DOCUMENTATION TECHNIQUE DU PROJET JC_SVG_Disque

06/07/2023

Partie 1

Projet

Projet

Code

Initialisation de JC_SVG_Disque

```
GLOBAL
    VG_EnCours_Rep        est une chaîne = ""           // en cours : répertoire chargé en cours
    VG_EnCours_LstFic est une chaîne = ""           // en cours : liste des fichiers chargés
    VG_EnCours_Recursif   est une chaîne = ""           // en cours : Récursif valeur
    VG_FicRep_Cachés  est une chaîne = ""           // en cours : Fichiers et répertoires cachés

    // définition des types de .ext
    wWord, wExcel, wPowerP, wRtf, wPdf, wTxt, wZip, wLog     sont des chaînes
    wPhotos, wVideos, wSons, wExe, wSgbd, wHtml, wSysteme    sont des chaînes
    wConfig, wLibre, wLeReste, wInutiles                     sont des chaînes


    VG_cheminBD           est une chaîne                // chemin des fichiers HyperFile

    // Intégration du fichier "WinConst.wl" qui contient les constantes standard de Windows
    EXTERNE "KeyConst.wl"
    EXTERNE "WinConst.wl"

// ---------------------------------------------------------------------------------------------

// A1 - on vérifie le répertoire : C:\ProgramData\HF_global
VG_cheminBD=ComplèteRep(fRepGlobalCommun())+"HF_global"
SI PAS fRepExiste(VG_cheminBD) ALORS
    SI PAS fRepCrée(VG_cheminBD) ALORS
        Erreur("Création de la BD impossible, finPrograme !")
        FinProgramme()
    FIN
FIN
// A2 - on vérifie le répertoire : C:\ProgramData\HF_global
VG_cheminBD=ComplèteRep(VG_cheminBD)+"SVGDD"
SI PAS fRepExiste(VG_cheminBD) ALORS
    SI PAS fRepCrée(VG_cheminBD) ALORS
        Erreur("Création de la BD impossible, finPrograme !")
        FinProgramme()
    FIN
FIN

HChangeRep(SVGDD_DD_TEMPS,VG_cheminBD)
HChangeRep(SVGDD_EXT,VG_cheminBD)
HCréationSiInexistant(SVGDD_DD_TEMPS)
HCréationSiInexistant(SVGDD_EXT)
```

Projet

Statistiques sur le code

	Lignes [1]	% comm. [1]	Lig./trait. [1]
ProcéduresGlobales	184	17	92
FEN_Popup_Saisie_Durée	0	0	0
Fen_JCsvgDD_QueFaire	77	8	6
Fen_ParseKA	44	14	14
fen_JCsvgDD_ListeFic	426	6	12
fen_JCsvgDD_ModifType	43	14	10
fen_JCsvgDD_StructureRep	90	11	9
fen_TimeLine	92	13	5
JC_SVG_Disque	48	21	48
	964	**11**	11

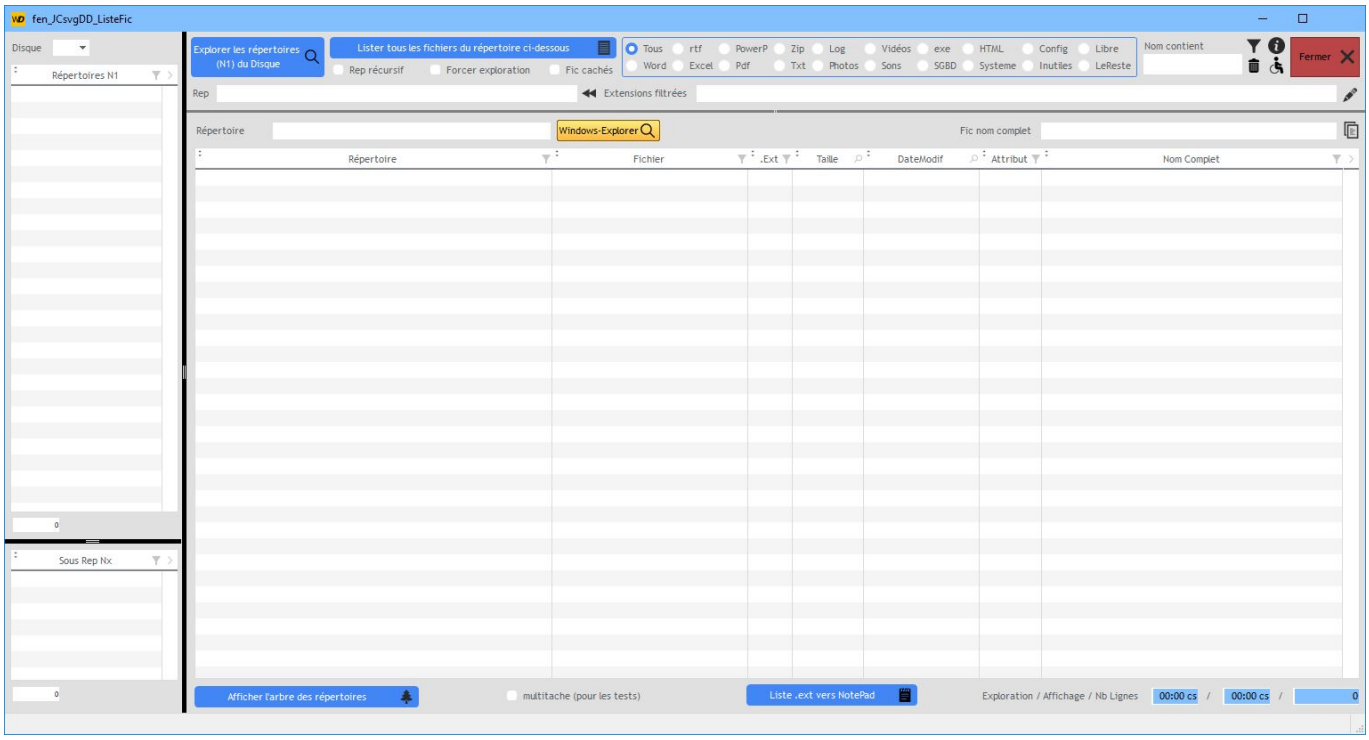
[1] Lignes : Nombre total de lignes de code. % comm. : Pourcentage de commentaires dans le code.
Lig./trait. : Nombre de lignes de code par traitement.

Partie 2

Fenêtre WINDEV

fen_JCsvgDD_ListeFic

Image



fen_JCsvgDD_ListeFic

Code

Déclarations globales de fen_JCsvgDD_ListeFic

```
PROCÉDURE MaFenêtre()
```

Fin d'initialisation de fen_JCsvgDD_ListeFic

```
//fen_JCsvgDD_ListeFic.x=1
//fen_JCsvgDD_ListeFic.Y=1

Maximise()
```

fen_JCsvgDD_ListeFic

Code des champs

Clic sur bou_Arbre

```
i,j sont des entiers
k            est un entier
wRep     est une chaîne
ta_Rep       est un tableau associatif d'entiers
wTrav        est une chaîne

k=TableOccurrence(Table_Fichier)

POUR TOUTE LIGNE i DE Table_Fichier

    Jauge(i,k)

    wRep=tf_rep[i]
    ta_Rep[wRep]++

FIN

wTrav=ChaîneConstruit("Les fichiers de type '%1' (%2)",sel_Filtre[sel_Filtre]..Libellé,sai_ext)

i=2
j=Taille(sai_ext)-1
wTrav=ChaîneConstruit(wTrav,sai_ext)

wTrav = Ouvre(fen_JCsvgDD_StructureRep,wTrav,ta_Rep)

SI wTrav<>"" ALORS
    sai_RepAexplorer=wTrav
    ExécuteTraitement(bou_explorer_DD_Fichier,trtClic)
FIN
```

Clic sur bou_explorer_DD

```
wListeRep        est une chaîne

TableSupprimeTout(Table_Rep1)
TableSupprimeTout(Table_SousRep) ; sai_nb_SousRep=0
TableSupprimeTout(Table_Fichier)
```

```
SI Combo_DD_source..ValeurAffichée="" ALORS
    Message("ERREUR : aucun disque source de sélectionné ...")
    RETOUR
FIN

wListeRep=fListeRépertoire(ComplèteRep(Combo_DD_source..ValeurAffichée),frNonRécursif)

POUR TOUTE CHAÎNE MonRep DE wListeRep SÉPARÉE PAR RC
    SI Contient(MonRep,"$")   ALORS CONTINUER      // c'est un rep système
    TableAjouteLigne(Table_Rep1,MonRep)
FIN

TableTrie(Table_Rep1,"tr1_rep")

TableSelectMoins(Table_Rep1)

sai_nb_rep=Table_Rep1..Occurrence

Message()
```

