

## UTILISATION DE LA LIBRAIRIE NSFCTUTIL DANS NS-DK

Afin de personnaliser l'outil NS-DESIGN et de l'adapter plus finement aux exigences des clients, il est possible, depuis la version 5, de définir des fonctions qui seront appelées pendant la phase de développement.

Ces fonctions de contrôles seront appelées

- Au chargement du projet
- Après une vérification sur du code NCL (touche [F8]) appartenant à une librairie ou à un événement.
- Avant et/ou après le lancement de la construction de l'application (Build)
- A partir de l'éditeur de code NCL (deux fonctions).
- Quel que soit le contexte (édition NCL, édition SCR, compilation, ...deux fonctions utilisateurs globales).

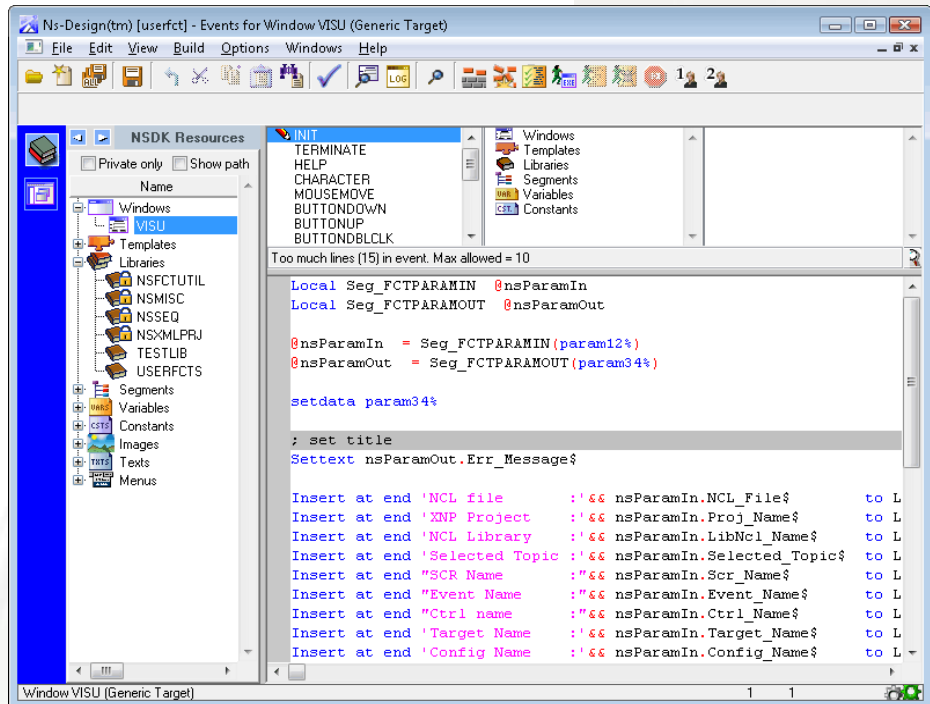
Le principe est le suivant : une DLL écrite par un développeur NS-DK exporte des fonctions qui sont appelées depuis l'outil NS-DESIGN. Cette DLL peut analyser le projet en cours, le NCL en cours d'édition, effectuer d'autres traitements puis rendre la main en renvoyant des messages et codes retour qui peuvent être affichés, dans la barre de statut ou la fenêtre de log.

Ces fonctions peuvent être très utiles, voici quelques exemples de leur utilisation :

- Au chargement d'un projet, lors de la conversion de V3 en V5, vérifier que les répertoires des ressources sont corrects et les changer si nécessaires.
- Introduire dans l'outil une vérification "Maison" de la syntaxe du NCL (Nombre de lignes dans un événement, nom des fonctions ...).
- Recopie automatique des exécutables générés après le BUILD.
- Installation d'une aide en ligne spécifique au projet ou aux librairies du client.

Nous verrons dans un exemple d'utilisation comment on peut (par exemple) :

- Compter le nombre de lignes dans un événement d'un contrôle et afficher une erreur, 'il est trop grand.
- Afficher une aide sur une fonction d'une librairie interne.

