Guide of POS_PC library (.NET)

POS_PC Library Info:

File Name : POS_PC.dll
Version : 1.4.25.0
Date : 1401/06/14

Dependency:

To use POS_PC library in your application, you must have Microsoft .NET Framework 2.0 or higher, installed on your Windows system.

Compatibility:

Current version support following functions and connection type:

Functions:

- Result **Debits_Goods_And_Service** (string RequestID, string PcID, string Amount, string PayerID, string MerchantMsg, string MerchantadditionalData,bool Settle = true)
- Result **Bill_Payment_Service** (string RequestID,string PcID, string BillID, string PayID, string MerchantMsg, string MerchantadditionalData)
- Result **Payment** (string RequestID,string PcID, string Amount, string PayerID, string AccountID, string MerchantadditionalData,bool Settle = true)
- Result **MultiPayment** (string RequestID,string PcID, string TotalAmount, MultiPaymentReqDataSet[] RequestList, byte PrintDetail, string MerchantadditionalData,bool Settle = true)
- Result **Inquery_Service**(string RequestID, string PcID,string MerchantadditionalData)
- Result **Last_Inquery_Service**(string PcID, string MerchantadditionalData)
- Result **Get_Card_Service**(string ProcessCode, string PcID, string MerchantadditionalData)
- Result **Get_Card_Debits_Goods_And_Service**(string RequestID, string PcID, string Amount, string PayerID, string MerchantMsg, string MerchantadditionalData)
- Result **Etebary_Debits_Goods_And_Service** (string RequestID, string PcID, string Amount, string PayerID, string MerchantMsg, string MerchantadditionalData)
- Result **EsfahanQR** (string PcID, string EsfahanQR, string MerchantMsg)
- Result **Navosh_Charge_Service** (string RequestID, string PcID, string Amount, string MerchantMsg, string MerchantadditionalData)

- Result **Navosh_Inventory_Service** `(string RequestID, string PcID, string MerchantMsg, string MerchantadditionalData)`
- Result **Verify_Service**`(string PcID,String ReferenceID)`
- Result **Reverse_Service**`(string PcID, String ReferenceID)`
- Result **Ready_Cancel**`("CANCEL")`
- Result **Ready_Cancel**`("Ready?")`

Connection type :

Serial port connection (RS232 / USB plug)
tcp/ip connection

Dll Class Names:

- **Globals**
  Includes dll information: `DllReleaseDate, dllVersion, dllType`
- **Result**
  Includes response of transaction from pos: `ReqID, SerialTransaction, TraceNumber, TerminalNo, TransactionDate, TransactionTime, PAN, AccountNo, ReasonCode, ReturnCode, PcID, TotalAmount, RequestId, Amount`
- **Transaction**
  Includes everything that needs for a transaction:
  - Structure :
    - `TransInfo`
    - `MultiPaymentReqDataSet`
  - Class:
    - `Connection`
    - `CRCTool`
  - Function:
    - `Debits_Goods_And_Service`
    - `Bill_Payment_Service`
    - `Payment`
    - `MultiPayment`
    - `Inquery`
    - `Last_Inquery`
    - `Get_Card`
    - `Get_Card_Debits_Goods_And_Service`
    -

Function Details:

❖Result Debits_Goods_And_Service (string RequestID,string PcID, string Amount, string PayerID, string MerchantMsg, string MerchantadditionalData,bool Settle = true)

Function is used to perform a debit (goods and service) transaction.

Parameters [in]:

Name	Format	Length (byte)	Existence	explain
RequestID	string ASCII numeric	Variable MAX:20	optional	Request id of transaction if necessary
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
Amount	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	amount of transaction
PayerID	string ASCII numeric	Variable MIN:3 MAX:17	optional	payer id of transaction if necessary
Merchant Msg	Hex string numeric /alphabetic	Variable MAX:160	optional	merchant message if necessary for print on client receipt.
MerchantadditionalData	string ASCII numeric	Variable MAX:100	optional	merchant data if necessary for sending to switch
Settle	bool	Defult : True AutoSettle: False	optional	Defult value is True but for send AutoSettle change to False

Detailed description:

For performing a debit (Goods and service) transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include the amount, Id Code and merchant custom message of transaction) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**RET_OK"** and the return parameters are accessible via the public properties.

