

# Retail connecté – Librairie Windows

## Manuel d'intégration



Version document : 1.5

Date : 16/07/2018

Reference : Retail connecté - Librairie Windows - Manuel d'intégration

## Historique des modifications

Version	Author	Description	Date
1.0	Nepting	Première version	18/03/2016
1.1	Renaud Le Gouhinec	Ajout paramétrage lecteur	22/03/2016
1.2	Nepting	Ajout numéro de terminal	13/06/2016
1.3	Nepting	Ajout récupération des données en fin de transaction	11/06/2018
1.4	Nepting	Ajout des méthodes doPing() et doLogin() Mise à jour codes d'erreur	15/06/2018
1.5	Nepting	Correction description doLogin()	16/07/2018

**Table des matières**

<b>1. Introduction</b>	<b>4</b>
1.1 Architecture logicielle	4
1.2 Principe de fonctionnement	5
<b>2. Paramétrage du lecteur</b>	<b>5</b>
2.1 Paramètres IP	5
2.2 Paramètres fonctionnels	6
<b>3. Signatures des fonctions</b>	<b>7</b>
3.1 DoTransaction	7
3.2 DoPing	7
3.3 DoLogin	8
3.4 Exemples	8
3.5 Codes d'erreur	12

## 1. Introduction

### 1.1 Architecture logicielle

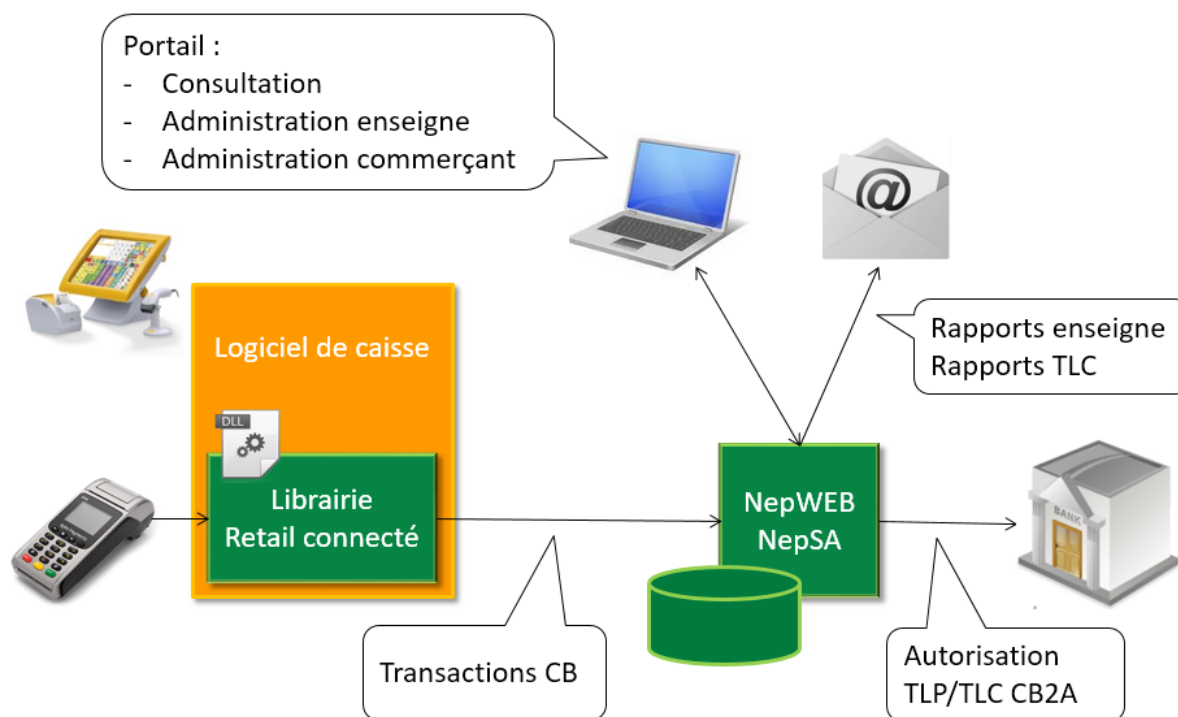
La solution Retail de Nepting est disponible sous deux architectures différentes en fonction des demandes :

- Solution Retail intégrée
- Solution Retail connectée

Avec la solution intégrée, les messages du caissier et le ticket de la carte sont gérés par la caisse. Dans la solution connectée, les messages du caissier s'affichent sur l'écran du TPE et le ticket est imprimé par le terminal.

