

Computer Network Protocol (CNP) Project

Table of Contents

Main Page	1
Module Index	2
Namespace Index	2
Hierarchical Index	2
Class Index	3
File Index	4
Module Documentation	5
CNP Protocol Messages	5
Server Messages	5
Client Messages	6
Message TypeDefs	6
cnp	8
cnp::prim	12
Class Documentation	13
cnp::prim::_BALANCE_QUERY_REQUEST	13
cnp::prim::_BALANCE_QUERY_RESPONSE	13
cnp::prim::_CONNECT_REQUEST	14
cnp::prim::_CONNECT_RESPONSE	15
cnp::prim::_CREATE_ACCOUNT_REQUEST	16
cnp::prim::_CREATE_ACCOUNT_RESPONSE	18
cnp::prim::_DEPOSIT_REQUEST	19
cnp::prim::_DEPOSIT_RESPONSE	20
cnp::prim::_LOGOFF_REQUEST	21
cnp::prim::_LOGOFF_RESPONSE	21
cnp::prim::_LOGON_REQUEST	22
cnp::prim::_LOGON_RESPONSE	23
cnp::prim::_STAMP_PURCHASE_REQUEST	23
cnp::prim::_STAMP_PURCHASE_RESPONSE	24
cnp::prim::_TRANSACTION_QUERY_REQUEST	25
cnp::prim::_TRANSACTION_QUERY_RESPONSE	25
cnp::prim::_WITHDRAWAL_REQUEST	27
cnp::prim::_WITHDRAWAL_RESPONSE	27
ACCOUNT_INFO	28
cnp::BALANCE_QUERY_REQUEST	30
cnp::BALANCE_QUERY_RESPONSE	32
CNP_Socket	34
cnp::CONNECT_REQUEST	39
cnp::CONNECT_RESPONSE	41
cnp::CREATE_ACCOUNT_REQUEST	43
cnp::CREATE_ACCOUNT_RESPONSE	46
cnp::DEPOSIT_REQUEST	48
cnp::DEPOSIT_RESPONSE	51
cnp::LOGOFF_REQUEST	52
cnp::LOGOFF_RESPONSE	55
cnp::LOGON_REQUEST	56
cnp::LOGON_RESPONSE	59
SESSION_INFO	61
cnp::STAMP_PURCHASE_REQUEST	62
cnp::STAMP_PURCHASE_RESPONSE	64
cnp::STD_HDR	66
TAutoLock<_Ty>	68
THREAD_INFO	69

TLock<_Ty>.....	70
cnp::TRANSACTION	72
TRANSACTION_INFO	74
cnp::TRANSACTION_QUERY_REQUEST.....	75
cnp::TRANSACTION_QUERY_RESPONSE.....	78
TTSQueue<_Ty>.....	80
cnp::WITHDRAWAL_REQUEST	83
cnp::WITHDRAWAL_RESPONSE	85
File Documentation	87
Client/CNP_Client.cpp	87
Client/CNP_Client.h.....	95
Include/CNP_Protocol.h.....	95
Server/CNP_Common.h	98
Server/CNP_Messaging.cpp.....	99
Server/CNP_Messaging.h.....	106
Server/CNP_Server.cpp.....	112
Server/CNP_Server.h.....	117
Server/CNP_ServerDB.cpp	117
Server/CNP_ServerDB.h.....	120
Server/CNP_Session.cpp	122
Server/CNP_Session.h.....	123
Server/CNP_Socket.cpp	124
Client/CNP_Socket.cpp	124
Server/CNP_Socket.h	125
Client/CNP_Socket.h.....	126
Server/FNV1A_Hash.cpp	126
Server/FNV1A_Hash.h.....	127
Server/ThreadMisc.h	128
Server/TSQueue.h.....	129
Index	130

Main Page

Author:

Mark L. Short

Date:

March 25, 2015

Course :

5580 Computer Networks

Objective:

Objective of this group activity is to design an application layer protocol standard that will be used by all groups for the project. The protocol must provide the basic ATM Banking functionality of:

- Establish a connection
- Creating an Account
 - First Name
 - Last Name
 - Email Address
 - SSN
 - Driver's License #
 - User selected PIN
- Logging On
 - PIN & First Name
- Deposit
 - Cash or Check
- Withdrawal
- Stamp Purchase
- Transaction History Query

Implementation Notes

1. In addition to those required functions, the following were implemented:
 - Logging Off (explicit)
 - Balance Query
2. Those types with the prefixed '_' are intentionally 'uglified' to discourage their direct use. Additionally, they have been wrapped in the 'prim' namespace to further obscure them from direct use.

This is a common naming procedure used to denote an 'internal' type within a published API specification or protocol to not directly use.

3. Regarding inheritance.

Even though, from an implementation stand-point, C++ inheritance would have afforded the ability to avoid a lot of "helper" method duplication; from the position of affording any byte-wise guarantee of consistency across various compilers and platforms, there was no way to insure that the use of inheritance would not, covertly, introduce additional hidden bytes or other compiler generated information.

So, it was explicitly avoided in the message protocol implementation.

Module Index

Modules

Here is a list of all modules:

CNP Protocol Messages	5
Server Messages	5
Client Messages.....	6
Message TypeDefs	6

Namespace Index

Namespace List

Here is a list of all namespaces with brief descriptions:

cnp	8
cnp::prim	12

Hierarchical Index

Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

cnp::prim::_BALANCE_QUERY_REQUEST	13
cnp::prim::_BALANCE_QUERY_RESPONSE	13
cnp::prim::_CONNECT_REQUEST.....	14
cnp::prim::_CONNECT_RESPONSE.....	15
cnp::prim::_CREATE_ACCOUNT_REQUEST	16
ACCOUNT_INFO	28
cnp::prim::_CREATE_ACCOUNT_RESPONSE.....	18
cnp::prim::_DEPOSIT_REQUEST	19
cnp::prim::_DEPOSIT_RESPONSE	20
cnp::prim::_LOGOFF_REQUEST	21
cnp::prim::_LOGOFF_RESPONSE.....	21
cnp::prim::_LOGON_REQUEST	22
cnp::prim::_LOGON_RESPONSE	23
cnp::prim::_STAMP_PURCHASE_REQUEST	23
cnp::prim::_STAMP_PURCHASE_RESPONSE	24
cnp::prim::_TRANSACTION_QUERY_REQUEST.....	25
cnp::prim::_TRANSACTION_QUERY_RESPONSE.....	25
cnp::prim::_WITHDRAWAL_REQUEST.....	27

cnp::prim::_WITHDRAWAL_RESPONSE.....	27
cnp::BALANCE_QUERY_REQUEST.....	30
cnp::BALANCE_QUERY_RESPONSE.....	32
CNP_Socket	34
cnp::CONNECT_REQUEST	39
cnp::CONNECT_RESPONSE	41
cnp::CREATE_ACCOUNT_REQUEST	43
cnp::CREATE_ACCOUNT_RESPONSE	46
cnp::DEPOSIT_REQUEST.....	48
cnp::DEPOSIT_RESPONSE.....	51
cnp::LOGOFF_REQUEST.....	52
cnp::LOGOFF_RESPONSE.....	55
cnp::LOGON_REQUEST	56
cnp::LOGON_RESPONSE	59
SESSION_INFO	61
cnp::STAMP_PURCHASE_REQUEST	62
cnp::STAMP_PURCHASE_RESPONSE	64
cnp::STD_HDR.....	66
THREAD_INFO	69
TLock< _Ty >	70
TAutoLock< _Ty >	68
 cnp::TRANSACTION.....	 72
TRANSACTION_INFO	74
 cnp::TRANSACTION_QUERY_REQUEST	 75
cnp::TRANSACTION_QUERY_RESPONSE	78
TTSQueue< _Ty >.....	80
cnp::WITHDRAWAL_REQUEST	83
cnp::WITHDRAWAL_RESPONSE	85

Class Index

Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<u>cnp::prim:: BALANCE_QUERY_REQUEST</u> (Balance Query Request Primitive)	13
<u>cnp::prim:: BALANCE_QUERY_RESPONSE</u> (Balance Query Response Primitive)	13
<u>cnp::prim:: CONNECT_REQUEST</u> (Connect Request Primitive)	14
<u>cnp::prim:: CONNECT_RESPONSE</u> (Connection Response Primitive)	15
<u>cnp::prim:: CREATE_ACCOUNT_REQUEST</u> (Create Account Request Primitive)	16
<u>cnp::prim:: CREATE_ACCOUNT_RESPONSE</u> (Create Account Response Primitive)	18
<u>cnp::prim:: DEPOSIT_REQUEST</u> (Deposit Request Primitive)	19
<u>cnp::prim:: DEPOSIT_RESPONSE</u> (Deposit Response Primitive)	20
<u>cnp::prim:: LOGOFF_REQUEST</u> (Logoff Request Primitive)	21

<u>cnp::prim:: LOGOFF_RESPONSE</u> (Logoff Response Primitive)	21
<u>cnp::prim:: LOGON_REQUEST</u> (Logon Request Primitive)	22
<u>cnp::prim:: LOGON_RESPONSE</u> (Logon Response Primitive)	23
<u>cnp::prim:: STAMP_PURCHASE_REQUEST</u> (Purchase Stamp Request Primitive)	23
<u>cnp::prim:: STAMP_PURCHASE_RESPONSE</u> (Stamp Purchase Response Primitive)	24
<u>cnp::prim:: TRANSACTION_QUERY_REQUEST</u> (Transaction Query Request Primitive)	25
<u>cnp::prim:: TRANSACTION_QUERY_RESPONSE</u> (Transaction Query Result Primitive)	25
<u>cnp::prim:: WITHDRAWAL_REQUEST</u> (Withdrawal Request Primitive)	27
<u>cnp::prim:: WITHDRAWAL_RESPONSE</u> (Withdrawal Response Primitive)	27
<u>ACCOUNT_INFO</u>	28
<u>cnp::BALANCE_QUERY_REQUEST</u> ([Client] Balance Query Request message)	30
<u>cnp::BALANCE_QUERY_RESPONSE</u> ([Server] Balance Query Response message)	32
<u>CNP Socket</u>	34
<u>cnp::CONNECT_REQUEST</u> ([Client] Connect Request message)	39
<u>cnp::CONNECT_RESPONSE</u> ([Server] Connect Response message)	41
<u>cnp::CREATE_ACCOUNT_REQUEST</u> ([Client] Create Account Request message)	43
<u>cnp::CREATE_ACCOUNT_RESPONSE</u> ([Server] Create Account Response message)	46
<u>cnp::DEPOSIT_REQUEST</u> ([Client] Deposit Request message)	48
<u>cnp::DEPOSIT_RESPONSE</u> ([Server] Deposit Response message)	51
<u>cnp::LOGOFF_REQUEST</u> ([Client] Logoff Request message)	52
<u>cnp::LOGOFF_RESPONSE</u> ([Server] Logoff Response message)	55
<u>cnp::LOGON_REQUEST</u> ([Client] Logon Request message)	56
<u>cnp::LOGON_RESPONSE</u> ([Server] Logon Response message)	59
<u>SESSION_INFO</u>	61
<u>cnp::STAMP_PURCHASE_REQUEST</u> ([Client] Stamp Purchase Request Message)	62
<u>cnp::STAMP_PURCHASE_RESPONSE</u> ([Server] Stamp Purchase Response message)	64
<u>cnp::STD_HDR</u> (CNP Standard Message Header)	66
<u>TAutoLock< Ty ></u> ("Stack-based" Template)	68
<u>THREAD_INFO</u>	69
<u>TLock< Ty ></u> ("Stack-based" Template)	70
<u>cnp::TRANSACTION</u> (A Customer Transaction Record)	72
<u>TRANSACTION_INFO</u>	74
<u>cnp::TRANSACTION_QUERY_REQUEST</u> ([Client] Transaction Query Request message)	75
<u>cnp::TRANSACTION_QUERY_RESPONSE</u> ([Server] Transaction Query Response message)	78
<u>TTSQueue< Ty ></u>	80
<u>cnp::WITHDRAWAL_REQUEST</u> ([Client] Withdrawal Request message)	83
<u>cnp::WITHDRAWAL_RESPONSE</u> ([Server] Withdrawal Response message)	85

File Index

File List

Here is a list of all files with brief descriptions:

<u>Client/CNP_Client.cpp</u>	87
--	----

Client/ CNP_Client.h	95
Client/ CNP_Socket.cpp (Client CNP_Socket class implementation)	124
Client/ CNP_Socket.h (Client CNP_Socket class interface)	126
Include/ CNP_Protocol.h (Contains type definitions required to support 5580 Computer Networks Project (CNP) Protocol)	95
Server/ CNP_Common.h (Common type definitions)	98
Server/ CNP_Messaging.cpp (Server Message processing implementation)	99
Server/ CNP_Messaging.h (Message processing function prototypes)	106
Server/ CNP_Server.cpp (Server Main)	112
Server/ CNP_Server.h	117
Server/ CNP_ServerDB.cpp (Server DB persistence implementation)	117
Server/ CNP_ServerDB.h (ACCOUNT_INFO & TRANSACTION_INFO struct definitions) ..	120
Server/ CNP_Session.cpp (SessionMap_t Global Instance)	122
Server/ CNP_Session.h (SESSION_INFO struct definition)	123
Server/ CNP_Socket.cpp (Server CNP_Socket class implementation)	124
Server/ CNP_Socket.h (Server CNP_Socket class interface)	125
Server/ FNV1A_Hash.cpp (FNV1A Hash function implementation)	126
Server/ FNV1A_Hash.h (FNV1A Hash function prototype)	127
Server/ ThreadMisc.h (Interface for the Thread Related Classes/Templates)	128
Server/ TSQueue.h (Interface for the TTSQueue class)	129

Module Documentation

CNP Protocol Messages

Modules

- [Server Messages](#)
- [Client Messages](#)
- [Message TypeDefs](#)

Detailed Description

Server Messages

Classes

- struct [cnp::CONNECT_RESPONSE](#)
- [Server] Connect Response message struct [cnp::CREATE_ACCOUNT_RESPONSE](#)
- [Server] Create Account Response message struct [cnp::LOGON_RESPONSE](#)
- [Server] Logon Response message struct [cnp::LOGOFF_RESPONSE](#)
- [Server] Logoff Response message struct [cnp::DEPOSIT_RESPONSE](#)

- [Server] Deposit Response message struct [cnp::WITHDRAWAL_RESPONSE](#)
 - [Server] Withdrawal Response message struct [cnp::BALANCE_QUERY_RESPONSE](#)
 - [Server] Balance Query Response message struct [cnp::TRANSACTION_QUERY_RESPONSE](#)
 - [Server] Transaction Query Response message struct [cnp::STAMP_PURCHASE_RESPONSE](#)
- [Server] Stamp Purchase Response message
-

Detailed Description

Client Messages

Classes

- struct [cnp::CONNECT_REQUEST](#)
 - [Client] Connect Request message struct [cnp::CREATE_ACCOUNT_REQUEST](#)
 - [Client] Create Account Request message struct [cnp::LOGON_REQUEST](#)
 - [Client] Logon Request message struct [cnp::LOGOFF_REQUEST](#)
 - [Client] Logoff Request message struct [cnp::DEPOSIT_REQUEST](#)
 - [Client] Deposit Request message struct [cnp::WITHDRAWAL_REQUEST](#)
 - [Client] Withdrawal Request message struct [cnp::BALANCE_QUERY_REQUEST](#)
 - [Client] Balance Query Request message struct [cnp::TRANSACTION_QUERY_REQUEST](#)
 - [Client] Transaction Query Request message struct [cnp::STAMP_PURCHASE_REQUEST](#)
- [Client] Stamp Purchase Request Message
-

Detailed Description

Message TypeDefs

Classes

- struct [cnp::TRANSACTION](#)
- A Customer Transaction Record. struct [cnp::STD_HDR](#)

CNP Standard Message Header. Enumerations

- enum [cnp::MSG_TYPE](#) { [cnp::MT_INVALID](#) = MAKE_MSG_TYPE(CMT_INVALID, CMS_INVALID), [cnp::MT_CONNECT_REQUEST](#) = MAKE_MSG_TYPE(CMT_CONNECT, CMS_REQUEST), [cnp::MT_CONNECT_RESPONSE](#) = MAKE_MSG_TYPE(CMT_CONNECT, CMS_RESPONSE), [cnp::MT_CREATE_ACCOUNT_REQUEST](#) = MAKE_MSG_TYPE(CMT_CREATE_ACCOUNT, CMS_REQUEST), [cnp::MT_CREATE_ACCOUNT_RESPONSE](#) = MAKE_MSG_TYPE(CMT_CREATE_ACCOUNT, CMS_RESPONSE), [cnp::MT_LOGON_REQUEST](#) = MAKE_MSG_TYPE(CMT_LOGON, CMS_REQUEST), [cnp::MT_LOGON_RESPONSE](#) = MAKE_MSG_TYPE(CMT_LOGON, CMS_RESPONSE), [cnp::MT_LOGOFF_REQUEST](#) = MAKE_MSG_TYPE(CMT_LOGOFF, CMS_REQUEST), [cnp::MT_LOGOFF_RESPONSE](#) = MAKE_MSG_TYPE(CMT_LOGOFF, CMS_RESPONSE), [cnp::MT_DEPOSIT_REQUEST](#) = MAKE_MSG_TYPE(CMT_DEPOSIT, CMS_REQUEST), [cnp::MT_DEPOSIT_RESPONSE](#) = MAKE_MSG_TYPE(CMT_DEPOSIT, CMS_RESPONSE), [cnp::MT_WITHDRAWAL_REQUEST](#) = MAKE_MSG_TYPE(CMT_WITHDRAWAL, CMS_REQUEST), [cnp::MT_WITHDRAWAL_RESPONSE](#) = MAKE_MSG_TYPE(CMT_WITHDRAWAL, CMS_RESPONSE),

[cnp::MT_BALANCE_QUERY_REQUEST](#) = MAKE_MSG_TYPE(CMT_BALANCE_QUERY, CMS_REQUEST), [cnp::MT_BALANCE_QUERY_RESPONSE](#) = MAKE_MSG_TYPE(CMT_BALANCE_QUERY, CMS_RESPONSE),
[cnp::MT_TRANSACTION_QUERY_REQUEST](#) = MAKE_MSG_TYPE(CMT_TRANSACTION_QUERY, CMS_REQUEST), [cnp::MT_TRANSACTION_QUERY_RESPONSE](#) = MAKE_MSG_TYPE(CMT_TRANSACTION_QUERY, CMS_RESPONSE),
[cnp::MT_PURCHASE_STAMPS_REQUEST](#) = MAKE_MSG_TYPE(CMT_PURCHASE_STAMPS, CMS_REQUEST), [cnp::MT_PURCHASE_STAMPS_RESPONSE](#) = MAKE_MSG_TYPE(CMT_PURCHASE_STAMPS, CMS_RESPONSE) }

- *Constructed Message Type IDs.* enum [cnp::CFC_TYPE](#) { [cnp::CFC_CONNECT](#), [cnp::CFC_CREDENTIALS](#), [cnp::CFC_FUNCTIONAL](#), [cnp::CFC_ACCOUNT](#), [cnp::CFC_UNDEFINED](#) }
- *Message Facility Code Types (CFC)* enum [cnp::DEPOSIT_TYPE](#) { [cnp::DT_INVALID](#) = 0, [cnp::DT_CASH](#) = 0x01, [cnp::DT_CHECK](#) = 0x02 }
- *CNP Deposit types (DT)* enum [cnp::TRANSACTION_TYPE](#) { [cnp::TT_INVALID](#) = 0, [cnp::TT_DEPOSIT](#) = 0x01, [cnp::TT_WITHDRAWAL](#) = 0x02, [cnp::TT_STAMP_PURCHASE](#) = 0x03 }

CNP Transaction types (TT)

Detailed Description

Enumeration Type Documentation

enum [cnp::MSG_TYPE](#)

Enumerator

MT_INVALID
MT_CONNECT_REQUEST
MT_CONNECT_RESPONSE
MT_CREATE_ACCOUNT_REQUEST
MT_CREATE_ACCOUNT_RESPONSE
MT_LOGON_REQUEST
MT_LOGON_RESPONSE
MT_LOGOFF_REQUEST
MT_LOGOFF_RESPONSE
MT_DEPOSIT_REQUEST
MT_DEPOSIT_RESPONSE
MT_WITHDRAWAL_REQUEST
MT_WITHDRAWAL_RESPONSE
MT_BALANCE_QUERY_REQUEST
MT_BALANCE_QUERY_RESPONSE
MT_TRANSACTION_QUERY_REQUEST
MT_TRANSACTION_QUERY_RESPONSE
MT_PURCHASE_STAMPS_REQUEST
MT_PURCHASE_STAMPS_RESPONSE

enum [cnp::CFC_TYPE](#)

Used in the creation of result codes returned to the client. The purpose is to help provide the client useful diagnostic information regarding associating specific errors with a particular facility or functional subsystem.

Enumerator

CFC_CONNECT Connection validation related issues.
CFC_CREDENTIALS Logon related issues.
CFC_FUNCTIONAL Invalid arguments or Client state.
CFC_ACCOUNT Account related errors related to balances, etc.
CFC_UNDEFINED Other error categories not explicitly defined.

enum [cnp::DEPOSIT_TYPE](#)

Enumerator

DT_INVALID for initialization and error checking
DT_CASH Cash Deposit.
DT_CHECK Check Deposit.

enum [cnp::TRANSACTION_TYPE](#)

Enumerator

TT_INVALID for initialization and error checking
TT_DEPOSIT Deposit Transaction.
TT_WITHDRAWAL Withdrawal Transaction.
TT_STAMP_PURCHASE Stamp Purchase Transaction.