❖Result Etebary_Debits_Goods_And_Service (string RequestID,string PcID, string Amount, string PayerID, string MerchantMsg, string MerchantadditionalData)

Function is used to perform a Etebary debit (goods and service) transaction.

Parameters [in]:

Name	Format	Length (byte)	Existence	explain
RequestID	string ASCII numeric	Variable MAX:20	optional	Request id of transaction if necessary
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
Amount	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	amount of transaction
PayerID	string ASCII numeric	Variable MIN:3 MAX:17	optional	payer id of transaction if necessary
Merchant Msg	Hex string numeric /alphabetic	Variable MAX:160	optional	merchant message if necessary for print on client receipt.
MerchantadditionalData	string ASCII numeric	Variable MAX:100	optional	merchant data if necessary for sending to switch

Detailed description:

For performing a Etebary debit (Goods and service) transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include the amount, Id Code and merchant custom message of transaction) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**RET_OK"** and the return parameters are accessible via the public properties.

❖Result Bill_Payment_Service(string RequestID,string PcID, string BillID, string PayID, string MerchantMsg, string MerchantadditionalData)

Function is used to perform a Bill Payment transaction.

Parameters [in]:

Name	Format	Length (byte)	Existence	explain
RequestID	string ASCII numeric	Variable MAX:20	optional	Request id of transaction if necessary
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
BillID	string ASCII numeric	Variable MIN:6 MAX:18	mandatory	bill identity of transaction
PayID	string ASCII numeric	Variable MIN:6 MAX:18	optional	payment identity of transaction
Amount	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	amount of transaction
Merchant Msg	Hex string numeric /alphabetic	Variable MAX:160	optional	Merchant message if necessary for print on client receipt.
MerchantadditionalData	string ASCII numeric	Variable MAX:100	optional	merchant data if necessary for sending to switch

Detailed description:

For performing a bill payment transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include the Bill ID, Payment ID, Amount and merchant custom message of transaction) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**RET_OK"** and the return parameters are accessible via the public properties.

❖ Result Payment(string RequestID,string PcID, string Amount, string PayerID, string AccountID, string MerchantadditionalData, bool Settle = true)

Function is used to perform a payment transaction.

Parameters [in]:

Name	Format	Length (byte)	Existence	explain
RequestID	string ASCII numeric	Variable MAX:20	optional	Request id of transaction if necessary
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
Amount	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	amount of transaction
PayerID	string ASCII numeric	Variable MIN:3 MAX:17	optional	payer id of transaction if necessary
AccountID	English string numeric /alphabetic	Variable MAX:24	mandatory	account number id of payment transaction.
MerchantadditionalData	string ASCII numeric	Variable MAX:100	optional	merchant data if necessary for sending to switch
Settle	bool	Defult : True AutoSettle: False	optional	Defult value is True but for send AutoSettle change to False

Detailed description:

For performing a payment transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include the amount, payer id and account id of transaction) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Note:

After successful transaction,

- The amount of the payment transaction will be transferred to the merchant account. Merchant account number determines from account id.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**RET_OK"** and the return parameters are accessible via the following public properties.

❖ Result MultiPayment(string RequestID, string PcID, string TotalAmount, MultiPaymentReqDataSet[] RequestList, byte PrintDetail, string MerchantadditionalData, bool Settle = true)

Function is used to perform a multiPayment transaction.