Clic sur bou_explorer_DD_Fichier

```
wListeFic                 est une chaîne

j,k                   sont des entiers

wRep, wFic, wNomComplet   sont des chaînes
wAttribut                 est une chaîne
wTaille                   est un entier sur 8 octets
wDateModif                est une DateHeure
wExtension                est une chaîne
MkExplorer                est un booléen
wType                     est une chaîne

wTempsE                   est un entier
wTempSA                   est un entier
nbTotalFic                est un entier

wTempsTotal               est un entier
wTrav                     est une chaîne

// Titre Filtre extensions recherchées
wType    = sel_Filtre[sel_Filtre]..Libellé
wType    = ChaîneFormate(wType,ccSansAccent+ccSansEspace+ccSansEspaceIntérieur+ccSansPonctuationNiEspace
+ccMajuscule)

// mise en forme des extensions à retenir
sai_ext=Majuscule(sai_ext)
SI Gauche(sai_ext,1)<>"," ALORS sai_ext=","+sai_ext
SI Droite(sai_ext,1)<>"," ALORS sai_ext=sai_ext+","
sai_ext=Remplace(sai_ext," ",",")
sai_ext=Remplace(sai_ext,",,",",")
sai_ext=ChaîneFormate(sai_ext,ccSansAccent+ccSansEspace+ccSansEspaceIntérieur+ccMajuscule)

POUR TOUTE CHAÎNE monExt DE sai_ext SÉPARÉE PAR ","
    SI monExt="" ALORS CONTINUER
    SI Gauche(monExt,1)<>"." ALORS
        Erreur("Ce n'est pas une extension de la forme '.x' : "+monExt)
        RETOUR
    FIN
FIN
```

```
SI sai_RepAexplorer="" ALORS
    Message("ERREUR : Sélectionner un répertoire ...")
    RETOUR
FIN

Sablier(Vrai)

TableSupprimeTout(Table_Fichier)

// pour éviter d'explorer plusieurs fois le DD si on peut se l'éviter ...
MkExplorer=Faux
SI sai_RepAexplorer<>VG_EnCours_Rep                                    ALORS MkExplorer=Vrai
SI VG_EnCours_LstFic=""                                                ALORS MkExplorer=Vrai
SI int_récursif=Vrai ET VG_EnCours_Recursif="FAUX"              ALORS MkExplorer=Vrai
SI int_récursif=Faux ET VG_EnCours_Recursif="VRAI"              ALORS MkExplorer=Vrai
SI int_FicRepCaché=Faux ET VG_FicRep_Cachés="VRAI"              ALORS MkExplorer=Vrai
SI int_FicRepCaché=Vrai ET VG_FicRep_Cachés="FAUX"              ALORS MkExplorer=Vrai
SI int_ForcerEploration=Vrai                                           ALORS
    MkExplorer=Vrai
    int_ForcerEploration=Faux                                          // remise à faux après une mise à vrai
FIN

SI MkExplorer=Vrai ALORS
    // là on est obligé d'explorer

    // Sauf si on pense que ca va être long ...
    HLitRecherchePremier(SVGDD_DD_TEMPS,SVGDD_REP,sai_RepAexplorer)
    SI PAS HTrouve(SVGDD_DD_TEMPS) ALORS
        // on se fait un DD entier que si validation ...
        SI Taille(sai_RepAexplorer)=2 ET sai_RepAexplorer[[2]]=":" ALORS
            SI PAS OuiNon(Non,"Un disque en entier : Tu es patient ?") ALORS
                RETOUR
            FIN
        FIN
    SINON
        // on demande si + de 15 secondes
        wTempsTotal=Val(ExtraitChaîne(SVGDD_DD_TEMPS.SVGDD_TEMPS_SS,1,"/"))+Val(ExtraitChaîne(
        SVGDD_DD_TEMPS.SVGDD_TEMPS_SS,2,"/"))
        SI wTempsTotal>15 ALORS
            wTrav=ChaîneConstruit("Tu veux explorer %1 ?"+RC+
            "La dernière fois on avait mis %2s pour %3 fichiers !",sai_RepAexplorer,wTempsTotal,
            SansEspace(NumériqueVersChaîne(Val(ExtraitChaîne(SVGDD_DD_TEMPS.SVGDD_TEMPS_SS,3,"/")),"
            12ds")))
            SI PAS OuiNon(Non,wTrav) ALORS
                RETOUR
            FIN
        FIN
    FIN

    Message("Exploration en cours : attendre ...")
    ChronoDébut(1)
    SI int_récursif=Vrai ALORS
        SI int_FicRepCaché=Faux ALORS
            wListeFic=fListeFichier(ComplèteRep(sai_RepAexplorer)+"*.*",frInformationComplète+
            frRécursif+frSansFichierCaché+frSansRépertoireCaché+frInterruptible)
        SINON
            wListeFic=fListeFichier(ComplèteRep(sai_RepAexplorer)+"*.*",frRécursif+frInterruptible)
            // pas d'infocomplete avec les fichiers cachés
        FIN
    SINON
        SI int_FicRepCaché=Faux ALORS
            wListeFic=fListeFichier(ComplèteRep(sai_RepAexplorer)+"*.*",frInformationComplète+
            frNonRécursif+frSansFichierCaché+frSansRépertoireCaché+frInterruptible)
        SINON
            wListeFic=fListeFichier(ComplèteRep(sai_RepAexplorer)+"*.*",frNonRécursif+frInterruptible)
            // pas d'infocomplete avec les fichiers cachés
```

```
        FIN
    FIN
    sai_DuréeExploration=ChronoFin(1)
    VG_EnCours_Rep=sai_RepAexplorer
    VG_EnCours_LstFic=wListeFic
    SI int_récursif=Vrai  ALORS VG_EnCours_Recursif="VRAI"   SINON VG_EnCours_Recursif="FAUX"
    SI int_FicRepCaché=Vrai ALORS VG_FicRep_Cachés="VRAI"        SINON VG_FicRep_Cachés="FAUX"
SINON
    wListeFic=VG_EnCours_LstFic
    sai_DuréeExploration=0
FIN

Message()

j=0
k=ChaîneOccurrence(wListeFic,RC)
nbTotalFic=k
ChronoDébut(1)

POUR TOUTE CHAÎNE Maligne DE wListeFic SÉPARÉE PAR RC

    SI Maligne=EOT            ALORS CONTINUER
    SI Maligne=""             ALORS CONTINUER

    j++ ; Jauge(j,k,"Chargement en cours : Attendre ...")

    // SI on veut faire Esc lors du Chargement : à chaque fois c'est TROP LONG (+- *100) : donc on le
    fait tous les 1.999 fichiers affichés

    SI PartieDécimale(j/1999)=0. ET int_multitache=Vrai ALORS
        Multitâche(-1)
        // Vérification de la touche espace
        SI ToucheEnfoncée(teEchap) = Vrai ALORS
            SI OuiNon(Non,"Touche Esc enfoncée : tu veux arrêter le chargement ?") ALORS
                sai_DuréeExploration=0
                SORTIR
            FIN
        FIN
    FIN

    SI int_FicRepCaché=Vrai ALORS
        wNomComplet  = Maligne
        wRep     = fExtraitChemin(wNomComplet,fDisque+fRépertoire)
        wFic     = fExtraitChemin(wNomComplet,fFichier+fExtension)
        wTaille      = 0
        wDateModif   = ""
        wAttribut    = ""
    SINON
        wNomComplet=ExtraitChaîne(Maligne,1,TAB)
        wRep=fExtraitChemin(wNomComplet,fDisque+fRépertoire)
        wFic=fExtraitChemin(wNomComplet,fFichier+fExtension)
        wTaille=Val(ExtraitChaîne(Maligne,2,TAB))
        wDateModif=ExtraitChaîne(Maligne,3,TAB)
        wAttribut=ExtraitChaîne(Maligne,4,TAB)
    FIN

    wExtension=","+Majuscule(fExtraitChemin(wNomComplet,fExtension))+","

    SI sel_Filtre<>1 ET PAS Contient(sai_ext,wExtension) ET wType<>"LERESTE"          ALORS CONTINUER
    // ce n'est pas ce que l'on veut
    SI wType="LERESTE" ET Contient(sai_ext,wExtension)                                    ALORS
    CONTINUER
    // on ne veut pas ces extensions
    SI sai_NomFic_Contient<>"" ET PAS Contient(ChaîneFormate(wFic,ccSansAccent+ccSansEspace+
    ccSansEspaceIntérieur+ccMajuscule),sai_NomFic_Contient) ALORS CONTINUER
    // filtre sur le nom ne matche pas
```

```
    TableAjouteLigne(Table_Fichier, wRep, wFic, Remplace(wExtension,",",""), wTaille, wDateModif,
    wAttribut, wNomComplet)

FIN
sai_DuréeAffichage=ChronoFin(1)

Jauge()

TableTrie(Table_Fichier,"tf_rep","tf_fic")

TableSelectMoins(Table_Fichier)

Sablier(Faux)

sai_nb_fic=Table_Fichier..Occurrence

SI sai_nb_fic=0 ALORS
    Message("Aucun fichier trouvé ...")
    TableAjouteLigne(Table_Fichier, "Aucun fichier trouvé ...")
SINON
    Message()
FIN

// ------------------------ on affiche les S/R de niveau 1
wListeRep        est une chaîne
TableSupprimeTout(Table_SousRep)
wListeRep=fListeRépertoire(sai_RepAexplorer,frNonRécursif)
POUR TOUTE CHAÎNE MonRep DE wListeRep SÉPARÉE PAR RC
    SI Contient(MonRep,"$")   ALORS CONTINUER      // c'est un rep système
    TableAjouteLigne(Table_SousRep,MonRep)
FIN
TableTrie(Table_SousRep,"tsr_rep")
TableSelectMoins(Table_SousRep)
sai_nb_SousRep=Table_SousRep..Occurrence
Message()


// on enregistre le temps mis ... sauf si on ne'Explore pas
SI sai_DuréeExploration="0000000000" ALORS
    SI EnModeTest() ALORS Message("Pas de temps enregistré ...")
    RETOUR
FIN

wTempsE=(Val(Milieu(sai_DuréeExploration,3,2))*60)+(Val(Milieu(sai_DuréeExploration,6,2)))
wTempSA=(Val(Milieu(sai_DuréeAffichage,3,2))*60)+(Val(Milieu(sai_DuréeAffichage,6,2)))

HLitRecherchePremier(SVGDD_DD_TEMPS,SVGDD_REP,sai_RepAexplorer)
SI PAS HTrouve(SVGDD_DD_TEMPS) ALORS
    HRAZ(SVGDD_DD_TEMPS)
    SVGDD_DD_TEMPS.SVGDD_REP      = sai_RepAexplorer
    SVGDD_DD_TEMPS.SVGDD_TEMPS_SS = wTempsE+"/"+wTempSA+"/"+nbTotalFic
    SVGDD_DD_TEMPS.SVGDD_DATE     = DateHeureSys()
    HAjoute(SVGDD_DD_TEMPS)
SINON
    SVGDD_DD_TEMPS.SVGDD_TEMPS_SS = wTempsE+"/"+wTempSA+"/"+nbTotalFic
    SVGDD_DD_TEMPS.SVGDD_DATE     = DateHeureSys()
    HModifie()
FIN
```

Clic sur bou_ext_vers_pp

```
wListe   est une chaîne = ","
wTrav    est une chaîne
j        est un entier = Table_Fichier..Occurrence

POUR TOUTE LIGNE i DE Table_Fichier

    Jauge(i,j)

    wTrav=Majuscule(","+tf_extension[i]+","    )

    SI PAS Contient(wListe,wTrav) ALORS
        wListe+=wTrav
    FIN
FIN

Jauge()

wListe=Remplace(wListe,",,",",")
wListe=GP_Trie_Liste(wListe)
wListe=Remplace(wListe,",",RC)

fSauveTexte("d:\temp\ext.txt",wListe)

LanceAppliAssociée("d:\temp\ext.txt")
```

Clic sur Bou_Selection_Rep

```
SI sai_rep="" ALORS RETOUR

LanceAppliAssociée(sai_rep)
```

Initialisation de Combo_DD_source

```
wListe   est une chaîne = fListeDisque()

ListeAjoute(Combo_DD_source, wListe)

SI Combo_DD_source..Occurrence>0 ALORS
    ListeSelectPlus(Combo_DD_source,1)
FIN

sai_RepAexplorer=Combo_DD_source..ValeurAffichée
ExécuteTraitement(bou_explorer_DD,trtClic)
```

Sélection d'une ligne de Combo_DD_source

```
sai_RepAexplorer=Combo_DD_source..ValeurAffichée
ExécuteTraitement(bou_explorer_DD,trtClic)
int_ForcerEploration=Vrai
```

Initialisation de Fermer

```
// Version 1
// Description
// Bouton permettant de fermer la fenêtre
```

Clic sur Fermer

```
FinProgramme()
```

Clic sur Image_appliquer

```
ExécuteTraitement(bou_explorer_DD_Fichier,trtClic)
```