Namespace Documentation

cnp Namespace Reference

Namespaces

- [prim](#)

Classes

- struct [BALANCE_QUERY_REQUEST](#)
- [Client] Balance Query Request message struct [BALANCE_QUERY_RESPONSE](#)
- [Server] Balance Query Response message struct [CONNECT_REQUEST](#)
- [Client] Connect Request message struct [CONNECT_RESPONSE](#)
- [Server] Connect Response message struct [CREATE_ACCOUNT_REQUEST](#)
- [Client] Create Account Request message struct [CREATE_ACCOUNT_RESPONSE](#)
- [Server] Create Account Response message struct [DEPOSIT_REQUEST](#)
- [Client] Deposit Request message struct [DEPOSIT_RESPONSE](#)
- [Server] Deposit Response message struct [LOGOFF_REQUEST](#)
- [Client] Logoff Request message struct [LOGOFF_RESPONSE](#)
- [Server] Logoff Response message struct [LOGON_REQUEST](#)
- [Client] Logon Request message struct [LOGON_RESPONSE](#)
- [Server] Logon Response message struct [STAMP_PURCHASE_REQUEST](#)
- [Client] Stamp Purchase Request Message struct [STAMP_PURCHASE_RESPONSE](#)

- [Server] Stamp Purchase Response message struct [STD_HDR](#)
- CNP Standard Message Header. struct [TRANSACTION](#)
- A Customer Transaction Record. struct [TRANSACTION_QUERY_REQUEST](#)
- [Client] Transaction Query Request message struct [TRANSACTION_QUERY_RESPONSE](#)
- [Server] Transaction Query Response message struct [WITHDRAWAL_REQUEST](#)
- [Client] Withdrawal Request message struct [WITHDRAWAL_RESPONSE](#)

[Server] Withdrawal Response message Typedefs

- typedef unsigned short [WORD](#)
16bit type definition
- typedef unsigned long [DWORD](#)
32bit type definition
- typedef unsigned long long [QWORD](#)
64bit type definition

Enumerations

- enum [CNP_MSG_TYPE](#) { [CMT_INVALID](#) = 0x00, [CMT_CONNECT](#) = 0x50, [CMT_CREATE_ACCOUNT](#) = 0x51, [CMT_LOGON](#) = 0x52, [CMT_LOGOFF](#) = 0x53, [CMT_DEPOSIT](#) = 0x54, [CMT_WITHDRAWAL](#) = 0x55, [CMT_BALANCE_QUERY](#) = 0x56, [CMT_TRANSACTION_QUERY](#) = 0x57, [CMT_PURCHASE_STAMPS](#) = 0x58 }
- Supported CNP Message Types (CMT_) enum [CNP_MSG_SUBTYPE](#) { [CMS_INVALID](#) = 0x00, [CMS_REQUEST](#) = 0x01, [CMS_RESPONSE](#) = 0x02 }
- Supported CNP Message Subtypes (CMS_) enum [MSG_TYPE](#) { [MT_INVALID](#) = MAKE_MSG_TYPE(CMT_INVALID, CMS_INVALID), [MT_CONNECT_REQUEST](#) = MAKE_MSG_TYPE(CMT_CONNECT, CMS_REQUEST), [MT_CONNECT_RESPONSE](#) = MAKE_MSG_TYPE(CMT_CONNECT, CMS_RESPONSE), [MT_CREATE_ACCOUNT_REQUEST](#) = MAKE_MSG_TYPE(CMT_CREATE_ACCOUNT, CMS_REQUEST), [MT_CREATE_ACCOUNT_RESPONSE](#) = MAKE_MSG_TYPE(CMT_CREATE_ACCOUNT, CMS_RESPONSE), [MT_LOGON_REQUEST](#) = MAKE_MSG_TYPE(CMT_LOGON, CMS_REQUEST), [MT_LOGON_RESPONSE](#) = MAKE_MSG_TYPE(CMT_LOGON, CMS_RESPONSE), [MT_LOGOFF_REQUEST](#) = MAKE_MSG_TYPE(CMT_LOGOFF, CMS_REQUEST), [MT_LOGOFF_RESPONSE](#) = MAKE_MSG_TYPE(CMT_LOGOFF, CMS_RESPONSE), [MT_DEPOSIT_REQUEST](#) = MAKE_MSG_TYPE(CMT_DEPOSIT, CMS_REQUEST), [MT_DEPOSIT_RESPONSE](#) = MAKE_MSG_TYPE(CMT_DEPOSIT, CMS_RESPONSE), [MT_WITHDRAWAL_REQUEST](#) = MAKE_MSG_TYPE(CMT_WITHDRAWAL, CMS_REQUEST), [MT_WITHDRAWAL_RESPONSE](#) = MAKE_MSG_TYPE(CMT_WITHDRAWAL, CMS_RESPONSE), [MT_BALANCE_QUERY_REQUEST](#) = MAKE_MSG_TYPE(CMT_BALANCE_QUERY, CMS_REQUEST), [MT_BALANCE_QUERY_RESPONSE](#) = MAKE_MSG_TYPE(CMT_BALANCE_QUERY, CMS_RESPONSE), [MT_TRANSACTION_QUERY_REQUEST](#) = MAKE_MSG_TYPE(CMT_TRANSACTION_QUERY, CMS_REQUEST), [MT_TRANSACTION_QUERY_RESPONSE](#) = MAKE_MSG_TYPE(CMT_TRANSACTION_QUERY, CMS_RESPONSE), [MT_PURCHASE_STAMPS_REQUEST](#) = MAKE_MSG_TYPE(CMT_PURCHASE_STAMPS, CMS_REQUEST), [MT_PURCHASE_STAMPS_RESPONSE](#) = MAKE_MSG_TYPE(CMT_PURCHASE_STAMPS, CMS_RESPONSE) }
- Constructed Message Type IDs. enum [CFC_TYPE](#) { [CFC_CONNECT](#), [CFC_CREDENTIALS](#), [CFC_FUNCTIONAL](#), [CFC_ACCOUNT](#), [CFC_UNDEFINED](#) }
- Message Facility Code Types (CFC) enum [CER_TYPE](#) { [CER_SUCCESS](#) = 0, [CER_AUTHENTICATION_FAILED](#) = MAKE_ERROR_RESULT(CFC_CONNECT, 0x01), [CER_UNSUPPORTED_PROTOCOL](#) = MAKE_ERROR_RESULT(CFC_CONNECT, 0x02), [CER_INVALID_CLIENT_ID](#) = MAKE_ERROR_RESULT(CFC_CREDENTIALS, 0x01), [CER_INVALID_NAME_PIN](#) = MAKE_ERROR_RESULT(CFC_CREDENTIALS, 0x02), [CER_INVALID_ARGUMENTS](#) = MAKE_ERROR_RESULT(CFC_FUNCTIONAL, 0x01),

[CER_CLIENT_NOT_LOGGEDON](#) = MAKE_ERROR_RESULT(CFC_FUNCTIONAL, 0x02),
[CER_DRAWER_BLOCKED](#) = MAKE_ERROR_RESULT(CFC_FUNCTIONAL, 0x03),
[CER_INSUFFICIENT_FUNDS](#) = MAKE_ERROR_RESULT(CFC_ACCOUNT, 0x01),
[CER_ACCOUNT_NOT_FOUND](#) = MAKE_ERROR_RESULT(CFC_ACCOUNT, 0x02),
[CER_ACCOUNT_EXISTS](#) = MAKE_ERROR_RESULT(CFC_ACCOUNT, 0x03), [CER_ERROR](#) = (~0) }

- *CNP Error Result Types (CER) enum* [DEPOSIT_TYPE](#) { [DT_INVALID](#) = 0, [DT_CASH](#) = 0x01, [DT_CHECK](#) = 0x02 }
- *CNP Deposit types (DT) enum* [TRANSACTION_TYPE](#) { [TT_INVALID](#) = 0, [TT_DEPOSIT](#) = 0x01, [TT_WITHDRAWAL](#) = 0x02, [TT_STAMP_PURCHASE](#) = 0x03 }

CNP Transaction types (TT) Functions

- bool [Succeeded](#) ([cnp::CER_TYPE](#) cerRR) throw ()

Variables

- const [WORD](#) [g_wMajorVersion](#) = 1
Global message sequence number.
- const [WORD](#) [g_wMinorVersion](#) = 1
Protocol minor version (i.e. x.3)
- const [DWORD](#) [g_dwValidationKey](#) = 0x00DEAD01
CNP Validation Key.
- const size_t [MAX_NAME_LEN](#) = 32
[first,last,email] name field lengths
- const [WORD](#) [INVALID_CLIENT_ID](#) = (~0)
Used for error checking & default initialization.
- const [WORD](#) [INVALID_PIN](#) = 0
Used for error checking & default initialization.

Typedef Documentation

typedef unsigned short [cnp::WORD](#)

typedef unsigned long [cnp::DWORD](#)

typedef unsigned long long [cnp::QWORD](#)

Enumeration Type Documentation

enum [cnp::CNP_MSG_TYPE](#)

Enumerator

CMT_INVALID used for initialization and error checking
CMT_CONNECT
CMT_CREATE_ACCOUNT
CMT_LOGON
CMT_LOGOFF
CMT_DEPOSIT

CMT_WITHDRAWAL
CMT_BALANCE_QUERY
CMT_TRANSACTION_QUERY
CMT_PURCHASE_STAMPS

enum [cnp::CNP_MSG_SUBTYPE](#)

Enumerator

CMS_INVALID used for initialization and error checking
CMS_REQUEST
CMS_RESPONSE

enum [cnp::CER_TYPE](#)

Enumerator

CER_SUCCESS Success!
CER_AUTHENTICATION_FAILED Invalid validation key.
CER_UNSUPPORTED_PROTOCOL Protocol version not supported.
CER_INVALID_CLIENT_ID Invalid client ID found.
CER_INVALID_NAME_PIN Invalid name or pin.
CER_INVALID_ARGUMENTS Invalid arguments used.
CER_CLIENT_NOT_LOGGEDON Client not logged-on.
CER_DRAWER_BLOCKED Mechanical Failure.
CER_INSUFFICIENT_FUNDS Insufficient funds available.
CER_ACCOUNT_NOT_FOUND Client account does not exist.
CER_ACCOUNT_EXISTS Prior account already exists.
CER_ERROR Generic error result.

Function Documentation

bool cnp::Succeeded ([cnp::CER_TYPE](#) cerRR) throw ()

Variable Documentation

const [WORD](#) cnp::g_wMajorVersion = 1

The following is a little complicated, but I will try to explain what is going on.

g_dwSequenceNumber is a static global variable intended to be auto-incremented by the client as part of constructing request messages. It needs to be defined such that it is accessible by various message constructors, but at the same time does not create a linker error as it can be included in multiple .cpp files. (resulting in multiple instances across various COMDATs)

So, we are telling the linker to just use one of the instances here if it finds multiple instance declarations.

Another complication is that MSVC++ & GNUC++ cannot agree on how to do this.

See also:

<https://gcc.gnu.org/onlinedocs/gcc/Vague-Linkage.html>

[https://msdn.microsoft.com/EN-US/library/5tkz6s71\(v=VS.120,d=hv.2\).aspx](https://msdn.microsoft.com/EN-US/library/5tkz6s71(v=VS.120,d=hv.2).aspx) CNP Protocol version
Protocol major version (i.e. 1.x)

const [WORD](#) cnp::g_wMinorVersion = 1

const [DWORD](#) cnp::g_dwValidationKey = 0x00DEAD01

const size_t cnp::MAX_NAME_LEN = 32

See also:

CNP_CREATE_ACCOUNT_REQUEST, CNP_LOGON_REQUEST

const [WORD](#) cnp::INVALID_CLIENT_ID = (~0)

const [WORD](#) cnp::INVALID_PIN = 0

cnp::prim Namespace Reference

Classes

- struct [BALANCE_QUERY_REQUEST](#)
 - *Balance Query Request Primitive.* struct [BALANCE_QUERY_RESPONSE](#)
 - *Balance Query Response Primitive.* struct [CONNECT_REQUEST](#)
 - *Connect Request Primitive.* struct [CONNECT_RESPONSE](#)
 - *Connection Response Primitive.* struct [CREATE_ACCOUNT_REQUEST](#)
 - *Create Account Request Primitive.* struct [CREATE_ACCOUNT_RESPONSE](#)
 - *Create Account Response Primitive.* struct [DEPOSIT_REQUEST](#)
 - *Deposit Request Primitive.* struct [DEPOSIT_RESPONSE](#)
 - *Deposit Response Primitive.* struct [LOGOFF_REQUEST](#)
 - *Logoff Request Primitive.* struct [LOGOFF_RESPONSE](#)
 - *Logoff Response Primitive.* struct [LOGON_REQUEST](#)
 - *Logon Request Primitive.* struct [LOGON_RESPONSE](#)
 - *Logon Response Primitive.* struct [STAMP_PURCHASE_REQUEST](#)
 - *Purchase Stamp Request Primitive.* struct [STAMP_PURCHASE_RESPONSE](#)
 - *Stamp Purchase Response Primitive.* struct [TRANSACTION_QUERY_REQUEST](#)
 - *Transaction Query Request Primitive.* struct [TRANSACTION_QUERY_RESPONSE](#)
 - *Transaction Query Result Primitive.* struct [WITHDRAWAL_REQUEST](#)
 - *Withdrawal Request Primitive.* struct [WITHDRAWAL_RESPONSE](#)
 - *Withdrawal Response Primitive.*
-

Class Documentation

cnp::prim::_BALANCE_QUERY_REQUEST Struct Reference

Balance Query Request Primitive.

Public Member Functions

- [_BALANCE_QUERY_REQUEST](#) ()
-

Constructor & Destructor Documentation

cnp::prim::_BALANCE_QUERY_REQUEST::_BALANCE_QUERY_REQUEST ()

cnp::prim::_BALANCE_QUERY_RESPONSE Struct Reference

Balance Query Response Primitive.

Public Member Functions

- [_BALANCE_QUERY_RESPONSE](#) ([DWORD](#) dwResult=[cnp::CER_ERROR](#), [DWORD](#) dwBalance=0)
- [DWORD](#) [get_Balance](#) (void) const

Public Attributes

- [DWORD](#) [m_dwResult](#)
Success or Error code from [cnp::CER_TYPE](#).
 - [DWORD](#) [m_dwBalance](#)
Current Client account balance.
-

Detailed Description

See also:

[cnp::CER_TYPE](#)

Constructor & Destructor Documentation

cnp::prim::_BALANCE_QUERY_RESPONSE::_BALANCE_QUERY_RESPONSE ([DWORD](#) dwResult = [cnp::CER_ERROR](#), [DWORD](#) dwBalance = 0)

Member Function Documentation

[DWORD](#) `cnp::prim::_BALANCE_QUERY_RESPONSE::get_Balance (void) const`

Member Data Documentation

[DWORD](#) `cnp::prim::_BALANCE_QUERY_RESPONSE::m_dwResult`

[DWORD](#) `cnp::prim::_BALANCE_QUERY_RESPONSE::m_dwBalance`

cnp::prim::_CONNECT_REQUEST Struct Reference

Connect Request Primitive.

Public Member Functions

- [_CONNECT_REQUEST](#) ()
Default constructor.
- [_CONNECT_REQUEST](#) ([WORD](#) wMajorVersion, [WORD](#) wMinorVersion, [DWORD](#) dwKey)
Initialization constructor.

Public Attributes

- [WORD m_wMajorVersion](#)
Client Major Protocol version number.
 - [WORD m_wMinorVersion](#)
Client Minor Protocol version number.
 - [DWORD m_dwValidationKey](#)
Used by Server to authenticate the connection.
-

Detailed Description

Field(s)	Begin Byte	End Byte
m_wMajorVersion	0	1
m_wMinorVersion	2	3
m_dwValidationKey	4	7

Constructor & Destructor Documentation

`cnp::prim::_CONNECT_REQUEST::_CONNECT_REQUEST ()`

`cnp::prim::_CONNECT_REQUEST::_CONNECT_REQUEST (WORD wMajorVersion, WORD wMinorVersion, DWORD dwKey)`

Member Data Documentation

[WORD](#) `cnp::prim::_CONNECT_REQUEST::m_wMajorVersion`

[WORD](#) `cnp::prim::_CONNECT_REQUEST::m_wMinorVersion`

[DWORD](#) `cnp::prim::_CONNECT_REQUEST::m_dwValidationKey`

cnp::prim::_CONNECT_RESPONSE Struct Reference

Connection Response Primitive.

Public Member Functions

- [_CONNECT_RESPONSE](#) ([DWORD](#) dwResult=[cnp::CER_ERROR](#), [WORD](#) wMajorVersion=0, [WORD](#) wMinorVersion=0, [WORD](#) wClientID=[INVALID_CLIENT_ID](#))
Initialization Constructor.

Public Attributes

- [DWORD](#) `m_dwResult`
Success or Error code from [cnp::CER_TYPE](#).
 - [WORD](#) `m_wMajorVersion`
Server Major Protocol version number.
 - [WORD](#) `m_wMinorVersion`
Server Minor Protocol version number.
 - [WORD](#) `m_wClientID`
-

Detailed Description

Field(s)	Begin Byte	End Byte
m_dwResult	0	3
m_wMajorVersion	4	5
m_wMinorVersion	6	7
m_wClientID	8	9

See also:

[cnp::CER_TYPE](#)

Constructor & Destructor Documentation

cnp::prim::_CONNECT_RESPONSE::_CONNECT_RESPONSE ([DWORD](#) *dwResult* = [cnp::CER_ERROR](#), [WORD](#) *wMajorVersion* = 0, [WORD](#) *wMinorVersion* = 0, [WORD](#) *wClientID* = [INVALID_CLIENT_ID](#))

Parameters:

in	<i>dwResult</i>	Server generated cnp::CER_TYPE result
in	<i>wMajorVersion</i>	Current Server major protocol version number
in	<i>wMinorVersion</i>	Current Server minor protocol version number
in	<i>wClientID</i>	Server generated ClientID

Member Data Documentation

[DWORD](#) **cnp::prim::_CONNECT_RESPONSE::m_dwResult**

[WORD](#) **cnp::prim::_CONNECT_RESPONSE::m_wMajorVersion**

[WORD](#) **cnp::prim::_CONNECT_RESPONSE::m_wMinorVersion**

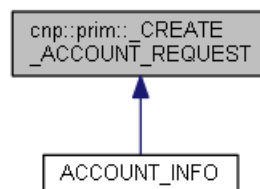
[WORD](#) **cnp::prim::_CONNECT_RESPONSE::m_wClientID**

generated by the Server and is required in all subsequent request messages by the Client

cnp::prim::_CREATE_ACCOUNT_REQUEST Struct Reference

Create Account Request Primitive.

Inheritance diagram for **cnp::prim::_CREATE_ACCOUNT_REQUEST**:



Public Member Functions

- [_CREATE_ACCOUNT_REQUEST](#) ()
Default constructor.

- [_CREATE_ACCOUNT_REQUEST](#) (const char *szFirstName, const char *szLastName, const char *szEmailAddress, [WORD](#) wPIN, [DWORD](#) dwSSNumber, [DWORD](#) dwDLNumber)
Initialization constructor.
- void [set_FirstName](#) (const char *szSet)
- void [set_LastName](#) (const char *szSet)
- void [set_EmailAddress](#) (const char *szSet)

Public Attributes

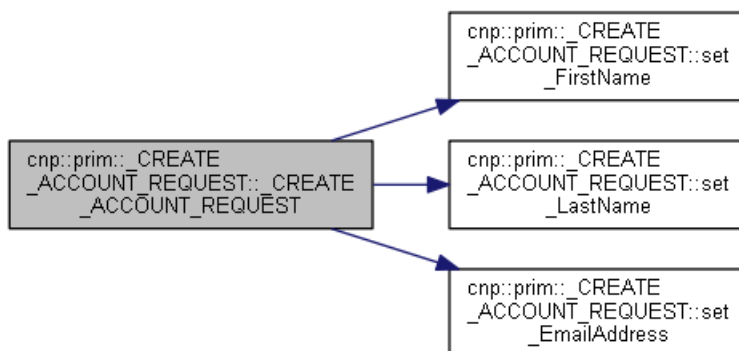
- char [m_szFirstName](#) [[MAX_NAME_LEN](#)]
User's First Name.
- char [m_szLastName](#) [[MAX_NAME_LEN](#)]
User's Last Name.
- char [m_szEmailAddress](#) [[MAX_NAME_LEN](#)]
User's Email Address.
- [WORD](#) [m_wPIN](#)
User's Personal Identification Number.
- [DWORD](#) [m_dwSSNumber](#)
(optional) User's Social Security Number
- [DWORD](#) [m_dwDLNumber](#)
(optional) User's Drivers License Number

Constructor & Destructor Documentation

cnpp::prim::_CREATE_ACCOUNT_REQUEST::_CREATE_ACCOUNT_REQUEST ()

cnpp::prim::_CREATE_ACCOUNT_REQUEST::_CREATE_ACCOUNT_REQUEST (const char * szFirstName, const char * szLastName, const char * szEmailAddress, [WORD](#) wPIN, [DWORD](#) dwSSNumber, [DWORD](#) dwDLNumber)

Here is the call graph for this function:



Member Function Documentation

void cnp::prim::_CREATE_ACCOUNT_REQUEST::set_FirstName (const char * szSet)

Parameters:

in	szSet	address containing null-terminated first name
----	-------	---

void cnp::prim::_CREATE_ACCOUNT_REQUEST::set_LastName (const char * szSet)

Parameters:

in	szSet	address containing null-terminated last name
----	-------	--

void cnp::prim::_CREATE_ACCOUNT_REQUEST::set_EmailAddress (const char * szSet)

Parameters:

in	szSet	address containing null-terminated email address
----	-------	--

Member Data Documentation

char cnp::prim::_CREATE_ACCOUNT_REQUEST::m_szFirstName[\[MAX_NAME_LEN\]](#)

char cnp::prim::_CREATE_ACCOUNT_REQUEST::m_szLastName[\[MAX_NAME_LEN\]](#)

char cnp::prim::_CREATE_ACCOUNT_REQUEST::m_szEmailAddress[\[MAX_NAME_LEN\]](#)

[WORD](#) cnp::prim::_CREATE_ACCOUNT_REQUEST::m_wPIN

[DWORD](#) cnp::prim::_CREATE_ACCOUNT_REQUEST::m_dwSSNumber

[DWORD](#) cnp::prim::_CREATE_ACCOUNT_REQUEST::m_dwDLNumber

cnp::prim::_CREATE_ACCOUNT_RESPONSE Struct Reference

Create Account Response Primitive.

Public Member Functions

- [_CREATE_ACCOUNT_RESPONSE](#) ([DWORD](#) dwResult=[cnp::CER_ERROR](#))

Public Attributes

- [DWORD](#) m_dwResult
Success or Error code from [cnp::CER_TYPE](#).

Detailed Description

See also:

[cnp::CER_TYPE](#)

Constructor & Destructor Documentation

`cnp::prim::_CREATE_ACCOUNT_RESPONSE::_CREATE_ACCOUNT_RESPONSE (DWORD
dwResult = cnp::CER_ERROR)`

Member Data Documentation

[DWORD](#) `cnp::prim::_CREATE_ACCOUNT_RESPONSE::m_dwResult`

cnp::prim::_DEPOSIT_REQUEST Struct Reference

Deposit Request Primitive.

Public Member Functions

- [_DEPOSIT_REQUEST](#) ([DWORD](#) *dwAmount*=0, [DEPOSIT_TYPE](#) *Type*=[DT_INVALID](#))
Initialization constructor.

Public Attributes

- [DWORD](#) `m_dwAmount`
Amount excluding decimal point (i.e. \$100.00 would be 10000)
 - [WORD](#) `m_wType`
[cnp::DT_CASH](#) or [cnp::DT_CHECK](#)
-

Detailed Description

See also:

[cnp::DEPOSIT_TYPE](#)

Constructor & Destructor Documentation

`cnp::prim::_DEPOSIT_REQUEST::_DEPOSIT_REQUEST (DWORD dwAmount = 0, DEPOSIT_TYPE Type = DT_INVALID)`

Member Data Documentation

[DWORD](#) `cnp::prim::_DEPOSIT_REQUEST::m_dwAmount`

[WORD](#) `cnp::prim::_DEPOSIT_REQUEST::m_wType`

cnp::prim::_DEPOSIT_RESPONSE Struct Reference

Deposit Response Primitive.

Public Member Functions

- [_DEPOSIT_RESPONSE](#) ([DWORD](#) dwResult=[cnp::CER_ERROR](#))

Public Attributes

- [DWORD](#) `m_dwResult`
Success or Error code from [cnp::CER_TYPE](#).
-

Detailed Description

See also:

[cnp::CER_TYPE](#)

Constructor & Destructor Documentation

`cnp::prim::_DEPOSIT_RESPONSE::_DEPOSIT_RESPONSE (DWORD dwResult = cnp::CER_ERROR)`

Member Data Documentation

[DWORD](#) `cnp::prim::_DEPOSIT_RESPONSE::m_dwResult`

cnp::prim::_LOGOFF_REQUEST Struct Reference

Logoff Request Primitive.

