

DataServer 2.10 et DataClient 2.10 pour Theos et Linux du 11-01-2024

Les serveurs et clients sont compatibles entre eux de la version 2.5 à la 2.10. Le client interroge le serveur pour savoir quelles sont les méthodes ajoutées qu'il supporte. Il n'est pas nécessaire que tous les clients connectés à un même serveur soient tous de la même version.

1 Bibliothèque dynamique DataClient

1.1 Fonctions de l'API

```
short DCLNewClient();
BOOL DCLDeleteClient(short client);
BOOL DCLAttach(short client, LPCTSTR logique, LPCTSTR physique, LPCTSTR options);
BOOL DCLBinary(short client, short canal, short champ, short binaire);
BOOL DCLCancel(short client);
BOOL DCLCanceled(short client);
BOOL DCLChDir(short client, LPCTSTR repertoire);
BOOL DCLClear(short client, LPCTSTR fichier);
BOOL DCLClose(short client, short canal);
BOOL DCLCloseAll(short client);
BOOL DCLCompressAll(short client, LPCTSTR extensions);
BOOL DCLCreate(short client, LPCTSTR fichier, LPCTSTR organisation, long enregistrements, long longueur_cle, short longueur_donnees);
BOOL DCLDelete(short client, short canal, LPCTSTR cle);
BOOL DCLDisconnect(short client);
BOOL DCLEof(short client, short canal);
BOOL DCLERase(short client, LPCTSTR fichier);
BOOL DCLExist(short client, LPCTSTR fichier);
BOOL DCLGet(short client, short canal);
BOOL DCLGetFile(short client, LPCTSTR destination, LPCTSTR source, LPCTSTR options);
BOOL DCLGetPipe(short client, LPCTSTR commande, LPCTSTR source, LPCTSTR options);
BOOL DCLInput(short client, short canal);
BOOL DCLInputKey(short client, short canal, LPCTSTR cle);
BOOL DCLIsConnected(short client);
BOOL DCLIsSequential(short client, short canal);
BOOL DCLLinput(short client, short canal);
BOOL DCLLinputKey(short client, short canal, LPCTSTR cle);
BOOL DCLLogOn(short client, LPCTSTR compte, LPCTSTR mot_de_passe);
BOOL DCLMkDir(short client, LPCTSTR repertoire);
BOOL DCLOptionCharSet(short client, LPCTSTR pszCSet);
BOOL DCLOptionLock(short client, short secondes);
BOOL DCLOptionMath(short client, LPCTSTR format);
BOOL DCLPrint(short client, short canal);
BOOL DCLPrintKey(short client, short canal, LPCTSTR cle);
BOOL DCLPrintLine(short client, short canal);
BOOL DCLPut(short client, short canal);
BOOL DCLPutFile(short client, LPCTSTR destination, LPCTSTR source, LPCTSTR options);
BOOL DCLPutPipe(short client, LPCTSTR commande, LPCTSTR source, LPCTSTR options);
BOOL DCLRead(short client, short canal);
BOOL DCLReadEnd(short client, short canal);
BOOL DCLReadKey(short client, short canal, LPCTSTR cle);
BOOL DCLReadNext(short client, short canal);
BOOL DCLReadPrev(short client, short canal);
BOOL DCLReadTop(short client, short canal);
BOOL DCLRename(short client, LPCTSTR ancien, LPCTSTR nouveau);
BOOL DCLRmDir(short client, LPCTSTR repertoire);
BOOL DCLSetField(short client, short canal, short champ, LPCTSTR valeur);
BOOL DCLSetFloat(short client, short canal, short champ, double valeur);
BOOL DCLSetFTime(short client, LPCTSTR fichier, unsigned long date_heure);
BOOL DCLSetInt(short client, short canal, short champ, short valeur);
BOOL DCLSetLong(short client, short canal, short champ, long valeur);
BOOL DCLStart(short client, LPCTSTR commande, short delai);
BOOL DCLSystem(short client, LPCTSTR commande);
BOOL DCLUnlock(short client, short canal);
BOOL DCLUserDef(short client, LPCTSTR commande);
BOOL DCLWrite(short client, short canal);
BOOL DCLWriteKey(short client, short canal, LPCTSTR cle);
LPCTSTR DCLClientAddress(short client);
LPCTSTR DCLClientVersion(short client);
LPCTSTR DCLDateTime(short client, LPCTSTR code);
LPCTSTR DCLErrorMessage(short client);
LPCTSTR DCLGetField(short client, short canal, short champ);
LPCTSTR DCLGetType(short client, short canal, short champ);
LPCTSTR DCLListTree(short client, LPCTSTR repertoire);
BOOL DCLPipeInput(short client, short canal);
LPCTSTR DCLSysEnv(short client, short code, LPCTSTR valeur);
double DCLGetFloat(short client, short canal, short champ);
long DCLFcntl(short client, short canal, short type, long valeur);
long DCLGetLong(short client, short canal, short champ);
long DCLReturnCode(short client);
```

```

short DCLConnect(short client, LPCTSTR serveur, LPCTSTR compte, LPCTSTR mot_de_passe, short port, LPCTSTR cle_privee);
short DCLErrorNumber(short client);
short DCLGetInt(short client, short canal, short champ);
short DCLGetSize(short client, short canal);
short DCLLockedBy(short client, short canal);
short DCLOpen(short client, LPCTSTR fichier, LPCTSTR options);
short DCLPipeOpen(short client, LPCTSTR commande, LPCTSTR options);
unsigned long DCLGetFTime(short client, LPCTSTR fichier);
void DCLClearRecord(short client, short canal);
void DCLOptionComma(short client, BOOL virgule);
void DCLOptionDelay(short client, short secondes);

typedef void(*callback_t)(short client, short event, size_t value);
void DCLRegisterCallBack(short client, callback_t CallBack);