Clic sur Image_appliquer1

```
wTrav    est une chaîne
i        est un entier

wTrav=sai_RepAexplorer

SI Droite(wTrav,1)="\"    ALORS wTrav=Gauche(wTrav,Taille(wTrav)-1)
SI Droite(wTrav,1)=":"    ALORS RETOUR

SI ChaîneOccurrence(wTrav,"\")<=1 ALORS RETOUR              // on ne reviens pas au disque !!!

i=Taille(wTrav)
TANTQUE wTrav[[i]]<>"\" ET i>2
    wTrav=Gauche(wTrav,i-1)
    i--
FIN

sai_RepAexplorer=wTrav

ExécuteTraitement(bou_explorer_DD_Fichier,trtClic)
```

Clic sur Image_ModifierType

```
wType    est une chaîne
i        est un entier = sel_Filtre

wType    = sel_Filtre[i]..Libellé
wType    = ChaîneFormate(wType,ccSansAccent+ccSansEspace+ccSansEspaceIntérieur+ccSansPonctuationNiEspace
+ccMajuscule)

HLitRecherchePremier(SVGDD_EXT,SVGDD_TYPE,wType)
SI PAS HTrouve(SVGDD_EXT) ALORS
    Erreur("Type "+wType+" : non modifiable en base ...")
    RETOUR
FIN

SI Ouvre(fen_JCsvgDD_ModifType,wType)=Vrai ALORS

    sel_Filtre=i
    ExécuteTraitement(sel_Filtre,trtModification)

FIN
```

Clic sur Image_raz_nfc

```
sai_NomFic_Contient=""
sai_NomFic_Contient..CouleurFond=Blanc
```

Clic sur Image_reste_A__faire

```
wTrav est une chaîne


wTrav=[
- copier un fichier dans le PP (pas que le nom)
- Esc sur le mode frinterruptible
- ...
]

Info (wTrav)
```

Clic sur Image_reste_A__faire1

```
Info("Fichiers dans "+VG_cheminBD)
```

Clic sur Image_reste_A__faire2

```
wImage       est une Image

SI sai_Fic="" ALORS RETOUR

wImage=sai_Fic
SI VersPressePapier(wImage,Faux) ALORS
    Message("Image dans le presse papier ...")
SINON
    Message("Ce n'est pas une image ...")
FIN
```

Initialisation de int_FicRepCaché

```
int_FicRepCaché=Faux
```

A chaque modification de int_FicRepCaché

```
SI int_FicRepCaché=Vrai ALORS
    Info(
    "Attention : avec l'exploration des fichiers cachés, les infos complémentaires (taille, date, ...)
    ne peuvent pas être remontées. "
)
FIN
```

Initialisation de int_ForcerEploration

```
int_ForcerEploration=Faux
```

Initialisation de int_multitache

```
MoiMême=Vrai

SI EnModeTest() ALORS
```

```
    MoiMême..Visible=Vrai
FIN
```

Initialisation de int_récursif

```
int_récursif=Vrai
```

Initialisation de sai_DuréeAffichage

```
// Version 2
// Description
// Saisie d'une durée au format J j H h M m S s
// En entrée de champ, une popup s'ouvre permettant la saisie des valeurs
```

Bouton gauche relâché (WM_LBUTTONUP) de sai_DuréeAffichage

```
// Ouvre la popup d'édition de la durée et lance le formatage
MonChamp = OuvrePopup(FEN_Popup_Saisie_Durée,MonChamp)
```

Initialisation de sai_DuréeExploration

```
// Version 2
// Description
// Saisie d'une durée au format J j H h M m S s
// En entrée de champ, une popup s'ouvre permettant la saisie des valeurs
```

Bouton gauche relâché (WM_LBUTTONUP) de sai_DuréeExploration

```
// Ouvre la popup d'édition de la durée et lance le formatage
MonChamp = OuvrePopup(FEN_Popup_Saisie_Durée,MonChamp)
```

Ajout d'un jeton dans sai_ext

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans sai_ext

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de sai_ext

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Bouton gauche double-clic (WM_LBUTTONDBLCLK) sur sai_ext

```
Info ("Tous les fichiers qui n'ont pas comme extension : ",sai_ext)
```

Ajout d'un jeton dans sai_Fic

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans sai_Fic

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de sai_Fic

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Ajout d'un jeton dans sai_nb_fic

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans sai_nb_fic

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de sai_nb_fic

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Ajout d'un jeton dans sai_nb_rep

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans sai_nb_rep

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de sai_nb_rep

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Ajout d'un jeton dans sai_nb_SousRep

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans sai_nb_SousRep

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de sai_nb_SousRep

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Initialisation de sai_NomFic_Contient

```
sai_NomFic_Contient=""
```

Sortie de sai_NomFic_Contient

```
sai_NomFic_Contient=ChaîneFormate(sai_NomFic_Contient,ccSansAccent+ccSansEspace+ccSansEspaceIntérieur+
ccMajuscule)
```

Ajout d'un jeton dans sai_NomFic_Contient

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans sai_NomFic_Contient

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de sai_NomFic_Contient

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

A chaque modification de sai_NomFic_Contient

```
SI sai_NomFic_Contient<>"" ALORS
    sai_NomFic_Contient..CouleurFond=RougeClair
SINON
    sai_NomFic_Contient..CouleurFond=Blanc
FIN
```

Ajout d'un jeton dans sai_rep

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans sai_rep

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de sai_rep

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Ajout d'un jeton dans sai_RepAexplorer

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans sai_RepAexplorer

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de sai_RepAexplorer

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Initialisation de sel_Filtre

```
sel_Filtre                = 1
sai_ext                   = ",.*,"
sai_ext..Etat             = Grisé
```

A chaque modification de sel_Filtre

```
wType                                                    est une chaîne

wType = sel_Filtre[sel_Filtre]..Libellé
wType=ChaîneFormate(wType,ccSansAccent+ccSansEspace+ccSansEspaceIntérieur+ccSansPonctuationNiEspace+
ccMajuscule)

GP_Initialise_Type()

sai_ext..Etat             = Actif
sai_ext..CouleurFond  = Blanc
sai_ext..PoliceBarrée = Faux
sai_ext..EnSaisie     = Vrai
```

```
SELON wType
    CAS "TOUS" :
        sai_ext                  = ",.*,"
        sai_ext..Etat            = Grisé
    CAS "WORD" :          sai_ext=wWord
    CAS "EXCEL" :         sai_ext=wExcel
    CAS "POWERP" :        sai_ext=wPowerP
    CAS "RTF" :       sai_ext=wRtf
    CAS "PDF" :       sai_ext=wPdf
    CAS "TXT" :       sai_ext=wTxt
    CAS "ZIP" :       sai_ext=wZip
    CAS "LOG" :       sai_ext=wLog
    CAS "PHOTOS" :        sai_ext=wPhotos
    CAS "VIDEOS" :        sai_ext=wVideos
    CAS "SONS" :          sai_ext=wSons
    CAS "EXE" :       sai_ext=wExe
    CAS "SGBD" :          sai_ext=wSgbd
    CAS "HTML" :          sai_ext=wHtml
    CAS "SYSTEME" :   sai_ext=wSysteme
    CAS "CONFIG" :        sai_ext=wConfig
    CAS "INUTILES" :  sai_ext=wInutiles
    CAS "LIBRE" :         sai_ext=wLibre
    CAS "LERESTE" :
        sai_ext=wLeReste
        sai_ext..CouleurFond  = JaunePastel
        sai_ext..PoliceBarrée = Vrai
        sai_ext..EnSaisie     = Faux

    AUTRE CAS :
        sai_ext=",.AUTRECAS,"
        sai_ext..CouleurFond  = OrangePastel

FIN

// on trie les extensions ...
wTable   est un tableau associatif d'entiers
POUR TOUTE CHAÎNE wMaChaine DE sai_ext SÉPARÉE PAR ","
    SI SansEspace(wMaChaine)="" OU SansEspace(wMaChaine)=EOT ALORS CONTINUER
    wTable[wMaChaine]++
FIN
TableauTrie(wTable,ttCroissant)
sai_ext=","
POUR TOUT ÉLÉMENT nOccurrence, nExt DE wTable
    sai_ext=sai_ext+nExt+","
FIN


SI wType<>"LIBRE" ALORS ExécuteTraitement(bou_explorer_DD_Fichier,trtClic)
```

Sélection d'une ligne de Table_Fichier

```
i est un entier
i=TableSelect(Table_Fichier)

sai_rep=Table_Fichier.tf_rep[i]
sai_Fic=Table_Fichier.tf_NomComplet[i]
```

Bouton gauche double-clic (WM_LBUTTONDBLCLK) sur Table_Fichier

```
i est un entier

wTrav est une chaîne = Ouvre(Fen_JCsvgDD_QueFaire)

Message()
SI wTrav="REP" ALORS
     i=TableSelect(Table_Fichier)
     sai_RepAexplorer=tf_rep[i]
     ExécuteTraitement(bou_explorer_DD_Fichier,trtClic)
FIN
SI wTrav="SUP" ALORS
     i=TableSelect(Table_Fichier)
     Message("Fichier supprimé : "+tf_NomComplet[i])
     TableSupprime(Table_Fichier,i)
     TableSelectPlus(Table_Fichier,i)
FIN
SI wTrav="COP" ALORS
     Message("Dans le PP : "+PressePapier())
FIN
```

Ajout d'un jeton dans tf_Attribut (Table_Fichier)

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans tf_Attribut (Table_Fichier)

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de tf_Attribut (Table_Fichier)

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Ajout d'un jeton dans tf_dateModif (Table_Fichier)

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans tf_dateModif (Table_Fichier)

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de tf_dateModif (Table_Fichier)

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Ajout d'un jeton dans tf_extension (Table_Fichier)

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans tf_extension (Table_Fichier)

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de tf_extension (Table_Fichier)

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Ajout d'un jeton dans tf_fic (Table_Fichier)

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans tf_fic (Table_Fichier)

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de tf_fic (Table_Fichier)

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Ajout d'un jeton dans tf_NomComplet (Table_Fichier)

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans tf_NomComplet (Table_Fichier)

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de tf_NomComplet (Table_Fichier)

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Ajout d'un jeton dans tf_rep (Table_Fichier)

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans tf_rep (Table_Fichier)

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de tf_rep (Table_Fichier)

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Ajout d'un jeton dans tf_taille (Table_Fichier)

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans tf_taille (Table_Fichier)

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de tf_taille (Table_Fichier)