Public Member Functions

- [_LOGOFF_REQUEST](#) ()
-

Constructor & Destructor Documentation

cnp::prim::_LOGOFF_REQUEST::_LOGOFF_REQUEST ()

cnp::prim::_LOGOFF_RESPONSE Struct Reference

Logoff Response Primitive.

Public Member Functions

- [_LOGOFF_RESPONSE](#) ([DWORD](#) dwResult=[cnp::CER_ERROR](#))

Public Attributes

- [DWORD](#) m_dwResult
Success or Error code from [cnp::CER_TYPE](#).
-

Detailed Description

See also:

[cnp::CER_TYPE](#)

Constructor & Destructor Documentation

cnp::prim::_LOGOFF_RESPONSE::_LOGOFF_RESPONSE ([DWORD](#) dwResult = [cnp::CER_ERROR](#))

Member Data Documentation

[DWORD](#) cnp::prim::_LOGOFF_RESPONSE::m_dwResult

cnp::prim::_LOGON_REQUEST Struct Reference

Logon Request Primitive.

Public Member Functions

- [_LOGON_REQUEST](#) ()
Default constructor.
- [_LOGON_REQUEST](#) (const char *szFirstName, [WORD](#) wPIN)
Initialization constructor.
- void [set_FirstName](#) (const char *szSet)

Public Attributes

- char [m_szFirstName](#) [[MAX_NAME_LEN](#)]
User's first name.
- [WORD](#) [m_wPIN](#)
Personal Identification Number.

Constructor & Destructor Documentation

cnp::prim::_LOGON_REQUEST::_LOGON_REQUEST ()

cnp::prim::_LOGON_REQUEST::_LOGON_REQUEST (const char * szFirstName, [WORD](#) wPIN)

Here is the call graph for this function:



Member Function Documentation

void cnp::prim::_LOGON_REQUEST::set_FirstName (const char * szSet)

Parameters:

in	szSet	address containing null terminated first name
----	-------	---

Member Data Documentation

char cnp::prim::_LOGON_REQUEST::m_szFirstName[[MAX_NAME_LEN](#)]

[WORD](#) cnp::prim::_LOGON_REQUEST::m_wPIN

cnp::prim::_LOGON_RESPONSE Struct Reference

Logon Response Primitive.

Public Member Functions

- [_LOGON_RESPONSE](#) ([DWORD](#) dwResult=[cnp::CER_ERROR](#))

Public Attributes

- [DWORD](#) m_dwResult
Success or Error code from [cnp::CER_TYPE](#).

Detailed Description

See also:

[cnp::CER_TYPE](#)

Constructor & Destructor Documentation

cnp::prim::_LOGON_RESPONSE::_LOGON_RESPONSE ([DWORD](#) dwResult = [cnp::CER_ERROR](#))

Member Data Documentation

[DWORD](#) cnp::prim::_LOGON_RESPONSE::m_dwResult

cnp::prim::_STAMP_PURCHASE_REQUEST Struct Reference

Purchase Stamp Request Primitive.

Public Member Functions

- [_STAMP_PURCHASE_REQUEST](#) ()
Default constructor.
- [_STAMP_PURCHASE_REQUEST](#) ([DWORD](#) dwAmount)
Initialization constructor.

Public Attributes

- [DWORD](#) m_dwAmount
Amount excluding decimal point (i.e. \$100.00 would be 10000)

Constructor & Destructor Documentation

`cnp::prim::_STAMP_PURCHASE_REQUEST::_STAMP_PURCHASE_REQUEST ()`

`cnp::prim::_STAMP_PURCHASE_REQUEST::_STAMP_PURCHASE_REQUEST (DWORD dwAmount)`

Member Data Documentation

[DWORD](#) `cnp::prim::_STAMP_PURCHASE_REQUEST::m_dwAmount`

cnp::prim::_STAMP_PURCHASE_RESPONSE Struct Reference

Stamp Purchase Response Primitive.

Public Member Functions

- [_STAMP_PURCHASE_RESPONSE](#) ([DWORD](#) *dwResult*=[cnp::CER_ERROR](#))

Public Attributes

- [DWORD](#) `m_dwResult`
Success or Error code from [cnp::CER_TYPE](#).
-

Detailed Description

See also:

[cnp::CER_TYPE](#)

Constructor & Destructor Documentation

`cnp::prim::_STAMP_PURCHASE_RESPONSE::_STAMP_PURCHASE_RESPONSE (DWORD dwResult = cnp::CER_ERROR)`

Member Data Documentation

[DWORD](#) `cnp::prim::_STAMP_PURCHASE_RESPONSE::m_dwResult`

cnp::prim::_TRANSACTION_QUERY_REQUEST Struct Reference

Transaction Query Request Primitive.

Public Member Functions

- [_TRANSACTION_QUERY_REQUEST](#) ()
Default constructor.
- [_TRANSACTION_QUERY_REQUEST](#) ([DWORD](#) dwStartID, [WORD](#) wTransactionCount)
Initialization constructor.

Public Attributes

- [DWORD](#) m_dwStartID
the transaction number to begin the query from
- [WORD](#) m_wTransactionCount
the number of transactions requested

Constructor & Destructor Documentation

cnp::prim::_TRANSACTION_QUERY_REQUEST::_TRANSACTION_QUERY_REQUEST ()

cnp::prim::_TRANSACTION_QUERY_REQUEST::_TRANSACTION_QUERY_REQUEST ([DWORD](#) dwStartID, [WORD](#) wTransactionCount)

Member Data Documentation

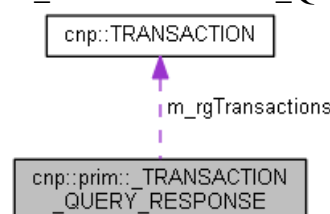
[DWORD](#) cnp::prim::_TRANSACTION_QUERY_REQUEST::m_dwStartID

[WORD](#) cnp::prim::_TRANSACTION_QUERY_REQUEST::m_wTransactionCount

cnp::prim::_TRANSACTION_QUERY_RESPONSE Struct Reference

Transaction Query Result Primitive.

Collaboration diagram for cnp::prim::_TRANSACTION_QUERY_RESPONSE:



Public Member Functions

- [TRANSACTION_QUERY_RESPONSE](#) ([DWORD](#) dwResult=[cnp::CER_ERROR](#), [WORD](#) wTransactionCount=0)
- [WORD](#) [get_TransactionCount](#) (void) const

Public Attributes

- [DWORD](#) [m_dwResult](#)
Success or Error code from [cnp::CER_TYPE](#).
 - [WORD](#) [m_wTransactionCount](#)
number of transactions returned in array
 - [TRANSACTION](#) [m_rgTransactions](#) []
unsized array of Transaction records
-

Detailed Description

See also:

[cnp::CER_TYPE](#)
[cnp::TRANSACTION](#)

Constructor & Destructor Documentation

[cnp::prim::_TRANSACTION_QUERY_RESPONSE::_TRANSACTION_QUERY_RESPONSE](#) ([DWORD](#) dwResult = [cnp::CER_ERROR](#), [WORD](#) wTransactionCount = 0)

Member Function Documentation

[WORD](#) [cnp::prim::_TRANSACTION_QUERY_RESPONSE::get_TransactionCount](#) (void) const

Member Data Documentation

[DWORD](#) [cnp::prim::_TRANSACTION_QUERY_RESPONSE::m_dwResult](#)

[WORD](#) [cnp::prim::_TRANSACTION_QUERY_RESPONSE::m_wTransactionCount](#)

[TRANSACTION](#) [cnp::prim::_TRANSACTION_QUERY_RESPONSE::m_rgTransactions](#) []

cnp::prim::_WITHDRAWAL_REQUEST Struct Reference

Withdrawal Request Primitive.

Public Member Functions

- [_WITHDRAWAL_REQUEST](#) ()
Default constructor.
- [_WITHDRAWAL_REQUEST](#) ([DWORD](#) dwAmount)
Initialization constructor.

Public Attributes

- [DWORD](#) m_dwAmount
Amount excluding decimal point (i.e. \$100.00 would be 10000)

Constructor & Destructor Documentation

cnp::prim::_WITHDRAWAL_REQUEST::_WITHDRAWAL_REQUEST ()

cnp::prim::_WITHDRAWAL_REQUEST::_WITHDRAWAL_REQUEST ([DWORD](#) dwAmount)

Member Data Documentation

[DWORD](#) cnp::prim::_WITHDRAWAL_REQUEST::m_dwAmount

cnp::prim::_WITHDRAWAL_RESPONSE Struct Reference

Withdrawal Response Primitive.

Public Member Functions

- [_WITHDRAWAL_RESPONSE](#) ([DWORD](#) dwResult=[cnp::CER_ERROR](#))

Public Attributes

- [DWORD](#) m_dwResult
Success or Error code from [cnp::CER_TYPE](#).

Detailed Description

See also:

[cnp::CER_TYPE](#)

Constructor & Destructor Documentation

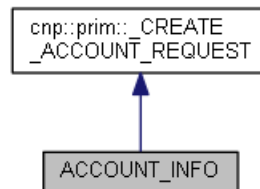
`cnp::prim::_WITHDRAWAL_RESPONSE::_WITHDRAWAL_RESPONSE (DWORD dwResult = cnp::CER_ERROR)`

Member Data Documentation

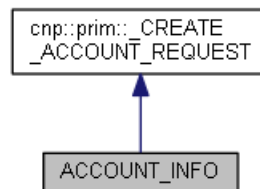
[DWORD](#) `cnp::prim::_WITHDRAWAL_RESPONSE::m_dwResult`

ACCOUNT_INFO Struct Reference

Inheritance diagram for ACCOUNT_INFO:



Collaboration diagram for ACCOUNT_INFO:



Public Types

- typedef [cnp::QWORD](#) [key_type](#)
- typedef [cnp::prim::CREATE_ACCOUNT_REQUEST](#) [_Base](#)

Public Member Functions

- [ACCOUNT_INFO](#) ()
Default Constructor.
- [ACCOUNT_INFO](#) (const [_Base](#) &base, const [cnp::QWORD](#) &qwID=[INVALID_CUSTOMER_ID](#), [cnp::DWORD](#) dwBalance=[INVALID_BALANCE](#))
Initialization Constructor.
- [ACCOUNT_INFO](#) (const [ACCOUNT_INFO](#) &rhs)
Copy Constructor.
- const [cnp::QWORD](#) & [get_PrimaryKey](#) (void) const throw ()
- const [cnp::QWORD](#) & [get_CustomerID](#) (void) const throw ()
- [cnp::DWORD](#) [get_Balance](#) (void) const throw ()

- void [set_Balance](#) ([cnp::DWORD](#) dwSet) throw ()
- void [decr_Balance](#) ([cnp::DWORD](#) dwSet) throw ()
- void [incr_Balance](#) ([cnp::DWORD](#) dwSet) throw ()

Public Attributes

- [cnp::QWORD](#) m_qwCustomerID
- [cnp::DWORD](#) m_dwBalance

Detailed Description

[ACCOUNT_INFO](#) is used to maintain and persist information as it relates to an individual customer. It uses the Customer ID as the key field.

Member Typedef Documentation

typedef [cnp::QWORD](#) [ACCOUNT_INFO::key_type](#)

typedef [cnp::prim:: CREATE_ACCOUNT_REQUEST](#) [ACCOUNT_INFO:: Base](#)

Constructor & Destructor Documentation

[ACCOUNT_INFO::ACCOUNT_INFO](#) ()

[ACCOUNT_INFO::ACCOUNT_INFO](#) (const [Base](#) & base, const [cnp::QWORD](#) & qwID = [INVALID_CUSTOMER_ID](#), [cnp::DWORD](#) dwBalance = [INVALID_BALANCE](#))

[ACCOUNT_INFO::ACCOUNT_INFO](#) (const [ACCOUNT_INFO](#) & rhs)

Member Function Documentation

const [cnp::QWORD](#) & [ACCOUNT_INFO::get_PrimaryKey](#) (void) const throw ()

This method is used to provide a generic interface to retrieve a record's primary key field. In this instance, it is a thin wrapper around [get_CustomerID\(\)](#).

Return values:

cnp::QWORD	containing the primary key value
----------------------------	----------------------------------

Here is the call graph for this function:



const [cnp::QWORD](#)& ACCOUNT_INFO::get_CustomerID (void) const throw ()

[cnp::DWORD](#) ACCOUNT_INFO::get_Balance (void) const throw ()

void ACCOUNT_INFO::set_Balance ([cnp::DWORD](#) dwSet) throw ()

void ACCOUNT_INFO::decr_Balance ([cnp::DWORD](#) dwSet) throw ()

void ACCOUNT_INFO::incr_Balance ([cnp::DWORD](#) dwSet) throw ()

Member Data Documentation

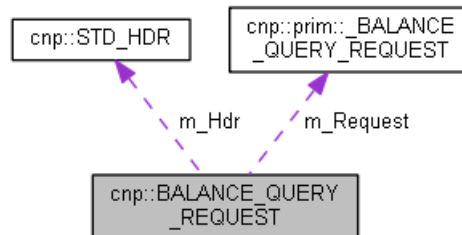
[cnp::QWORD](#) ACCOUNT_INFO::m_qwCustomerID

[cnp::DWORD](#) ACCOUNT_INFO::m_dwBalance

cnp::BALANCE_QUERY_REQUEST Struct Reference

[Client] Balance Query Request message

Collaboration diagram for cnp::BALANCE_QUERY_REQUEST:



Public Member Functions

- [BALANCE_QUERY_REQUEST](#) ()
Default constructor.
- [BALANCE_QUERY_REQUEST](#) ([WORD](#) wClientID, [DWORD](#) dwContext=0)
Initialization constructor.
- `size_t` [get_Size](#) (void) const
- [DWORD](#) [get_MsgType](#) (void) const
- [WORD](#) [get_ClientID](#) (void) const
- [DWORD](#) [get_Sequence](#) (void) const
- [DWORD](#) [get_Context](#) (void) const

Public Attributes

- [STD_HDR](#) [m_Hdr](#)
- [prim::BALANCE_QUERY_REQUEST](#) [m_Request](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15

Constructor & Destructor Documentation

cnp::BALANCE_QUERY_REQUEST::BALANCE_QUERY_REQUEST ()

cnp::BALANCE_QUERY_REQUEST::BALANCE_QUERY_REQUEST ([WORD](#) *wClientID*, [DWORD](#) *dwContext* = 0)

Note:

auto increments the Client's global sequence number

Parameters:

<i>wClientID</i>	Server generated Client ID
<i>dwContext</i>	[Optional] Client provided field

Member Function Documentation

size_t cnp::BALANCE_QUERY_REQUEST::get_Size (void) const

Return values:

<i>size_t</i>	containing the size of the message in bytes
---------------	---

[DWORD](#) cnp::BALANCE_QUERY_REQUEST::get_MsgType (void) const

Here is the call graph for this function:



[WORD](#) cnp::BALANCE_QUERY_REQUEST::get_ClientID (void) const

Here is the call graph for this function:



[DWORD](#) cnp::BALANCE_QUERY_REQUEST::get_Sequence (void) const

Here is the call graph for this function:



[DWORD](#) cnp::BALANCE_QUERY_REQUEST::get_Context (void) const

Here is the call graph for this function:



Member Data Documentation

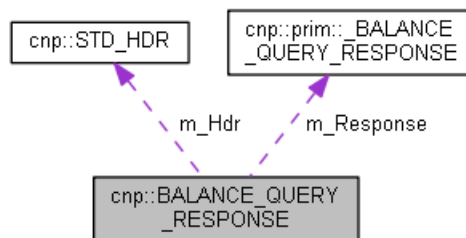
[STD_HDR](#) cnp::BALANCE_QUERY_REQUEST::m_Hdr

[prim:: BALANCE_QUERY_REQUEST](#) cnp::BALANCE_QUERY_REQUEST::m_Request

cnp::BALANCE_QUERY_RESPONSE Struct Reference

[Server] Balance Query Response message

Collaboration diagram for cnp::BALANCE_QUERY_RESPONSE:



Public Member Functions

- **[BALANCE_QUERY_RESPONSE](#)** ([DWORD](#) dwResult, [WORD](#) wClientID, [DWORD](#) dwBalance, [DWORD](#) dwContext, [DWORD](#) dwSequence)
Initialization Constructor.
- **[DWORD](#) get_MsgType** (void) const

- [DWORD get_ResponseResult](#) (void) const
- [DWORD get_Balance](#) (void) const
- [size_t get_Size](#) (void) const

Public Attributes

- [STD_HDR m_Hdr](#)
- [prim::BALANCE_QUERY_RESPONSE m_Response](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Response	m_dwResult	16	19
m_Response	m_dwBalance	20	23

See also:

[cnp::BALANCE_QUERY_REQUEST](#)

Constructor & Destructor Documentation

cnp::BALANCE_QUERY_RESPONSE::BALANCE_QUERY_RESPONSE ([DWORD](#) *dwResult*, [WORD](#) *wClientID*, [DWORD](#) *dwBalance*, [DWORD](#) *dwContext*, [DWORD](#) *dwSequence*)

Parameters:

<i>dwResult</i>	Server generated cnp::CER_TYPE result
<i>wClientID</i>	Copied from BALANCE_QUERY_REQUEST
<i>dwBalance</i>	Client's current account balance
<i>dwContext</i>	Copied from BALANCE_QUERY_REQUEST
<i>dwSequence</i>	Copied from BALANCE_QUERY_REQUEST

Member Function Documentation

[DWORD](#) **cnp::BALANCE_QUERY_RESPONSE::get_MsgType** (void) const

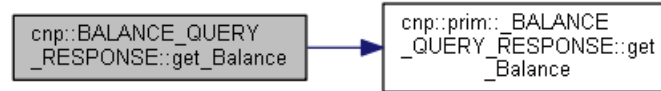
Here is the call graph for this function:



[DWORD](#) `cnp::BALANCE_QUERY_RESPONSE::get_ResponseResult (void) const`

[DWORD](#) `cnp::BALANCE_QUERY_RESPONSE::get_Balance (void) const`

Here is the call graph for this function:



`size_t` `cnp::BALANCE_QUERY_RESPONSE::get_Size (void) const`

Member Data Documentation

[STD_HDR](#) `cnp::BALANCE_QUERY_RESPONSE::m_Hdr`

[prim::_BALANCE_QUERY_RESPONSE](#) `cnp::BALANCE_QUERY_RESPONSE::m_Response`

CNP_Socket Class Reference

Public Member Functions

- [CNP_Socket](#) ()
Default Constructor.
- [CNP_Socket](#) (SOCKET hSocket, const sockaddr_in &remoteAddr)
Initialization Constructor.
- [~CNP_Socket](#) ()
- void [Close](#) (void)
- bool [Create](#) (unsigned short wPort)
- bool [Connect](#) (const char *szHostAddress, unsigned short wPort)
- bool [Listen](#) (int iBackLog)
- bool [Accept](#) (SOCKET &hSocket, sockaddr_in &remoteAddr)
- int [Receive](#) (void *pData, size_t cbLen, int iFlags=0)
Receives data from the underlying connected socket or a bound connectionless socket.
- int [Send](#) (const void *pData, size_t cbLen, int iFlags=0)
- int [SetSocketOption](#) (int iLevel, int iOption, const void *pVal, size_t cbLen)
Sets the underlying socket option.
- bool [SetBlocking](#) (bool bBlocking=true)
- bool [Shutdown](#) (int iHow)
disables sends or receives on underlying socket
- int [GetError](#) (void) const throw ()
- [CNP_Socket](#) ()
Default Constructor.
- [CNP_Socket](#) (SOCKET hSocket, const sockaddr_in &remoteAddr)

Initialization Constructor.

- [~CNP_Socket](#) ()
- void [Close](#) (void)
- bool [Create](#) (unsigned short wPort)
- bool [Connect](#) (const char *szHostAddress, unsigned short wPort)
- bool [Listen](#) (int iBackLog)
- bool [Accept](#) (SOCKET &hSocket, sockaddr_in &remoteAddr)
- int [Receive](#) (void *pData, size_t cbLen, int iFlags=0)
Receives data from the underlying connected socket or a bound connectionless socket.
- int [Send](#) (const void *pData, size_t cbLen, int iFlags=0)
- int [SetSocketOption](#) (int iLevel, int iOption, const void *pVal, size_t cbLen)
Sets the underlying socket option.
- bool [SetBlocking](#) (bool bBlocking=true)
- bool [Shutdown](#) (int iHow)
disables sends or receives on underlying socket
- int [GetError](#) (void) const throw ()

Private Attributes

- SOCKET [m_hSocket](#)
- unsigned short [m_wPort](#)
- sockaddr_in [m_LocalAddr](#)
- sockaddr_in [m_RemoteAddr](#)
- int [m_iError](#)

Constructor & Destructor Documentation

CNP_Socket::CNP_Socket (void)

Performs member data initialization to default values only

CNP_Socket::CNP_Socket (SOCKET hSocket, const sockaddr_in & remoteAddr)

Performs member data initialization to default values, with the exception of the parameters

CNP_Socket::~~CNP_Socket (void)

Here is the call graph for this function:



CNP_Socket::CNP_Socket ()

Performs member data initialization to default values only

CNP_Socket::CNP_Socket (SOCKET hSocket, const sockaddr_in & remoteAddr)

Performs member data initialization to default values, with the exception of the parameters

CNP_Socket::~CNP_Socket ()

Member Function Documentation

void CNP_Socket::Close (void)

bool CNP_Socket::Create (unsigned short wPort)

Here is the call graph for this function:



bool CNP_Socket::Connect (const char * szHostAddress, unsigned short wPort)

bool CNP_Socket::Listen (int iBackLog)

places the underlying socket in a state in which it is listening for an incoming connection

Parameters:

in	<i>iBackLog</i>	The maximum length of the queue of pending connections. If set to SOMAXCONN, the underlying service provider responsible for socket s will set the backlog to a maximum reasonable value. There is no standard provision to obtain the actual backlog value
----	-----------------	---

Return values:

<i>true</i>	on success
<i>false</i>	on failure

bool CNP_Socket::Accept (SOCKET & hSocket, sockaddr_in & remoteAddr)

int CNP_Socket::Receive (void * pData, size_t cbLen, int iFlags = 0)

Parameters:

out	<i>pData</i>	A pointer to the buffer to receive the incoming data
in	<i>cbLen</i>	The length, in bytes, of the buffer pointed to by the pData parameter
in	<i>iFlags</i>	Optional parameter that influences the behavior of this function

Return values:

<i>int</i>	containing the number of bytes received and the buffer pointed to by the pData parameter will contain this data received
<i>0</i>	if the connection has been gracefully closed
<i>SOCKET_ERROR</i>	on failure call GetError() to retrieve the specific error code

int CNP_Socket::Send (const void * pData, size_t cbLen, int iFlags = 0)

int CNP_Socket::SetSocketOption (int iLevel, int iOption, const void * pVal, size_t cbLen)

Parameters:

in	<i>iLevel</i>	The level at which the option is defined (for example,
----	---------------	--

		SOL_SOCKET).
in	<i>iOption</i>	The socket option for which the value is to be set (for example, SO_BROADCAST). The iOption parameter must be a socket option defined within the specified level, or behavior is undefined
in	<i>pVal</i>	A pointer to the buffer in which the value for the requested option is specified
in	<i>cbLen</i>	The size, in bytes, of the buffer pointed to by the pVal parameter