Parameters [in]:

Name	Format	Length	Existence	explain
RequestID	string ASCII numeric	Variable MAX:20	optional	Request id of transaction if necessary
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
Amount	string ASCII numeric	Variable(byte) MIN:1 MAX:12	mandatory	Total amount of multipayment transaction
RequestList	Array of MultiPaymentReqDataSet structure	Variable(index) MIN:1 MAX:10	mandatory	Request list of multipayment data set
PrintDetail	numeric	Fixed (1byte)	mandatory	Print Details of multipayment on receipt Value = 0 : Print on both receipt Value = 1: Print on customer receipt Value = 2: Print on merchant receipt Value = 3: Not printed on the receipt
MerchantadditionalData	string ASCII numeric	Variable MAX:100	optional	merchant data if necessary for sending to switch
Settle	bool	Defult : True AutoSettle: False	optional	Defult value is True but for send AutoSettle change to False

MultiPaymentReqDataSet structure fields:

Name	Format	Length (byte)	Existence	explain
AccountID	string ASCII numeric	Variable MIN:1 MAX:10	mandatory	account number of request data set
Amount	string ASCII numeric	Variable MIN:1 MAX:10	mandatory	amount of request data set
PayerID	string ASCII numeric	Variable MIN:3 MAX:17	optional	payer id of request data set

Detailed description:

For performing a multiPayment transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include the total amount and list of request data set (max 10 records) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**RET_OK"** and the return parameters are accessible via the public properties.

❖Result Result Inquery_Service(string RequestID, string PcID, string MerchantadditionalData)

Function is used to perform a Inquery transaction.

Parameters [in]:

Name	Format	Length (byte)	Existence	explain
RequestID	string ASCII numeric	Variable MAX:20	optional	Request id of transaction if necessary
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
MerchantadditionalData	string ASCII numeric	Variable MAX:100	optional	merchant data if necessary for sending to switch

Detailed description:

For performing a Inquery transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include the RequestID, Id Code and merchant custom message of transaction) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**RET_OK"** and the return parameters are accessible via the public properties.

❖Result Last_Inquery_Service(string PcID, string MerchantadditionalData)

Function is used to perform a Last_Inquery transaction.

Parameters [in]:

Name	Format	Length (byte)	Existence	explain
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
MerchantadditionalData	string ASCII numeric	Variable MAX:100	optional	merchant data if necessary for sending to switch

Detailed description:

For performing a Last_Inquery transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include Id Code and merchant custom message of transaction) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**RET_OK"** and the return parameters are accessible via the public properties.

❖Result Get_Card_Service (string ProcessCode, string PcID, string MerchantadditionalData)

Function is used to perform a Get_Card transaction.

Parameters [in]:

Name	Format	Length (byte)	Existence	explain
ProcessCode	string ASCII numeric	Variable MAX:2	mandatory	ProcessCode
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
MerchantadditionalData	string ASCII numeric	Variable MAX:100	optional	merchant data if necessary for sending to switch

Detailed description:

For performing a Get_Card transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include ProcessCode,Id Code and merchant custom message of transaction) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**RET_OK"** and the return parameters are accessible via the public properties.

❖Result Get_Card_Debits_Goods_And_Service (string RequestID,string PcID, string Amount, string PayerID, string MerchantMsg, string MerchantadditionalData)

Function is used to perform a debit (goods and service) transaction.

Parameters [in]:

Name	Format	Length (byte)	Existence	explain
RequestID	string ASCII numeric	Variable MAX:20	optional	Request id of transaction if necessary
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
Amount	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	amount of transaction
PayerID	string ASCII numeric	Variable MIN:3 MAX:17	optional	payer id of transaction if necessary
Merchant Msg	Hex string numeric /alphabetic	Variable MAX:160	optional	merchant message if necessary for print on client receipt.
MerchantadditionalData	string ASCII numeric	Variable MAX:100	optional	merchant data if necessary for sending to switch

Detailed description:

For performing a Get_Card _debit (Goods and service) transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include the amount, Id Code and merchant custom message of transaction) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**RET_OK"** and the return parameters are accessible via the public properties.

❖Result Verify_Service (string PcID,String ReferenceID)

Function is used to perform a (Verify_Service) transaction.

Parameters [in]:

Name	Format	Length (byte)	Existence	explain
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
Amount	string ASCII numeric	Variable MAX:12	mandatory	POS_PC_Reference ID

Detailed description:

For performing a (Verify_Service) transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include the amount, Id Code and merchant custom message of transaction) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**OK"** and the return parameters are accessible via the public properties.

❖Result Reverse _Service (string PcID,String ReferenceID)

Function is used to perform a (Reverse_Service) transaction.