```

- Les noms de fonctions sont ceux de l'ActiveX précédés de DCL.
- Les fonctions reçoivent un paramètre supplémentaire, le numéro de client renvoyé par DCLNewClient.
- Les événements COM/OLE de l'ActiveX sont remplacés par des call back.
- Les chaînes de caractères sont des chaînes C, terminées par un caractère nul. Les chaînes binaires incluant des caractères nuls ne sont pas supportées.

1.2 Paramètres de la fonction call back

Valeur 16 bits	Code événement 16 bits	Information 32 (DLL32) ou 64 bits (DLL64)
Numéro de client	'T' délai de réponse du serveur dépassé	0
Numéro de client	'O' stdout de DCLStart	Pointeur vers une chaîne de caractères
Numéro de client	'R' stderr de DCLStart	Pointeur vers une chaîne de caractères
Numéro de client	'Y' inoccupation de DCLStart	0
Numéro de client	'B' début de transfert	DCLGetFile ou DCLPutFile = longueur du fichier DCLGetPipe ou DCLPutPipe = 0
Numéro de client	'P' progression du transfert	Nombre d'octets transférés, 32 bits significatifs
Numéro de client	'E' fin de transfert	0 = erreur, 1 = succès

Utilisation :

Étape	Exemple de code
Inclure le fichier d'en-tête	<code>#include "DataCIntDll.h"</code>
Écrire une fonction call back si nécessaire	<pre> callback_t DCLCallBack(short client, short event, size_t value) { // traitement des call back de DataClient switch (event) { case 'T': ... break; case 'O': { LPCTSTR text = (LPCTSTR) value; ... break; } case 'R': { LPCTSTR text = (LPCTSTR) value; ... break; } case 'Y': ... break; case 'B': ... break; case 'P': ... break; case 'E': ... break; } } </pre>
Créer une instance de DataClient	<pre> short client = DCLNewClient(); if (client > 0) // succès else // erreur </pre>
Inscrire la fonction call back si nécessaire	<code>DCLRegisterCallBack(client, DCLCallBack);</code>
Se connecter à un serveur	<pre> short pid_distant = DCLConnect(client, "192.168.1.1", "gimini", "gimini", 0, ""); if (pid_distant > 0) // succès else // erreur </pre>
Utiliser la connexion	<code>BOOL status = DCLGetFile(client, "page.pcl", "page.pcl", "");</code>
Se déconnecter	<code>DCLDisconnect(client);</code>
Détruire l'instance	<code>DCLDeleteClient(client);</code>