Dans les deux cas, il est possible de connecter le TPE en IP, avec un port USB ou un port série.

Cette documentation et la librairie « Retail connecté » (DLL) sont distribuées avec la solution Retail connectée :



La solution Retail connectée de Nepting se compose des logiciels suivants :

- La librairie (DLL) disponible sous Windows qui permet d'effectuer des transactions (débit, crédit, annulation) et de tenir à jour le terminal de paiement
- Un web service front (NepWEB) recevant les demandes de transaction
- Le serveur d'acceptation (NepSA)

## 1.2 Principe de fonctionnement

La librairie « Retail connecté » expose une fonction très simple qui permet d'effectuer un paiement sur un terminal : « DoTransaction ». Les paramètres de cette fonction sont les suivants :

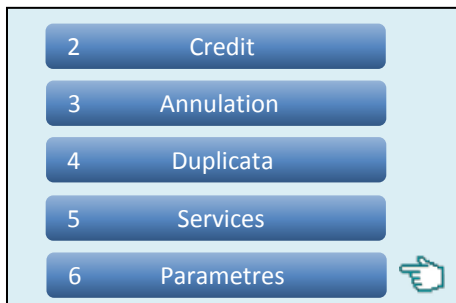
- Le point d'entrée du lecteur (une IP et un port ou un port com)
- Le type de la transaction (Debit, Refund, Reversal)
- Le montant en centimes
- Le code de la devise (978 pour l'euro)
- Le numéro de caisse (vide si non géré)
- Une référence de transaction (vide si non géré)

En retour, la fonction donne le statut de la transaction :

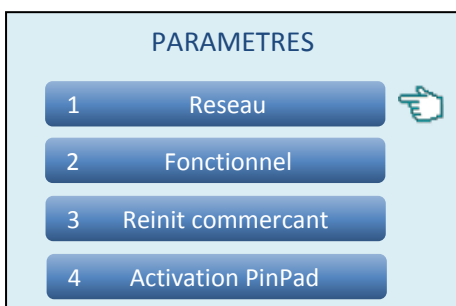
- 0 pour une transaction acceptée, un code d'erreur dans les autres cas

## 2. Paramétrage du lecteur

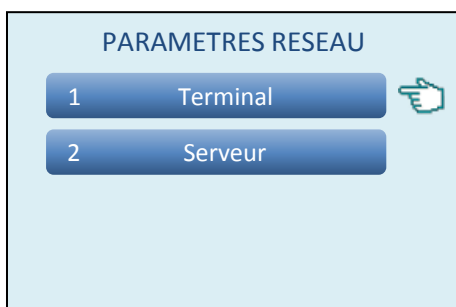
### 2.1 Paramètres IP



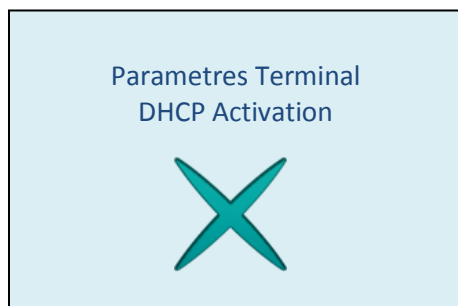
- Pour accéder au menu, appuyer sur la touche du milieu (Carré).
- Sélectionner "Parametres" avec la touche "6" ou utiliser les touches de navigation puis valider.



- Sélectionner "Reseau".

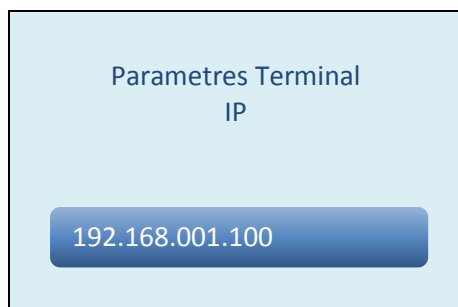


- Sélectionner "Terminal".



Le premier paramètre est l'activation "V" ou la non activation "X" du mode DHCP du terminal. Pour modifier ce paramètre il faut appuyer sur le bouton jaune "clear".

Dans le cas du mode DHCP actif le reste des paramètres ne peut être changés, mais une validation jusqu'au bout est nécessaire pour valider la modification.