Parameters [in]:

Name	Format	Length (byte)	Existence	explain
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
Amount	string ASCII numeric	Variable MAX:12	mandatory	POS_PC_Reference ID

Detailed description:

For performing a (Reverse _Service) transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include the amount, Id Code and merchant custom message of transaction) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**NO"** and the return parameters are accessible via the public properties.

❖Result EsfahanQR(string PcID, string EsfahanQR, string MerchanMsg)

Function is used to perform a **EsfahanQR** transaction.

Parameters [in]:

Name	Format	Length (byte)	Existence	explain
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
EsfahanQR	string ASCII numeric	Variable MAX:500	mandatory	EsfahanQR from PC
MerchantMsg	Hex string numeric /alphabetic	Variable MAX:160	optional	merchant message if necessary for print on client receipt.

Detailed description:

For performing a EsfahanQR transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include the RequestID, Id Code and merchant custom message of transaction) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**RET_OK"** and the return parameters are accessible via the public properties.

❖Result Navosh_Charge_Service (string RequestID,string PcID, string Amount, string MerchantMsg, string MerchantadditionalData)

Function is used to perform a Navosh_Charge_Service transaction.

Parameters [in]:

Name	Format	Length (byte)	Existence	explain
RequestID	string ASCII numeric	Variable MAX:20	optional	Request id of transaction if necessary
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
Amount	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	amount of transaction
Merchant Msg	Hex string numeric /alphabetic	Variable MAX:160	optional	merchant message if necessary for print on client receipt.
MerchantadditionalData	string ASCII numeric	Variable MAX:100	optional	merchant data if necessary for sending to switch

Detailed description:

For performing a a Navosh_Charge_Service transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include the amount, Id Code and merchant custom message of transaction) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**RET_OK"** and the return parameters are accessible via the public properties.

❖Result Navosh_Inventory_Service (string RequestID,string PcID, string MerchantMsg, string MerchantadditionalData)

Function is used to perform a Navosh_ Inventory _Service transaction.

Parameters [in]:

Name	Format	Length (byte)	Existence	explain
RequestID	string ASCII numeric	Variable MAX:20	optional	Request id of transaction if necessary
PcID	string ASCII numeric	Variable MIN:1 MAX:12	mandatory	Request ID from PC
Merchant Msg	Hex string numeric /alphabetic	Variable MAX:160	optional	merchant message if necessary for print on client receipt.
MerchantadditionalData	string ASCII numeric	Variable MAX:100	optional	merchant data if necessary for sending to switch

Detailed description:

For performing a a Navosh_ Inventory _Service transaction this function opens a serial or tcp/ip port (depending on connection type) at first, write a specific data (a message for pos which include the amount, Id Code and merchant custom message of transaction) on the port and then wait for receiving data (a message receive from pos) on it, then close the port and finally return a result for showing that if the transaction has done successfully or not.

Return Values:

Refer to Appendix "A"

- If the transaction is approved successfully, the return value is "**RET_OK"** and the return parameters are accessible via the public properties.

Parameter Details:

❖ Merchant Additional Data

Detailed description:

Additional data of the acceptor is sent to the payment switch and returned to the acceptor in reporting systems and other acceptor data separated by a comma (2C Hex).

APPID:"APPID",MMID:"Multi Merchant ID", MerchantData

Example:

APPID 1000000 and MMID 1 and other additional data acceptor have the following two values:

0080694283-21998033511961477810

Merchant Additional Data sample:

APPID:1000000,MMID:1,219980335119614778l0,0080694283

Important explanation:

- It is necessary to enter the value of APPID constant string at the beginning and comma after the ID of the acceptor software. This ID is provided to the calling software by the company for payment.
- If you need to use the Multi Merchant feature, the value of the fixed MMID string and the multi-merchant ID of the desired acceptor will be added to the structure of this variable. The default value is always 1.
- If the additional data of the receiver has more than one value, the values are separated by commas.
- If the caller request has no additional data, only the software ID value will be sent for the reporting process.

