Microsoft Visual Basic 5.0
Les fichiers de ressources

Novembre 1998

David Rousse

Les fichiers de ressources sous Visual Basic 5.0

I – GENERALITES

a) Fichiers de ressources et Ressources de chaîne

Les fichiers de ressources ont été ajouté à Visual Basic pour faciliter l'internationalisation des applications. Avant la disponibilité de ce type de fichiers, il fallait éditer tous les titres dans le projet Visual Basic lui-même. Les fichiers de ressources permettent d'isoler toutes les chaînes d'une application, ainsi que tous les fichiers (images, icônes, …) qui concernent directement le projet.

Les **ressources de chaîne** regroupent la totalité du texte qui apparaît dans l'interface utilisateur de l'application. Menus, boîtes de dialogues, messages d'erreur, d'avertissement et d'information en constituent une liste non exhaustive. Si une application est destinée à être utilisée avec des paramètres régionaux autres que ceux avec lesquels elle a été développée, ces ressources doivent être traduites.

b) Modèle d'adaptation

Sur le schéma suivant, le bloc de données représente le «composant Interface utilisateur » et le bloc de code le «composant Application». Ainsi, on a :

Bloc de données + bloc de code = Produit

Le bloc de données contient toutes les ressources de chaîne de l'interface utilisateur mais aucun code. À l'inverse, **le bloc de code** ne contient que le code de l'application commun à tous les paramètres régionaux.

c) Avantages liés à la conception d'un logiciel multilingue

- **Efficacité**. Le développement d'une version dans une autre langue de l'application implique simplement la création d'un nouveau fichier de ressources. En effet, toutes les versions utilisent le même bloc de code. Il est ainsi possible de créer directement des versions en plusieurs langues de votre application Visual Basic.
- **Sécurité accrue**. Que vous décidiez de traduire votre application en interne ou d'utiliser les services d'une autre société, il n'est pas nécessaire d'accéder au code source pour développer des versions en d'autres langues de votre application. Cette approche diminue également le nombre de tests nécessaires pour valider la version en langue étrangère.
- **Adaptation de meilleure qualité**. Les ressources de chaîne figurant toutes dans un même fichier, la traduction en est facilitée et les risques de laisser des chaînes non traduites sont moindres.

II – LES FICHIERS DE RESSOURCES

a) Avantages du stockage des chaînes dans des fichiers de ressource

Lorsque vous rédigez un code Visual Basic, vous pouvez utiliser les **fonctions LoadResString, LoadResPicture et LoadResData** au lieu de références à des données, images et chaînes littérales. Le stockage de ce type d'éléments dans un fichier de ressources présente deux avantages :

- Les performances et les fonctionnalités s'en trouvent accrues car les chaînes, images bitmap, icônes et données peuvent être chargées sur demande depuis le fichier de ressources, au lieu d'être toutes chargées simultanément lorsqu'une feuille ou un module est chargé.
- Les ressources qui doivent être traduites sont isolées dans un fichier de ressources unique. Il n'est ainsi pas nécessaire d'avoir accès au code source ou de recompiler l'application.

b) Pour créer un fichier de ressources

- Créez un fichier source de ressources (**.rc**) contenant toutes les ressources de chaîne de votre application. La syntaxe permettant de créer le fichier source de ressources est présentée dans le fichier Resource.txt du dossier \Tools dans le dossier principal de Visual Basic.
  Vous devez associer un identificateur (ID) à chaque ressource, puis ajouter une référence à chaque identificateur dans votre code.
- Utilisez un compilateur de ressources pour convertir le fichier source de ressources en fichier de ressources (**.res**). Vous pouvez utiliser le compilateur de ressources (Rc.exe) figurant dans le dossier \Tools\Resource du dossier principal de Visual Basic.
- **Exemple** :

```
// FRES.RC - Fichier de ressources pour l'exemple
// SYNTAXE DU COMPILATEUR
// E:\TOOLS\RESOURCE\RC /R /FO FRES.RES FRES.RC

// Icons
2  ICON  "win95ico/35floppy.ico"
3  ICON  "win95ico/525flop1.ico"
4  ICON  "win95ico/audio.ico"
5  ICON  "win95ico/cddrive.ico"
6  ICON  "win95ico/clsdfold.ico"
7  ICON  "win95ico/ctrpanel.ico"
8  ICON  "win95ico/desktop.ico"
9  ICON  "win95ico/drive.ico"

// Chaînes
STRINGTABLE MOVEABLE DISCARDABLE
BEGIN
2  "Disquette 3'1/2"
3  "Disquette 5'1/4"
4  "Audio"
5  "Lecteur de disque"
6  "Répertoire"
7  "Panneau de configuration"
8  "Bureau"
9  "Lecteur"
END
```

