ainsi que des deux composants qui vont servir à les afficher ou à les modifier : la zone de liste **Exemple** et le **combo box**.

Il s'agira donc ici de gèrer les exemples d'un mot donné du dictionnaire: les afficher, en ajouter, en supprimer, créer un nouveau fichier exemple. Ce mot sera toujours le mot sélectionné dans la zone de liste Index, nous l'appelerons par la suite le **mot courant**.

Question 1 : Prise en compte des exemples dans la procédure ChargerMot

Pour afficher la définition d'un exemple, ainsi que son éventuelle image, on utilisera encore une fois la procédure **ChargerMot**, mais cette fois-ci avec **Exemple = true**.

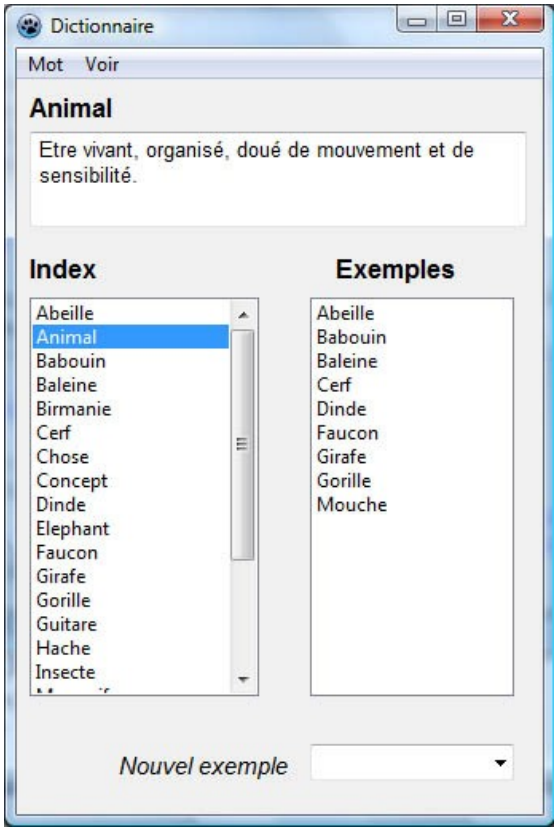
Mais pour que tout fonctionne correctement, **ChargerMot** doit à présent réaliser deux actions supplémentaires lorsque le mot n'est pas un exemple:

- Elle définit ce mot comme le mot courant ou autrement dit, elle le sélectionne dans la zone de liste **Index**.
- S'il existe un fichier d'exemple pour ce mot, elle le charge dans la zone de liste Exemple.

Effectuez les modifications nécessaires dans cette procédure. Pour vérifiez qu'elle fonctionne cliquez sur le mot Animal dans la zone de liste Index. Si tout fonctionne bien:

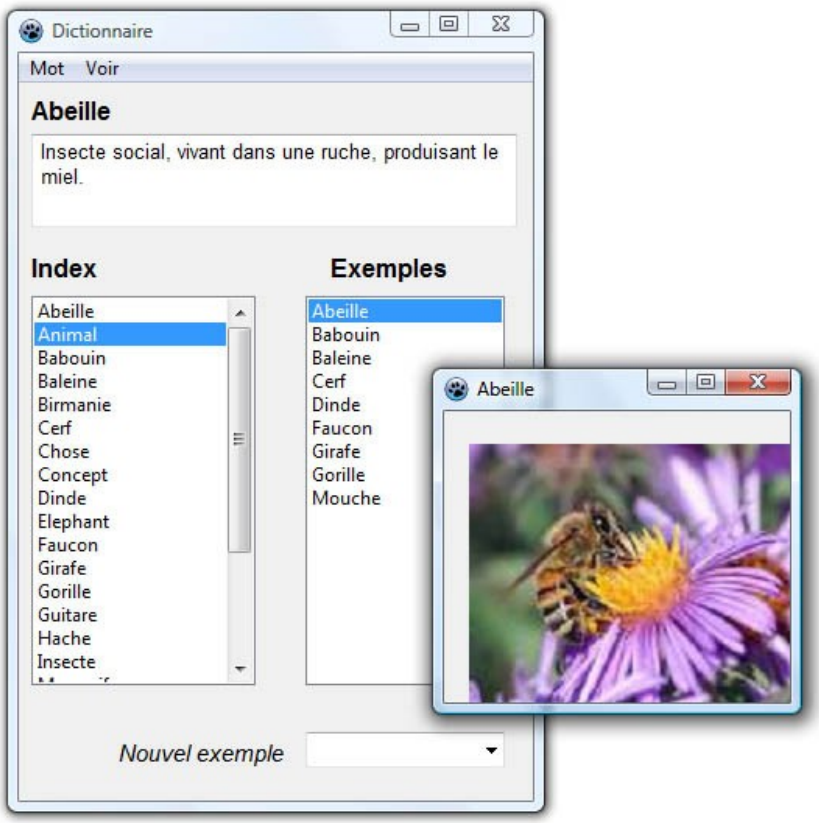
- Les exemples d'animaux du dictionnaire devrait alors s'afficher dans la zone de liste Exemple.
- Le mot Animal est sélectionné dans la zone de liste Index.
- La définition d'un Animal est affichée dans le mémo.

