



**Forum: WD 11**

**Topic: Fichier .Rep**

**Subject: Re: Fichier .Rep**

Posté par: R&B

Contribution le : 18/4/2008 10:24:33

Bonjour,

Lors de la mise à jour de nos projet, nous supprimons le .rep (eh oui) et lançons WDMODFIC en ligne de commande depuis un exe en charge de piloter l'installation.

gcRepertoireDesDonnées est le répertoire des données à mettre à jour.

```
LanceAppli(fCheminCourt(CompleteRep(fRepEnCours()))+"WDModFic.EXE -subdir -noanaguid  
-nobackup /WDD="+fCheminCourt(CompleteRep(fRepEnCours()))+"MonAnalyse.wdd  
/DIR="+fCheminCourt(gcRepertoireDesDonnées),exeActif,exeBloquant,fCheminCourt(CompleteRep(fRepEnCours())))
```

Cette ligne de commande traite TOUS les fichiers de l'application dans les sous répertoire pour ceux qu'on joigne l'analyse de la nouvelle version (le .wdd).

Note : -noanaguid permet d'intégrer les fichiers dont wdmofic met un (!) en mode interactif et qui ne serait mis à jour autrement. La mise à jour automatique des données ignore aussi ces fichier, ce qui peut être la cause de votre problème. Avec cette ligne de commande (et donc un exe qui pilote l'installation), on passe outre ce type de problèmes.

Sinon il est possible de reconstituer le .rep (en exclusivité WDForge)

Dans une fenêtre : 2 champs tables dont les colonnes sont des chaînes

TABLE\_REP[TLOGIQUE, TPHYSIQUE, TGUIDA, TGUIDF]

TABLE\_ALIAS[TLOGIQUE, TPHYSIQUE, TGUIDA, TGUIDF]

Les procédures suivantes ou gcRepDonnées est le répertoire du projet et gProjetNom le nom du projet/exe.

rbREP\_Init va charger les tables

```
// Monte dans une table la liste des fichiers créés directement de l'analyse dans un répertoire  
// contenu dans la globale gcRepDonnée // Analyse la liste des fichiers logiques et les recherche  
dans le répertoire // Si ouverture possible, récupération sdu GUID analyse et GUID fichier. // Ce sont  
des GUID qui seront envoyés au rep PROCEDURE rbREP_Init(pcNomTable,pcListe,pcLibelle="") SI  
pcLibelle ALORS {pcLibelle,indChamp}=gcRepDonnees {pcNomTable,indChamp}..Visible=Faux  
TableSupprimeTout(pcNomTable) Sablier cListeFichier est une chaîne cListeFichier = HListeFichier()  
iFic est un entier = 1 hFic est un entier cChemin est une chaîne cGUID_Ana,cGUID_Fic sont des  
chaînes cFichier est une chaîne = ExtraitChaîne(cListeFichier,iFic,RC) ss est une Source de  
Données TANTQUE cFichier<>EOT cChemin=gcRepDonnees+cFichier+".Fic" SI  
fRep(cChemin,frFichier)<>"" ALORS hFic=fOuvre(cChemin,foLecture) SI hFic<>-1 ALORS  
fPositionne(hFic,20,fpDébut) cGUID_Ana = fLit(hFic,32) fPositionne(hFic,53,fpDébut)  
cGUID_Fic= fLit(hFic,32) fFerme(hFic)  
TableAjouteLigne(pcNomTable,cFichier,cChemin,cGUID_Ana,cGUID_Fic)  
ListeAjoute(pcListe,cChemin) FIN FIN iFic++ cFichier = ExtraitChaîne(cListeFichier,iFic,RC) FIN  
Sablier(Faux) {pcNomTable,indChamp}..Visible=Vrai
```

```

rbRep_Lst qui va lister les fichiers physiques
// récupération des fichiers aliassés PROCEDURE rbRep_Lst(Repertoire, Nom, Change, Nb)
cChemin est une chaîne = ComplèteRep(Repertoire)+Nom // le fichier existe déjà dans la liste SI
ListeCherche(Liste_rep,cChemin)=-1 ALORS // non alors on va l'ajouter SI
fRep(cChemin,frFichier)<>" ALORS hFic est un entier=fOuvre(cChemin,foLecture) SI hFic <> -1
ALORS fPositionne(hFic,20,fpDébut) cGUID_Ana est une chaîne= fLit(hFic,32)
fPositionne(hFic,53,fpDébut) cGUID_Fic est une chaîne = fLit(hFic,32) fFerme(hFic) // si
(sansespace(cGUID_Ana)+sansespace(cGUID_Fic))<>" alors
// trace(nom,cchemin,cGUID_Ana,cguid_fic)
TableAjouteLigne(TABLE_ALIAS,Nom,cChemin,cGUID_Ana,cGUID_Fic)
// listeajoute(liste_rep,cchemin) // fin FIN FIN FIN Renvoyer Vrai

```

```

rbREP_Construit qui construit le .REP
PROCEDURE rbREP_Construit(pcT1,pcT2) i est un entier Ind est un entier cLogique est une chaîne
cFic est une chaîne = ComplèteRep(fRepEnCours()+gProjetNom+".REP" eFic est un entier =
fOuvre(cFic,foCréation+foEcriture) cLigne est une chaîne SI eFic=-1 ALORS RETOUR POUR i=1 A
TableOccurrence(pcT1) cLigne="" SI i=1 ALORS
cLigne="ANALYSISGUID="+pcT1[i,3]+TAB+";Analysis GUID"+RC FIN
cLigne+="LOCALIZATION="+pcT1[i,4]+TAB+pcT1[i,1]+TAB+pcT1[i,2] fEcritLigne(eFic,cLigne) FIN
POUR i=1 A TableOccurrence(pcT2) cLigne="" Ind =
TableCherche(TABLE_REP.TGUIDF,pcT2[i,4]) SI Ind<>-1 ALORS
cLigne+="LOCALIZATION="+pcT2[i,4]+TAB+pcT1[Ind,1]+TAB+pcT2[i,2] fEcritLigne(eFic,cLigne)
FIN FIN fFerme(eFic)

```

et finalement le code du bouton pour créer le .REP :

```

// récupère les définitions pour les fichiers normaux dans le répertoire des données
rbREP_Init(TABLE_REP..Nom,Liste_rep..Nom,LIB_REP..Nom) // complète avec les fichiers issus
d'alias TableSupprimeTout(TABLE_ALIAS) fListeFichier(gcRepDonnees+"*.fic","rbRep_Lst") //
Génère le REP à partir des données rbREP_Construit(TABLE_REP..Nom, TABLE_ALIAS..Nom)
Ferme()

```