- **Commentaires** :

La syntaxe du compilateur est RC /R /FO FRES.RES FRES.RC où :

- r signifie que l'on effectue une compilation seule.

- fo signifie que l'on désigne un nouveau nom pour le fichier de ressource compilé.

L'option MOVABLE signifie que la ressource pourra être déplacé en mémoire si besoin.
L'option DISCARDABLE signifie que la ressource peut être déchargée si elle n'est plus utilisée.

- **Notes :**

  - Les chaînes sont définies dans une ou plusieurs tables de chaînes (STRINGTABLE).

  - Dans Visual Basic, la ressource dotée de l'identificateur 1 est réservée à l'icône d'application. Il est donc impossible d'inclure une ressource avec cet identificateur dans votre fichier .res. Visual Basic génère un message d'erreur si votre code tente de charger cet identificateur de ressource.

  - De plus, il faut inclure un seul fichier de ressource par projet.

c) Pour traduire un fichier de ressources

- Chargez le fichier de ressources dans un éditeur de ressources, comme Microsoft Visual C++ ou Borland C++ ou un simple éditeur de texte.
- Une fois le fichier chargé, traduisez les entrées. Créez autant de versions des chaînes, images bitmap, icônes et données que vous souhaitez disposer de langues.

d) Pour ajouter un fichier de ressources à votre projet

- Dans le menu **Projet**, cliquez sur **Ajouter un fichier** (CTRL+D).
- Dans la boîte de dialogue **Ajouter le fichier**, sélectionnez **Fichiers de ressource (*.res)** dans la zone **Type**.
- Sélectionnez le fichier de ressources à ajouter au projet, puis cliquez sur **Ouvrir**. Visual Basic reconnaît les fichiers de ressources grâce à l'extension .res. Si le fichier de ressources ne comporte pas l'extension de fichier appropriée, Visual Basic ne le charge pas. À l'inverse, tout fichier utilisant l'extension .res est interprété par Visual Basic comme étant un fichier de ressources lorsqu'il est ajouté au projet. Si le fichier ne respecte pas le format standard des fichiers de ressources, Visual Basic génère un message d'erreur la première fois que vous tentez d'utiliser les fonctions de support de fichier de ressources (LoadResString, LoadResPicture et LoadResData), ou lorsque vous tentez de créer un fichier .exe. Visual Basic génère le même message d'erreur si vous tentez d'ajouter un fichier de ressources 16 bits à un projet.
- Une fois le fichier de ressources ajouté au projet, le fichier .res apparaît dans la fenêtre Projet. À la différence d'une feuille ou d'un module, vous ne pouvez toutefois pas visualiser le fichier de ressources dans Visual Basic. Lorsque vous cliquez sur Créer *nomprojet*.exe dans le menu Fichier, Visual Basic compile toutes les ressources de ce fichier dans le fichier .exe en tant que ressources Windows.

e) Points particuliers à étudier

La taille du texte n'étant par identique dans toutes les langues, il est conseillé de faire attention aux composants suivants lors de la conception de l'interface utilisateur.

- messages :

  La taille des chaînes de texte varie selon la langue employée. En règle générale, l'anglais est ainsi plus court que le français ou l'allemand. Vous trouverez dans l'aide de Visual Basic un tableau de coefficients de foisonnement à respecter en prenant l'anglais comme référence.