Return values:

0	on success
SOCKET_ERROR	on failure call GetError() to retrieve the specific error code

bool CNP_Socket::SetBlocking (bool *bBlocking* = true)

bool CNP_Socket::Shutdown (int *iHow*)

Parameters:

in	<i>iHow</i>	A platform specific flag that describes what types of operation will no longer be allowed.
----	-------------	--

Return values:

<i>true</i>	on success
<i>false</i>	on failure

int CNP_Socket::GetError (void) const throw ()

Return values:

<i>int</i>	containing the most recent error code
------------	---------------------------------------

void CNP_Socket::Close (void)

bool CNP_Socket::Create (unsigned short *wPort*)

bool CNP_Socket::Connect (const char * *szHostAddress*, unsigned short *wPort*)

bool CNP_Socket::Listen (int *iBackLog*)

places the underlying socket in a state in which it is listening for an incoming connection

Parameters:

in	<i>iBackLog</i>	The maximum length of the queue of pending connections. If set to SOMAXCONN, the underlying service provider responsible for socket s will set the backlog to a maximum reasonable value. There is no standard provision to obtain the actual backlog value
----	-----------------	---

Return values:

<i>true</i>	on success
<i>false</i>	on failure

bool CNP_Socket::Accept (SOCKET & *hSocket*, sockaddr_in & *remoteAddr*)

int CNP_Socket::Receive (void * *pData*, size_t *cbLen*, int *iFlags* = 0)

Parameters:

out	<i>pData</i>	A pointer to the buffer to receive the incoming data
in	<i>cbLen</i>	The length, in bytes, of the buffer pointed to by the <i>pData</i> parameter
in	<i>iFlags</i>	Optional parameter that influences the behavior of this function

Return values:

<i>int</i>	containing the number of bytes received and the buffer pointed to by the <i>pData</i> parameter will contain this data received
0	if the connection has been gracefully closed
<i>SOCKET_ERROR</i>	on failure call GetError() to retrieve the specific error code

int CNP_Socket::Send (const void * *pData*, size_t *cbLen*, int *iFlags* = 0)

int CNP_Socket::SetSocketOption (int *iLevel*, int *iOption*, const void * *pVal*, size_t *cbLen*)

Parameters:

in	<i>iLevel</i>	The level at which the option is defined (for example, SOL_SOCKET).
in	<i>iOption</i>	The socket option for which the value is to be set (for example, SO_BROADCAST). The <i>iOption</i> parameter must be a socket option defined within the specified level, or behavior is undefined
in	<i>pVal</i>	A pointer to the buffer in which the value for the requested option is specified
in	<i>cbLen</i>	The size, in bytes, of the buffer pointed to by the <i>pVal</i> parameter

Return values:

0	on success
<i>SOCKET_ERROR</i>	on failure call GetError() to retrieve the specific error code

bool CNP_Socket::SetBlocking (bool *bBlocking* = true)

bool CNP_Socket::Shutdown (int *iHow*)

Parameters:

in	<i>iHow</i>	A platform specific flag that describes what types of operation will no longer be allowed.
----	-------------	--

Return values:

<i>true</i>	on success
<i>false</i>	on failure

int CNP_Socket::GetError (void) const throw ()

Return values:

<i>int</i>	containing the most recent error code
------------	---------------------------------------

Member Data Documentation

SOCKET CNP_Socket::m_hSocket[private]

unsigned short CNP_Socket::m_wPort[private]

sockaddr_in CNP_Socket::m_LocalAddr[private]

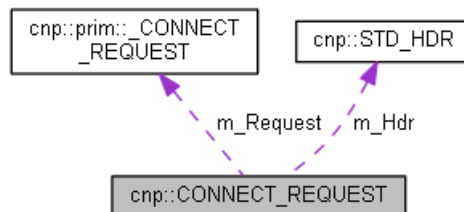
sockaddr_in CNP_Socket::m_RemoteAddr[private]

int CNP_Socket::m_iError[private]

cnp::CONNECT_REQUEST Struct Reference

[Client] Connect Request message

Collaboration diagram for cnp::CONNECT_REQUEST:



Public Member Functions

- [CONNECT_REQUEST](#) ([WORD](#) wClientID=0, [WORD](#) wMajorVersion=[g_wMajorVersion](#), [WORD](#) wMinorVersion=[g_wMinorVersion](#), [DWORD](#) dwValidationKey=[g_dwValidationKey](#), [DWORD](#) dwContext=0) *Initialization Constructor.*
- [size_t](#) [get_Size](#) (void) const
- [DWORD](#) [get_MsgType](#) (void) const
- [WORD](#) [get_ClientID](#) (void) const
- [DWORD](#) [get_Sequence](#) (void) const
- [DWORD](#) [get_Context](#) (void) const
- [WORD](#) [get_ClientMajorVersion](#) (void) const
- [WORD](#) [get_ClientMinorVersion](#) (void) const
- [DWORD](#) [get_ClientValidationKey](#) (void) const

Public Attributes

- [STD_HDR](#) [m_Hdr](#)
 - [prim::CONNECT_REQUEST](#) [m_Request](#)
-

Detailed Description

The connect request message establishes an authenticated connection with the Server. No Client ID is required for this message, but is provided by Server in [CONNECT_RESPONSE](#) message.

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Request	m_dwValidationKey	16	19
m_Request	m_wMajorVersion	20	21
m_Request	m_wMinorVersion	22	23

Constructor & Destructor Documentation

cnp::CONNECT_REQUEST::CONNECT_REQUEST ([WORD](#) *wClientID* = 0, [WORD](#) *wMajorVersion* = [g_wMajorVersion](#), [WORD](#) *wMinorVersion* = [g_wMinorVersion](#), [DWORD](#) *dwValidationKey* = [g_dwValidationKey](#), [DWORD](#) *dwContext* = 0)

Parameters:

in	<i>wClientID</i>	[Optional] This value is ignored by server on initial connect
in	<i>wMajorVersion</i>	[Optional] Defaulted to g_wMajorVersion
in	<i>wMinorVersion</i>	[Optional] Defaulted to g_wMinorVersion
in	<i>dwValidationKey</i>	[Optional] Default to g_dwValidationKey
in	<i>dwContext</i>	[Optional] field for Client's use

Note:

auto increments the Client's global sequence number

Member Function Documentation

size_t cnp::CONNECT_REQUEST::get_Size (void) const

Return values:

<i>size_t</i>	containing the size of the message in bytes
---------------	---

[DWORD](#) **cnp::CONNECT_REQUEST::get_MsgType (void) const**

Here is the call graph for this function:



[WORD](#) **cnp::CONNECT_REQUEST::get_ClientID (void) const**

Here is the call graph for this function:



[DWORD](#) **cnp::CONNECT_REQUEST::get_Sequence (void) const**

Here is the call graph for this function:



[DWORD](#) **cnp::CONNECT_REQUEST::get_Context (void) const**

Here is the call graph for this function:



[WORD](#) **cnp::CONNECT_REQUEST::get_ClientMajorVersion (void) const**

[WORD](#) **cnp::CONNECT_REQUEST::get_ClientMinorVersion (void) const**

[DWORD](#) **cnp::CONNECT_REQUEST::get_ClientValidationKey (void) const**

Member Data Documentation

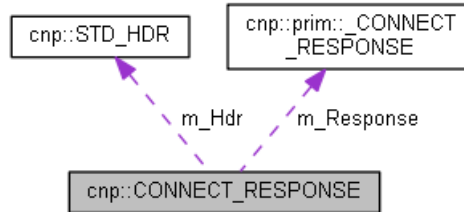
[STD_HDR](#) **cnp::CONNECT_REQUEST::m_Hdr**

[prim:: CONNECT_REQUEST](#) **cnp::CONNECT_REQUEST::m_Request**

cnp::CONNECT_RESPONSE Struct Reference

[Server] Connect Response message

Collaboration diagram for `cnp::CONNECT_RESPONSE`:



Public Member Functions

- [CONNECT_RESPONSE](#) ()
Default Constructor.
- [CONNECT_RESPONSE](#) ([DWORD](#) dwResult, [WORD](#) wClientID, [WORD](#) wMajorVersion, [WORD](#) wMinorVersion, [DWORD](#) dwSequence, [DWORD](#) dwContext)
Initialization Constructor.
- [DWORD](#) [get_MsgType](#) (void) const
- [WORD](#) [get_ClientID](#) (void) const
- [DWORD](#) [get_ResponseResult](#) (void) const
- [size_t](#) [get_Size](#) (void) const

Public Attributes

- [STD_HDR](#) [m_Hdr](#)
- [prim::CONNECT_RESPONSE](#) [m_Response](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Response	m_dwResult	16	19
m_Response	m_wMajorVersion	20	21
m_Response	m_wMinorVersion	22	23
m_Response	m_wClientID	24	25

See also:

[cnp::CONNECT_REQUEST](#)

Constructor & Destructor Documentation

cnp::CONNECT_RESPONSE::CONNECT_RESPONSE ()

cnp::CONNECT_RESPONSE::CONNECT_RESPONSE ([DWORD](#) *dwResult*, [WORD](#) *wClientID*, [WORD](#) *wMajorVersion*, [WORD](#) *wMinorVersion*, [DWORD](#) *dwSequence*, [DWORD](#) *dwContext*)

Parameters:

in	<i>dwResult</i>	Server generated cnp::CER_TYPE result
in	<i>wClientID</i>	Provided by Server on a successful connection response
in	<i>wMajorVersion</i>	current Server major version
in	<i>wMinorVersion</i>	current Server minor version
in	<i>dwSequence</i>	copied from CONNECT_REQUEST
in	<i>dwContext</i>	copied from CONNECT_REQUEST

Member Function Documentation

[DWORD](#) **cnp::CONNECT_RESPONSE::get_MsgType (void) const**

Here is the call graph for this function:



[WORD](#) **cnp::CONNECT_RESPONSE::get_ClientID (void) const**

[DWORD](#) **cnp::CONNECT_RESPONSE::get_ResponseResult (void) const**

size_t **cnp::CONNECT_RESPONSE::get_Size (void) const**

Return values:

<i>size_t</i>	containing the size of the message in bytes
---------------	---

Member Data Documentation

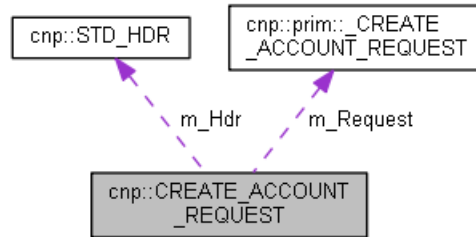
[STD_HDR](#) **cnp::CONNECT_RESPONSE::m_Hdr**

[prim::CONNECT_RESPONSE](#) **cnp::CONNECT_RESPONSE::m_Response**

cnp::CREATE_ACCOUNT_REQUEST Struct Reference

[Client] Create Account Request message

Collaboration diagram for cnp::CREATE_ACCOUNT_REQUEST:



Public Member Functions

- [CREATE_ACCOUNT_REQUEST](#) ()
Default constructor.
- [CREATE_ACCOUNT_REQUEST](#) ([WORD](#) wClientID, const char *szFirstName, const char *szLastName, const char *szEmailAddress, [WORD](#) wPIN, [DWORD](#) dwSSN=0, [DWORD](#) dwDLN=0, [DWORD](#) dwClientContext=0)
Initialization constructor.
- [size_t get_Size](#) (void) const
- [DWORD get_MsgType](#) (void) const
- [WORD get_ClientID](#) (void) const
- [DWORD get_Sequence](#) (void) const
- [DWORD get_Context](#) (void) const
- const char * [get_FirstName](#) (void) const
- const char * [get_LastName](#) (void) const
- const char * [get_EmailAddress](#) (void) const
- [WORD get_PIN](#) (void) const
- [DWORD get_SSNumber](#) (void) const
- [DWORD get_DLNumber](#) (void) const

Public Attributes

- [STD_HDR m_Hdr](#)
- [prim::CREATE_ACCOUNT_REQUEST m_Request](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Request	m_szFirsName	16	47
m_Request	m_szLastName	48	79
m_Request	m_szEmailAddress	80	111
m_Request	m_wPIN	112	113
m_Request	m_dwSSNumber	114	117
m_Request	m_dwDLNumber	118	121

Constructor & Destructor Documentation

cnp::CREATE_ACCOUNT_REQUEST::CREATE_ACCOUNT_REQUEST ()

cnp::CREATE_ACCOUNT_REQUEST::CREATE_ACCOUNT_REQUEST ([WORD](#) *wClientID*, const char * *szFirstName*, const char * *szLastName*, const char * *szEmailAddress*, [WORD](#) *wPIN*, [DWORD](#) *dwSSN* = 0, [DWORD](#) *dwDLN* = 0, [DWORD](#) *dwClientContext* = 0)

Parameters:

in	<i>wClientID</i>	Provided by Server on a successful connection response
in	<i>szFirstName</i>	Client's first name (used in LOGON messages)
in	<i>szLastName</i>	Client's last name
in	<i>szEmailAddress</i>	Client's email address
in	<i>wPIN</i>	Client Generated PIN (used in LOGON messages)
in	<i>dwSSN</i>	[Optional] Social Security Number field
in	<i>dwDLN</i>	[Optional] Driver's License Number field
in	<i>dwClientContext</i>	[Optional] field for the Client's use

Note:

auto increments the Client's global sequence number

Member Function Documentation

size_t cnp::CREATE_ACCOUNT_REQUEST::get_Size (void) const

Return values:

<i>size_t</i>	containing the size of the message in bytes
---------------	---

[DWORD](#) **cnp::CREATE_ACCOUNT_REQUEST::get_MsgType (void) const**

Here is the call graph for this function:



[WORD](#) **cnp::CREATE_ACCOUNT_REQUEST::get_ClientID (void) const**

Here is the call graph for this function:



[DWORD](#) `cnp::CREATE_ACCOUNT_REQUEST::get_Sequence (void) const`

Here is the call graph for this function:



[DWORD](#) `cnp::CREATE_ACCOUNT_REQUEST::get_Context (void) const`

Here is the call graph for this function:



`const char* cnp::CREATE_ACCOUNT_REQUEST::get_FirstName (void) const`

`const char* cnp::CREATE_ACCOUNT_REQUEST::get_LastName (void) const`

`const char* cnp::CREATE_ACCOUNT_REQUEST::get_EmailAddress (void) const`

[WORD](#) `cnp::CREATE_ACCOUNT_REQUEST::get_PIN (void) const`

[DWORD](#) `cnp::CREATE_ACCOUNT_REQUEST::get_SSNumber (void) const`

[DWORD](#) `cnp::CREATE_ACCOUNT_REQUEST::get_DLNumber (void) const`

Member Data Documentation

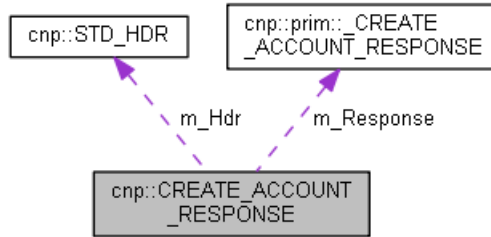
[STD_HDR](#) `cnp::CREATE_ACCOUNT_REQUEST::m_Hdr`

[prim:: CREATE_ACCOUNT_REQUEST](#) `cnp::CREATE_ACCOUNT_REQUEST::m_Request`

`cnp::CREATE_ACCOUNT_RESPONSE` Struct Reference

[Server] Create Account Response message

Collaboration diagram for `cnp::CREATE_ACCOUNT_RESPONSE`:



Public Member Functions

- [CREATE_ACCOUNT_RESPONSE](#) ()
Default constructor.
- [CREATE_ACCOUNT_RESPONSE](#) ([DWORD](#) dwResult, [WORD](#) wClientID, [DWORD](#) dwSequence, [DWORD](#) dwContext)
Initialization Constructor.
- [DWORD](#) [get_MsgType](#) (void) const
- [DWORD](#) [get_ResponseResult](#) (void) const
- [size_t](#) [get_Size](#) (void) const

Public Attributes

- [STD_HDR](#) [m_Hdr](#)
- [prim::CREATE_ACCOUNT_RESPONSE](#) [m_Response](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Response	m_dwResult	16	19

See also:

[cnp::CREATE_ACCOUNT_REQUEST](#)

Constructor & Destructor Documentation

cnp::CREATE_ACCOUNT_RESPONSE::CREATE_ACCOUNT_RESPONSE ()

cnp::CREATE_ACCOUNT_RESPONSE::CREATE_ACCOUNT_RESPONSE ([DWORD](#) dwResult, [WORD](#) wClientID, [DWORD](#) dwSequence, [DWORD](#) dwContext)

Parameters:

in	<i>dwResult</i>	Server generated cnp::CER_TYPE result
----	-----------------	---

in	<i>wClientID</i>	Copied from CREATE_ACCOUNT_REQUEST
in	<i>dwSequence</i>	Copied from CREATE_ACCOUNT_REQUEST
in	<i>dwContext</i>	Copied from CREATE_ACCOUNT_REQUEST

Member Function Documentation

[DWORD](#) [cnp::CREATE_ACCOUNT_RESPONSE::get_MsgType \(void \) const](#)

Here is the call graph for this function:



[DWORD](#) [cnp::CREATE_ACCOUNT_RESPONSE::get_ResponseResult \(void \) const](#)

[size_t](#) [cnp::CREATE_ACCOUNT_RESPONSE::get_Size \(void \) const](#)

Return values:

<i>size_t</i>	containing the size of the message in bytes
---------------	---

Member Data Documentation

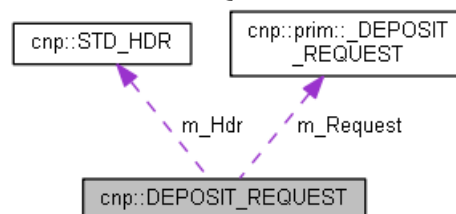
[STD_HDR](#) [cnp::CREATE_ACCOUNT_RESPONSE::m_Hdr](#)

[prim:: CREATE_ACCOUNT_RESPONSE](#) [cnp::CREATE_ACCOUNT_RESPONSE::m_Response](#)

cnp::DEPOSIT_REQUEST Struct Reference

[Client] Deposit Request message

Collaboration diagram for [cnp::DEPOSIT_REQUEST](#):



Public Member Functions

- [DEPOSIT_REQUEST](#) ([WORD](#) wClientID, [DWORD](#) dwAmount, [DEPOSIT_TYPE](#) Type, [DWORD](#) dwContext=0)

Initialization constructor.

- `size_t` [get_Size](#) (void) const
- [DWORD](#) [get_MsgType](#) (void) const
- [WORD](#) [get_ClientID](#) (void) const
- [DWORD](#) [get_Sequence](#) (void) const
- [DWORD](#) [get_Context](#) (void) const
- [DWORD](#) [get_Amount](#) (void) const
- [WORD](#) [get_DepositType](#) (void) const

Public Attributes

- [STD_HDR](#) [m_Hdr](#)
- [prim::DEPOSIT_REQUEST](#) [m_Request](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Request	m_dwAmount	16	19
m_Request	m_wType	20	21

Constructor & Destructor Documentation

`cnp::DEPOSIT_REQUEST::DEPOSIT_REQUEST` ([WORD](#) *wClientID*, [DWORD](#) *dwAmount*, [DEPOSIT_TYPE](#) *Type*, [DWORD](#) *dwContext* = 0)

Note:

auto increments the Client's global sequence number

Parameters:

<i>wClientID</i>	Server generated Client ID
<i>dwAmount</i>	Amount to deposit (in cents)
<i>Type</i>	cnp::DT_CASH or cnp::DT_CHECK
<i>dwContext</i>	[Optional] Client provided field

Member Function Documentation

`size_t` `cnp::DEPOSIT_REQUEST::get_Size` (void) const

Return values:

<code>size_t</code>	containing the size of the message in bytes
---------------------	---

[DWORD](#) `cnp::DEPOSIT_REQUEST::get_MsgType (void) const`

Here is the call graph for this function:



[WORD](#) `cnp::DEPOSIT_REQUEST::get_ClientID (void) const`

Here is the call graph for this function:



[DWORD](#) `cnp::DEPOSIT_REQUEST::get_Sequence (void) const`

Here is the call graph for this function:



[DWORD](#) `cnp::DEPOSIT_REQUEST::get_Context (void) const`

Here is the call graph for this function:



[DWORD](#) `cnp::DEPOSIT_REQUEST::get_Amount (void) const`

[WORD](#) `cnp::DEPOSIT_REQUEST::get_DepositType (void) const`

Member Data Documentation

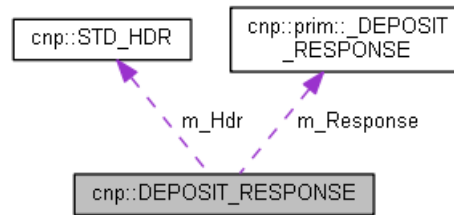
[STD_HDR](#) `cnp::DEPOSIT_REQUEST::m_Hdr`

[prim:: DEPOSIT_REQUEST](#) `cnp::DEPOSIT_REQUEST::m_Request`

cnp::DEPOSIT_RESPONSE Struct Reference

[Server] Deposit Response message

Collaboration diagram for cnp::DEPOSIT_RESPONSE:



Public Member Functions

- [DEPOSIT_RESPONSE](#) ()
Default Constructor.
- [DEPOSIT_RESPONSE](#) ([DWORD](#) dwResult, [WORD](#) wClientID, [DWORD](#) dwSequence, [DWORD](#) dwContext)
Initialization Constructor.
- [DWORD](#) [get_MsgType](#) (void) const
- [DWORD](#) [get_ResponseResult](#) (void) const
- [size_t](#) [get_Size](#) (void) const

Public Attributes

- [STD_HDR](#) [m_Hdr](#)
- [prim::DEPOSIT_RESPONSE](#) [m_Response](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Response	m_dwResult	16	19

See also:

[cnp::DEPOSIT_REQUEST](#)

Constructor & Destructor Documentation

cnp::DEPOSIT_RESPONSE::DEPOSIT_RESPONSE ()

cnp::DEPOSIT_RESPONSE::DEPOSIT_RESPONSE ([DWORD](#) *dwResult*, [WORD](#) *wClientID*, [DWORD](#) *dwSequence*, [DWORD](#) *dwContext*)

Parameters:

<i>dwResult</i>	Server generated cnp::CER_TYPE result
<i>wClientID</i>	Copied from DEPOSIT_REQUEST
<i>dwSequence</i>	Copied from DEPOSIT_REQUEST
<i>dwContext</i>	Copied from DEPOSIT_REQUEST

Member Function Documentation

[DWORD](#) **cnp::DEPOSIT_RESPONSE::get_MsgType (void) const**

Here is the call graph for this function:



[DWORD](#) **cnp::DEPOSIT_RESPONSE::get_ResponseResult (void) const**

size_t cnp::DEPOSIT_RESPONSE::get_Size (void) const

Member Data Documentation

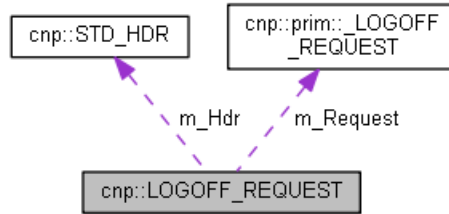
[STD_HDR](#) **cnp::DEPOSIT_RESPONSE::m_Hdr**