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Bouton gauche double-clic (WM_LBUTTONDBLCLK) sur Table_Rep1

```
i est un entier = TableSelect(Table_Rep1)

SI i<1 ALORS RETOUR

wListeRep        est une chaîne
TableSupprimeTout(Table_SousRep)
wListeRep=fListeRépertoire(tr1_rep[i],frNonRécursif)
POUR TOUTE CHAÎNE MonRep DE wListeRep SÉPARÉE PAR RC
    SI Contient(MonRep,"$")   ALORS CONTINUER      // c'est un rep système
    TableAjouteLigne(Table_SousRep,MonRep)
FIN
TableTrie(Table_SousRep,"tsr_rep")
TableSelectMoins(Table_SousRep)
sai_nb_SousRep=Table_SousRep..Occurrence
Message()


sai_RepAexplorer=tr1_rep[i]

ExécuteTraitement(bou_explorer_DD_Fichier,trtClic)
```

Ajout d'un jeton dans tr1_rep (Table_Rep1)

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans tr1_rep (Table_Rep1)

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de tr1_rep (Table_Rep1)

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Bouton gauche double-clic (WM_LBUTTONDBLCLK) sur Table_SousRep

```
i est un entier = TableSelect(Table_SousRep)

SI i<1 ALORS RETOUR

sai_RepAexplorer=tsr_rep[i]

ExécuteTraitement(bou_explorer_DD_Fichier,trtClic)
```

Ajout d'un jeton dans tsr_rep (Table_SousRep)

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans tsr_rep (Table_SousRep)

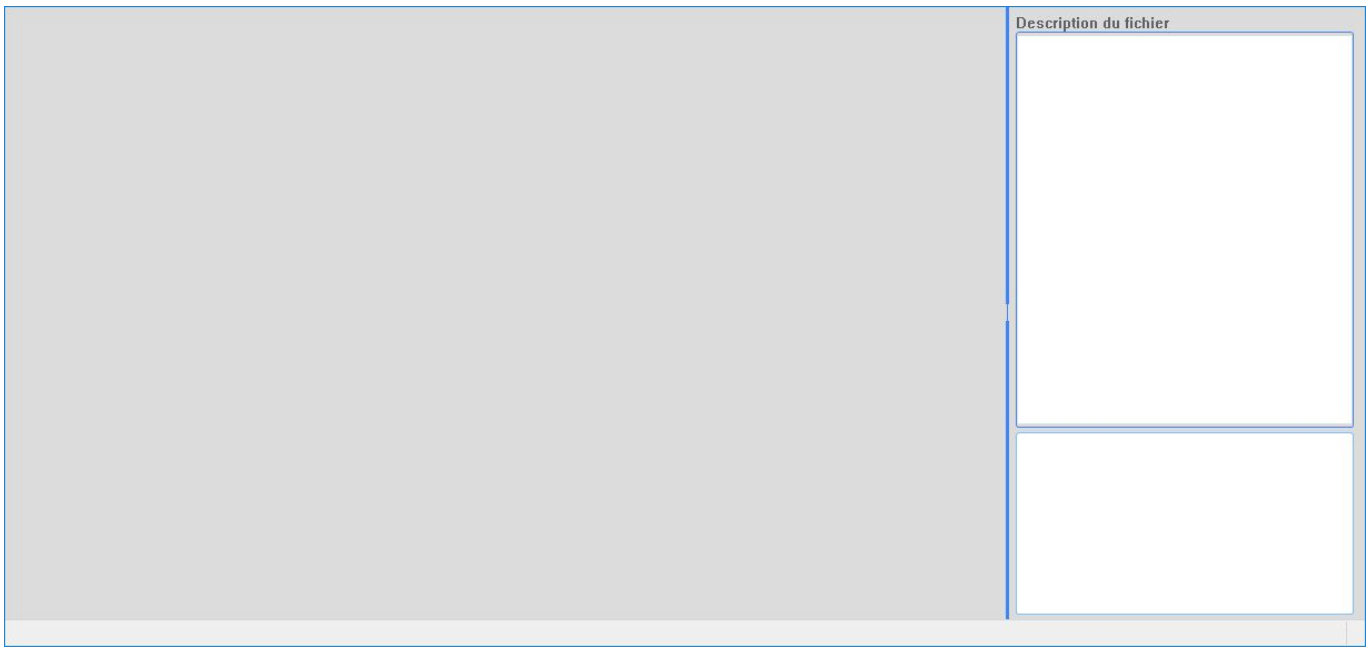
```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

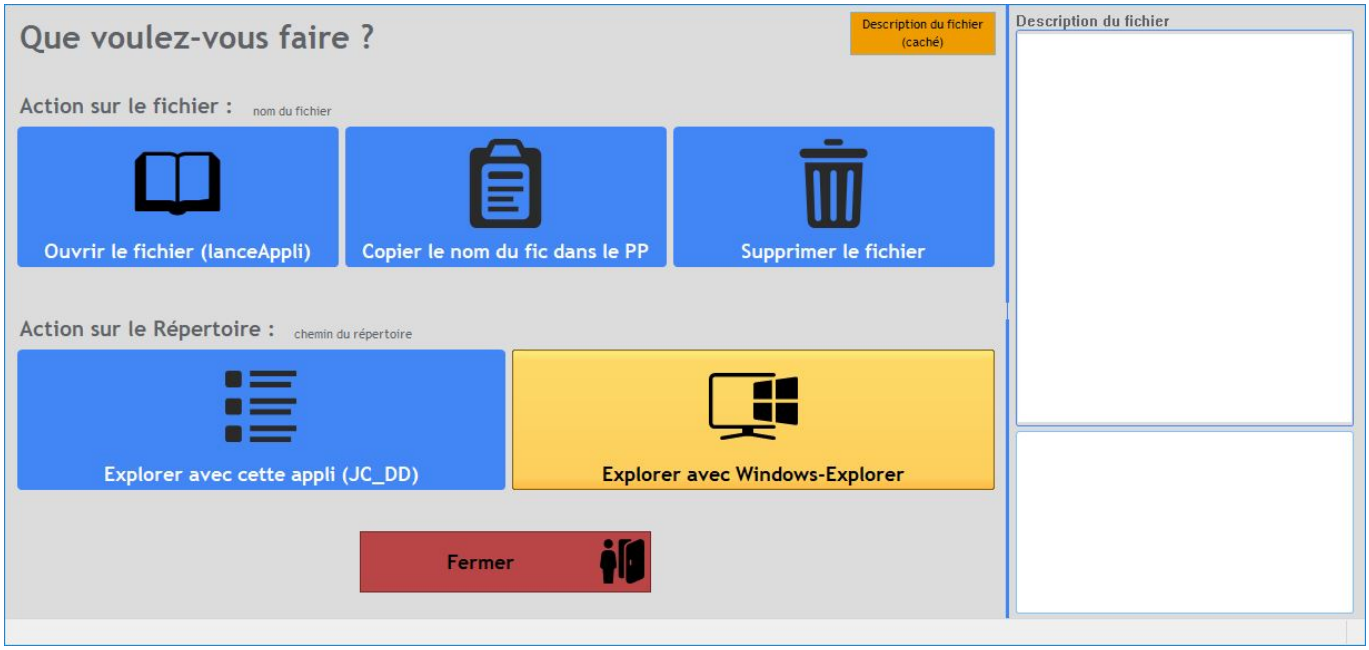
Clic sur un jeton de tsr_rep (Table_SousRep)

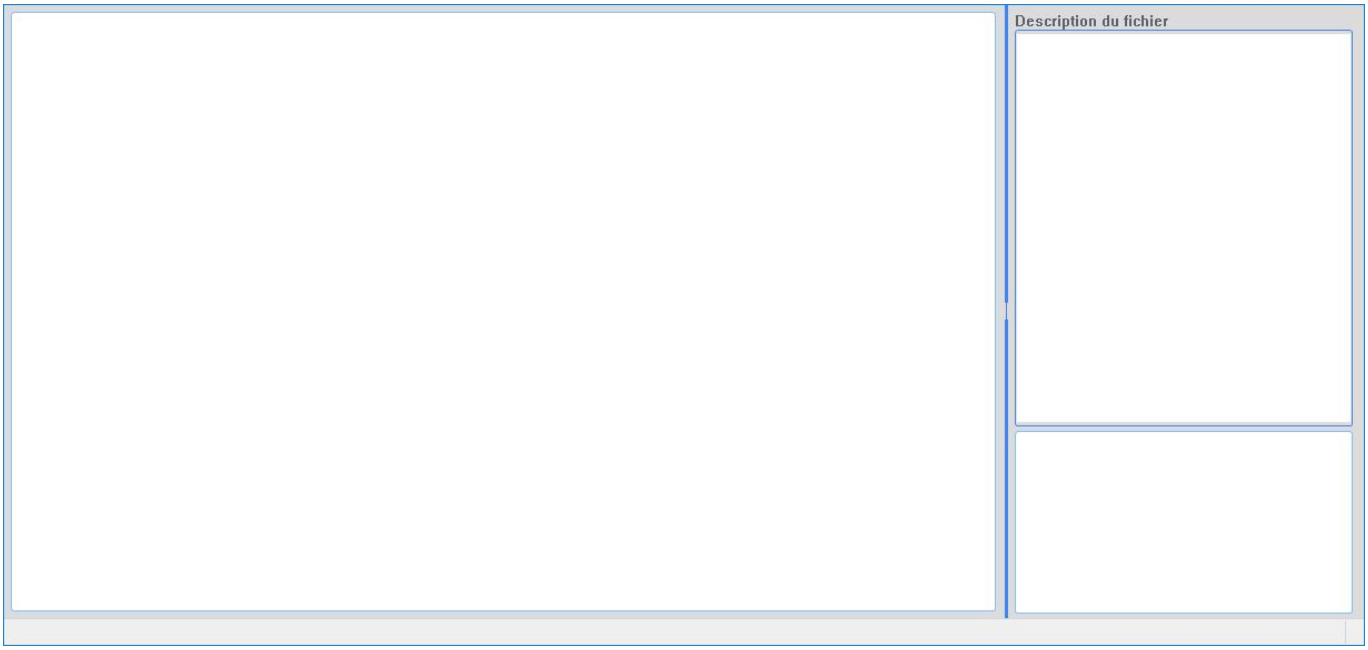
```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Fen_JCsvgDD_QueFaire

Image




Que voulez-vous faire ?
Description du fichier (caché)
Description du fichier
Action sur le fichier :
nom du fichier
Ouvrir le fichier (lanceAppli)
Copier le nom du fic dans le PP
Supprimer le fichier
Action sur le Répertoire :
chemin du répertoire
Explorer avec cette appli (JC_DD)
Explorer avec Windows-Explorer
Fermer



Fen_JCsvgDD_QueFaire

Code

Déclarations globales de Fen_JCsvgDD_QueFaire

```
PROCÉDURE MaFenêtre()
```

Fin d'initialisation de Fen_JCsvgDD_QueFaire

```
MaFenêtre.Plan=1

i est un entier = TableSelect(fen_JCsvgDD_ListeFic.Table_Fichier)


Lib_fichier=fen_JCsvgDD_ListeFic.tf_fic[i]
Lib_Répertoire=fen_JCsvgDD_ListeFic.tf_rep[i]

ExécuteTraitement(bou_desc,trtClic)
```