❖ Result properties

SerialTransaction:
The local transaction number generated by POS for each transaction request.

TraceNumber:
The unique transaction number which is get from switch for the approved transactions.

PAN:
The permanent account number (the last 4 digit of card number).

BIN:
The bank identity number (the first 6 digit of card number).

TransactionDate:
The date of transaction in shamsi format (yyyy/mm/dd)

TransactionTime:
The time of transaction

AccountNo:
The merchant account number.

TerminalNo:
Terminal identity number.

ReasonCode:

If the transaction failed because of error number 109 (**ERR_POS_FAILED_TRANSACTION OR RET_NOK**), the reason of error will be accessible via this public properties.

See **ReasonCode**.**pdf** for more information.

Appendix A

Return Values:

Response Code	Error Define	Description	Transaction Status
100	RET_*OK*	The process for send and receive data performs successfully	OK
101	ERR_PC_INVALID_REC_SIZE	invalid message size received from pos	Failed
102	ERR_POS_INVALID_DATA	invalid message received from pc (message size, process code)	Failed
103	ERR_PC_INVALID_REC_PROCESS_CODE	invalid process code received from pos in response	Failed
104	ERR_PC_INVALID_AMOUNT	invalid input amount or invalid received amount from pos in response	Failed
105	ERR_PC_INVALID_INPUT_PAYERID	invalid input payer id	Failed
106	ERR_PC_INVALID_INPUT_TIMEOUT	invalid input timeout for waiting to receive data from pos on serial port (the value of it must be : Min = 2000 ms Max = 600000 ms)	Failed
107	ERR_PC_PORT_TIMEOUT_FOR_REC	timeout occurred while waiting for receive data from pos on serial port	Failed
108	ERR_POS_RESPONSE_RECEIVED_TOO_LATE	transaction failed: not receive response from server because of timeout or connection failed	Failed
109	ERR_POS_FAILED_TRANSACTION	transaction failed: credit is not sufficient, server is down or …	Failed
110	ERR_POS_PRINTER	transaction failed because of printer error	Failed
Response Code	**Error Define**	**Description**	**Transaction Status**

111	ERR_POS_COMMUNICATION	transaction failed because of connection error	Failed
112	ERR_POS_TO_SEND_TRANSACTION	transaction failed because of error in send settlement, send reversal or generate transaction	Failed
113	ERR_PC_INVALID_INPUT_PORTNAME	invalid input serial port name	Failed
114	ERR_POS_USER_ABORT	transaction failed because of aborting by user	Failed
115	ERR_PC_INVALID_INPUT_BILLID	invalid input bill id for bill payment transaction	Failed
116	ERR_PC_INVALID_INPUT_PAYID	invalid input payment id for bill payment transaction	Failed
117	ERR_PC_PORT_OPEN_FAILED	error occurred while opening selected serial port	Failed
118	ERR_PC_PORT_ACCESS_FAILED	I/O error or a specific type of security error : the serial port can't write data	Failed
119	ERR_PC_INVALID_PORT_STATE	I/O error occurs: the selected port isn't serial port	Failed
120	ERR_PC_INVALID_PORT_PARAMETERS	argument out of range exception: one or more of the properties for this instance are invalid(on serial port)	Failed
121	ERR_PC_INVALID_PORT_NAME	arguments provided to a method is not valid: the serial port name is invalid	Failed
122	ERR_PC_NULL_STR_TO_WRITE_IN_PORT	Argument Null Exception on serial port	Failed
123	ERR_PC_PORT_TIMEOUT_FOR_SEND	timeout occurred while send data on serial port	Failed
Response Code	**Error Define**	**Description**	**Transaction Status**

