



## Editeur de recherche et liste hiérarchique

---

Par

Jean-Luc PELLERIN, Formation 4D

Note technique 4D-200006-18-FR

Version 1

Date 1 Juin 2000

### Résumé

---

Cette note technique vous a présenté une autre manière de réaliser un éditeur de recherche à l'aide d'une liste hiérarchique simple. L'objectif étant de comprendre le fonctionnement d'une liste hiérarchique simple.

### 4D Notes techniques

---

Copyright © 1985-2004 4D SA - Tous droits réservés

Tous les efforts ont été faits pour que le contenu de cette note technique présente le maximum de fiabilité possible. Néanmoins, les différents éléments composant cette note technique, et le cas échéant, le code, sont fournis sans garantie d'aucune sorte. L'auteur et 4D S.A. déclinent donc toute responsabilité quant à l'utilisation qui pourrait être faite de ces éléments, tant à l'égard de leurs utilisateurs que des tiers.

Les informations contenues dans ce document peuvent faire l'objet de modifications sans préavis et ne sauraient en aucune manière engager 4D SA. La fourniture du logiciel décrit dans ce document est régie par un octroi de licence dont les termes sont précisés par ailleurs dans la licence électronique figurant sur le support du Logiciel et de la Documentation afférente. Le logiciel et sa documentation ne peuvent être utilisés, copiés ou reproduits sur quelque support que ce soit et de quelque manière que ce soit, que conformément aux termes de cette licence.

Aucune partie de ce document ne peut être reproduite ou recopiée de quelque manière que ce soit, électronique ou mécanique, y compris par photocopie, enregistrement, archivage ou tout autre procédé de stockage, de traitement et de récupération d'informations, pour d'autres buts que l'usage personnel de l'acheteur, et ce exclusivement aux conditions contractuelles, sans la permission explicite de 4D SA.

4D, 4D Calc, 4D Draw, 4D Write, 4D Insider, 4ème Dimension ®, 4D Server, 4D Compiler ainsi que les logos 4e Dimension, sont des marques enregistrées de 4D SA.

Windows, Windows NT, Win 32s et Microsoft sont des marques enregistrées de Microsoft Corporation.

Apple, Macintosh, Power Macintosh, LaserWriter, ImageWriter, QuickTime sont des marques enregistrées ou des noms commerciaux de Apple Computer, Inc.

Mac2Win Software Copyright © 1990-2002 est un produit de Altura Software, Inc.

4D Write contient des éléments de "MacLink Plus file translation", un produit de DataViz, Inc, 55 Corporate drive, Trumbull, CT, USA.

XTND Copyright 1992-2002 © 4D SA. Tous droits réservés.

XTND Technology Copyright 1989-2002 © Claris Corporation.. Tous droits réservés ACROBAT © Copyright 1987-2002, Secret Commercial Adobe Systems Inc. Tous droits réservés. ACROBAT est une marque enregistrée d'Adobe Systems Inc.

Tous les autres noms de produits ou appellations sont des marques déposées ou des noms commerciaux appartenant à leurs propriétaires respectifs.

---

## Introduction

---

L'éditeur de recherche standard de recherche présente de nombreuses fonctionnalités, mais il a le défaut de proposer une interface peut conviviale.

La maîtrise des listes hiérarchiques permet de proposer aux utilisateurs des outils de recherche plus adaptés.

L'objectif est de comprendre le fonctionnement d'une liste hiérarchique simple.

## Principe de la base exemple

---

Le principe est de présenter sous forme de listes hiérarchiques des valeurs hiérarchisées. Un simple clic sur un élément permet de faire la sélection correspondante.

## La base exemple

---

La base exemple présente une liste d'entreprises contenant un champ région et un champ département.

## La programmation

---

### La phase d'initialisation

Dans un premier temps, il faut initialiser la variable qui contient la référence à la liste dans la méthode base

```
``Méthode base sur ouverture  
◇LH:=0
```

À l'ouverture de l'interface, on initialise la liste hiérarchique sur la méthode objet du bouton initialiser.