[prim:: DEPOSIT_RESPONSE](#) **cnp::DEPOSIT_RESPONSE::m_Response**

cnp::LOGOFF_REQUEST Struct Reference

[Client] Logoff Request message

Collaboration diagram for `cnp::LOGOFF_REQUEST`:



Public Member Functions

- [LOGOFF_REQUEST](#) ()
Default Constructor.
- [LOGOFF_REQUEST](#) ([WORD](#) wClientID, [DWORD](#) dwContext=0)
Initialization Constructor.
- [size_t](#) [get_Size](#) (void) const
- [DWORD](#) [get_MsgType](#) (void) const
- [WORD](#) [get_ClientID](#) (void) const
- [DWORD](#) [get_Sequence](#) (void) const
- [DWORD](#) [get_Context](#) (void) const

Public Attributes

- [STD_HDR](#) [m_Hdr](#)
- [prim::LOGOFF_REQUEST](#) [m_Request](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15

Constructor & Destructor Documentation

cnp::LOGOFF_REQUEST::LOGOFF_REQUEST ()

cnp::LOGOFF_REQUEST::LOGOFF_REQUEST ([WORD](#) wClientID, [DWORD](#) dwContext = 0)

Note:

auto increments the Client's global sequence number

Parameters:

wClientID	Server generated Client ID
-----------	----------------------------

<i>dwContext</i>	[Optional] Client provided field
------------------	----------------------------------

Member Function Documentation

size_t **cnp::LOGOFF_REQUEST::get_Size (void) const**

Return values:

<i>size_t</i>	containing the size of the message in bytes
---------------	---

DWORD **cnp::LOGOFF_REQUEST::get_MsgType (void) const**

Here is the call graph for this function:



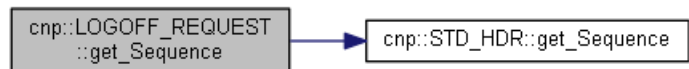
WORD **cnp::LOGOFF_REQUEST::get_ClientID (void) const**

Here is the call graph for this function:



DWORD **cnp::LOGOFF_REQUEST::get_Sequence (void) const**

Here is the call graph for this function:



DWORD **cnp::LOGOFF_REQUEST::get_Context (void) const**

Here is the call graph for this function:



Member Data Documentation

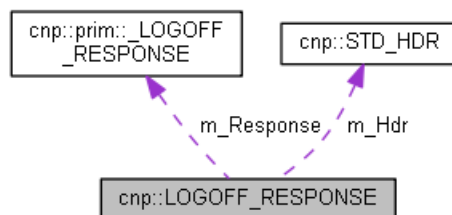
[STD_HDR](#) `cnp::LOGOFF_REQUEST::m_Hdr`

[prim:: LOGOFF_REQUEST](#) `cnp::LOGOFF_REQUEST::m_Request`

cnp::LOGOFF_RESPONSE Struct Reference

[Server] Logoff Response message

Collaboration diagram for `cnp::LOGOFF_RESPONSE`:



Public Member Functions

- [LOGOFF_RESPONSE](#) ()
Default Constructor.
- [LOGOFF_RESPONSE](#) ([DWORD](#) dwResult, [WORD](#) wClientID, [DWORD](#) dwSequence, [DWORD](#) dwContext)
Initialization Constructor.
- [DWORD](#) [get_MsgType](#) (void) const
- [DWORD](#) [get_ResponseResult](#) (void) const
- [size_t](#) [get_Size](#) (void) const

Public Attributes

- [STD_HDR](#) [m_Hdr](#)
- [prim:: LOGOFF_RESPONSE](#) [m_Response](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Response	m_dwResult	16	19

See also:

[cnp::LOGOFF_REQUEST](#)

Constructor & Destructor Documentation

cnp::LOGOFF_RESPONSE::LOGOFF_RESPONSE ()

cnp::LOGOFF_RESPONSE::LOGOFF_RESPONSE ([DWORD](#) *dwResult*, [WORD](#) *wClientID*, [DWORD](#) *dwSequence*, [DWORD](#) *dwContext*)

Parameters:

<i>dwResult</i>	Server generated cnp::CER_TYPE result
<i>wClientID</i>	Copied from LOGOFF_REQUEST
<i>dwSequence</i>	Copied from LOGOFF_REQUEST
<i>dwContext</i>	Copied from LOGOFF_REQUEST

Member Function Documentation

[DWORD](#) **cnp::LOGOFF_RESPONSE::get_MsgType (void) const**

[DWORD](#) **cnp::LOGOFF_RESPONSE::get_ResponseResult (void) const**

size_t cnp::LOGOFF_RESPONSE::get_Size (void) const

Return values:

<i>size_t</i>	containing the size of the message in bytes
---------------	---

Member Data Documentation

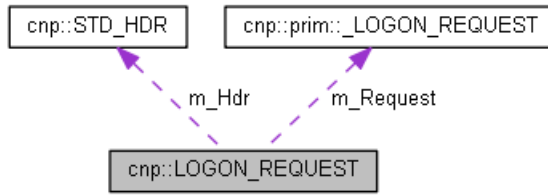
[STD_HDR](#) **cnp::LOGOFF_RESPONSE::m_Hdr**

[prim:: LOGOFF_RESPONSE](#) **cnp::LOGOFF_RESPONSE::m_Response**

cnp::LOGON_REQUEST Struct Reference

[Client] Logon Request message

Collaboration diagram for cnp::LOGON_REQUEST:



Public Member Functions

- [LOGON_REQUEST](#) ()
Default constructor.
- [LOGON_REQUEST](#) ([WORD](#) wClientID, const char *szFirstName, [WORD](#) wPIN, [DWORD](#) dwContext=0)
Initialization constructor.
- size_t [get_Size](#) (void) const
- [DWORD](#) [get_MsgType](#) (void) const
- [WORD](#) [get_ClientID](#) (void) const
- [DWORD](#) [get_Sequence](#) (void) const
- [DWORD](#) [get_Context](#) (void) const
- const char * [get_FirstName](#) (void) const
- [WORD](#) [get_PIN](#) (void) const

Public Attributes

- [STD_HDR m_Hdr](#)
- [prim::_LOGON_REQUEST m_Request](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Request	m_szFirstName	16	47
m_Request	m_wPIN	48	49

Constructor & Destructor Documentation

cnp::LOGON_REQUEST::LOGON_REQUEST ()

cnp::LOGON_REQUEST::LOGON_REQUEST ([WORD](#) wClientID, const char * szFirstName, [WORD](#) wPIN, [DWORD](#) dwContext = 0)

Note:

auto increments the Client's global sequence number

Parameters:

<i>wClientID</i>	Server provided Client ID
<i>szFirstName</i>	Client field used in CREATE ACCOUNT REQUEST
<i>wPIN</i>	Client field used in CREATE ACCOUNT REQUEST
<i>dwContext</i>	[Optional] field provided for Client's use

Member Function Documentation

size_t [cnp::LOGON_REQUEST::get_Size \(void \) const](#)

Return values:

<i>size_t</i>	containing the size of the message in bytes
---------------	---

DWORD [cnp::LOGON_REQUEST::get_MsgType \(void \) const](#)

Here is the call graph for this function:



WORD [cnp::LOGON_REQUEST::get_ClientID \(void \) const](#)

Here is the call graph for this function:



DWORD [cnp::LOGON_REQUEST::get_Sequence \(void \) const](#)

Here is the call graph for this function:



DWORD [cnp::LOGON_REQUEST::get_Context \(void \) const](#)

Here is the call graph for this function:



const char* cnp::LOGON_REQUEST::get_FirstName (void) const

[WORD](#) cnp::LOGON_REQUEST::get_PIN (void) const

Member Data Documentation

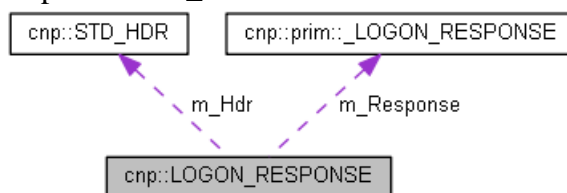
[STD_HDR](#) cnp::LOGON_REQUEST::m_Hdr

[prim:: LOGON_REQUEST](#) cnp::LOGON_REQUEST::m_Request

cnp::LOGON_RESPONSE Struct Reference

[Server] Logon Response message

Collaboration diagram for cnp::LOGON_RESPONSE:



Public Member Functions

- [LOGON_RESPONSE](#) ()
Default Constructor.
- [LOGON_RESPONSE](#) ([DWORD](#) dwResult, [WORD](#) wClientID, [DWORD](#) dwSequence, [DWORD](#) dwContext)
Initialization Constructor.
- [DWORD](#) [get_MsgType](#) (void) const
- [DWORD](#) [get_ResponseResult](#) (void) const
- [size_t](#) [get_Size](#) (void) const

Public Attributes

- [STD_HDR](#) [m_Hdr](#)
- [prim:: LOGON_RESPONSE](#) [m_Response](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11

m_Hdr	m_dwContext	12	15
m_Response	m_dwResult	16	19

See also:

[cnp::LOGON_REQUEST](#)

Constructor & Destructor Documentation

cnp::LOGON_RESPONSE::LOGON_RESPONSE ()

cnp::LOGON_RESPONSE::LOGON_RESPONSE ([DWORD](#) dwResult, [WORD](#) wClientID, [DWORD](#) dwSequence, [DWORD](#) dwContext)

Parameters:

<i>dwResult</i>	Server generated cnp::CER_TYPE result
<i>wClientID</i>	Copied from LOGON_REQUEST
<i>dwSequence</i>	Copied from LOGON_REQUEST
<i>dwContext</i>	Copied from LOGON_REQUEST

Member Function Documentation

[DWORD](#) **cnp::LOGON_RESPONSE::get_MsgType (void) const**

Here is the call graph for this function:



[DWORD](#) **cnp::LOGON_RESPONSE::get_ResponseResult (void) const**

size_t cnp::LOGON_RESPONSE::get_Size (void) const

Return values:

<i>size_t</i>	containing the size of the message in bytes
---------------	---

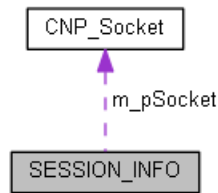
Member Data Documentation

[STD_HDR](#) **cnp::LOGON_RESPONSE::m_Hdr**

[prim:: LOGON_RESPONSE](#) **cnp::LOGON_RESPONSE::m_Response**

SESSION_INFO Struct Reference

Collaboration diagram for SESSION_INFO:



Public Types

- typedef [cnp::WORD](#) [key_type](#)

Public Member Functions

- [SESSION_INFO](#) ([cnp::WORD](#) wClientID, [SESSION_STATE](#) sState, [CNP_Socket](#) *pSocket=nullptr)
Initialization Constructor.
- [cnp::WORD](#) [get_ClientID](#) (void) const throw ()
- void [set_ClientID](#) ([cnp::WORD](#) wSet) throw ()
- [cnp::WORD](#) [get_State](#) (void) const throw ()
- void [set_State](#) ([SESSION_STATE](#) sSet) throw ()
- const [cnp::QWORD](#) & [get_CustomerID](#) (void) const throw ()
- void [set_CustomerID](#) (const [cnp::QWORD](#) &qwSet) throw ()

Public Attributes

- [cnp::WORD](#) [m_wClientID](#)
Key field.
- [cnp::WORD](#) [m_wState](#)
- [CNP_Socket](#) * [m_pSocket](#)
- [cnp::QWORD](#) [m_qwCustomerID](#)

Detailed Description

[SESSION_INFO](#) is a runtime only data-structure used to maintain an association between Client ID, session state, socket connection & Customer ID.

Member Typedef Documentation

typedef [cnp::WORD](#) [SESSION_INFO::key_type](#)

Constructor & Destructor Documentation

[SESSION_INFO::SESSION_INFO](#) ([cnp::WORD](#) wClientID, [SESSION_STATE](#) sState, [CNP_Socket](#) * pSocket = nullptr)

Member Function Documentation

[cnp::WORD](#) SESSION_INFO::get_ClientID (void) const throw)

void SESSION_INFO::set_ClientID ([cnp::WORD](#) wSet) throw)

[cnp::WORD](#) SESSION_INFO::get_State (void) const throw)

void SESSION_INFO::set_State ([SESSION_STATE](#) sSet) throw)

const [cnp::QWORD](#)& SESSION_INFO::get_CustomerID (void) const throw)

void SESSION_INFO::set_CustomerID (const [cnp::QWORD](#) & qwSet) throw)

Member Data Documentation

[cnp::WORD](#) SESSION_INFO::m_wClientID

[cnp::WORD](#) SESSION_INFO::m_wState

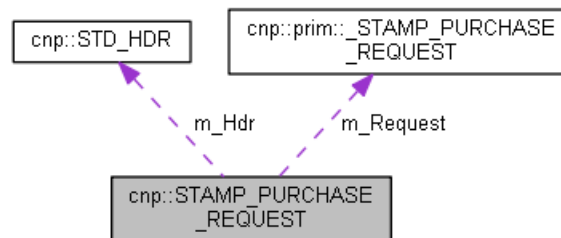
[CNP_Socket*](#) SESSION_INFO::m_pSocket

[cnp::QWORD](#) SESSION_INFO::m_qwCustomerID

cnp::STAMP_PURCHASE_REQUEST Struct Reference

[Client] Stamp Purchase Request Message

Collaboration diagram for cnp::STAMP_PURCHASE_REQUEST:



Public Member Functions

- [STAMP_PURCHASE_REQUEST](#) ([WORD](#) wClientID, [DWORD](#) dwAmount, [DWORD](#) dwContext=0)
Initialization constructor.
- `size_t` [get_Size](#) (void) const
- [DWORD](#) [get_MsgType](#) (void) const
- [WORD](#) [get_ClientID](#) (void) const

- [DWORD get_Sequence](#) (void) const
- [DWORD get_Context](#) (void) const
- [DWORD get_Amount](#) (void) const

Public Attributes

- [STD_HDR m_Hdr](#)
- [prim::STAMP_PURCHASE_REQUEST m_Request](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Request	m_dwAmount	16	19

Constructor & Destructor Documentation

cnp::STAMP_PURCHASE_REQUEST::STAMP_PURCHASE_REQUEST ([WORD](#) *wClientID*, [DWORD](#) *dwAmount*, [DWORD](#) *dwContext* = 0)

Parameters:

in	<i>wClientID</i>	Server generated Client ID
in	<i>dwAmount</i>	Cost of stamps attempting to purchase (in cents) (i.e. 1000 = \$10.00)
in	<i>dwContext</i>	[Optional] field provided by the Client

Note:

auto increments the Client's global sequence number

Member Function Documentation

size_t cnp::STAMP_PURCHASE_REQUEST::get_Size (void) const

[DWORD](#) **cnp::STAMP_PURCHASE_REQUEST::get_MsgType** (void) const

Here is the call graph for this function:



WORD `cnp::STAMP_PURCHASE_REQUEST::get_ClientID (void) const`

Here is the call graph for this function:



DWORD `cnp::STAMP_PURCHASE_REQUEST::get_Sequence (void) const`

Here is the call graph for this function:



DWORD `cnp::STAMP_PURCHASE_REQUEST::get_Context (void) const`

Here is the call graph for this function:



DWORD `cnp::STAMP_PURCHASE_REQUEST::get_Amount (void) const`

Member Data Documentation

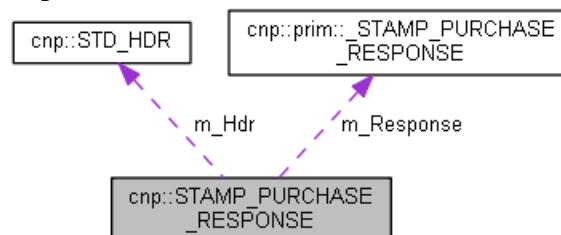
STD_HDR `cnp::STAMP_PURCHASE_REQUEST::m_Hdr`

prim:: STAMP_PURCHASE_REQUEST `cnp::STAMP_PURCHASE_REQUEST::m_Request`

cnp::STAMP_PURCHASE_RESPONSE Struct Reference

[Server] Stamp Purchase Response message

Collaboration diagram for `cnp::STAMP_PURCHASE_RESPONSE`:



Public Member Functions

- [STAMP_PURCHASE_RESPONSE](#) ([DWORD](#) dwResult, [WORD](#) wClientID, [DWORD](#) dwSequence, [DWORD](#) dwContext)
Initialization Constructor.
- [DWORD](#) [get_MsgType](#) (void) const
- [DWORD](#) [get_ResponseResult](#) (void) const
- [size_t](#) [get_Size](#) (void) const

Public Attributes

- [STD_HDR](#) m_Hdr
- [prim::STAMP_PURCHASE_RESPONSE](#) m_Response

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Response	m_dwResult	16	19

See also:

[cnp::STAMP_PURCHASE_REQUEST](#)

Constructor & Destructor Documentation

[cnp::STAMP_PURCHASE_RESPONSE::STAMP_PURCHASE_RESPONSE](#) ([DWORD](#) dwResult, [WORD](#) wClientID, [DWORD](#) dwSequence, [DWORD](#) dwContext)

Parameters:

in	dwResult	Server generated cnp::CER_TYPE result
in	wClientID	Copied from STAMP_PURCHASE_REQUEST
in	dwSequence	Copied from STAMP_PURCHASE_REQUEST
in	dwContext	Copied from STAMP_PURCHASE_REQUEST

Member Function Documentation

[DWORD](#) [cnp::STAMP_PURCHASE_RESPONSE::get_MsgType](#) (void) const

Here is the call graph for this function:



[DWORD](#) cnp::STAMP_PURCHASE_RESPONSE::get_ResponseResult (void) const

size_t cnp::STAMP_PURCHASE_RESPONSE::get_Size (void) const

Member Data Documentation

[STD_HDR](#) cnp::STAMP_PURCHASE_RESPONSE::m_Hdr

[prim:: STAMP_PURCHASE_RESPONSE](#) cnp::STAMP_PURCHASE_RESPONSE::m_Response

cnp::STD_HDR Struct Reference

CNP Standard Message Header.

Public Member Functions

- [STD_HDR](#) ()
Default constructor.
- [STD_HDR](#) ([DWORD](#) dwMsgType, [WORD](#) wDataLen, [WORD](#) wClientID=[INVALID_CLIENT_ID](#), [DWORD](#) dwSequence=0, [DWORD](#) dwContext=0)
Initialization constructor.
- [STD_HDR](#) (const [STD_HDR](#) &rhs)
Copy Constructor.
- [DWORD](#) [get_MsgType](#) (void) const
- [WORD](#) [get_ClientID](#) (void) const
- [DWORD](#) [get_Sequence](#) (void) const
- [DWORD](#) [get_Context](#) (void) const

Public Attributes

- [DWORD](#) [m_dwMsgType](#)
Message Type.
 - [WORD](#) [m_wDataLen](#)
Message data length excluding this header.
 - [WORD](#) [m_wClientID](#)
 - [DWORD](#) [m_dwSequence](#)
 - [DWORD](#) [m_dwContext](#)
[Optional] field, reserved for the Client's use
-

Detailed Description

This is the header message that is provided as part of all CNP Request and Response messages. The m_wDataLen field is subsequently calculated as:

```
sizeof(m_Request) or sizeof(m_Response)
```

to get the size of message excluding the size of the header.

The `m_dwContext` field is exclusively reserved for application use. The contents of this field is returned to the user in the corresponding result message structure without modification.

The `m_wClientID` value is return by the server in the `CONNECTION_RESPONSE` message and is required in all subsequent messages sent by the client to the server.

See also:

[CONNECT_RESPONSE](#)

Field(s)	Begin Byte	End Byte
<code>m_dwMsgType</code>	0	3
<code>m_wDataLen</code>	4	5
<code>m_wClientID</code>	6	7
<code>m_dwSequence</code>	8	11
<code>m_dwContext</code>	12	15

Constructor & Destructor Documentation

`cnp::STD_HDR::STD_HDR ()`

`cnp::STD_HDR::STD_HDR (DWORD dwMsgType, WORD wDataLen, WORD wClientID = INVALID_CLIENT_ID, DWORD dwSequence = 0, DWORD dwContext = 0)`

Parameters:

in	<i>dwMsgType</i>	MSG_TYPE
in	<i>wDataLen</i>	Message data length excluding this header
in	<i>wClientID</i>	Client ID, initially set by the Server & used by Client in subsequent messages
in	<i>dwSequence</i>	Incremented by the Client, used to match Server responses to Client requests
in	<i>dwContext</i>	[Optional] field for Client's use

`cnp::STD_HDR::STD_HDR (const STD_HDR & rhs)`

Member Function Documentation

[DWORD](#) `cnp::STD_HDR::get_MsgType (void) const`

[WORD](#) `cnp::STD_HDR::get_ClientID (void) const`

[DWORD](#) `cnp::STD_HDR::get_Sequence (void) const`

[DWORD](#) `cnp::STD_HDR::get_Context (void) const`

Member Data Documentation

[DWORD](#) `cnp::STD_HDR::m_dwMsgType`

[WORD](#) `cnp::STD_HDR::m_wDataLen`

[WORD](#) `cnp::STD_HDR::m_wClientID`

Client ID, initially set by the Server & used by Client in subsequent messages

[DWORD](#) `cnp::STD_HDR::m_dwSequence`

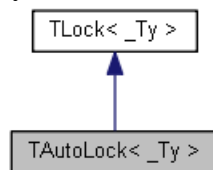
Incremented by the Client, used to match Server responses to Client requests

[DWORD](#) `cnp::STD_HDR::m_dwContext`

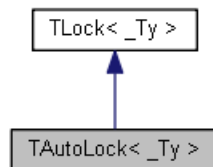
TAutoLock< _Ty > Class Template Reference

"Stack-based" Template

Inheritance diagram for TAutoLock< _Ty >:



Collaboration diagram for TAutoLock< _Ty >:



Public Member Functions

- [TAutoLock](#) (`_Ty *pCS`)
- [~TAutoLock](#) ()

Protected Member Functions

- [TAutoLock](#) ()
 - [TAutoLock](#) (const [TAutoLock](#)< _Ty > &o)
-

Detailed Description

`template<class _Ty>class TAutoLock< _Ty >`

[TAutoLock](#) encapsulates some of the mundane operations of its base class - [TLock](#).

Is meant to be instantiated on the Stack so that on return of the 'Locked' function, it is Unlocked with the destruction of the object.