Vous devez donc voir ceci:



Question 2 : Affichage d'un exemple

L'affichage d'un exemple d'un mot, c'est à dire de sa définition et de son image, se fait en cliquant sur cet exemple dans la zone de liste Exemple. Voici par exemple, l'affichage de l'exemple **Abeille** du mot **Animal**:



Vous pouvez réaliser ceci en appelant la procédure **ChargerMot** avec **Exemple = true**, dans le gestionnaire de l'évènement **Click** de la zone de liste **Exemple**.

Des clics virtuels !

Lorsque ce gestionnaire d'évènement fonctionne, faites l'expérience suivante: cliquez sur un exemple, puis sur les touches flèches (vers le haut ou vers le bas). Vous constaterez que pouvez ainsi afficher rapidement tous les exemples de la liste. Cela s'explique de la manière suivante : le fait d'appuyer sur la touche flèche vers le bas (respectivement vers le haut) sélectionne automatiquement l'élément suivant (respectivement précédent) et génère un évènement **Click** sur cet élément. Tout se passe donc comme si l'utilisateur avait cliqué sur l'élément précédent ou sur l'élément suivant. Ceci montre qu'un évènement **Click**, n'est pas nécessairement produit par un clic !

Question 3 : Ajout d'un nouvel exemple

Question 3 - A - Initialisation et mise à jour de la liste du combobox

Pour ajouter un nouvel exemple à un mot, nous allons utiliser le combobox. La sélection d'un nouvel exemple se fera via la liste de ce combobox.

Question 3 - A - 1 - Initialisation

Il faudra donc la remplir avec tous les mots du dictionnaire dès le démarrage du programme. Ajoutez l'instruction nécessaire dans le gestionnaire **FormCreate** de l'unité principale.

Question 3 - A - 2 - Mise à jour

Lorsque l'on regénère l'index, la liste du combobox doit également être réactualisée. Ajoutez l'instruction nécessaire dans la procédure **RegenererIndex**.

Question 3 - B - Utilisation du combobox pour ajouter un exemple

Pour ajouter un exemple au mot courant, il suffit de le sélectionner dans le combobox. Il est alors automatiquement ajouté dans la zone de liste exemple et le fichier exemple du mot courant est mis à jour. Pour réaliser ceci écrivez le gestionnaire de l'évènement **EditingDone** du combobox.

Mettez y les instructions permettant:

- d'ajouter le mot dans la zone de liste exemple. Attention à ne pas ajouter un mot qui s'y trouve déjà !
- de sauvegarder le contenu de la zone de liste Exemple dans le fichier d'exemple du mot courant.

On prendra également garde à interdire l'ajout d'exemple si le mot courant est **Chose** (car **Chose.exp** est regénéré automatiquement !).

Question 4 : Suppression d'un nouvel exemple

La suppression d'un exemple se fait en double-cliquant dessus dans la zone de liste Exemple. Ecrivez le gestionnaire de cet évènement afin qu'il effectue les opérations suivantes:

- suppression du mot dans la zone de liste exemple.
- sauvegarde de la zone de liste dans le fichier Exemple du mot courant.

On prendra également garde à interdire la suppression d'un exemple si le mot courant est **Chose** (car **Chose.exp** est regénéré automatiquement !).

Question 5 : Mise à jour du mot courant lors de l'ajout d'un nouveau mot

Souvenez vous que pour ajouter un nouveau mot au dictionnaire, l'utilisateur sélectionne d'abord l'entrée **Nouveau Mot**. Il saisi ensuite la définition de ce mot dans le mémo. Puis il sauvegarde cette définition dans un fichier **.def**. Pour terminer cet exercice, il nous reste à prendre en compte deux choses:

1. Lorsque l'utilisateur clique sur l'entrée **Nouveau Mot**, il se peut qu'il existe déjà un mot courant, c'est à dire un mot sélectionné dans la zone de liste Index. Comme le nouveau mot est alors inconnu, il faut le déselectionner.
2. Ce n'est que lorsque l'utilisateur enregistre la définition, que le nouveau mot est connu (souvenez vous qu'il est déduit du nom du fichier). Ce nouveau mot doit alors devenir le mot courant.