124	ERR_POS_CARD_SWIPE_FAILED	the card has not been swiped on pos	Failed
125	ERR_PC_INVALID_INPUT_ACCOUNT_ID	invalid input account id for payment transaction	Failed
126	ERR_POS_INVALID_INPUT_ACCOUNT_ID	invalid received account id from pc	Failed
127	ERR_POS_INVALID_INPUT_PAYERID	invalid received payer id from pc	Failed
128	ERR_POS_INVALID_INPUT_AMOUNT	invalid received amount from pc	Failed
129	ERR_POS_INVALID_INPUT_REFERENCE_NUMBER	invalid received reference number from pc	Failed
130	ERR_POS_INVALID_INPUT_BILL_ID	invalid received bill id from pc	Failed
131	ERR_POS_INVALID_INPUT_PAYMENT_ID	invalid received payment id from pc	Failed
132	ERR_POS_INVALID_INPUT_ADDITIONALDATA	invalid received additional data from pc	Failed
133	ERR_POS_INVALID_INPUT_MULTI_PAYMENT_AMOUNT	invalid received multi payment total amount from pc	Failed
134	ERR_POS_UNCONFIRM_REC_DATA	unconfirmed received data from pc by user	Failed
161	ERR_PC_INVALID_INPUT_MULTI_PAYMENT_REQUEST_LIST	Invalid input multi payment request data set list (max 10 records)	Failed
162	ERR_PC_INVALID_INPUT_MULTI_PAYMENT_AMOUNT	invalid input multi payment amount	Failed
163	ERR_PC_INVALID_INPUT_REFERENCE_NUMBER	invalid input reference number	Failed
164	ERR_POS_PC_CRCERROR_INVALID_DATA	Invalid received data on pos or pc (crc error)	Failed

Response Code	Error Define	Description	Transaction Status

165	ERR_PC_INVALID_POSPC_COMMUNICATION_TYPE	invalid input connection type of pos-pc (the value of it must be Serial or tcp/ip)	Failed
166	ERR_PC_INVALID_INPUT_TCP_SOCKET_PORT	invalid input tcp/ip connection port number (the value of it must be : MIN : 1024 MAX: 65535)	Failed
167	ERR_PC_INVALID_INPUT_TCP_SOCKET_RECEIVED_TIMEOUT	invalid input timeout for waiting to receive data from pos on tcp/ip connection (the value of it must be : Min = 2000 ms Max = 600000 ms)	Failed
168	ERR_PC_TCP_SOCKET_FAILED	error occurred while creating socket on pc in tcp/ip communication	Failed
169	ERR_PC_TCP_SOCKET_SEND_MSG_FAILED	error occurred while send data to pos in tcp/ip communication	Failed
170	ERR_PC_TCP_SOCKET_RECEIVED_MSG_FAILED	error occurred while receiving data from pos on tcp/ip communication because of timeout or …	Failed
171	ERR_PC_INVALID_INPUT_MERCHANT_MESSAGE	invalid format of input merchant message	Failed
172	ERR_PC_PREPARE_TLV_MSG_FAILED	error occurred while prepare TLV message for send	Failed
173	ERR_PC_PORT_EXCEPTION_FOR_REC	Serial port exception for receiving data from pos	Failed
174	ERR_PC_NULL_STR_IN_READ_PORT	Serial port exception for receiving data from pos	Failed
175	ERR_PC_CALCULATE_CRC_ERROR	error occurred while calculate crc of message	Failed
176	ERR_PC_INVALID_INPUT_MERCHANT_FIELD	invalid input additional data of merchant	Failed
200	ERR_POS_PC_OTHER	Other exception	Failed