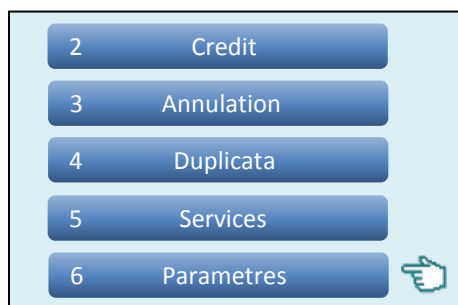


Le reste des paramètres sont les informations suivantes à saisir :

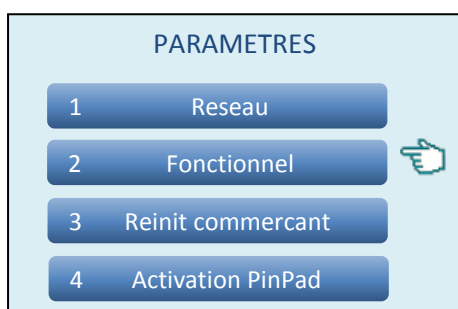
- IP
- Masque
- Passerelle
- DNS Primaire
- DNS Secondaire

En règles générales, le paramétrage du TPE est en "IP fixe", c'est-à-dire en mode DHCP inactif. Cela permet de maîtriser les adresses IP du réseau.

## 2.2 Paramètres fonctionnels



- Pour accéder au menu, appuyer sur la touche du milieu (Carré).
- Sélectionner "Parametres" avec la touche "6" ou utiliser les touches de navigation puis valider.



- Sélectionner "Fonctionnel".

Les paramètres suivants vont défiler les uns après les autres :

- Numéro de caisse
- Identifiant du Store
- Numéro de série (**Remarque** : ce paramètre est récupéré sur le terminal, il est donc verrouillé)
- Heure de mise à jour
- Mode d'impression pour les TNA (Choix multiple)

- Mode d'impression pour les TA (Choix multiple)

Seulement pour un terminal Wi-Fi ou GPRS :

- Délai de mise en veille
- Saisie manuelle (Actif / Inactif)

Seulement pour les terminaux de type SPc50 :

- Protocole de dialogue avec la caisse (Choix multiple) -> **Sélectionner "Nepting"**

Seulement si le protocole Nepting est sélectionné :

- Passerelle IP -> **Sélectionner "X" inactif**
- Port de la caisse -> **Sélectionner "IP"**
- Port d'IP d'écoute -> **Saisir le port d'écoute (ex : 8888)**

### 3. Signatures des fonctions

#### 3.1 DoTransaction

Nom	Type	Description
eftAddress	Char *	Port COM ou IP:port
Type	Char *	Debit, Refund, Reversal
Amount	Char *	Montant en centimes
currencyCode	Char *	Code numérique de la norme ISO 4217
posNumber	Char *	Numéro de caisse (ex :1)
merchantTransactionId	Char *	Référence de transaction (de préférence taille max de 12 caractères alpha-numériques ou espace)
eftNumber	Char *	Numéro de terminal (rien ou numéro de caisse ou numéro de terminal quand plusieurs caisses partagent le même terminal).

Le retour est le suivant :

Nom	Type	Description
status	Int	0 : transaction acceptée >0 : code d'erreur

#### 3.2 DoPing

Nom	Type	Description
eftAddress	Char *	Port COM ou IP:port

Le retour est le suivant :

Nom	Type	Description
status	Int	0 : Lecteur connecté >0 : code d'erreur

### 3.3 DoLogin

Nom	Type	Description
eftAddress	char*	Port COM ou IP:port
user	char*	Nom de l'utilisateur (NULL si non géré)
password	char*	Mot de passe de l'utilisateur (NULL si non géré)
url	char*	URL du web service NepWEB Ex : <a href="https://qualif.nepting.com/nepweb/ws?wsdl">https://qualif.nepting.com/nepweb/ws?wsdl</a> (NULL si non géré)
merchantCode	char*	Code commerçant (NULL si non géré)
posNumber	char*	Numéro de caisse (ex :1)

Le retour est le suivant :