Fen_JCsvgDD_QueFaire

Code des champs

Clic sur bou_Copier

```
i est un entier = TableSelect(fen_JCsvgDD_ListeFic.Table_Fichier)

VersPressePapier(fen_JCsvgDD_ListeFic.tf_NomComplet[i])
Ferme(Fen_JCsvgDD_QueFaire,"COP")
```

Clic sur bou_desc

```
wTrav                est une chaîne
wNomFic              est une chaîne    = Lib_Répertoire+Lib_fichier
wEext                est une chaîne    = ","+Majuscule(fExtraitChemin(wNomFic,fExtension))+","

wListNonOuvrable est une chaîne    = ",.EXE,.COM,.BAT,.DLL,"
wListOuvrable        est une chaîne    = ",.LOG,.TXT,"

SI (Contient(wListNonOuvrable,wEext) OU fTypeMIME(wNomFic)="") ET PAS Contient(wListOuvrable,wEext)
ALORS
    bou_Ouvre_Fic..Etat=Grisé
SINON
    bou_Ouvre_Fic..Etat=Actif
FIN

wTrav="Description du fichier :"+RC+RC
wTrav+=[RC]+"Nom : "+Lib_fichier
wTrav+=[RC]+"Répertoire : "+Lib_Répertoire
wTrav+=[RC]+"Rep. court : "+fCheminCourt(wNomFic)
wTrav+=[RC]+"Extension : "+fExtraitChemin(wNomFic,fExtension)
wTrav+=[RC]+"Disque : "+fExtraitChemin(wNomFic,fDisque)
wTrav+=[RC]+"Date Création : "+DateHeureVersChaîne(fDateHeure(wNomFic,"",fCréation))
wTrav+=[RC]+"Date Modification : "+DateHeureVersChaîne(fDateHeure(wNomFic,"",fModification))
wTrav+=[RC]+"Date Accès : "+DateHeureVersChaîne(fDateHeure(wNomFic,"",fAccès))
wTrav+=[RC]+"Attributs : "+fAttribut(wNomFic)
wTrav+=[RC]+"Taille : "+SansEspace(NumériqueVersChaîne(fTaille(wNomFic),"12dS"))+" Octets"
wTrav+=[RC]+"Type MIME : "+fTypeMIME(wNomFic)
SI fTypeMIME(wNomFic)[="image" ALORS
    Image_fic        = wNomFic
    Image_fic_plan2  = ""
SINON
    Image_fic        = ""
    Image_fic_plan2  = ""
FIN
SI fTaille(wNomFic)<10000000 ALORS
    SI fEstUneImage(wNomFic)=Vrai ALORS
        wTrav+=[RC]+"Type image : Oui"
    SINON
        wTrav+=[RC]+"Type image : Non"
    FIN
FIN

sai_desc=wTrav
```

Clic sur bou_Lancer_explorer

```
i est un entier = TableSelect(fen_JCsvgDD_ListeFic.Table_Fichier)
```

```
LanceAppliAssociée(fen_JCsvgDD_ListeFic.tf_rep[i])
Ferme(Fen_JCsvgDD_QueFaire,"")
```

Clic sur bou_Ouvre_Fic

```
i est un entier = TableSelect(fen_JCsvgDD_ListeFic.Table_Fichier)

SI PAS LanceAppliAssociée(fen_JCsvgDD_ListeFic.tf_NomComplet[i]) ALORS
    Erreur("Impossible de lancer une appli pour ce fichier ...")
FIN
```

Clic sur bou_Rep

```
Ferme(Fen_JCsvgDD_QueFaire,"REP")
```

Clic sur bou_Rien

```
Ferme(Fen_JCsvgDD_QueFaire,"")
```

Clic sur bou_SupprimerFic

```
i est un entier = TableSelect(fen_JCsvgDD_ListeFic.Table_Fichier)

SI PAS OuiNon(Non,"Voulez vous supprimer le fichier : "+Lib_fichier+" ?") ALORS
    RETOUR
FIN

SI fSupprime(fen_JCsvgDD_ListeFic.Table_Fichier.tf_NomComplet[i]) ALORS
    Ferme(Fen_JCsvgDD_QueFaire,"SUP")
SINON
    Message("Suppression impossible du fichier "+Lib_fichier+" ...")
FIN
```

Clic sur Image_fic

```
SI Image_fic<>"" ET MaFenêtre..Plan=1 ALORS
    Image_fic_plan2=Image_fic
    MaFenêtre..Plan=2
SINON
    MaFenêtre..Plan=1
FIN
```

Clic sur Image_fic_plan2

```
MaFenêtre..Plan=1
Image_fic_plan2=""
```

Initialisation de Lib_fichier

```
// Version 1
// Description
// Libellé simple
```

Initialisation de Lib_Répertoire

```
// Version 1
// Description
// Libellé simple
```

Initialisation de Libellé

```
// Version 1
// Description
// Libellé pour écrire un titre
```

Initialisation de Libellé1

```
// Version 1
// Description
// Libellé1 pour écrire un titre
```

Initialisation de Libellé2

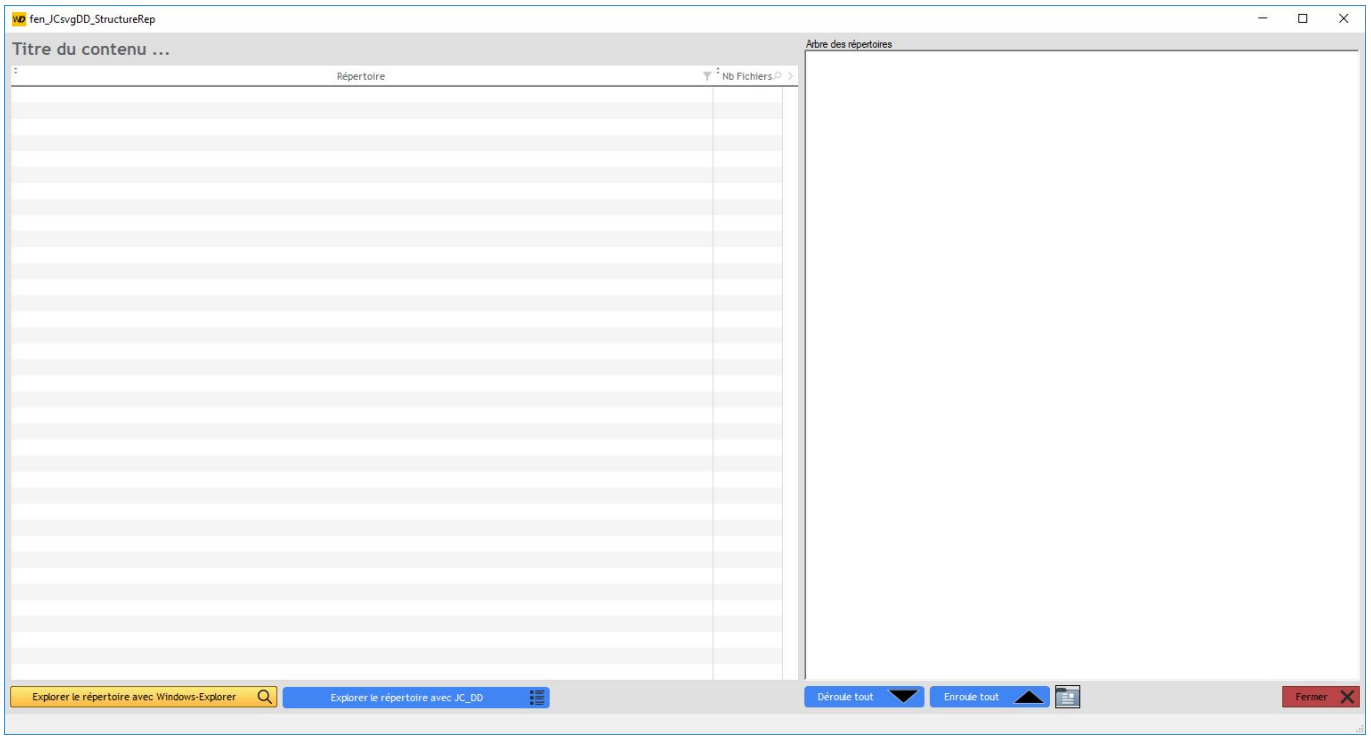
```
// Version 1
// Description
// Libellé2 pour écrire un titre
```

Initialisation de sai_desc

```
// Version 1
// Description
// Champ de saisie pour un texte multiligne simple, sans mise en forme
```

fen_JCsvgDD_StructureRep

Image



fen_JCsvgDD_StructureRep

Code

Déclarations globales de fen_JCsvgDD_StructureRep

```
PROCÉDURE MaFenêtre(parTitre est une chaîne, parTableRep est un tableau associatif d'entiers)
```

Fin d'initialisation de fen_JCsvgDD_StructureRep

```
lib_Titre=parTitre

// on créé l'arbre
wTrav            est une chaîne
i,j              sont des entiers
wCnGroupe        est une chaîne
wCarSep          est une chaîne


TableSupprimeTout(Table_SR)
ArbreSupprimeTout(ArbreGroupe)
Sablier(Vrai)

//
=============================================================================================================
======= on charge la table

TableauTrie(parTableRep,ttDécroissant)
POUR TOUT ÉLÉMENT nOccurrence, nRépertoire DE parTableRep
    TableAjouteLigne(Table_SR,nRépertoire,nOccurrence)
FIN
TableTrie(Table_SR,"tsr_rep")
TableSelectMoins(Table_SR)

//
=============================================================================================================
======= INIT de l'arbre


// j'ajoute la racine
wTrav="Disque"
ArbreAjoute(ArbreGroupe,"Disque",aDéfaut,aDéfaut,"",aDernier)

j=TableOccurrence(Table_SR)
POUR i=1 À j

    wCnGroupe    = tsr_rep[i]

    wCarSep      = "\"
    wTrav        = "Disque"
    POUR TOUTE CHAÎNE machaine DE wCnGroupe SÉPARÉE PAR wCarSep
        wTrav+=[TAB]+machaine
    FIN

    // ArbreAjoute(ArbreGroupe,wTrav,aDéfaut,aDéfaut,wCnGroupe,aDernier)
    // ArbreAjoute(ArbreGroupe,wTrav,aDéfaut,aDéfaut,wCnGroupe,aDernier)
    ArbreAjoute(ArbreGroupe,wTrav,image_fichier,image_fichier,wCnGroupe,aDernier)
FIN

ArbreTrie(ArbreGroupe,Null,aAscendant)
ArbreDéroule(ArbreGroupe,"Disque")
ArbreSelectMoins(ArbreGroupe)

Wait(100)
Sablier(Faux)
```

fen_JCsvgDD_StructureRep

Code des champs

Sélection d'une ligne de ArbreGroupe

```
resChemin         est une chaîne
wCnGroupe         est une chaîne
i                 est un entier

// on récupère la structure de l'arbre sur laquelle on a cliqué
resChemin     = ArbreSelect(ArbreGroupe)
wCnGroupe     = ArbreRécupèreIdentifiant(ArbreGroupe,resChemin)
wCnGroupe     = SansEspace(wCnGroupe)

SI wCnGroupe<>"" ALORS
    i=TableCherche(tsr_rep,wCnGroupe)
    SI i>0 ALORS
        TableSelectPlus(Table_SR,i)
        TableAffiche(Table_SR,i)
    SINON
        TableSelectMoins(Table_SR)
    FIN
FIN
```