- icônes et images bitmap :

Les icônes et les images bitmap permettent souvent de décrire certaines fonctions sans utiliser de texte. Il faut toutefois tenir compte des points suivants :

- Evitez d'utiliser des images spécifiques à un pays ou à une région.
- Evitez d'utiliser des images contenant du texte
- Assurez-vous que les images bitmap et les icônes n'ont pas de connotation négative dans les différentes langues et cultures auxquelles elles sont destinées.

- menus et boîtes de dialogue :

À l'instar des messages, la longueur du texte des menus et des boîtes de dialogue peut varier selon les langues. Sachant que la longueur du texte peut augmenter, prévoyez dès le départ un espace suffisant ou positionner à True la propriété Autosize des contrôles afin de ne pas être obligé de réorganiser certains éléments une fois traduits.

- touches d'accès rapide et de raccourci.

La configuration des claviers est différente selon les pays. En effet, l'alphabet n'est pas le même dans toutes les langues.

Il est donc préférable de ne pas utiliser les touches d'accès rapide et de raccourci qui ne sont pas disponibles dans toutes les langues ou qui sont spécifiques à certaines Editions de Windows.

Par contre, il est recommandé d'employer des chiffres et des touches de fonction (F1, F2, etc.). Cette méthode est certes moins explicite, mais elle ne nécessite aucune adaptation car les chiffres et les touches de fonction figurent dans pratiquement toutes les configurations clavier.

Note : Les caractères DBCS ne peuvent pas être utilisés dans les touches d'accès rapide ou de raccourci.

f) Concaténation de chaînes

Pour réduire la taille d'une chaîne, vous pouvez recourir au procédé dit de concaténation de chaînes. Cette méthode vous permet d'utiliser la même ressource dans plusieurs chaînes. Elle présente toutefois quelques dangers, illustrés dans l'exemple suivant :

Anglais	**Français**
Chaîne1: one after the other	Chaîne1: l'un après l'autre
Chaîne2: The controls will be deleted.	Chaîne2: Les contrôles seront supprimés.
Chaîne3: The forms will be deleted.	Chaîne3: Les feuilles seront supprimées

Considérées séparément, Chaîne1, Chaîne2 et Chaîne3 ne posent aucun problème d'adaptation. Si votre code effectue la concaténation Chaîne2 + Chaîne1 ou Chaîne3 + Chaîne1, la chaîne que vous obtiendrez sera correcte en anglais. Dans sa version française, en revanche, la chaîne obtenue comportera une erreur d'accord en genre lors de la concaténation Chaîne3 + Chaîne1; « feuille » étant un nom féminin, il faudrait en effet remplacer « l'un après l'autre » par « l'une après l'autre ». Ce type de problème peut survenir dans d'autres langues étrangères. Le seul moyen de l'éviter ici est de

regrouper les deux chaînes Chaîne2 et Chaîne1 ainsi que les deux chaînes Chaîne3 et Chaîne1 dans le fichier de ressources.

g) Chargement des ressources depuis le fichier

Il existe des options qui contrôlent le moment où chaque ressource est effectivement chargée en mémoire (voir le fichier Tools\Ressource.txt sur le CD de Visual Basic 5.0). Toutes les ressources dont les numéros sont dans le même bloc de 16 membres. (celle dont les 12 bits les plus significatifs des numéros de ressource – entiers de 16 bits – sont identiques) seront chargées ensemble par le système dés que l'une d'elles est chargée par Visual Basic.