Sample Code :

```
/* pos-pc via Serial port */

POS_PC.Transaction.Connection Connect = new POS_PC.Transaction.Connection();
POS_PC.Result retCode = new POS_PC.Result();
Connect.CommunicationType = “serial”;

Connect.PC_PORT_Name = "COM1";
Connect.PC_PORT_BaudRate = 115200;
Connect.PC_PORT_DataBits = PC_PORT.DataBits;
Connect.PC_PORT_Parity = PC_PORT.Parity;
Connect.PC_PORT_StopBits = PC_PORT.StopBits;

POS_PC.Transaction TXN = new POS_PC.Transaction(Connect);
        /* Debit */
retCode = TXN.Debits_Goods_And_Service(RequestID,"1", Amount, PayerID, MerchantMsg,
MerchantadditionalData);

OR      /* Debit with AutoSettle */
retCode = TXN.Debits_Goods_And_Service(RequestID,"1", Amount, PayerID, MerchantMsg,
MerchantadditionalData,false);

OR      /* BillPayment */
retCode = TXN.Bill_Payment_Service(RequestID,"1", strBillId, strPayCode, strAmount,   strMerchantMsg,
strMerchantAddit);

OR      /* Payment */
retCode = TXN.Payment(RequestID,"1", strAmount, strPayerId, strAcountId, strMerchantAddit);

OR       /*MultiPayment */
retCode = TXN.MultiPayment(RequestID,"1", strTotalAmount, RequestList, PrintDetail,
strMerchantadditionalData);

OR       /* Inquery */
retCode = TXN.Inquery_Service(RequestID,"1", MerchantadditionalData);

OR       /* Last_Inquery */
retCode = TXN. Last_Inquery_Service("1", MerchantadditionalData);

OR       /* Get_Card */
retCode = TXN.Get_Card_Service ("20", "1", MerchantadditionalData);

OR       /* Get_Card_Debits */
retCode = TXN. Get_Card_Debits_Goods_And_Service(RequestID,"1", Amount, PayerID, MerchantMsg,
MerchantadditionalData);


POS_PC.Result.return_codes returncode = (POS_PC.Result.return_codes)retCode.ReturnCode;

if (returncode == Result.return_codes.RET_OK)
{
        /* the transaction has been done */
}
else
{
        MessageBox.Show(retCode.ToString());
         /* If the transaction failed because of error number 109
        (ERR_POS_FAILED_TRANSACTION),   the reason of error will be accessible via Result.ReasonCode */
}
```

Sample Code:

```
/* pos-pc via tcp/ip */

POS_PC.Transaction.Connection Connect = new POS_PC.Transaction.Connection();
POS_PC.Result retCode = new POS_PC.Result();
Connect.CommunicationType = “serial”;

Connect.POSPC_TCPCOMMU_SocketRecTimeout = 60000;
Connect.POS_PORTtcp = Convert.ToInt16(“1024”);
Connect.POS_IP =”172.20.120.193”;

Transaction TXN = new Transaction();
POS_PC.Transaction.return_codes retCode = Transaction.return_codes.ERR_POS_PC_OTHER;
POS_PC.Transaction TXN = new POS_PC.Transaction(Connect);
        /* Debit */
retCode = TXN.Debits_Goods_And_Service(RequestID,"1", Amount, PayerID, MerchantMsg,
MerchantadditionalData);

OR     /* Debit with AutoSettle */
retCode = TXN.Debits_Goods_And_Service(RequestID,"1", Amount, PayerID, MerchantMsg,
MerchantadditionalData,false);

OR      /* BillPayment */
retCode = TXN.Bill_Payment_Service(RequestID,"1", strBillId, strPayCode, strAmount,   strMerchantMsg,
strMerchantAddit);

OR      /* Payment */
retCode = TXN.Payment(RequestID,"1", strAmount, strPayerId, strAcountId, strMerchantAddit);

OR      /*MultiPayment */
retCode = TXN.MultiPayment(RequestID,"1", strTotalAmount, RequestList, PrintDetail,
strMerchantadditionalData)

OR      /* Inquery */
retCode = TXN.Inquery_Service(RequestID,"1", MerchantadditionalData);

OR      /* Last_Inquery */
retCode = TXN. Last_Inquery_Service("1", MerchantadditionalData);

OR      /* Get_Card */
retCode = TXN.Get_Card_Service ("20", "1", MerchantadditionalData);

OR      /* Get_Card_Debits */
retCode = TXN. Get_Card_Debits_Goods_And_Service(RequestID,"1", Amount, PayerID, MerchantMsg,
MerchantadditionalData);


POS_PC.Result.return_codes returncode = (POS_PC.Result.return_codes)retCode.ReturnCode;

if (returncode == Result.return_codes.RET_OK)
{
        /* the transaction has been done */
}
else
{
        MessageBox.Show(retCode.ToString());
         /* If the transaction failed because of error number 109
        (ERR_POS_FAILED_TRANSACTION),   the reason of error will be accessible via Result.ReasonCode */
}
```