1/ Nous supprimons la liste hiérarchique si elle existe

```
` Liste_H ("Initialiser_Interface")  
  Liste_H ("Supprimer_Liste")`Suppression de l'ancienne liste éventuelle  
  
` Liste_H("Supprimer_Liste")  
  Si (Liste existante (◇LH))  
    SUPPRIMER LISTE (◇LH ;*)  
  
  Fin de si
```

2/ À partir de la sélection, on crée un tableau local afin de récupérer la liste des régions

```
TOUT SELECTIONNER ([Clients])
```

```
`La sélection comprend tous les enregistrements
```

```
VALEURS DISTINCTES ([Clients] Région ;$T_Region)
```

```
`Le tableau contient la liste de toutes les régions
```

```
REDUIRE SELECTION ([Clients] ;0)
```

3/ Nous créons la liste hiérarchique puis nous ajoutons de nouveaux éléments à la liste pour présenter la liste des régions. Chaque élément de la liste sera affecté par un élément de tableau et vaudra une valeur unique délivrée par la routine Hasard.

```
◇LH :=Nouvelle liste
```

```
Boucle ($i ;1 ;Taille tableau ($T_Region))
```

```
  AJOUTER A LISTE (◇LH ;$T_Region {$i}; Hasard)
```

```
Fin de boucle
```

## La phase d'utilisation

Un clic sur un élément de la liste hiérarchique devra provoquer les actions suivantes :

1/ Connaître le niveau dans la liste, en effet on ne fera pas la même chose si l'élément est au premier niveau ou au second.

Pour connaître le niveau on va boucler à la recherche de l'élément père. Lorsque la routine Element parent retourne la valeur zéro, nous sommes en présence d'un élément de niveau un.

```
`` Liste_H("Liste_Niveau")
```

```
$Element_Sel :=Element selectionne (◇LH)
```

```
INFORMATION ELEMENT (◇LH ;$Element_Sel ;$Element_Ref ;$Tx)
```

```
$Element_Ref :=Element parent (◇LH ;$Element_Ref)
```

```
$0:=1
```

```
Tant que ($Element_Ref>0)
```

```
  $Element_Ref :=Element parent (◇LH ;$Element_Ref)
```

```
  $0:=$0+1
```

```
Fin tant que
```

Après avoir obtenu le niveau de l'élément, si nous sommes au premier niveau de la liste, nous allons rechercher les entreprises du département concerné, puis construire une sous-liste contenant ces éléments, à la condition qu'elle n'existe pas.

Cette sous-liste sera associée à l'élément du niveau supérieur à l'aide de la routine CHANGER ELEMENT.

Il est préférable de demander explicitement le redessinement de la liste hiérarchique avec la routine REDESSINER LISTE.

Lorsque l'on se trouve en présence d'un élément de niveau 2, on fait une recherche des entreprises du département.

### Au cas ou

```
: ($Niveau=1) `Faire liste des Dpt
$Element_Sel :=Element selectionne (◇LH)
INFORMATION ELEMENT (◇LH ;$Element_Sel ;$Element_Ref ;$Tx ;$Sous_Liste)
CHERCHER ([Clients] ;[Clients] Région=$Tx)

Si ($Sous_Liste=0)
VALEURS DISTINCTES ([Clients] Département ;$T_Dpt)
$Sous_Liste :=Nouvelle liste
Boucle ($i ;1 ;Taille tableau ($T_Dpt))
AJOUTER A LISTE ($Sous_Liste ;$T_Dpt {$i};Hasard)
Fin de boucle
CHANGER ELEMENT (◇LH ;$Element_Ref ;$Tx ;$Element_Ref ;$Sous_Liste ;Vrai)
Fin de si
REDESSINER LISTE (◇LH)

: ($Niveau=2) `Faire liste des entreprises
$Element_Sel :=Element selectionne (◇LH)
INFORMATION ELEMENT (◇LH ;$Element_Sel ;$Element_Ref ;$Tx ;$Sous_Liste)
CHERCHER ([Clients] ;[Clients] Département=$Tx)
```

**Fin de cas**

## La phase de sortie

À la sortie du formulaire, on ne doit pas oublier de supprimer la liste hiérarchique de la mémoire vive. En effet, ce qui est contenu dans le variable retourné par nouvelle liste c'est la référence à la liste.

Lorsque le process est tué, la variable process disparaîtra avec la référence de la liste hiérarchique mais non la liste elle-même. La liste hiérarchique restera en mémoire vive et sera inaccessible.

La suppression se fera avec la routine SUPPRIMER LISTE ajoutée d'une\* afin que toutes les sous-listes soient aussi effacées de la mémoire.

```
`` Liste_H("Supprimer_Liste")
Si (Liste existante (◇LH))
SUPPRIMER LISTE (◇LH ;*)
Fin de si
```

La variable ◇LH contenant la référence à la liste hiérarchique est variable interprocess afin que cette référence n'existe qu'une seule fois dans la mémoire vive.

L'utilisation d'une variable process aurait conduit à la duplication de cette variable dans tous les process.