III – EXEMPLE D'UTILISATION DE FICHIERS DE RESSOURCES

a) Exemple n°1

Le code suivant, issu du fichier FrmInput.frm, charge des ressources stockées dans le fichier Atm32.res (exemple d'application contenue dans l'aide en ligne de Visual Basic 5.0), qui contient les chaînes traduites dans toutes les langues.

```
Sub Form_Load()
     imgFlag = LoadResPicture(I, vbResBitmap)
     Caption = LoadResString(I)
     lblPINCode = LoadResString(1 + I)
     fraAccount = LoadResString(2 + I)
     optChecking.Caption = LoadResString(3 + I)
     optSavings.Caption = LoadResString(4 + I)
     lblAmount = LoadResString(5 + I)
     cmdOK.Caption = LoadResString(6 + I)
     SetCursor cmdOK
End Sub

Sub cmdOK_click()
     ' Affiche un message de processus.
     MsgBox LoadResString(7 + I)
     frmAmountWithdrawn.Show vbModal
     Unload Me
End Sub
```

Au moment de l'exécution, ce code lit la section appropriée du fichier de ressources, en fonction d'un décalage initialisé lorsque l'utilisateur sélectionne une langue dans l'écran de présentation. Le **décalage** est une variable publique déclarée dans le module standard indiquant la distance à laquelle se trouve un élément spécifique par rapport à un point de départ. Dans l'exemple d'application ATM, la variable de décalage est `I`.

Dans le fichier de ressources, les identificateurs de ressources allant de 16 à 47 sont réservés à l'anglais, de 48 à 79 au français, de 80 à 111 à l'allemand, etc. À chaque langue correspondent des entrées traduites qui constituent le bloc de données de l'exemple d'application. Ce bloc contient ici les onze ressources spécifiques à chaque langue.

Cette exemple d'application, qui contient plusieurs blocs de données, s'oppose à l'utilisation d'un fichier de ressources spécifique par langue et ne comportant à chaque fois qu'un seul bloc de données. Selon la nature de l'application que vous développez, vous pouvez choisir d'utiliser un fichier de ressources par langue pour chaque version de votre application ou un seul fichier de ressources contenant tous les blocs de données traduits.

b) Exemple n°2

J'ai crée cette petite application pour montrer l'utilisation d'un fichier de ressources. Ce projet, appelé prjiti102.vbp, comporte donc un fichier de ressources appelé langue4.res, plus quelques autres composants (frmsoul1.02.frm, frminf obulle.frm, general.bas, initialise.bas, registre.cls) qui permettent à ce projet de :

- Lire Win.ini pour connaître la langue utilisée mais seulement sous Win 32 bits car on utilise ici une fonction API version 32 bits.
- Gérer la dimensions des contrôles de la feuille quelle que soit la résolution utilisée.
- Déclaration des fonctions 32 bits seulement si l'environnement est 32 bits.
- Utiliser des bulles d'info bulles si on est sous un système 32 bits et d'info bulles avec le formulaire frminfobulle.frm pour simuler les info bulles sous un environnement 16 bits.
- Lire et écrire d'informations dans la base de registre Windows lors de démarrage et de la fermeture de l'application.
- Disposer de boutons « graphiques » avec références des images dans le fichier de ressources.

On utilisera un fichier de ressources avec les spécifications suivantes :

- On attribue 6550 identificateurs par version ;on pourra alors avoir 5 langues au total.
- En partant avec une base de 30 feuilles à afficher, on aura ici 210 identificateurs disponibles par feuilles.

En ouvrant ce projet sous Visual Basic 5.0, vous pourrez voir que l'on a 3 langues utilisées, mais que les propriétés Caption des contrôles sont vides et ne seront « remplies » qu'à la demande à partir du fichier de ressources par le procédure chargernoms (dans le formulaire frmmain.frm).

A noter que le décalage est appelé version : version est une variable publique de type Long déclarée dans le module general.bas.

La détermination de la langue à utiliser est cherché lors du premier démarrage dans le fichier Win.ini . Si aucune information n'y est trouvée, la 1° démarrage se fait en français. Puis on utilise la base de registre Windows pour lire ou sauvegarder la dernière langue utilisée (en fait pour sauver la valeur de version).

On peut conclure enfin en soulignant que cette petite application ne comporte qu'un seul écran mais elle à le mérite de mettre en œuvre quelques techniques intéressantes pour le développement d'applications en grandeur nature.