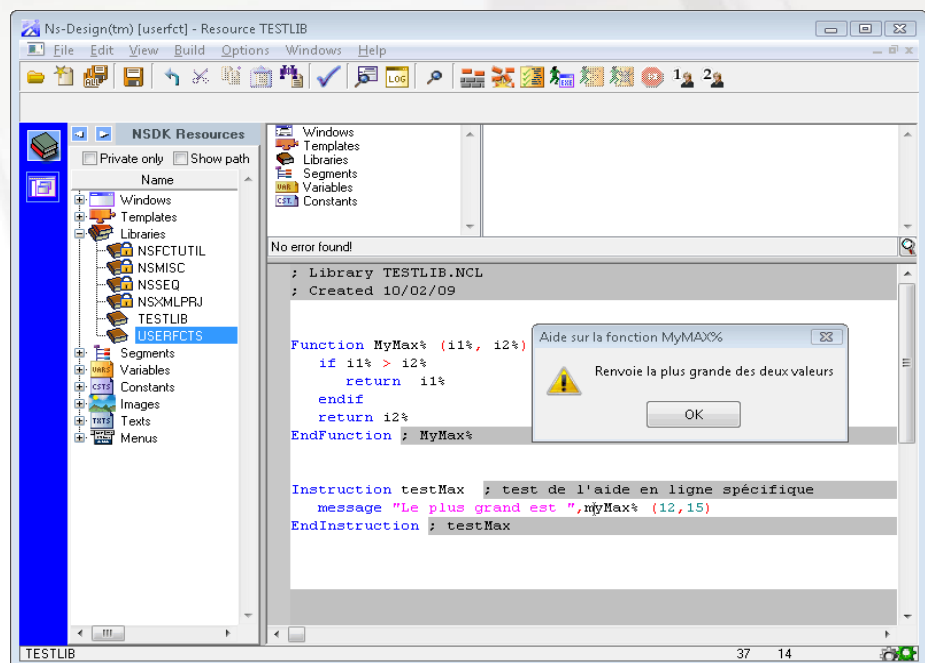


Dans l'exemple ci-dessus, on peut voir que :

- deux icônes ont été rajoutées dans la barre d'outils,
- Un contrôle spécifique du NCL de l'événement a été rajouté : s'il y a plus de 10 lignes de code NCL dans l'événement, le Check du NCL renvoie faux et un message d'erreur apparaît dans la status-bar de l'éditeur.

Dans l'exemple ci-dessus, on peut voir une aide en ligne spécifique.

L'utilisateur a positionné le curseur sur le mot myMax% et a appuyé sur la combinaison de touches Ctrl-F2, une aide en ligne spécifique, à l'intention des développeurs NS-DK a été écrite et apparaît.



## Mise en œuvre de la solution

### Écriture d'une DLL d'extension de NS-DK

En premier lieu, il faut créer un projet NS-DK qui pourra générer une DLL. Ce projet doit importer la librairie NSFCTUTIL et exporter un ensemble de fonctions décrites dans cette librairie.

Il y a 5 fonctions potentiellement exportables par la DLL, dont une seule est obligatoire.

Ces fonctions reçoivent principalement comme paramètre d'appel :

- un segment SEG\_FCTPARAMIN en entrée,
- et un segment SEG\_FCTPARAMOUT en sortie.

Le segment d'entrée contient toutes les informations relatives au NCL édité et au projet en cours.

Le segment de sortie contiendra les codes retour, les messages à afficher dans l'outil ainsi qu'éventuellement une liste d'erreurs ou de "warning" à afficher dans la fenêtre de LOG.

### Détail de la fonction NS\_USERFUNCT\_QUERYACTIONNAME\$

Cette fonction détaille la liste des fonctions utilisateur implémentées ainsi que les libellés devant apparaître dans les menus ou les bulles d'aide.

Elle reçoit comme paramètre d'appel une des constantes suivantes.

```

; Constantes passées à la fonction
; USERFUNCT_QUERYACTIONNAME$

; Check du NCL
Const NS_USRFCT_NCLCHECK% 1
; Avant le Build
Const NS_USRFCT_PREBUILD% 2
; après le Build
Const NS_USRFCT_POSTBUILD% 3
; Commande 1 de l'éditeur NCL
Const NS_USRFCT_NCLEDITOR1% 4
; Commande 2 de l'éditeur NCL
Const NS_USRFCT_NCLEDITOR2% 5
; fonction globale N°1
Const NS_USRFCT_GLOBAL1% 6
; fonction globale N°2
Const NS_USRFCT_GLOBAL2% 7
; Chargement du projet
Const NS_USRFCT_LOADPROJECT% 8
    
```

### Exemple d'implémentation.

```

; Définition des fonctions utilisateurs implémentées.
; ainsi que des libellés devant apparaître.
;-----
Function NS_USERFUNCT_QUERYACTIONNAME$ \
(Int Function_kind%) Return Cstring
Local Cstring ActionName$
    
```

```

Evaluate Function_kind%
Where NS_USRFCT_NCLCHECK%
ActionName$ = "Check"
EndWhere
Where NS_USRFCT_NCLEDITOR1%
ActionName$ = "Aide personnalisée"
EndWhere
Where NS_USRFCT_POSTBUILD%
ActionName$ = "PostBuild"
EndWhere
else
ActionName$ = ""
EndEvaluate
Return ActionName$
EndFunction ; NS_USERFUNCT_QUERYACTIONNAME$
    
```

Nous pouvons voir dans cet exemple que 3 fonctions utilisateurs seront implémentées : le check du NCL, une fonction de l'éditeur (aide en ligne personnalisée) et une action après le Build (si celui-ci s'est bien passé).