Nom	Type	Description
status	Int	0 : login acceptée >0 : code d'erreur

### 3.4 Exemples

```
typedef int (__stdcall *doTransaction_ptr)(const char*, const char*, const char*, const char*, const char*,
const char*, const char*);
```

```
typedef char* (__stdcall *GetNepField_ptr)(const char*, const char*);
```

```
typedef char* (__stdcall *GetRepeatableNepField_ptr)(const char*, const char*, const int);
```

```
typedef char* (__stdcall *GetNepFieldNameAt_ptr)(const char*, const int);
```

```
typedef int (__stdcall *doPing_ptr)(const char*);
```

```
typedef int (__stdcall *doLogin_ptr)(const char*, const char*, const char*, const char*, const char*, const
char*);
```

```
int main() {
```

```
    HINSTANCE hGetProcIDDLL = LoadLibrary("NepPosDLL64.dll");
```

```
    std::cout << "main start" << std::endl;
```

```
    if (!hGetProcIDDLL) {
```



```
std::cout << "could not load the dynamic library" << std::endl;

return EXIT_FAILURE;

}

// resolve function address here

doTransaction_ptr doTransaction = (doTransaction_ptr)GetProcAddress(hGetProcIDDLL,
"doTransaction");

doLogin_ptr doLogin = (doLogin_ptr)GetProcAddress(hGetProcIDDLL, "doLogin");

doPing_ptr doPing = (doPing_ptr)GetProcAddress(hGetProcIDDLL, "doPing");

GetNepField_ptr GetNepField = (GetNepField_ptr)GetProcAddress(hGetProcIDDLL, "GetNepField");

GetRepeatableNepField_ptr GetRepeatableNepField =
(GetRepeatableNepField_ptr)GetProcAddress(hGetProcIDDLL, "GetRepeatableNepField");

GetNepFieldNameAt_ptr GetNepFieldNameAt =
(GetNepFieldNameAt_ptr)GetProcAddress(hGetProcIDDLL, "GetNepFieldNameAt");

if (!doTransaction || !doLogin || !doPing || !GetNepField || !GetRepeatableNepField ||
!GetNepFieldNameAt) {

std::cout << "could not locate the functions" << std::endl;

FreeLibrary(hGetProcIDDLL);

return EXIT_FAILURE;

}

{

const char* eftAddress = "192.168.1.108:8888";

int ret = doPing(eftAddress);

if(ret == 0) {

std::cout << "PING OK" << std::endl;

} else {

std::cout << "PING KO" << std::endl;

}

}

{

const char* eftAddress = "192.168.1.108:8888";
```

```
const char * user = NULL;

const char * password = NULL;


const char * url = NULL;

//const char * url = "qualif.nepting.com:443/nepweb/ws?wsdl";
//const char * url = "nepsa1.nepting.com:443/nepweb/trx?wsdl";


const char * merchantCode = NULL;

//const char * merchantCode = "10002";


const char* posNumber = "1";


int ret = doLogin(eftAddress, user, password, url, merchantCode, posNumber);
if(ret == 0) {
    std::cout << "Login OK" << std::endl;
} else {
    std::cout << "Login KO" << std::endl;
}
}

const char* eftAddress = "192.168.1.108:8888";
//const char* eftAddress = "COM4";
const char* type = "Debit";
const char* amount = "100";
const char* currencyCode = "978";
const char* posNumber = "1";
const char* merchantTransactionId = "0123456";
const char* eftNumber = "1";


int ret = doTransaction(eftAddress, type, amount, currencyCode, posNumber, merchantTransactionId,
eftNumber);
```

```
if(ret == 0) {  
    std::cout << "Payment accepted" << std::endl;  
} else {  
    std::cout << "Payment refused" << std::endl;  
}  
  
char* ticket = GetNepField(NULL, NepTag::HOLDER_TICKET);  
  
if(ticket != NULL) {  
    std::cout << "HOLDER_TICKET=" << ticket << std::endl;  
}  
  
char* finalAmount = GetNepField(NULL, NepTag::POS_FINAL_AMOUNT);  
  
if(finalAmount != NULL) {  
    std::cout << "POS_FINAL_AMOUNT=" << finalAmount << std::endl;  
}  
  
// To list all the fields ( For repeatable fields the first value is repeated) (  
int index = 0;  
  
char* fieldName;  
  
do {  
    fieldName = GetNepFieldNameAt(NULL, index++);  
  
    if(fieldName != NULL) {  
        char* fieldValue = GetNepField(NULL, fieldName);  
  
        if(fieldValue != NULL) {  
            std::cout << fieldName << "=" << fieldValue << std::endl;  
        }  
    }  
}  
  
while (fieldName != NULL);  
  
// To list all the results (A repeatable field)  
  
index = 0;
```

```

char* extendedResult = NULL;

do {

    char* extendedResult = GetRepeatableNepField(NULL, NepTag::EXTENDED_RESULT, index++);

    if (extendedResult != NULL) {

        int status = atoi(extendedResult);

        std::cout << "EXTENDED_RESULT=" << status << std::endl;

    }

}

while (extendedResult != NULL);

FreeLibrary(hGetProcIDDLL);

return EXIT_SUCCESS;

}