Clic sur Bou_Selection_Rep

```
i est un entier = TableSelect(Table_SR)

SI i<1 ALORS
    Message("Sélectionner une ligne répertoire ... ")
    RETOUR
FIN

LanceAppliAssociée(tsr_rep[i])
```

Clic sur Bou_Selection_Rep1

```
i est un entier = TableSelect(Table_SR)

SI i<1 ALORS
    Message("Sélectionner une ligne répertoire ... ")
    RETOUR
FIN

Ferme(fen_JCsvgDD_StructureRep,tsr_rep[i])
```

Clic sur Déroule_tout

```
ArbreDérouleTout(ArbreGroupe,"Disque")
```

Clic sur Déroule_tout1

```
ArbreEnrouleTout(ArbreGroupe,"Disque")
```

Initialisation de Fermer

```
// Version 1
// Description
// Bouton permettant de fermer la fenêtre
```

Clic sur Fermer

```
Ferme(fen_JCsvgDD_StructureRep,"")
```

Initialisation de lib_Titre

```
// Version 1
// Description
// Libellé pour écrire un titre
```

Sélection d'une ligne de Table_SR

```
i            est un entier    = TableSelect(Table_SR)
wResultat    est une chaîne

wResultat=ArbreCherche(ArbreGroupe,tsr_rep[i],Null,aFeuille)

SI wResultat<>"" ALORS
    ArbreSelectPlus(ArbreGroupe,wResultat)
    ArbreAffiche(ArbreGroupe)
FIN
```

Ajout d'un jeton dans tsr_nb (Table_SR)

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans tsr_nb (Table_SR)

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de tsr_nb (Table_SR)

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Ajout d'un jeton dans tsr_rep (Table_SR)

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
```

```
RENVOYER Vrai
```

Suppression d'un jeton dans tsr_rep (Table_SR)

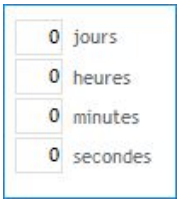
```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de tsr_rep (Table_SR)

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

FEN_Popup_Saisie_Durée

Image



FEN_Popup_Saisie_Durée

Code

Déclarations globales de FEN_Popup_Saisie_Durée

```
//--- Fenêtre Popup ---
// Pour accéder au champ ou à la combo popup qui a ouvert cette fenêtre
// vous pouvez utiliser : MonChampPopup
PROCÉDURE MaFenêtre(LOCAL gduValeur est une Durée)

 SC_Durée = gduValeur
 DonneFocus(SAI_Jours)
```

Fermeture de FEN_Popup_Saisie_Durée

```
RENVOYER  SC_Durée
```

FEN_Popup_Saisie_Durée

Code des champs

Clic sur BTN_Ferme

Affectation de la propriété Valeur de SC_Durée

```
SAI_Jours = gduValeur..Jour
SAI_Heures = gduValeur..Heure
SAI_Minutes = gduValeur..Minute
SAI_Secondes = gduValeur..Seconde
```

Récupération de la propriété Valeur de SC_Durée

```
duValeur est une Durée
duValeur..Jour = SAI_Jours
duValeur..Heure = SAI_Heures
duValeur..Minute = SAI_Minutes
duValeur..Seconde = SAI_Secondes

RENVOYER duValeur
```

Sortie de SAI_Heures (SC_Durée)

```
Maj()
```

Sortie de SAI_Jours (SC_Durée)

```
Maj()
```

Sortie de SAI_Minutes (SC_Durée)

```
Maj()
```

Sortie de SAI_Secondes (SC_Durée)

```
Maj()
```

FEN_Popup_Saisie_Durée

Procédures

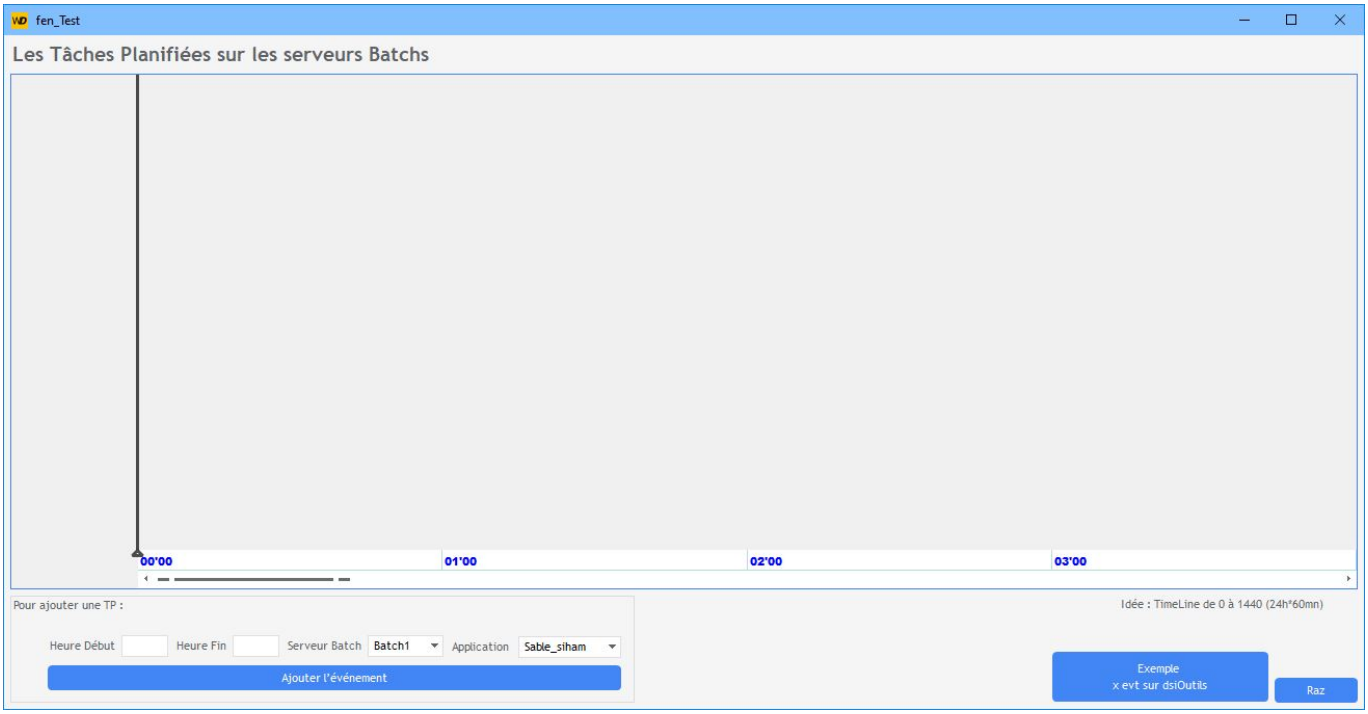
Procédure locale Maj (SC_Durée)

```
// Résumé : <indiquez ici ce que fait la procédure>
// Syntaxe :
// Maj ()
//
// Paramètres :
//   Aucun
// Valeur de retour :
//   Aucune
//
// Exemple :
// Indiquez ici un exemple d'utilisation.
//
PROCÉDURE Maj()

// Le type durée fait la correction en cas de dépassement des valeurs par composante de temps
duValeur est une Durée
duValeur..EnJours += SAI_Jours
duValeur..EnHeures += SAI_Heures
duValeur..EnMinutes += SAI_Minutes
duValeur..EnSecondes += SAI_Secondes

// Met à jour l'affichage
SAI_Jours = duValeur..Jour
```

```
SAI_Heures = duValeur..Heure
SAI_Minutes = duValeur..Minute
SAI_Secondes = duValeur..Seconde
```

fen_TimeLine

Image



fen_TimeLine

Code

Déclarations globales de fen_TimeLine

```
PROCÉDURE MaFenêtre()

VG_ListePiste est une chaîne = ""
```

Fin d'initialisation de fen_TimeLine

```
TimelineSupprimeTout(TL_TP)

VG_ListePiste=""

Maximise()
```

fen_TimeLine

Code des champs

Clic sur bou_Ajouter

```
Evt est un EvénementTimeline

SI Sai_Debut>=Sai_Fin ALORS
    Erreur("Une tache qui se termine avant d'avoir commencé ?","Minimum = 1 mn")
    Sai_Fin=EntierVersHeure(HeureVersEntier(Sai_Debut)+6000)
    RETOUR
FIN


SI PAS Contient(VG_ListePiste,";"+Combo_appli..ValeurAffichée+";") ALORS
    TimelineAjoutePiste(TL_TP, Combo_appli..ValeurAffichée)
    VG_ListePiste+=";"+Combo_appli..ValeurAffichée+";"
FIN

Evt.Piste        = Combo_appli..ValeurAffichée
Evt.Titre        = Combo_batch..ValeurAffichée
Evt.Début        = GP_heure_vers_minute(Sai_Debut)
Evt.Fin          = GP_heure_vers_minute(Sai_Fin)
Evt.CouleurFond  = OrangePastel
SI Combo_batch..ValeurAffichée="Batch0" ALORS Evt.CouleurFond    = JaunePastel
SI Combo_batch..ValeurAffichée="Batch1" ALORS Evt.CouleurFond    = VertPastel
SI Combo_batch..ValeurAffichée="Batch2" ALORS Evt.CouleurFond    = RougePastel
SI Combo_batch..ValeurAffichée="Batch3" ALORS Evt.CouleurFond    = BleuPastel
Evt.Bulle=ChaîneConstruit("Traitement %1 de %2 à %3 sur %4",Combo_appli..ValeurAffichée,HeureVersChaîne
(Sai_Debut,"HH:MM"),HeureVersChaîne(Sai_Fin,"HH:MM"),Combo_batch..ValeurAffichée)
TimelineAjouteEvénement(TL_TP, Evt)


PROCÉDURE INTERNE GP_heure_vers_minute(parHeure)

    i,j  sont des entiers

    i=Val(parHeure[[1 À 2]])*60
    j=Val(parHeure[[3 À 4]])
    i=i+j

    RENVOYER i

FIN
```