Warning:

This template class expects 'class _Ty' to support the Lock/Unlock methods

Constructor & Destructor Documentation

`template<class _Ty> TAutoLock< _Ty >::TAutoLock () [protected]`

Declare the Default and Copy Constructor protected so the class cannot be instantiated in this manner

`template<class _Ty> TAutoLock< _Ty >::TAutoLock (const TAutoLock< _Ty > & o) [protected]`

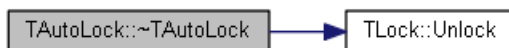
`template<class _Ty> TAutoLock< _Ty >::TAutoLock (_Ty * pCS) [explicit]`

Here is the call graph for this function:



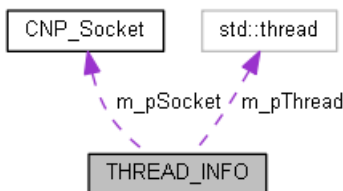
`template<class _Ty> TAutoLock< _Ty >::~~TAutoLock ()`

Here is the call graph for this function:



THREAD_INFO Struct Reference

Collaboration diagram for THREAD_INFO:



Public Member Functions

- [THREAD_INFO](#) (void)

- [~THREAD_INFO](#) (void)

Public Attributes

- std::atomic< bool > [m_bTerminate](#)
- [CNP_Socket](#) * [m_pSocket](#)
- std::thread * [m_pThread](#)

Private Member Functions

- [THREAD_INFO](#) (const [THREAD_INFO](#) &)
- [THREAD_INFO](#) & [operator=](#) (const [THREAD_INFO](#) &)

Detailed Description

A basic data-structure passed to the thread function

Constructor & Destructor Documentation

`THREAD_INFO::THREAD_INFO (void)`

`THREAD_INFO::~~THREAD_INFO (void)`

`THREAD_INFO::THREAD_INFO (const THREAD_INFO &) [private]`

Member Function Documentation

[THREAD_INFO](#)& `THREAD_INFO::operator= (const THREAD_INFO &) [private]`

Member Data Documentation

`std::atomic<bool> THREAD_INFO::m_bTerminate`

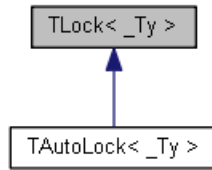
[CNP_Socket](#)* `THREAD_INFO::m_pSocket`

`std::thread* THREAD_INFO::m_pThread`

TLock< _Ty > Class Template Reference

"Stack-based" Template

Inheritance diagram for TLock< _Ty >:



Public Member Functions

- [TLock](#) ()
- void [Lock](#) (void)
- void [Unlock](#) (void)
- void [SetLock](#) (_Ty *pCS)

Protected Member Functions

- [TLock](#) (const [TLock](#)<_Ty> &)
- [TLock](#) (_Ty *pCS)

Private Attributes

- _Ty * [m_pCS](#)

Detailed Description

template<class _Ty>class TLock<_Ty>

This implementation does NOT actually contain a synchronization object, but is used to manipulate an existing one.

Warning:

This template class expects 'class _Ty' to support the Lock/Unlock methods

Constructor & Destructor Documentation

template<class _Ty> [TLock](#)<_Ty>::[TLock](#) (const [TLock](#)<_Ty> &) [protected]

Declare the Default and Copy Constructor protected so the class cannot be instantiated in this manner

template<class _Ty> [TLock](#)<_Ty>::[TLock](#) (_Ty * pCS) [protected]

template<class _Ty> [TLock](#)<_Ty>::[TLock](#) ()

Member Function Documentation

```
template<class _Ty > void TLock<_Ty>::Lock (void )
```

```
template<class _Ty > void TLock<_Ty>::Unlock (void )
```

```
template<class _Ty > void TLock<_Ty>::SetLock (_Ty * pCS)
```

Here is the call graph for this function:



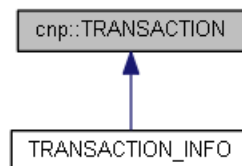
Member Data Documentation

```
template<class _Ty > _Ty* TLock<_Ty>::m_pCS[private]
```

cnp::TRANSACTION Struct Reference

A Customer Transaction Record.

Inheritance diagram for cnp::TRANSACTION:



Public Member Functions

- [TRANSACTION](#) (void)
- [TRANSACTION](#) ([DWORD](#) dwID, const [QWORD](#) &qwDateTime, [DWORD](#) dwAmount, [WORD](#) wType)
- [DWORD](#) [get_ID](#) (void) const
- [DWORD](#) [get_Amount](#) (void) const
- const [QWORD](#) & [get_DateTime](#) (void) const
- [TRANSACTION_TYPE](#) [get_Type](#) (void) const

Public Attributes

- [DWORD](#) [m_dwID](#)
A Server generated unique sequential ID associated with each transaction.
- [QWORD](#) [m_qwDateTime](#)
a 64bit UTC value that represents number of seconds since Epoch
- [DWORD](#) [m_dwAmount](#)
Amount excluding decimal point (i.e. \$100.00 would be 10000)

- [WORD m_wType](#)
The transaction type, represented as TT_DEPOSIT or TT_WITHDRAWAL.

Detailed Description

Field(s)	Begin Byte	End Byte
m_dwID	0	3
m_qwDateTime	4	11
m_dwAmount	12	15
m_wType	16	17

See also:

[TRANSACTION_TYPE](#)

Constructor & Destructor Documentation

`cnp::TRANSACTION::TRANSACTION (void)`

`cnp::TRANSACTION::TRANSACTION (DWORD dwID, const QWORD & qwDateTime, DWORD dwAmount, WORD wType)`

Member Function Documentation

[DWORD](#) `cnp::TRANSACTION::get_ID (void) const`

[DWORD](#) `cnp::TRANSACTION::get_Amount (void) const`

const [QWORD](#)& `cnp::TRANSACTION::get_DateTime (void) const`

[TRANSACTION_TYPE](#) `cnp::TRANSACTION::get_Type (void) const`

Member Data Documentation

[DWORD](#) `cnp::TRANSACTION::m_dwID`

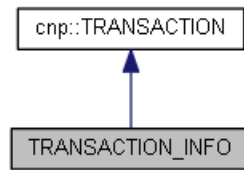
[QWORD](#) `cnp::TRANSACTION::m_qwDateTime`

[DWORD](#) `cnp::TRANSACTION::m_dwAmount`

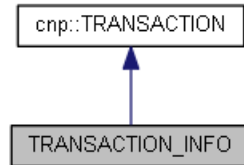
[WORD](#) `cnp::TRANSACTION::m_wType`

TRANSACTION_INFO Struct Reference

Inheritance diagram for TRANSACTION_INFO:



Collaboration diagram for TRANSACTION_INFO:



Public Types

- typedef [cnp::DWORD](#) [key_type](#)

Public Member Functions

- [TRANSACTION_INFO](#) (void)
Default Constructor.
- [TRANSACTION_INFO](#) ([cnp::DWORD](#) dwID, const [cnp::QWORD](#) &qwDateTime, [cnp::DWORD](#) dwAmount, [cnp::WORD](#) wType, const [cnp::QWORD](#) &qwCustomerID)
Initialization Constructor.
- [TRANSACTION_INFO](#) (const [TRANSACTION_INFO](#) &rhs)
Copy Constructor.
- [cnp::DWORD](#) [get_PrimaryKey](#) (void) const throw ()
- const [cnp::QWORD](#) & [get_CustomerID](#) (void) const throw ()

Public Attributes

- [cnp::QWORD](#) [m_qwCustomerID](#)

Detailed Description

[TRANSACTION_INFO](#) is used to maintain a listing of all transactions related to a specific customer. The Transaction ID is used as the primary key.

Member Typedef Documentation

typedef [cnp::DWORD](#) [TRANSACTION_INFO::key_type](#)

Constructor & Destructor Documentation

TRANSACTION_INFO::TRANSACTION_INFO (void)

TRANSACTION_INFO::TRANSACTION_INFO ([cnp::DWORD](#) *dwID*, const [cnp::QWORD](#) & *qwDateTime*, [cnp::DWORD](#) *dwAmount*, [cnp::WORD](#) *wType*, const [cnp::QWORD](#) & *qwCustomerID*)

Parameters:

in	<i>dwID</i>	A Server generated unique sequential ID associated with each transaction
in	<i>qwDateTime</i>	64bit UTC value that represents number of seconds since Epoch
in	<i>dwAmount</i>	Amount excluding decimal point (i.e. \$100.00 would be 10000)
in	<i>wType</i>	The transaction type, represented as cnp::TT_DEPOSIT or cnp::TT_WITHDRAWAL
in	<i>qwCustomerID</i>	Unique customer ID associated with the transaction

TRANSACTION_INFO::TRANSACTION_INFO (const [TRANSACTION_INFO](#) & *rhs*)

Member Function Documentation

[cnp::DWORD](#) TRANSACTION_INFO::get_PrimaryKey (void) const throw ()

This method is used to provide a generic interface to retrieve [TRANSACTION_INFO](#)'s primary key field. In this implementation, it is a wrapper around [cnp::TRANSACTION.get_ID\(\)](#).

Return values:

cnp::DWORD	containing the primary key value
----------------------------	----------------------------------

Here is the call graph for this function:



const [cnp::QWORD](#) & TRANSACTION_INFO::get_CustomerID (void) const throw ()

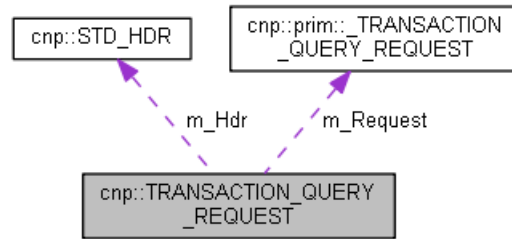
Member Data Documentation

[cnp::QWORD](#) TRANSACTION_INFO::m_qwCustomerID

cnp::TRANSACTION_QUERY_REQUEST Struct Reference

[Client] Transaction Query Request message

Collaboration diagram for `cnp::TRANSACTION_QUERY_REQUEST`:



Public Member Functions

- [TRANSACTION_QUERY_REQUEST](#) ([WORD](#) wClientID, [DWORD](#) dwStartID, [WORD](#) wTransactionCount, [DWORD](#) dwContext=0)
Initialization constructor.
- [size_t](#) [get_Size](#) (void) const
- [DWORD](#) [get_MsgType](#) (void) const
- [WORD](#) [get_ClientID](#) (void) const
- [DWORD](#) [get_Sequence](#) (void) const
- [DWORD](#) [get_Context](#) (void) const
- [DWORD](#) [get_StartID](#) (void) const
- [WORD](#) [get_TransactionCount](#) (void) const

Public Attributes

- [STD_HDR](#) [m_Hdr](#)
- [prim::TRANSACTION_QUERY_REQUEST](#) [m_Request](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Request	m_dwStartID	16	19
m_Request	m_wTransactionCount	20	21

Constructor & Destructor Documentation

cnp::TRANSACTION_QUERY_REQUEST::TRANSACTION_QUERY_REQUEST ([WORD](#) wClientID, [DWORD](#) dwStartID, [WORD](#) wTransactionCount, [DWORD](#) dwContext = 0)

Note:

auto increments the Client's global sequence number

Parameters:

<i>wClientID</i>	Server generated Client ID
<i>dwStartID</i>	Transaction Record ID to begin query from
<i>wTransactionCount</i>	Number of Records requested
<i>dwContext</i>	[Optional] Client provided field

Member Function Documentation

size_t cnp::TRANSACTION_QUERY_REQUEST::get_Size (void) const

Return values:

<i>size_t</i>	containing the size of the message in bytes
---------------	---

DWORD cnp::TRANSACTION_QUERY_REQUEST::get_MsgType (void) const

Here is the call graph for this function:



WORD cnp::TRANSACTION_QUERY_REQUEST::get_ClientID (void) const

Here is the call graph for this function:



DWORD cnp::TRANSACTION_QUERY_REQUEST::get_Sequence (void) const

Here is the call graph for this function:



DWORD cnp::TRANSACTION_QUERY_REQUEST::get_Context (void) const

Here is the call graph for this function:



[DWORD](#) `cnp::TRANSACTION_QUERY_REQUEST::get_StartID (void) const`

[WORD](#) `cnp::TRANSACTION_QUERY_REQUEST::get_TransactionCount (void) const`

Member Data Documentation

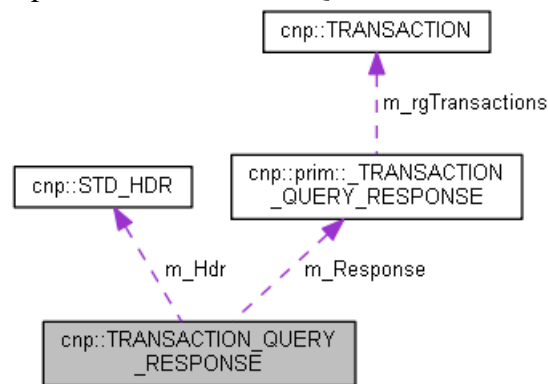
[STD_HDR](#) `cnp::TRANSACTION_QUERY_REQUEST::m_Hdr`

[prim:: TRANSACTION_QUERY_REQUEST](#) `cnp::TRANSACTION_QUERY_REQUEST::m_Request`

cnp::TRANSACTION_QUERY_RESPONSE Struct Reference

[Server] Transaction Query Response message

Collaboration diagram for `cnp::TRANSACTION_QUERY_RESPONSE`:



Public Member Functions

- [TRANSACTION_QUERY_RESPONSE \(DWORD dwResult, WORD wClientID, WORD wTransactionCount, DWORD dwSequence, DWORD dwContext\)](#)
Initialization Constructor.
- `size_t` [get_Size](#) (void) const
- [DWORD](#) [get_MsgType](#) (void) const
- [DWORD](#) [get_ResponseResult](#) (void) const
- [WORD](#) [get_TransactionCount](#) (void) const

Public Attributes

- [STD_HDR](#) `m_Hdr`
- [prim:: TRANSACTION_QUERY_RESPONSE](#) `m_Response`

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Response	m_dwResult	16	19
m_Response	m_wTransactionCount	20	21
m_Response	m_rgTransactions[]	22	...

See also:

[cnp::TRANSACTION_QUERY_REQUEST](#)
[cnp::TRANSACTION](#)

Constructor & Destructor Documentation

cnp::TRANSACTION_QUERY_RESPONSE::TRANSACTION_QUERY_RESPONSE ([DWORD](#) *dwResult*, [WORD](#) *wClientID*, [WORD](#) *wTransactionCount*, [DWORD](#) *dwSequence*, [DWORD](#) *dwContext*)

Parameters:

<i>dwResult</i>	Server generated cnp::CER_TYPE result
<i>wClientID</i>	Copied from TRANSACTION_QUERY_REQUEST
<i>wTransactionCount</i>	Actual number of records returned
<i>dwSequence</i>	Copied from TRANSACTION_QUERY_REQUEST
<i>dwContext</i>	Copied from TRANSACTION_QUERY_REQUEST

Member Function Documentation

size_t cnp::TRANSACTION_QUERY_RESPONSE::get_Size (void) const

Here is the call graph for this function:



[DWORD](#) cnp::TRANSACTION_QUERY_RESPONSE::get_MsgType (void) const

Here is the call graph for this function:



[DWORD](#) `cnp::TRANSACTION_QUERY_RESPONSE::get_ResponseResult (void) const`

[WORD](#) `cnp::TRANSACTION_QUERY_RESPONSE::get_TransactionCount (void) const`

Member Data Documentation

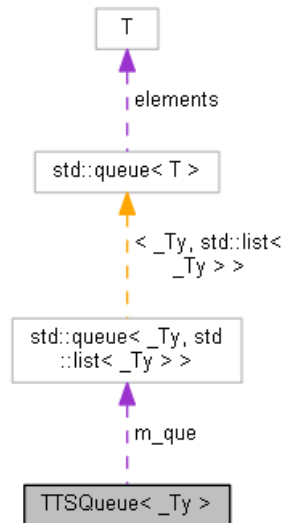
[STD_HDR](#) `cnp::TRANSACTION_QUERY_RESPONSE::m_Hdr`

[prim:: TRANSACTION_QUERY_RESPONSE](#)

`cnp::TRANSACTION_QUERY_RESPONSE::m_Response`

TTSQueue< _Ty > Class Template Reference

Collaboration diagram for TTSQueue< _Ty >:



Public Member Functions

- [TTSQueue](#) ()
- [TTSQueue](#) (const [_Myt](#) &o)
- [_Myt](#) & [operator=](#) (const [_Myt](#) &rhs)
- virtual [~TTSQueue](#) ()
- void [Lock](#) (void) const
- void [Unlock](#) (void) const
- void [Init](#) (bool bLock=true)
Initializes the que by popping all elements until empty.
- bool [GetItems](#) ([items_type](#) &Items, bool bLock=true) const
Returns a snapshot of queued-items. return == true if queue is not empty.
- const [que_type](#) & [GetItems](#) (void) const
Returns a const reference to private stl::queue.

- `const _Ty & Front` (void) const
- `void Push` (const _Ty &Item, bool bLock=true)
Adds an element to the back of the queue.
- `bool Pop` (bool bLock=true)
- `bool PopFront` (_Ty &Item, bool bLock=true)
- `bool IsEmpty` (void) const
Tests if a queue is empty.
- `size_type Size` (void) const
Returns the number of elements in the queue.

Private Types

- `typedef std::queue<_Ty, std::list<_Ty>> que_type`
- `typedef que_type::size_type size_type`
- `typedef std::vector<_Ty> items_type`
- `typedef TTSQueue<_Ty> _Myt`
- `typedef TLock<_Myt> _Lock`
- `typedef TAutoLock<_Myt> _AutoLock`

Private Attributes

- `CAutoCriticalSection m_cs`
- `que_type m_que`

Member Typedef Documentation

```
template<class _Ty> typedef std::queue<_Ty, std::list<_Ty>> TTSQueue<_Ty>
>::que\_type [private]
```

```
template<class _Ty> typedef que_type::size_type TTSQueue<_Ty>::size\_type [private]
```

```
template<class _Ty> typedef std::vector<_Ty> TTSQueue<_Ty>::items\_type [private]
```

```
template<class _Ty> typedef TTSQueue<_Ty> TTSQueue<_Ty>::\_Myt [private]
```

```
template<class _Ty> typedef TLock<\_Myt> TTSQueue<_Ty>::\_Lock [private]
```

```
template<class _Ty> typedef TAutoLock<\_Myt> TTSQueue<_Ty>::\_AutoLock [private]
```

Constructor & Destructor Documentation

```
template<class _Ty> TTSQueue<_Ty>::TTSQueue ()
```

```
template<class _Ty> TTSQueue<_Ty>::TTSQueue (const \_Myt & o)
```

```
template<class _Ty> virtual TTSQueue<_Ty>::~~TTSQueue () [virtual]
```

Member Function Documentation

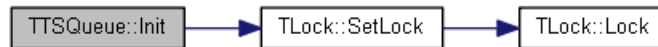
template<class _Ty > [_Myt](#)& [TTSQueue](#)< _Ty >::operator= (const [_Myt](#) & *rhs*)

template<class _Ty > void [TTSQueue](#)< _Ty >::Lock (void) const

template<class _Ty > void [TTSQueue](#)< _Ty >::Unlock (void) const

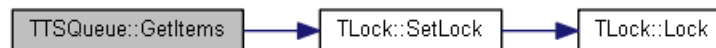
template<class _Ty > void [TTSQueue](#)< _Ty >::Init (bool *bLock* = true)

Here is the call graph for this function:



template<class _Ty > bool [TTSQueue](#)< _Ty >::GetItems ([items_type](#) & *Items*, bool *bLock* = true) const

Here is the call graph for this function:



template<class _Ty > const [que_type](#)& [TTSQueue](#)< _Ty >::GetItems (void) const

template<class _Ty > const _Ty& [TTSQueue](#)< _Ty >::Front (void) const

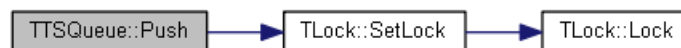
Returns a const reference to the first element at the front of the queue.

Warning:

If the queue is empty, the return value is undefined.

template<class _Ty > void [TTSQueue](#)< _Ty >::Push (const _Ty & *Item*, bool *bLock* = true)

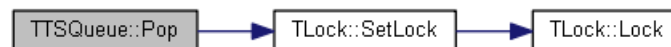
Here is the call graph for this function:



template<class _Ty > bool [TTSQueue](#)< _Ty >::Pop (bool *bLock* = true)

Removes an element from the front of the queue. Returns true if an element was removed.

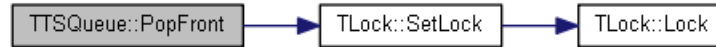
Here is the call graph for this function:



template<class _Ty > bool [TTSQueue](#)< _Ty >::PopFront (_Ty & *Item*, bool *bLock* = true)

Combines the operations of Pop & Front by returning a copy of the 'popped' element.