```

### 3.5 Codes d'erreur

Nom	Code	Description
Sys_Nothing	1	Pas de résultat
Sys_Refused	3	L'opération a été refusée par le serveur
Sys_Error	4	Erreur technique
Sys_Fatal	5	Erreur grave
Sys_MandatoryParameterMissing	6	Un paramètre non optionnel est manquant
Sys_BadRequest	7	Le format de la requête est incorrect
Sys_ChangesDone	8	L'opération n'a pas aboutie mais des informations ont été modifiées en base
Sys_UpdateNeeded	9	Un nouveau login est nécessaire avant de répéter l'opération
CurrencyNotManaged	12	La devise n'est pas supportée par le contrat
ContextError	13	Information contextuelle incorrecte
TransactionNotAllowed	14	Transaction non permise
PanLengthIncorrect	15	Longueur du PAN incorrecte
LuhnKeyIncorrect	16	Longueur de la clé de Luhn incorrecte
EndOfValidityError	17	Date de fin de validité expirée
BinForbidden	18	BIN interdit
BinRefused	19	BIN refusé
CardForbidden	20	Carte interdite
CardRefused	21	Carte refusé

AuthorizationIncident	22	Incident d'autorisation
AuthorizationRefused	23	Refus d'autorisation
AuthorizationForbidden	24	Carte interdite suite à autorisation
AmountTooHigh	25	Montant trop élevé
AmountTooLow	26	Montant trop faible
TrsToReverseNotFound	27	Transaction à annuler non trouvée
TrsAlreadyVoided	28	Transaction à annuler déjà annulée
EffectiveDateNotReached	29	Date de validité non atteinte
UserAbort	30	Abandon
MerchantAbort	31	Abandon caisse
CvvInvalid	32	Cryptogramme invalide
CardRemoved	33	Carte arrachée
OfflineTrs	34	Transaction dégradée
ForeignCard	35	Carte étrangère
AidUnknown	36	AID inconnu
DueDateAfterEndOfValidity	37	Date requise après la date de fin validité
BinUnknown	38	BIN inconnu
AuthenticationFailed	60	Mot de passe ou nom d'utilisateur incorrect
AccountBlocked	61	Compte bloqué
SessionExpired	62	Jeton temporaire expiré
AuthenticationError	63	Erreur technique d'authentification
AccountUnknown	64	Compte commerçant inconnu
NoAccess	65	Niveau de privilège requis insuffisant
PasswordExpired	66	La date de validité du mot de passe a expiré
TerminalUnknown	70	Terminal inconnu
TerminalModelUnknown	71	Modèle de terminal inconnu
TerminalInactive	72	Terminal inactif
StoreUnknown	80	Ilot inconnu
StoreEmpty	81	Ilot non renseigné
UnknownKey	90	Clé cryptographique inconnue
MACError	91	Problème d'intégrité de message entre le terminal et le serveur
Sys_Already	101	Objet existant déjà dans le système
Sys_Forbidden	102	Opération demandée interdite
Sys_Inactive	103	Activation de contrat en échec
Sys_Timeout	104	Pas de réponse dans le délai imparti
Sys_ConnectionError	105	Echec de connexion au terminal
Sys_CommunicationError	106	Echec de dialogue avec le terminal
Sys_Busy	107	Transaction déjà en cours
Sys_NepsaConnectionError	108	Echec de connexion au serveur Nepting
Reference_NotFound	109	Dossier non trouvé

<b>Reference_NotClosed</b>	110	Dossier non clôturé
<b>Sys_NepsaTimeoutCompletion</b>	111	Pas de réponse dans le délai imparti lors de l'enregistrement de la transaction
<b>OfflineRequirementsNotMet</b>	112	Prérequis mode dégradé non atteint
<b>InvalidMerchantTransactionID</b>	113	Identifiant de transaction invalide