Clic sur Bou_exemple

```
Combo_appli=4

Combo_batch=2
Sai_Debut="0150"
Sai_Fin="0155"
ExécuteTraitement(bou_Ajouter,trtClic)

Combo_batch  = 1
Sai_Debut    = "0351"
Sai_Fin      = "0423"
ExécuteTraitement(bou_Ajouter,trtClic)

Combo_batch  = 3
Sai_Debut    = "0425"
Sai_Fin      = "0426"
ExécuteTraitement(bou_Ajouter,trtClic)

Combo_batch  = 3
Sai_Debut    = "0523"
Sai_Fin      = "0526"
ExécuteTraitement(bou_Ajouter,trtClic)

Combo_batch  = 3
Sai_Debut    = "0525"
Sai_Fin      = "0533"
ExécuteTraitement(bou_Ajouter,trtClic)
```

Clic sur bou_Raz

```
TimelineSupprimeTout(TL_TP)

VG_ListePiste=""
```

Initialisation de Combo_appli

```
// Version 1
// Description
// Combo liste simple
```

Initialisation de Combo_batch

```
// Version 1
// Description
// Combo liste simple
```

Initialisation de Libellé

```
// Version 1
// Description
// Libellé simple
```

Initialisation de Libellé1

```
// Version 1
// Description
// Libellé pour écrire un titre
```

Initialisation de Sai_Debut

```
// Version 1
// Description
// Saisie d'une heure

MoiMême="0100"
```

Initialisation de Sai_Fin

```
// Version 1
// Description
// Saisie d'une heure

Sai_Fin="0115"
```

Avant création de l'événement dans TL_TP

```
PROCÉDURE AvantCréation(evtCréé est un EvénementTimeline)
```

Entrée en saisie d'un événement de TL_TP

```
PROCÉDURE EntréeEnSaisie(evtEdité est un EvénementTimeline)
```

Sortie de saisie de l'événement de TL_TP

```
PROCÉDURE SortieDeSaisie(evtModifié est un EvénementTimeline)
```

Sélection d'un événement de TL_TP

```
PROCÉDURE Sélection(evtSélectionné est un EvénementTimeline)
```

Déplacement d'un événement dans TL_TP

```
PROCÉDURE Déplacement(evtDéplacé est un EvénementTimeline)
```

Redimensionnement d'un événement de TL_TP

```
PROCÉDURE Redimensionnement(evtRedimensionné est un EvénementTimeline)
```

Suppression d'un événement de TL_TP

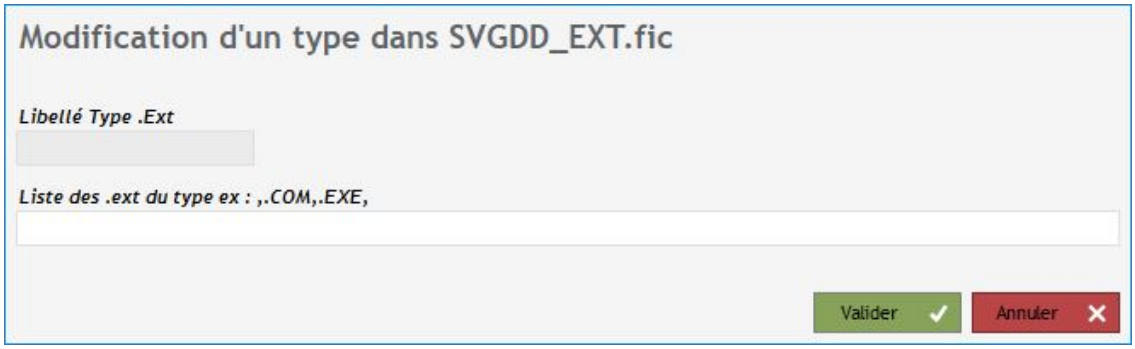
```
PROCÉDURE Suppression(evtSupprimé est un EvénementTimeline)
```

Réaffectation d'un événement dans TL_TP

```
PROCÉDURE DéplacementPiste(evtPisteDéplacé est un EvénementTimeline)
```

fen_JCsvgDD_ModifType

Image



fen_JCsvgDD_ModifType

Code

Déclarations globales de fen_JCsvgDD_ModifType

```
//--- Fenêtre Popup ---
// Pour accéder au champ ou à la combo popup qui a ouvert cette fenêtre
// vous pouvez utiliser : MonChampPopup
PROCÉDURE MaFenêtre(partype)
```

Fin d'initialisation de fen_JCsvgDD_ModifType

```
HLitRecherchePremier(SVGDD_EXT,SVGDD_TYPE,partype)
SI PAS HTrouve(SVGDD_EXT) ALORS
    Erreur(partype+" : non trouvé dans le fichier SVGDD_EXT.fic ...")
    Ferme(fen_JCsvgDD_ModifType,Faux)
FIN

sai_SVGDD_TYPE=SVGDD_EXT.SVGDD_TYPE
sai_Ext=SVGDD_EXT.SVGDD_LISTE_EXT
```

fen_JCsvgDD_ModifType

Code des champs

Initialisation de Annuler

```
// Version 1
```

```
// Description
// Bouton en annulation
```

Clic sur Annuler

```
Ferme(fen_JCsvgDD_ModifType,Faux)
```

Initialisation de Libellé

```
// Version 1
// Description
// Libellé pour écrire un titre
```

Ajout d'un jeton dans sai_Ext

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans sai_Ext

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de sai_Ext

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Ajout d'un jeton dans sai_SVGDD_TYPE

```
PROCÉDURE AjoutJeton (MonJeton est un Jeton)

//RENVOYER Faux pour interdire l'ajout du jeton
RENVOYER Vrai
```

Suppression d'un jeton dans sai_SVGDD_TYPE

```
PROCÉDURE SuppressionJeton(MonJeton est un Jeton)

//RENVOYER Faux pour interdire la suppression du jeton
RENVOYER Vrai
```

Clic sur un jeton de sai_SVGDD_TYPE

```
PROCÉDURE ClicJeton (MonJeton est un Jeton)
```

Initialisation de Valider

```
// Version 1
```

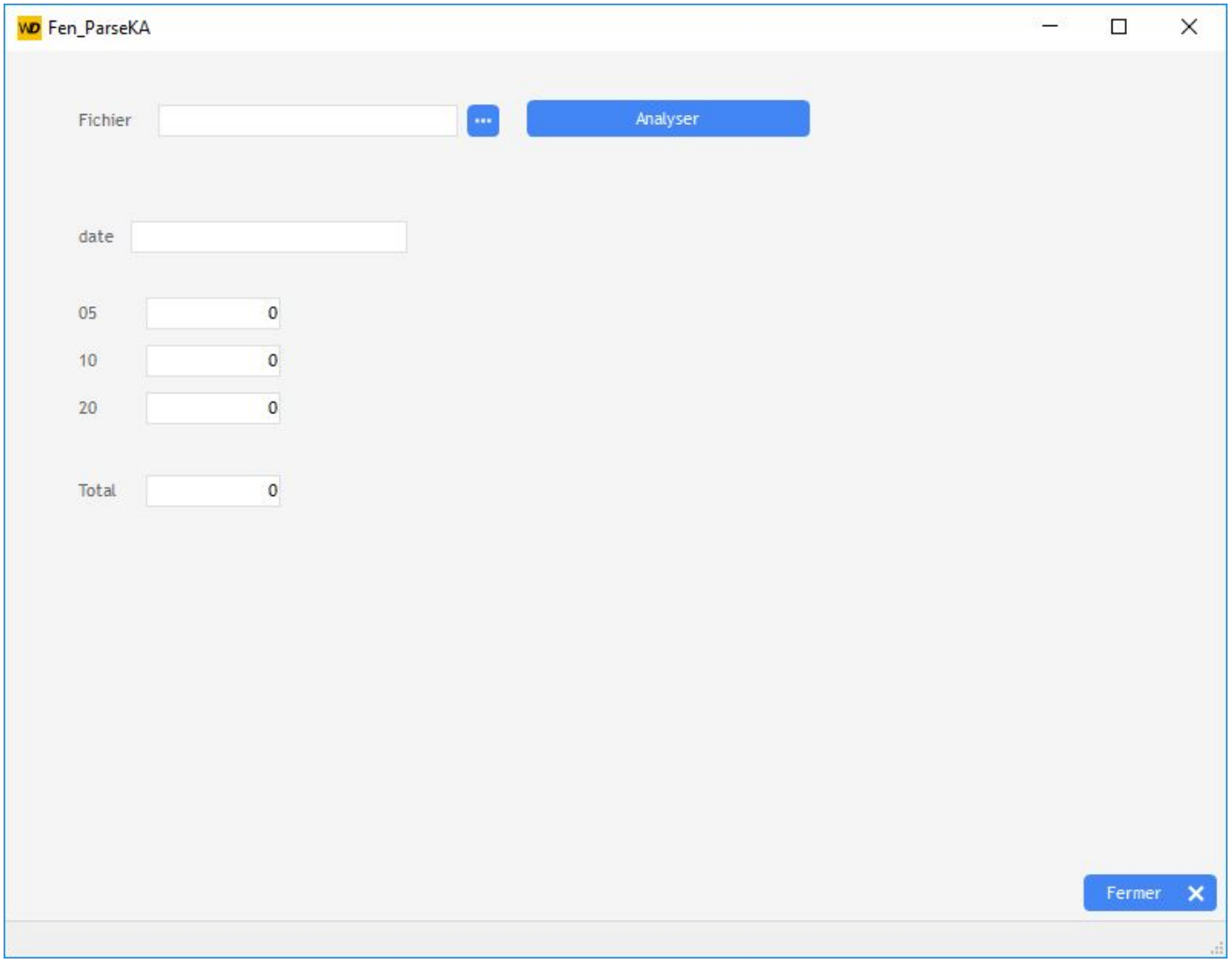
```
// Description
// Bouton en validation
```

Clic sur Valider

```
// on contrôle
sai_Ext=Majuscule(sai_Ext)
SI Gauche(sai_Ext,1)<>"," ALORS sai_Ext=","+sai_Ext
SI Droite(sai_Ext,1)<>"," ALORS sai_Ext=sai_Ext+","
sai_Ext  = Remplace(sai_Ext," ",",")
sai_Ext  = Remplace(sai_Ext,",,",",")
sai_Ext  = ChaîneFormate(sai_Ext,ccSansAccent+ccSansEspace+ccSansEspaceIntérieur+ccMajuscule)

POUR TOUTE CHAÎNE monExt DE sai_Ext SÉPARÉE PAR ","
    SI monExt="" ALORS CONTINUER
    SI Gauche(monExt,1)<>"." ALORS
        Erreur("Ce n'est pas une extension de la forme '.x' : "+monExt)
        RETOUR
    FIN
FIN

// on enregistre
HLitRecherchePremier(SVGDD_EXT,SVGDD_TYPE,partype)
SVGDD_EXT.SVGDD_LISTE_EXT= sai_Ext
HModifie(SVGDD_EXT)

// on ferme
Ferme(fen_JCsvgDD_ModifType,Vrai)
```