Here is the call graph for this function:



```
template<class _Ty > bool TTSQueue< _Ty >::IsEmpty (void ) const
```

```
template<class _Ty > size\_type TTSQueue< _Ty >::Size (void ) const
```

Member Data Documentation

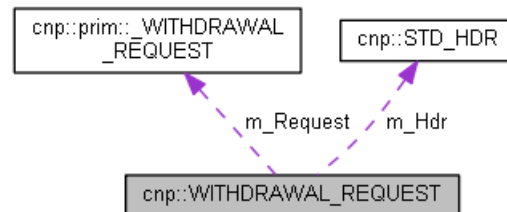
```
template<class _Ty > CAutoCriticalSection TTSQueue< _Ty >::m_cs[mutable], [private]
```

```
template<class _Ty > que\_type TTSQueue< _Ty >::m_que[private]
```

cnp::WITHDRAWAL_REQUEST Struct Reference

[Client] Withdrawal Request message

Collaboration diagram for cnp::WITHDRAWAL_REQUEST:



Public Member Functions

- [WITHDRAWAL_REQUEST](#) ([WORD](#) wClientID, [DWORD](#) dwAmount, [DWORD](#) dwContext=0)
Initialization constructor.
- [size_t](#) [get_Size](#) (void) const
- [DWORD](#) [get_MsgType](#) (void) const
- [WORD](#) [get_ClientID](#) (void) const
- [DWORD](#) [get_Sequence](#) (void) const
- [DWORD](#) [get_Context](#) (void) const
- [DWORD](#) [get_Amount](#) (void) const

Public Attributes

- [STD_HDR](#) [m_Hdr](#)
- [prim::WITHDRAWAL_REQUEST](#) [m_Request](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Request	m_dwAmount	16	19

Constructor & Destructor Documentation

cnp::WITHDRAWAL_REQUEST::WITHDRAWAL_REQUEST ([WORD](#) *wClientID*, [DWORD](#) *dwAmount*, [DWORD](#) *dwContext* = 0)

Note:

auto increments the Client's global sequence number

Parameters:

<i>wClientID</i>	Server generated Client ID
<i>dwAmount</i>	Amount Client wants to withdraw (in cents)
<i>dwContext</i>	[Optional] Client provided field

Member Function Documentation

size_t cnp::WITHDRAWAL_REQUEST::get_Size (void) const

Return values:

<i>size_t</i>	containing the size of the message in bytes
---------------	---

[DWORD](#) **cnp::WITHDRAWAL_REQUEST::get_MsgType** (void) const

Here is the call graph for this function:



[WORD](#) **cnp::WITHDRAWAL_REQUEST::get_ClientID** (void) const

Here is the call graph for this function:



[DWORD](#) [cnp::WITHDRAWAL_REQUEST::get_Sequence \(void \) const](#)

Here is the call graph for this function:



[DWORD](#) [cnp::WITHDRAWAL_REQUEST::get_Context \(void \) const](#)

Here is the call graph for this function:



[DWORD](#) [cnp::WITHDRAWAL_REQUEST::get_Amount \(void \) const](#)

Member Data Documentation

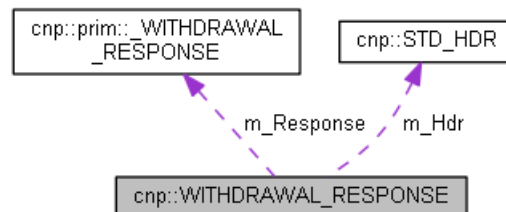
[STD_HDR](#) [cnp::WITHDRAWAL_REQUEST::m_Hdr](#)

[prim:: WITHDRAWAL_REQUEST](#) [cnp::WITHDRAWAL_REQUEST::m_Request](#)

cnp::WITHDRAWAL_RESPONSE Struct Reference

[Server] Withdrawal Response message

Collaboration diagram for `cnp::WITHDRAWAL_RESPONSE`:



Public Member Functions

- [WITHDRAWAL_RESPONSE](#) ([DWORD](#) dwResult, [WORD](#) wClientID, [DWORD](#) dwSequence, [DWORD](#) dwContext)
- [DWORD](#) [get_MsgType](#) (void) const
- [DWORD](#) [get_ResponseResult](#) (void) const
- [size_t](#) [get_Size](#) (void) const

Public Attributes

- [STD_HDR m_Hdr](#)
- [prim::WITHDRAWAL_RESPONSE m_Response](#)

Detailed Description

Message Members	Field	Begin Byte	End Byte
m_Hdr	m_dwMsgType	0	3
m_Hdr	m_wDataLen	4	5
m_Hdr	m_wClientID	6	7
m_Hdr	m_dwSequence	8	11
m_Hdr	m_dwContext	12	15
m_Response	m_dwResult	16	19

See also:

[cnp::WITHDRAWAL_REQUEST](#)

Constructor & Destructor Documentation

cnp::WITHDRAWAL_RESPONSE::WITHDRAWAL_RESPONSE ([DWORD](#) *dwResult*, [WORD](#) *wClientID*, [DWORD](#) *dwSequence*, [DWORD](#) *dwContext*)

Parameters:

<i>dwResult</i>	Server generated cnp::CER_TYPE result
<i>wClientID</i>	Copied from WITHDRAWAL_REQUEST
<i>dwSequence</i>	Copied from WITHDRAWAL_REQUEST
<i>dwContext</i>	Copied from WITHDRAWAL_REQUEST

Member Function Documentation

[DWORD](#) **cnp::WITHDRAWAL_RESPONSE::get_MsgType** (void) const

Here is the call graph for this function:



[DWORD](#) **cnp::WITHDRAWAL_RESPONSE::get_ResponseResult** (void) const

[size_t](#) **cnp::WITHDRAWAL_RESPONSE::get_Size** (void) const

Member Data Documentation

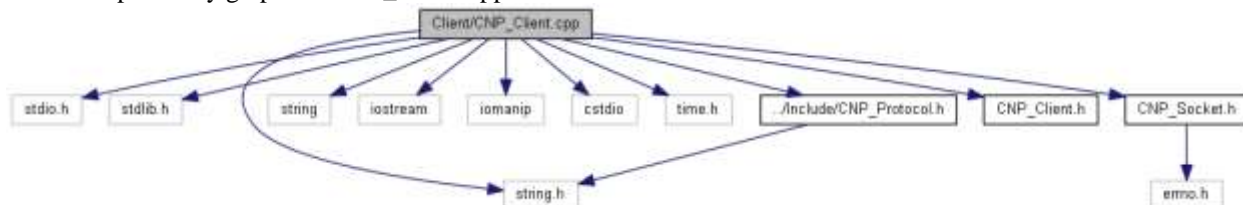
[STD_HDR](#) `cnp::WITHDRAWAL_RESPONSE::m_Hdr`

[prim:: WITHDRAWAL_RESPONSE](#) `cnp::WITHDRAWAL_RESPONSE::m_Response`

File Documentation

Client/CNP_Client.cpp File Reference

Include dependency graph for CNP_Client.cpp:



Macros

- `#define` [CASE_CERTYPE](#)(cer)

Functions

- `const char *` [CerTypeToString](#) (`cnp::CER_TYPE` cerType)
- `const char *` [TransTypeToString](#) (`cnp::TRANSACTION_TYPE` tType)
- `std::string &` [RawTimeToLocalTimeString](#) (`time_t` tRawTime, `std::string &`strLocalTime)
- `void` [PrintBankMenu](#) (`void`)
- `cnp::CER_TYPE` [SendConnect](#) (`CNP_Socket` &socket, `cnp::WORD` &wClientID)
- `cnp::CER_TYPE` [SendCreateAccount](#) (`CNP_Socket` &socket, `cnp::WORD` wClientID)
- `cnp::CER_TYPE` [SendLogIn](#) (`CNP_Socket` &socket, `cnp::WORD` wClientID)
- `cnp::CER_TYPE` [SendLogOut](#) (`CNP_Socket` &socket, `cnp::WORD` wClientID)
- `cnp::CER_TYPE` [SendDeposit](#) (`CNP_Socket` &socket, `cnp::WORD` wClientID)
- `cnp::CER_TYPE` [SendWithdrawal](#) (`CNP_Socket` &socket, `cnp::WORD` wClientID)
- `cnp::CER_TYPE` [SendBalance](#) (`CNP_Socket` &socket, `cnp::WORD` wClientID)
- `cnp::CER_TYPE` [SendTransaction](#) (`CNP_Socket` &socket, `cnp::WORD` wClientID)
- `cnp::CER_TYPE` [SendStampPurchase](#) (`CNP_Socket` &socket, `cnp::WORD` wClientID)
- `int` [main](#) (`int` argc, `char *`argv[])

Variables

- `char` [g_szBuffer](#) [512] = { 0 }

Detailed Description

Author:

Mark L. Short

Date:

April 10, 2015

Macro Definition Documentation

#define CASE_CERTYPE(cer)

```
Value: case cer: \
    return (const char*)#cer;
```

Function Documentation

const char* CerTypeToString ([cnp::CER_TYPE](#) cerType)

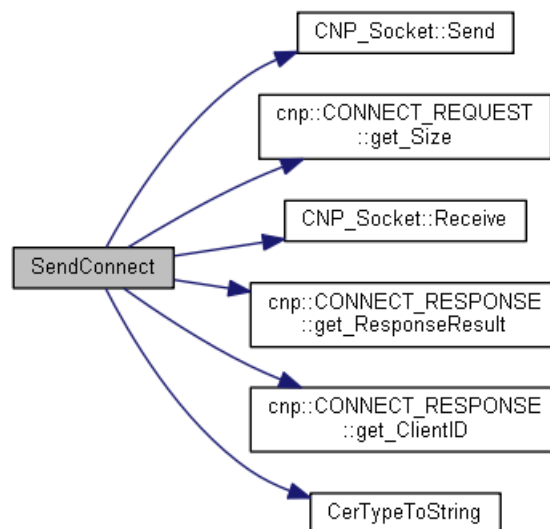
const char* TransTypeToString ([cnp::TRANSACTION_TYPE](#) tType)

std::string& RawTimeToLocalTimeString (time_t tRawTime, std::string & strLocalTime)

void PrintBankMenu (void)

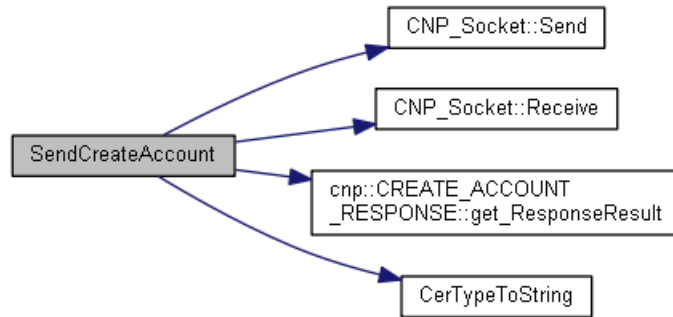
[cnp::CER_TYPE](#) SendConnect ([CNP_Socket](#) & socket, [cnp::WORD](#) & wClientID)

Here is the call graph for this function:



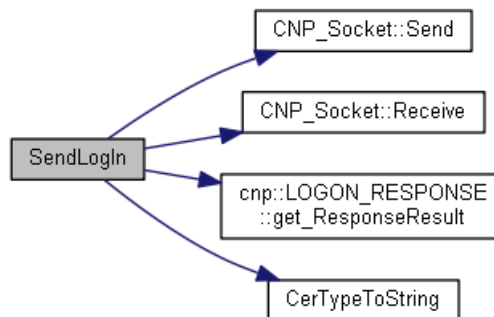
[cnp::CER_TYPE](#) SendCreateAccount ([CNP_Socket](#) & socket, [cnp::WORD](#) wClientID)

Here is the call graph for this function:



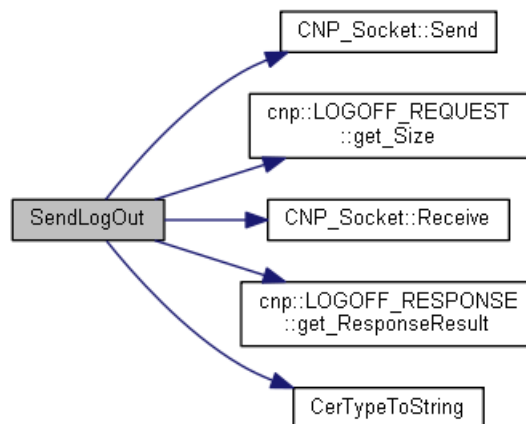
cnp::CER_TYPE SendLogin (CNP_Socket & *socket*, cnp::WORD *wClientID*)

Here is the call graph for this function:



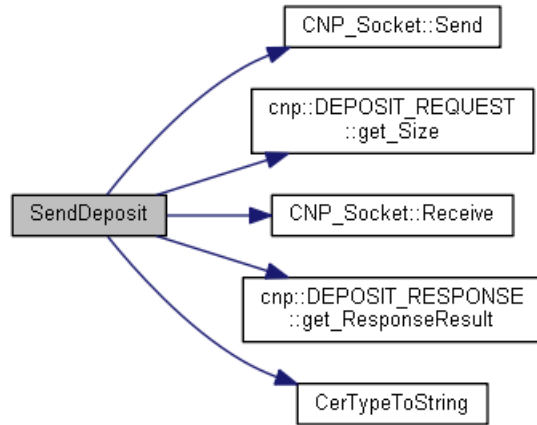
cnp::CER_TYPE SendLogOut (CNP_Socket & *socket*, cnp::WORD *wClientID*)

Here is the call graph for this function:



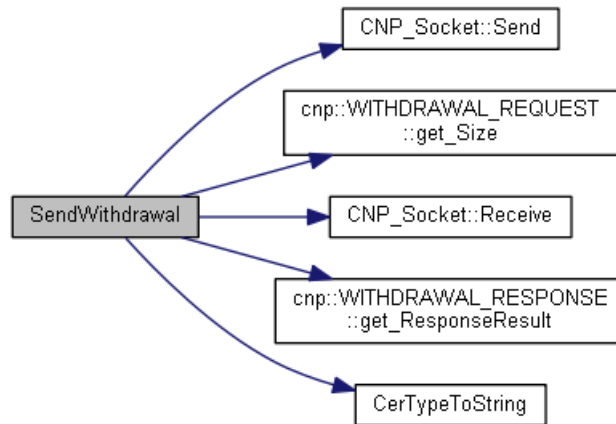
cnp::CER_TYPE SendDeposit (CNP_Socket & *socket*, cnp::WORD *wClientID*)

Here is the call graph for this function:



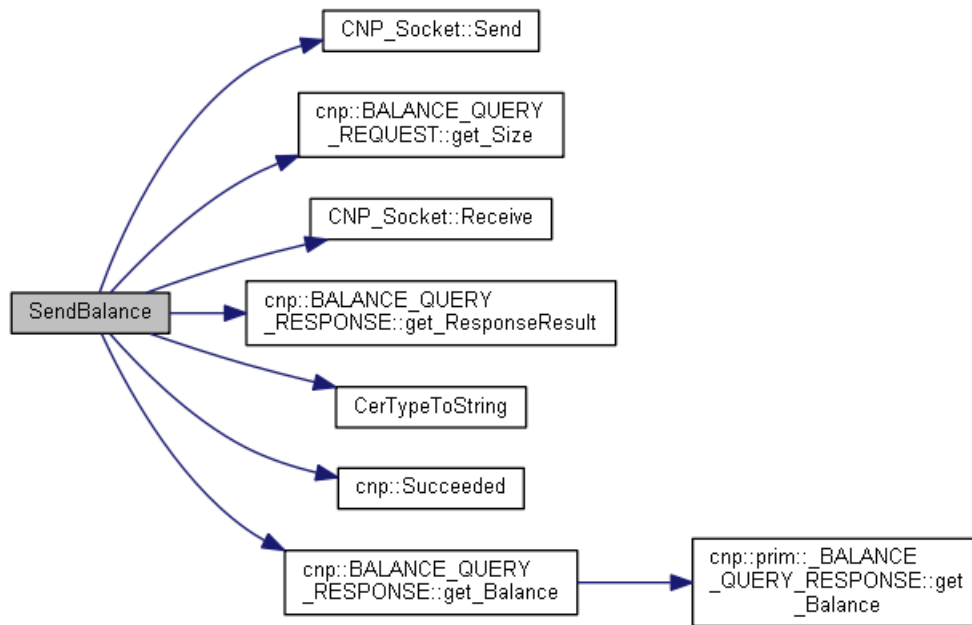
`cnp::CER_TYPE` `SendWithdrawal` (`CNP_Socket` & `socket`, `cnp::WORD` `wClientID`)

Here is the call graph for this function:



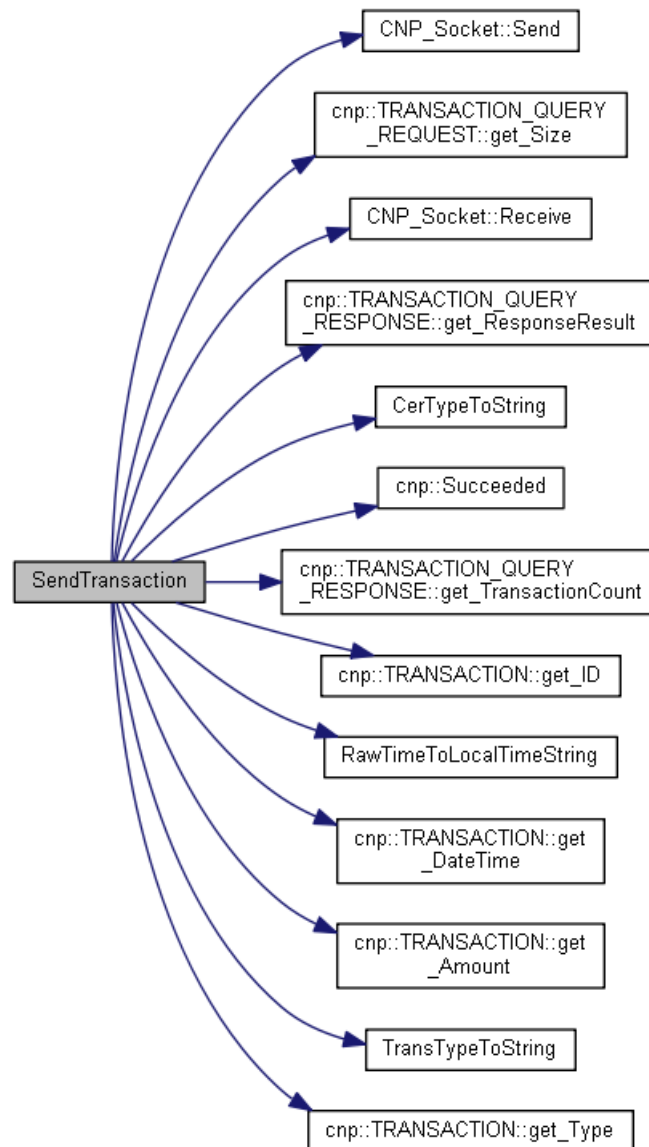
`cnp::CER_TYPE` `SendBalance` (`CNP_Socket` & `socket`, `cnp::WORD` `wClientID`)

Here is the call graph for this function:



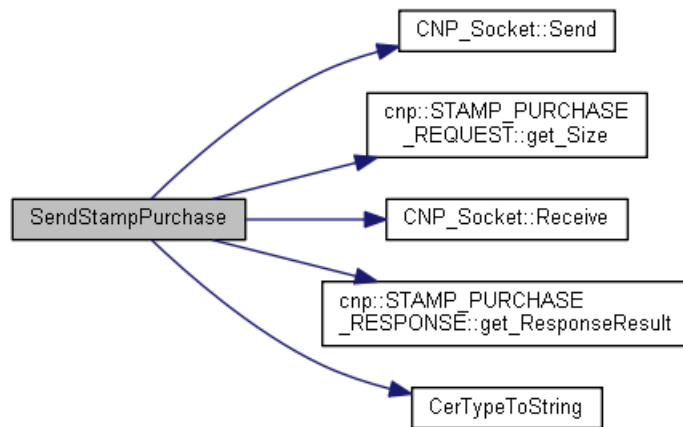
cnp::CER_TYPE SendTransaction (CNP_Socket & socket, cnp::WORD *wClientID*)

Here is the call graph for this function:



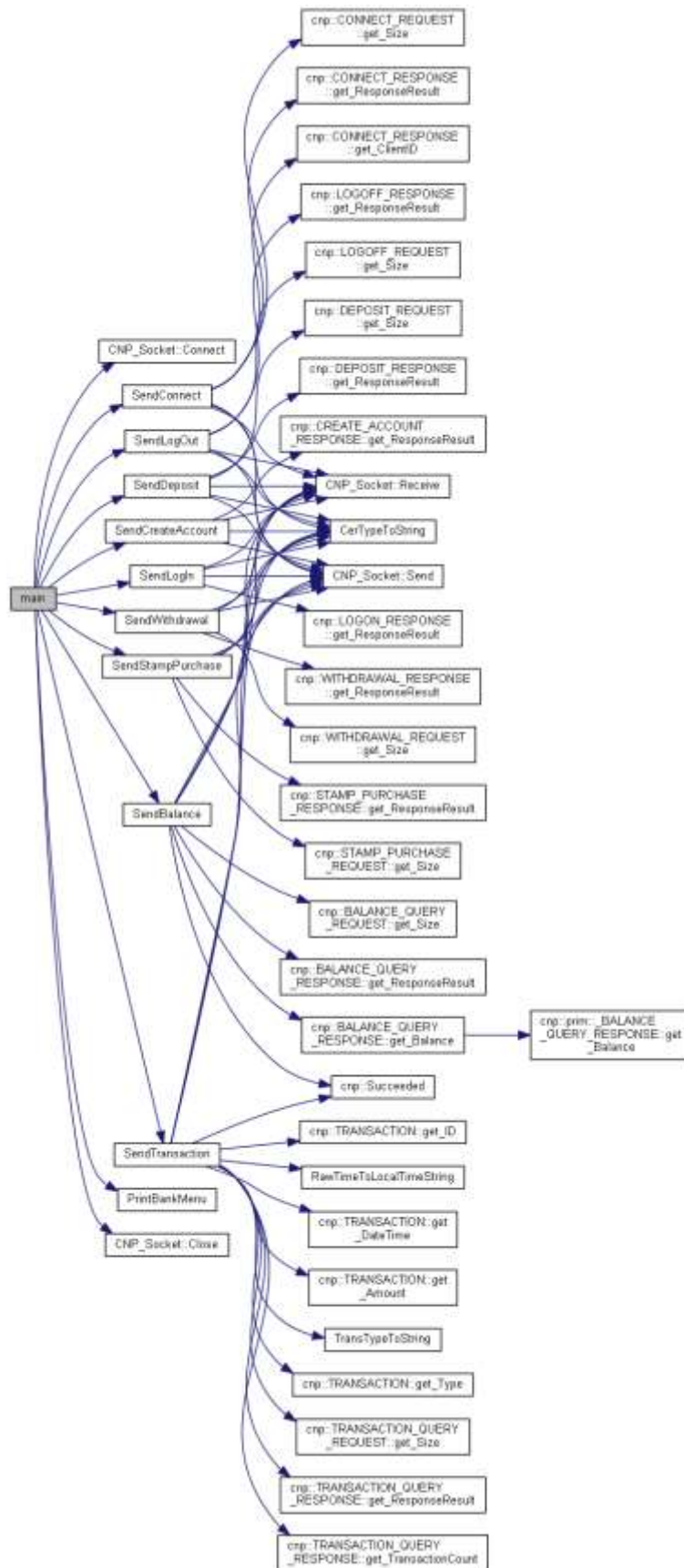
[cnp::CER_TYPE](#) SendStampPurchase ([CNP_Socket](#) & *socket*, [cnp::WORD](#) *wClientID*)

Here is the call graph for this function:



```
int main (int  argc, char *  argv[])
```

Here is the call graph for this function:



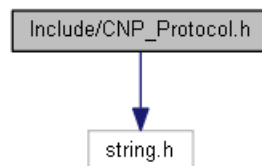
Variable Documentation

```
char g_szBuffer[512] = { 0 }
```

Client/CNP_Client.h File Reference

Include/CNP_Protocol.h File Reference

Contains type definitions required to support 5580 Computer Networks Project (CNP) Protocol.
Include dependency graph for CNP_Protocol.h:



Classes

- struct [cnp::TRANSACTION](#)
- *A Customer Transaction Record.* struct [cnp::prim:: CONNECT_REQUEST](#)
- *Connect Request Primitive.* struct [cnp::prim:: CONNECT_RESPONSE](#)
- *Connection Response Primitive.* struct [cnp::prim:: CREATE_ACCOUNT_REQUEST](#)
- *Create Account Request Primitive.* struct [cnp::prim:: CREATE_ACCOUNT_RESPONSE](#)
- *Create Account Response Primitive.* struct [cnp::prim:: LOGON_REQUEST](#)
- *Logon Request Primitive.* struct [cnp::prim:: LOGON_RESPONSE](#)
- *Logon Response Primitive.* struct [cnp::prim:: LOGOFF_REQUEST](#)
- *Logoff Request Primitive.* struct [cnp::prim:: LOGOFF_RESPONSE](#)
- *Logoff Response Primitive.* struct [cnp::prim:: DEPOSIT_REQUEST](#)
- *Deposit Request Primitive.* struct [cnp::prim:: DEPOSIT_RESPONSE](#)
- *Deposit Response Primitive.* struct [cnp::prim:: WITHDRAWAL_REQUEST](#)
- *Withdrawal Request Primitive.* struct [cnp::prim:: WITHDRAWAL_RESPONSE](#)
- *Withdrawal Response Primitive.* struct [cnp::prim:: STAMP_PURCHASE_REQUEST](#)
- *Purchase Stamp Request Primitive.* struct [cnp::prim:: STAMP_PURCHASE_RESPONSE](#)
- *Stamp Purchase Response Primitive.* struct [cnp::prim:: TRANSACTION_QUERY_REQUEST](#)
- *Transaction Query Request Primitive.* struct [cnp::prim:: TRANSACTION_QUERY_RESPONSE](#)
- *Transaction Query Result Primitive.* struct [cnp::prim:: BALANCE_QUERY_REQUEST](#)
- *Balance Query Request Primitive.* struct [cnp::prim:: BALANCE_QUERY_RESPONSE](#)
- *Balance Query Response Primitive.* struct [cnp::STD_HDR](#)
- *CNP Standard Message Header.* struct [cnp::CONNECT_REQUEST](#)
- *[Client] Connect Request message* struct [cnp::CONNECT_RESPONSE](#)