2 Fonctionnalités depuis la version 2.9 de DataServer et 2.10 de DataClient

2.1 Mode de transfert rapide

Le mode de transfert rapide s'applique automatiquement aux transferts de données si le client et le serveur le supportent tous les deux.

2.2 Compression de transfert

Les méthodes de transfert de données peuvent utiliser la compression-décompression à la volée. Par défaut seuls les fichiers d'extension *txt*, *csv*, *doc* et *pcl* sont compressés. Les méthodes de transfert acceptent l'option COMPRESS qui force l'utilisation de la compression.

La compression-décompression à la volée s'applique automatiquement aux transferts de données si le client et le serveur le supportent tous les deux.

2.3 Indication de la liste des types de fichiers à compresser

Méthode CompressAll :

Syntaxe C++ : `BOOL DCLCompressAll(short client, LPCTSTR extensions);`

Syntaxe VB : `DCLCompressAll(client As Integer, extensions As String) As Boolean`

Exemple :

```
BOOL result = DCLCompressAll(client, "txt, csv, doc, pcl, bmp, tif");
```

La liste indiquée remplace la liste par défaut.

2.4 Transfert de fichier constitué à la volée

Méthode GetPipe :

Redirige la sortie standard d'un programme exécuté sur le serveur vers un fichier local.

Syntaxe C++ : `BOOL DCLGetPipe(short client, LPCTSTR localfile, LPCTSTR command, LPCTSTR options);`

Syntaxe VB : `DCLGetPipe(client As Integer, localfile As String, command As String, options As String) as Boolean`

client = numéro de client renvoyé par `DCLNewClient`

winfile = nom du fichier côté client

command = commande à exécuter côté serveur pour constituer le fichier à la volée

options = les mêmes que `GetFile`

Exemple :

```
BOOL result = DCLGetPipe("fichier.pdf", "pcl2pdf fichier.pcl -", "");
```

Redirige la sortie standard de *pcl2pdf* vers le fichier local *fichier.pdf*. *pcl2pdf* utilise le fichier *fichier.pcl* en entrée.

Méthode PutPipe :

Redirige un fichier local vers l'entrée standard d'un programme exécuté vers le serveur.

Syntaxe C++ : `BOOL DCLPutPipe(short client, LPCTSTR localfile, LPCTSTR command, LPCTSTR options);`

Syntaxe VB : `DCLPutPipe(client As Integer, localfile As String, command As String, options As String) As Boolean`

client = numéro de client renvoyé par `DCLNewClient`

winfile = nom du fichier côté client

command = commande à exécuter côté serveur pour constituer le fichier à la volée

options = les mêmes que `GetFile`

Exemple :

```
BOOL result = DCLPutPipe(client, "fichier.pcl", "pcl2pdf - fichier.pdf", "");
```

Redirige le fichier local *fichier.pcl* vers l'entrée standard de la commande *pcl2pdf* qui écrit le résultat de son traitement dans le fichier distant *fichier.pdf*.

Méthodes PipeOpen et PipeInput :

`DCLPipeOpen` ouvre un tube pour une lecture ligne par ligne avec `DCLPipeInput`. `DCLPipeInput` est l'équivalent de `Linput` après `Open`.