Fen_ParseKA

Image



Fen_ParseKA

Code

Déclarations globales de Fen_ParseKA

```
PROCÉDURE MaFenêtre()
```

Fen_ParseKA

Code des champs

Clic sur Bouton1

```
wKA est une chaîne

i,j sont des entiers

wNbLigne est un entier =0
wNb05        est un entier =0
wNb10        est un entier =0
wNb20        est un entier =0
wNbx     est un entier =0


wKA=fChargeTexte(sai_Fichier)

i=0 ; j=ChaîneOccurrence(wKA,RC)+1

POUR TOUTE CHAÎNE MaLigne DE wKA SÉPARÉE  PAR RC

    SI SansEspace(MaLigne)="" OU SansEspace(MaLigne)=EOT ALORS CONTINUER

    Jauge(i,j)

    // si sai_date="" alors sai_date=MaLigne[[35 À 36]]+"-"+MaLigne[[33 À 34]]+"-"+MaLigne[[29 À 32]]
    sai_date=MaLigne[[35 À 36]]+"-"+MaLigne[[33 À 34]]+"-"+MaLigne[[29 À 32]]

    wNbLigne++
    SI MaLigne[[37 À 38]]="05" ALORS wNb05++
    SI MaLigne[[37 À 38]]="10" ALORS wNb10++
    SI MaLigne[[37 À 38]]="20" ALORS wNb20++

FIN


sai_TOT=wNbLigne
sai_05=wNb05
sai_10=wNb10
sai_20=wNb20
```

Initialisation de Fermer

```
// Version 1
// Description
// Bouton permettant de fermer la fenêtre
```

Clic sur Fermer

Clic sur Fichier1

```
sFichier est une chaîne
```

```
// Ouvre le sélecteur de fichiers
soit sCheminOuverture = fExtraitChemin(sai_Fichier,fDisque+fRépertoire)
soit sFichierInitial = fExtraitChemin(sai_Fichier,fFichier+fExtension)
soit sTitre="Sélectionner un fichier..."
soit sFiltre="Tous fichiers (*.*)"+TAB+"*.*"
// ouvre le sélecteur
sFichier = fSélecteur(sCheminOuverture, sFichierInitial, sTitre, sFiltre, "", fselOuvre + fselExiste)

SI sFichier<>"" ALORS
    sai_Fichier =  sFichier
    ExécuteTraitement(sai_Fichier,trtModification)
FIN
```

Initialisation de sai_05

```
// Version 1
// Description
// Saisie d'un nombre entier
```

Initialisation de sai_10

```
// Version 1
// Description
// Saisie d'un nombre entier
```

Initialisation de sai_20

```
// Version 1
// Description
// Saisie d'un nombre entier
```

Initialisation de sai_date

```
// Version 1
// Description
// Champ de saisie pour un texte simple sur une ligne
```

Initialisation de sai_Fichier

```
// Version 1
// Description
// Sélecteur de fichier

MoiMême="D:\Temp\KA\PAY_KX_A131"
```

Initialisation de sai_TOT

```
// Version 1
// Description
// Saisie d'un nombre entier
```

Partie 3

Collection de procédures

ProcéduresGlobales

Code

Procédure globale GP_Initialise_Type

```
// Résumé : <indiquez ici ce que fait la procédure>
// Syntaxe :
// GP_Initialise_Type ()
//
// Paramètres :
//  Aucun
// Valeur de retour :
//  Aucune
//
// Exemple :
// <Indiquez ici un exemple d'utilisation>
//
PROCÉDURE GP_Initialise_Type()

// 1
HLitRechercherPremier(SVGDD_EXT,SVGDD_TYPE,"WORD")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="WORD"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.DOC,.DOCX,.ODT,"
    HAjoute(SVGDD_EXT)
FIN
wWord=SVGDD_EXT.SVGDD_LISTE_EXT
// 2
HLitRechercherPremier(SVGDD_EXT,SVGDD_TYPE,"EXCEL")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="EXCEL"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.XLS,.XLSX,.XLSM,.CSV,.ODS,"
    HAjoute(SVGDD_EXT)
FIN
wExcel=SVGDD_EXT.SVGDD_LISTE_EXT
// 3
HLitRechercherPremier(SVGDD_EXT,SVGDD_TYPE,"POWERP")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="POWERP"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.PPT,.PPSX,.PPTX,.ODP,.PUB,"
    HAjoute(SVGDD_EXT)
FIN
wPowerP=SVGDD_EXT.SVGDD_LISTE_EXT
// 4
HLitRechercherPremier(SVGDD_EXT,SVGDD_TYPE,"RTF")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="RTF"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.RTF,"
    HAjoute(SVGDD_EXT)
FIN
wRtf=SVGDD_EXT.SVGDD_LISTE_EXT
// 5
HLitRechercherPremier(SVGDD_EXT,SVGDD_TYPE,"PDF")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
```

```
    SVGDD_EXT.SVGDD_TYPE="PDF"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.PDF,.SVG,.ODG,"
    HAjoute(SVGDD_EXT)
FIN
wPdf=SVGDD_EXT.SVGDD_LISTE_EXT
// 6
HLitRechercherPremier(SVGDD_EXT,SVGDD_TYPE,"TXT")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="TXT"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.TXT,.LST,.TMP,"
    HAjoute(SVGDD_EXT)
FIN
wTxt=SVGDD_EXT.SVGDD_LISTE_EXT
// 7
HLitRechercherPremier(SVGDD_EXT,SVGDD_TYPE,"ZIP")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="ZIP"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.ZIP,.TAR,.RAR,.7Z,.GZ,.TARGZ,.JAR,.BIAR,.WAR,.BACKUP,"
    HAjoute(SVGDD_EXT)
FIN
wZip=SVGDD_EXT.SVGDD_LISTE_EXT
// 8
HLitRechercherPremier(SVGDD_EXT,SVGDD_TYPE,"LOG")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="LOG"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.LOG,.OLD,"
    HAjoute(SVGDD_EXT)
FIN
wLog=SVGDD_EXT.SVGDD_LISTE_EXT
// 9
HLitRechercherPremier(SVGDD_EXT,SVGDD_TYPE,"PHOTOS")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="PHOTOS"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.JPG,.JPEG,.GIF,.TIF,.PNG,.SVG,.BMP,.TIFF,.ICO,"
    HAjoute(SVGDD_EXT)
FIN
wPhotos=SVGDD_EXT.SVGDD_LISTE_EXT
// 10
HLitRechercherPremier(SVGDD_EXT,SVGDD_TYPE,"VIDEOS")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="VIDEOS"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.MP4,.M4V,.MOV,.AVI,.FLV,.MPEG,.MPG,.VOB,.WMV,.MTS,.M4A,"
    HAjoute(SVGDD_EXT)
FIN
wVideos=SVGDD_EXT.SVGDD_LISTE_EXT
// 11
HLitRechercherPremier(SVGDD_EXT,SVGDD_TYPE,"SONS")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="SONS"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.MP3,.M4C,.AVI,.MPEG,.MPG,.WMA,,.M3A,"
    HAjoute(SVGDD_EXT)
FIN
wSons=SVGDD_EXT.SVGDD_LISTE_EXT
// 12
HLitRechercherPremier(SVGDD_EXT,SVGDD_TYPE,"EXE")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="EXE"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.EXE,.COM,.MSI,"
    HAjoute(SVGDD_EXT)
```

```
FIN
wExe=SVGDD_EXT.SVGDD_LISTE_EXT
// 13
HLitRecherchePremier(SVGDD_EXT,SVGDD_TYPE,"SGBD")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="SGBD"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.DAT,.DATA,.DB,.PDB,.FIC,.DBF,.NDX,.IDX,.ORA,"
    HAjoute(SVGDD_EXT)
FIN
wSgbd=SVGDD_EXT.SVGDD_LISTE_EXT
// 14
HLitRecherchePremier(SVGDD_EXT,SVGDD_TYPE,"HTML")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="HTML"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.HTML,.HTM,.XML,.CSS,.SCSS,.XCSS,.JSON,.XSL,.UML,.WSDL,.XSD,.JSP,.PHP,"
    HAjoute(SVGDD_EXT)
FIN
wHtml=SVGDD_EXT.SVGDD_LISTE_EXT
// 15
HLitRecherchePremier(SVGDD_EXT,SVGDD_TYPE,"SYSTEME")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="SYSTEME"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.BIN,.BAT,.DLL,.CMD,.SH,.SYS,.SYSTEM,"
    HAjoute(SVGDD_EXT)
FIN
wSysteme=SVGDD_EXT.SVGDD_LISTE_EXT
// 16
HLitRecherchePremier(SVGDD_EXT,SVGDD_TYPE,"CONFIG")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="CONFIG"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.INI,.PROPERTIES,.CONF,.CONFIG,.INFO,.INF,"
    HAjoute(SVGDD_EXT)
FIN
wConfig=SVGDD_EXT.SVGDD_LISTE_EXT
// 17
HLitRecherchePremier(SVGDD_EXT,SVGDD_TYPE,"INUTILES")
SI PAS HTrouve(SVGDD_EXT) ALORS
    HRAZ(SVGDD_EXT)
    SVGDD_EXT.SVGDD_TYPE="INUTILES"
    SVGDD_EXT.SVGDD_LISTE_EXT=",.BAK,.OLD,.XXX,.SUPPR,"
    HAjoute(SVGDD_EXT)
FIN
wInutiles=SVGDD_EXT.SVGDD_LISTE_EXT
//

// Tous sauf les 17 types
wLeReste=wWord+wExcel+wPowerP+wRtf+wPdf+wTxt+wZip+wLog+wPhotos+wVideos+wSons+wExe+wSgbd+wHtml+wSysteme+
wConfig+wInutiles
wLibre   =",.libre,"
```

Procédure globale GP_Trie_Liste

```
// Résumé : <indiquez ici ce que fait la procédure>
// Syntaxe :
//[ <Résultat> = ] GP_Trie_Liste (<parListe>)
//
```

```
// Paramètres :
//  parListe : <indiquez ici le rôle de parListe>
// Valeur de retour :
//  chaîne ANSI : <indiquez ici le rôle de la valeur de retour>
//
// Exemple :
// <Indiquez ici un exemple d'utilisation>
//
PROCÉDURE GP_Trie_Liste(parListe)

wResultat    est un chaîne

// on trie les extensions ...
wTable   est un tableau associatif d'entiers
POUR TOUTE CHAÎNE wMaChaine DE parListe SÉPARÉE PAR ","
    SI SansEspace(wMaChaine)="" OU SansEspace(wMaChaine)=EOT ALORS CONTINUER
    wTable[wMaChaine]++
FIN
TableauTrie(wTable,ttCroissant)
wResultat=","
POUR TOUT ÉLÉMENT nOccurrence, nExt DE wTable
    wResultat=wResultat+nExt+","
FIN

RENVOYER wResultat
```

Partie 4

Table des matières

Projet JC_SVG_Disque