### Détail de la fonction NS\_USERFUNCT\_NCLCHECK%

Cette fonction lit le texte NCL en cours d'édition, s'il s'agit d'un événement et pas d'une librairie, elle compte le nombre de lignes. Si ce nombre est supérieur à 10

```

;-----
; Fonction appelée après le check NCL
; exemple d'implémentation, les événements
; ne doivent pas comporter plus de 10 ligne de NCL
; (commentaires non compris)
;-----
Function NS_USERFUNCT_NCLCHECK% \
(SEG_FCTPARAMIN @nsParamIn, \
SEG_FCTPARAMOUT @nsParamOut) Return Int

local h%,count%
local dynStr ds$

nsParamOut.Col_Number% = 1
nsParamOut.Line_Number% = 1

; test : est-ce une librairie NCL
; si oui pas de test
if nsParamIn.LibNCL_Name$ <> ""
return true%
endif

; c'est un événement, alors
; ouverture du fichier intermédiaire
h% = t_open% (nsParamIn.NCL_file$)
if miscerror% <> 0
nsParamOut.Err_Message$ = "Can't find file " \
& nsParamIn.NCL_file$
return false%
endif
count% = 0
; on compte les lignes (sauf les commentaires)
while (not T_EOF% (h%))
ds$ = SKIP t_readLnEx% (h%)
if (ds$ <> "") and (copy$ (ds$,1,1) <> ";")
count% = count% + 1
endif
endwhile
t_close (h%)

; plus de 10 ligne? on renvoie l'erreur
; avec un message pour la StatusBar
if count% > 10
nsParamOut.Err_Message$ = "Too much lines (" & \
count% & ") in event. Max allowed = 10"
return false%
endif

return true%

EndFunction ; NS_USERFUNCT_NCLCHECK%
    
```

Noter la dynString et l'appel à la fonction T\_ReadLnEx () pour lire une chaîne de longueur > à 255 caractères.

### Détail de la fonction NS\_USERFUNCT\_NCLEDITOR%

Cette fonction est appelée depuis l'éditeur NCL en cours d'édition, il lui est passé un mot-clé en paramètre (celui qui est sous le curseur). Elle permet d'afficher de l'aide sous forme de message, d'appel à l'aide Windows, ou bien dans la barre de statut de l'éditeur NCL.

```

;-----
;Fonction appelée lors de l'édition du NCL (CTRL+F2)
;-----
Function NS_USERFUNCT_NCLEDITOR% \
(Int FctNum%, \
SEG_FCTPARAMIN @nsParamIn, \
SEG_FCTPARAMOUT @nsParamOut) Return Int

; (Ctrl-F2), affichage d'un message sur le mot-clé
; sélectionné (simulation d'aide spécifique)
Evaluate FctNum%
Where NS_USRFCT_NCLEDITOR1%
Evaluate uppercase nsParamIn.selected_topic$
Where ""
EndWhere
Where "MYMAX%"
message "Aide sur la fonction MyMax%",\
"Renvoie la plus grande des deux valeurs"
EndWhere
else
message "Aide spécifique sur le Topic",\
nsParamIn.selected_topic$
EndEvaluate
EndWhere
EndEvaluate
Return true%
EndFunction ; NS_USERFUNCT_NCLEDITOR%
    
```

Noter que dans ce cas, on n'a codé que l'aide sur le mot-clé MYMAX%, mais on aurait pu faire le branchement sur un système d'aide plus sophistiqué, ou bien afficher une aide simplifiée dans la barre de statut.

## Intégration à l'outil NS-DESIGN

Il suffit, dans NS-DESIGN, d'ouvrir le menu Options / Setup et cliquer sur le bouton « User Functions Call », puis de préciser la DLL qui a été développée. Les nouveaux raccourcis clavier, menu et appels se mettent en place automatiquement.

## Domaine d'utilisation de ces fonctions et conclusion :

On l'aura compris, cette librairie est extrêmement puissante, et permet une multitude d'extensions personnalisées au produit NS-DESIGN.