Syntaxe C++ : `short DCLPipeOpen(short client, LPCTSTR command, LPCTSTR options);`

Syntaxe VB : `DCLPipeOpen(client As Integer, command As String, options As String) As Integer`

Syntaxe C++ : `CString DCLPipeInput(short client, short file, short field);`

Syntaxe VB : `DCLPipeInput(client As Integer, file As Integer, field As Integer) As String`

Exemple :

```
short file = DCLPipeOpen(client, "filelist \"*. *.*\"", "");
if (file >= 0) {
    while (DCLPipeInput(client, file)) {
        CString str = DCLGetStr(client, file, 1);
        ...
    }
    DCLClose(client, file);
}
```

N.B. : Sous Theos, l'utilisation des tubes et de la compression passe par un stockage temporaire sur le disque désigné par la variable d'environnement `WORK`. L'envoi du fichier résultat ne s'effectue qu'après la fin de l'exécution de la commande passée en argument. Sous Linux, l'envoi s'effectue au fur et à mesure de sa constitution sans stockage sur disque.

2 Fonctionnalités depuis la version 2.10 de DataServer

2.1 Fonction keep alive

DCLSysEnv(client, 17, "DS_KEEPLIVE=délai") indique un délai d'inactivité à DataServer. Le délai est exprimé en secondes. DataServer ramène le délai dans l'intervalle de 0 à 3600. Zéro indique l'absence de délai.

Cette fonctionnalité est entièrement supportée par DataServer. Elle est donc compatible avec toutes les versions de DataClient.

3 Codes d'erreur

Numéro	Signification	Renvoyé par
1	Fin de fichier ou enregistrement non trouvé	Serveur
2	Fichier ou enregistrement verrouillé	Serveur
3	Code de commande inconnu	Serveur
4	Erreur de syntaxe	Serveur
5	Connexion refusée	Serveur
6	Nombre maxi d'utilisateurs connectés atteint	Serveur
7	Non connecté	Serveur
8	Fichier non trouvé	Serveur
9	Fichier ouvert en lecture seule	Serveur
10	Fichier ouvert en écriture seule	Serveur
15	Fichier non ouvert	Serveur
16	Erreur d'écriture	Serveur
17	Erreur de lecture	Serveur
18	Impossible de créer le fichier de destination du transfert	Serveur
19	Fichier à transférer non trouvé	Serveur
20	Pas assez de mémoire	Serveur
21	Code SysEnv non supporté	Serveur
22	Transfert interrompu	Serveur
23	Opération non autorisée	Serveur
24	En-tête de données utilisateur incorrect	Serveur
25	Fonction utilisateur inconnue	Serveur
26	Fonction restreinte par une clé développeur	Serveur
27	Chroot impossible	Serveur
28	Setuid impossible	Serveur
29	Chdir impossible	Serveur
30	Mkdir impossible	Serveur
31	Rmdir impossible	Serveur
32	Création de fichier impossible	Serveur
33	Suppression de fichier impossible	Serveur
34	Vidage de fichier impossible	Serveur
35	Rename impossible	Serveur
37	Fichier déjà existant	Serveur
40	Nom de fichier incorrect	Serveur
50	Non connecté	Client
51	Numéro de fichier incorrect	Client
52	Fichier non ouvert	Client
53	Mode d'accès incorrect	Client

54	Clé tronquée	Client
55	Enregistrement tronqué	Client
56	Déjà connecté	Client
57	Paramètre BCD ou IEEE requis	Client
58	Paramètre THEOS, ISO ou OEM requis	Client
59	Nombre maxi de fichiers ouverts atteint	Client
60	Nom de compte obligatoire	Client
61	Fonction interdite à partir d'un événement	Client
62	Aucune fonction à interrompre	Client
63	Serveur inconnu	Client
64	Paramètre manquant	Client
65	Paramètre inconnu	Client
66	Types de champs non concordants	Client
67	Pas assez de mémoire	Client
68	Numéro de série incorrect	Client
91	Réponse incorrecte du serveur	Client
97	Connexion RAS non établie	Client
98	Erreur réseau	Serveur
99	Erreur réseau	Client