- [Server] Connect Response message struct [cnp::CREATE_ACCOUNT_REQUEST](#)
- [Client] Create Account Request message struct [cnp::CREATE_ACCOUNT_RESPONSE](#)
- [Server] Create Account Response message struct [cnp::LOGON_REQUEST](#)
- [Client] Logon Request message struct [cnp::LOGON_RESPONSE](#)
- [Server] Logon Response message struct [cnp::LOGOFF_REQUEST](#)
- [Client] Logoff Request message struct [cnp::LOGOFF_RESPONSE](#)
- [Server] Logoff Response message struct [cnp::DEPOSIT_REQUEST](#)
- [Client] Deposit Request message struct [cnp::DEPOSIT_RESPONSE](#)
- [Server] Deposit Response message struct [cnp::WITHDRAWAL_REQUEST](#)
- [Client] Withdrawal Request message struct [cnp::WITHDRAWAL_RESPONSE](#)
- [Server] Withdrawal Response message struct [cnp::BALANCE_QUERY_REQUEST](#)
- [Client] Balance Query Request message struct [cnp::BALANCE_QUERY_RESPONSE](#)
- [Server] Balance Query Response message struct [cnp::TRANSACTION_QUERY_REQUEST](#)
- [Client] Transaction Query Request message struct [cnp::TRANSACTION_QUERY_RESPONSE](#)
- [Server] Transaction Query Response message struct [cnp::STAMP_PURCHASE_REQUEST](#)
- [Client] Stamp Purchase Request Message struct [cnp::STAMP_PURCHASE_RESPONSE](#)

[Server] Stamp Purchase Response message Namespaces

- [cnp](#)
- [cnp::prim](#)

Macros

- #define [COUNTOF](#)(_array) (sizeof(_array) / sizeof(_array[0]))
Helper macro that calculates count of elements in an array.
- #define [MAKE_MSG_TYPE](#)(type, sub) ((sub << 16) + type)
- #define [MAKE_ERROR_RESULT](#)(facility, sub) ((facility << 16) + sub)

Typedefs

- typedef unsigned short [cnp::WORD](#)
16bit type definition
- typedef unsigned long [cnp::DWORD](#)
32bit type definition
- typedef unsigned long long [cnp::QWORD](#)
64bit type definition

Enumerations

- enum [cnp::CNP_MSG_TYPE](#) { [cnp::CMT_INVALID](#) = 0x00, [cnp::CMT_CONNECT](#) = 0x50, [cnp::CMT_CREATE_ACCOUNT](#) = 0x51, [cnp::CMT_LOGON](#) = 0x52, [cnp::CMT_LOGOFF](#) = 0x53, [cnp::CMT_DEPOSIT](#) = 0x54, [cnp::CMT_WITHDRAWAL](#) = 0x55, [cnp::CMT_BALANCE_QUERY](#) = 0x56, [cnp::CMT_TRANSACTION_QUERY](#) = 0x57, [cnp::CMT_PURCHASE_STAMPS](#) = 0x58 }
- *Supported CNP Message Types (CMT_)* enum [cnp::CNP_MSG_SUBTYPE](#) { [cnp::CMS_INVALID](#) = 0x00, [cnp::CMS_REQUEST](#) = 0x01, [cnp::CMS_RESPONSE](#) = 0x02 }
- *Supported CNP Message Subtypes (CMS_)* enum [cnp::MSG_TYPE](#) { [cnp::MT_INVALID](#) = [MAKE_MSG_TYPE](#)(CMT_INVALID, CMS_INVALID), [cnp::MT_CONNECT_REQUEST](#) = [MAKE_MSG_TYPE](#)(CMT_CONNECT, CMS_REQUEST), [cnp::MT_CONNECT_RESPONSE](#) = [MAKE_MSG_TYPE](#)(CMT_CONNECT, CMS_RESPONSE), [cnp::MT_CREATE_ACCOUNT_REQUEST](#) = [MAKE_MSG_TYPE](#)(CMT_CREATE_ACCOUNT, CMS_REQUEST), [cnp::MT_CREATE_ACCOUNT_RESPONSE](#) = [MAKE_MSG_TYPE](#)(CMT_CREATE_ACCOUNT, CMS_RESPONSE), [cnp::MT_LOGON_REQUEST](#) = [MAKE_MSG_TYPE](#)(CMT_LOGON, CMS_REQUEST), [cnp::MT_LOGON_RESPONSE](#) = [MAKE_MSG_TYPE](#)(CMT_LOGON, CMS_RESPONSE), [cnp::MT_LOGOFF_REQUEST](#) = [MAKE_MSG_TYPE](#)(CMT_LOGOFF,

- CMS_REQUEST), [cnp::MT_LOGOFF_RESPONSE](#) = MAKE_MSG_TYPE(CMT_LOGOFF, CMS_RESPONSE), [cnp::MT_DEPOSIT_REQUEST](#) = MAKE_MSG_TYPE(CMT_DEPOSIT, CMS_REQUEST), [cnp::MT_DEPOSIT_RESPONSE](#) = MAKE_MSG_TYPE(CMT_DEPOSIT, CMS_RESPONSE), [cnp::MT_WITHDRAWAL_REQUEST](#) = MAKE_MSG_TYPE(CMT_WITHDRAWAL, CMS_REQUEST), [cnp::MT_WITHDRAWAL_RESPONSE](#) = MAKE_MSG_TYPE(CMT_WITHDRAWAL, CMS_RESPONSE), [cnp::MT_BALANCE_QUERY_REQUEST](#) = MAKE_MSG_TYPE(CMT_BALANCE_QUERY, CMS_REQUEST), [cnp::MT_BALANCE_QUERY_RESPONSE](#) = MAKE_MSG_TYPE(CMT_BALANCE_QUERY, CMS_RESPONSE), [cnp::MT_TRANSACTION_QUERY_REQUEST](#) = MAKE_MSG_TYPE(CMT_TRANSACTION_QUERY, CMS_REQUEST), [cnp::MT_TRANSACTION_QUERY_RESPONSE](#) = MAKE_MSG_TYPE(CMT_TRANSACTION_QUERY, CMS_RESPONSE), [cnp::MT_PURCHASE_STAMPS_REQUEST](#) = MAKE_MSG_TYPE(CMT_PURCHASE_STAMPS, CMS_REQUEST), [cnp::MT_PURCHASE_STAMPS_RESPONSE](#) = MAKE_MSG_TYPE(CMT_PURCHASE_STAMPS, CMS_RESPONSE) }
- *Constructed Message Type IDs.* enum [cnp::CFC_TYPE](#) { [cnp::CFC_CONNECT](#), [cnp::CFC_CREDENTIALS](#), [cnp::CFC_FUNCTIONAL](#), [cnp::CFC_ACCOUNT](#), [cnp::CFC_UNDEFINED](#) }
 - *Message Facility Code Types (CFC)* enum [cnp::CER_TYPE](#) { [cnp::CER_SUCCESS](#) = 0, [cnp::CER_AUTHENTICATION_FAILED](#) = MAKE_ERROR_RESULT(CFC_CONNECT, 0x01), [cnp::CER_UNSUPPORTED_PROTOCOL](#) = MAKE_ERROR_RESULT(CFC_CONNECT, 0x02), [cnp::CER_INVALID_CLIENT_ID](#) = MAKE_ERROR_RESULT(CFC_CREDENTIALS, 0x01), [cnp::CER_INVALID_NAME_PIN](#) = MAKE_ERROR_RESULT(CFC_CREDENTIALS, 0x02), [cnp::CER_INVALID_ARGUMENTS](#) = MAKE_ERROR_RESULT(CFC_FUNCTIONAL, 0x01), [cnp::CER_CLIENT_NOT_LOGGEDON](#) = MAKE_ERROR_RESULT(CFC_FUNCTIONAL, 0x02), [cnp::CER_DRAWER_BLOCKED](#) = MAKE_ERROR_RESULT(CFC_FUNCTIONAL, 0x03), [cnp::CER_INSUFFICIENT_FUNDS](#) = MAKE_ERROR_RESULT(CFC_ACCOUNT, 0x01), [cnp::CER_ACCOUNT_NOT_FOUND](#) = MAKE_ERROR_RESULT(CFC_ACCOUNT, 0x02), [cnp::CER_ACCOUNT_EXISTS](#) = MAKE_ERROR_RESULT(CFC_ACCOUNT, 0x03), [cnp::CER_ERROR](#) = (~0) }
 - *CNP Error Result Types (CER)* enum [cnp::DEPOSIT_TYPE](#) { [cnp::DT_INVALID](#) = 0, [cnp::DT_CASH](#) = 0x01, [cnp::DT_CHECK](#) = 0x02 }
 - *CNP Deposit types (DT)* enum [cnp::TRANSACTION_TYPE](#) { [cnp::TT_INVALID](#) = 0, [cnp::TT_DEPOSIT](#) = 0x01, [cnp::TT_WITHDRAWAL](#) = 0x02, [cnp::TT_STAMP_PURCHASE](#) = 0x03 }

CNP Transaction types (TT) Functions

- bool [cnp::Succeeded](#) ([cnp::CER_TYPE](#) cerRR) throw ()

Variables

- const [WORD](#) [cnp::g_wMajorVersion](#) = 1
Global message sequence number.
- const [WORD](#) [cnp::g_wMinorVersion](#) = 1
Protocol minor version (i.e. x.3)
- const [DWORD](#) [cnp::g_dwValidationKey](#) = 0x00DEAD01
CNP Validation Key.
- const size_t [cnp::MAX_NAME_LEN](#) = 32
[first,last,email] name field lengths
- const [WORD](#) [cnp::INVALID_CLIENT_ID](#) = (~0)
Used for error checking & default initialization.
- const [WORD](#) [cnp::INVALID_PIN](#) = 0
Used for error checking & default initialization.

Macro Definition Documentation

```
#define COUNTOF( _array) (sizeof(_array) / sizeof(_array[0]))
```

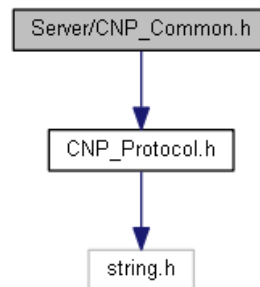
```
#define MAKE_MSG_TYPE( type, sub) ((sub << 16) + type)
```

```
#define MAKE_ERROR_RESULT( facility, sub) ((facility << 16) + sub)
```

Server/CNP_Common.h File Reference

Common type definitions.

Include dependency graph for CNP_Common.h:



Functions

- bool [IsValidCustomerID](#) (const [cnp::QWORD](#) &qwID) throw ()
Validation helper function.
- bool [IsValidPIN](#) ([cnp::WORD](#) wPIN) throw ()
Validation helper function.
- bool [IsValidName](#) (const char *szName) throw ()
Validation helper function.

Variables

- const [cnp::QWORD INVALID_CUSTOMER_ID](#) = 0
for error checking and data initialization
- const [cnp::DWORD INVALID_BALANCE](#) = (~0)
for error checking and data initialization
- const [cnp::WORD g_wServerMajorVersion](#) = 1
- const [cnp::WORD g_wServerMinorVersion](#) = 1

Detailed Description

Author:

Mark L. Short

Date:

April 10, 2015

April 25, 2015 updated code comments

Function Documentation

bool IsValidCustomerID (const [cnp::QWORD](#) & *qwID*) throw ()

bool IsValidPIN ([cnp::WORD](#) *wPIN*) throw ()

bool IsValidName (const char * *szName*) throw ()

Variable Documentation

const [cnp::QWORD](#) INVALID_CUSTOMER_ID = 0

const [cnp::DWORD](#) INVALID_BALANCE = (~0)

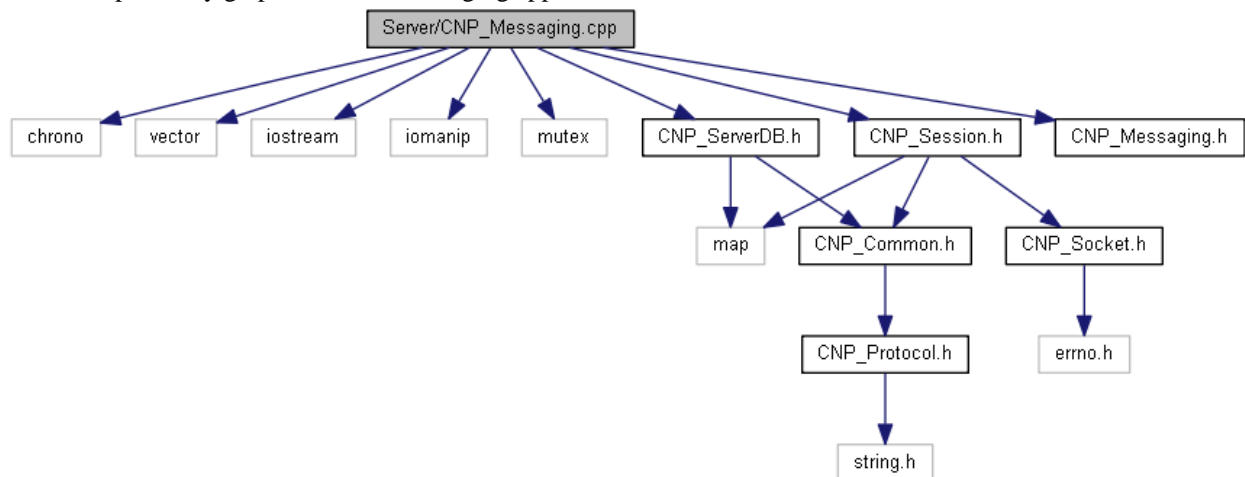
const [cnp::WORD](#) g_wServerMajorVersion = 1

const [cnp::WORD](#) g_wServerMinorVersion = 1

Server/CNP_Messaging.cpp File Reference

Server Message processing implementation.

Include dependency graph for CNP_Messaging.cpp:



Functions

- [cnp::WORD ProcessConnectRequest](#) (const void *pMsg, size_t cbMsgLen, [CNP_Socket](#) *pSocket)
- bool [ProcessCreateAccountRequest](#) (const void *pMsg, size_t cbMsgLen)
- bool [ProcessLogonRequest](#) (const void *pMsg, size_t cbMsgLen)
- bool [ProcessLogoffRequest](#) (const void *pMsg, size_t cbMsgLen)
- bool [ProcessDepositRequest](#) (const void *pMsg, size_t cbMsgLen)
- bool [ProcessWithdrawalRequest](#) (const void *pMsg, size_t cbMsgLen)
- bool [ProcessBalanceQueryRequest](#) (const void *pMsg, size_t cbMsgLen)
- bool [ProcessTransactionQueryRequest](#) (const void *pMsg, size_t cbMsgLen)
- bool [ProcessStampPurchaseRequest](#) (const void *pMsg, size_t cbMsgLen)
- bool [ProcessDisconnect](#) ([cnp::WORD](#) wClientID)

Variables

- [SessionMap_t g_SessionInfo](#)
Global SessionMap_t instance.
- std::mutex [g_SessionMutex](#)
- [AccountMap_t g_AccountInfo](#)
- std::mutex [g_AccountMutex](#)
- [TransactionMap_t g_TransactionInfo](#)
- std::mutex [g_TransactionMutex](#)

Detailed Description

Author:

Mark L. Short

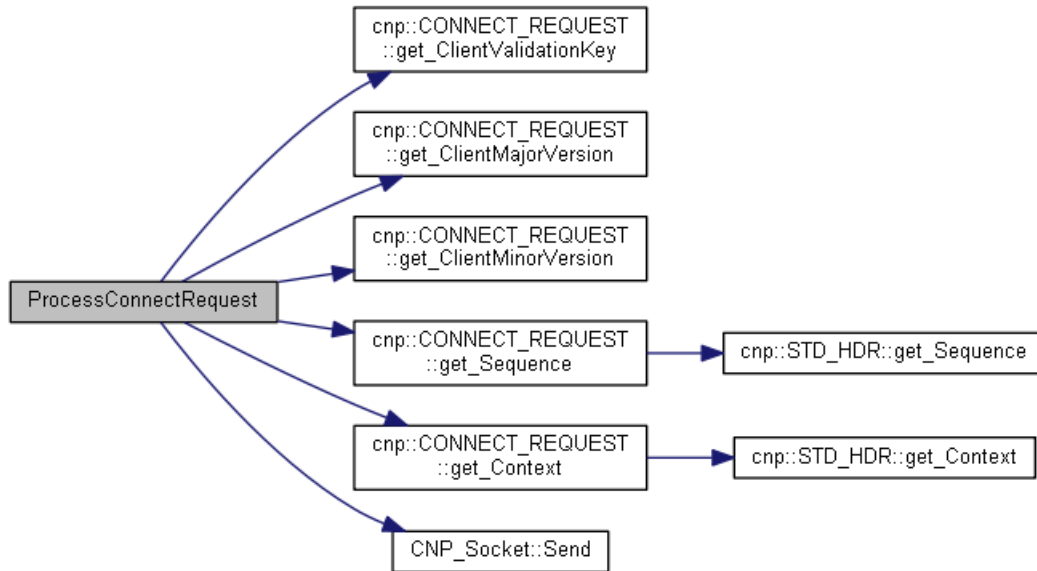
Date:

April 10, 2015

Function Documentation

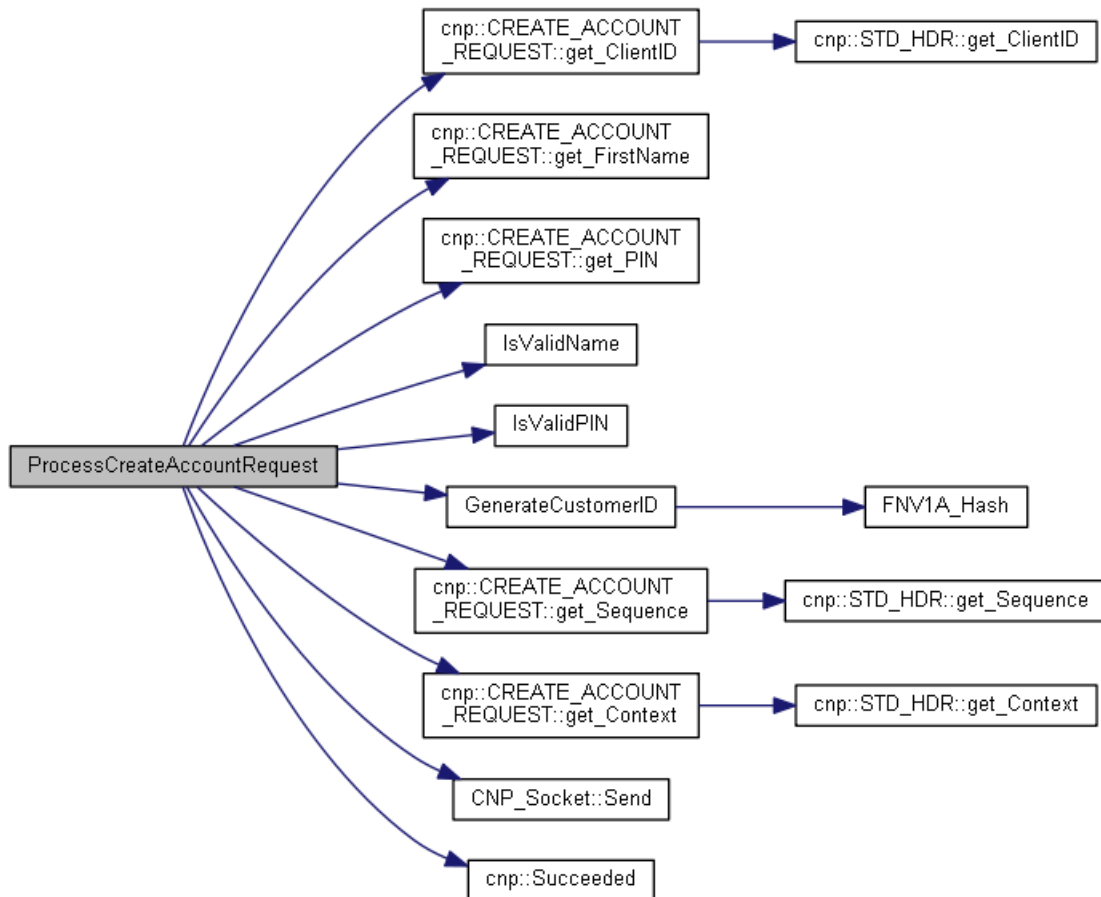
[cnp::WORD](#) [ProcessConnectRequest](#) (const void * *pMsg*, size_t *cbMsgLen*, [CNP_Socket](#) * *pSocket*)

Here is the call graph for this function:



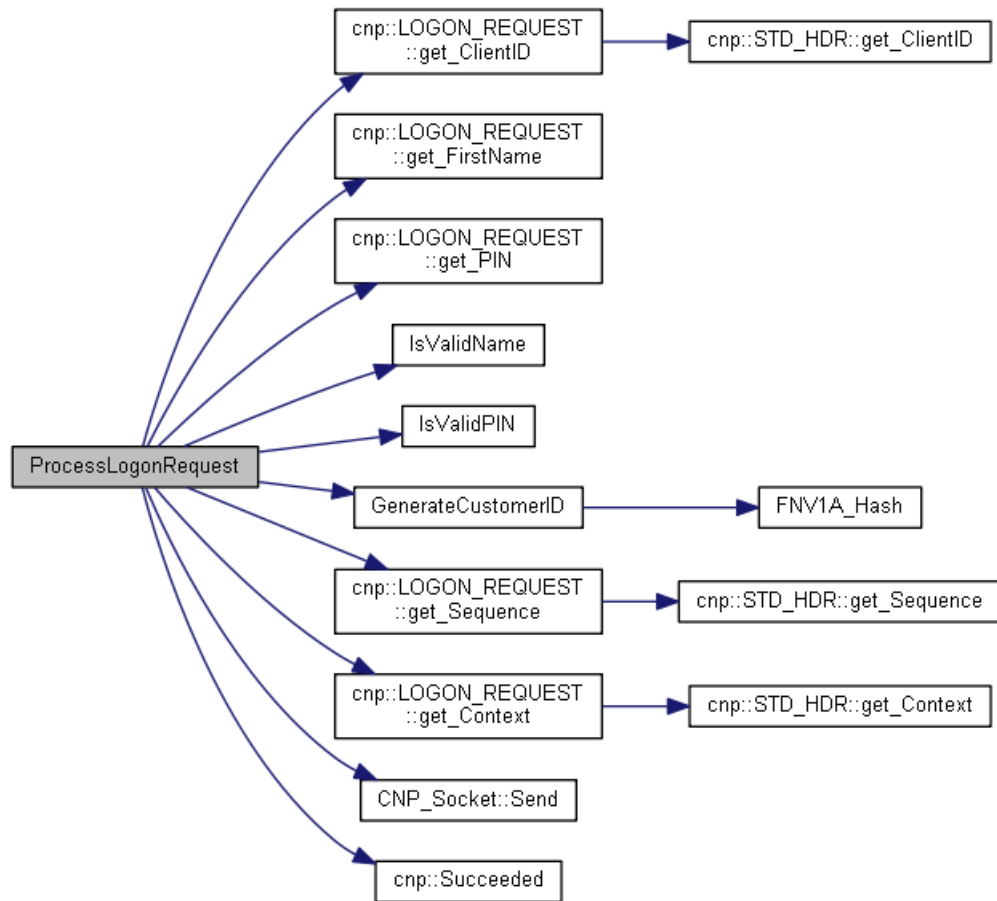
bool ProcessCreateAccountRequest (const void * *pMsg*, size_t *cbMsgLen*)

Here is the call graph for this function:



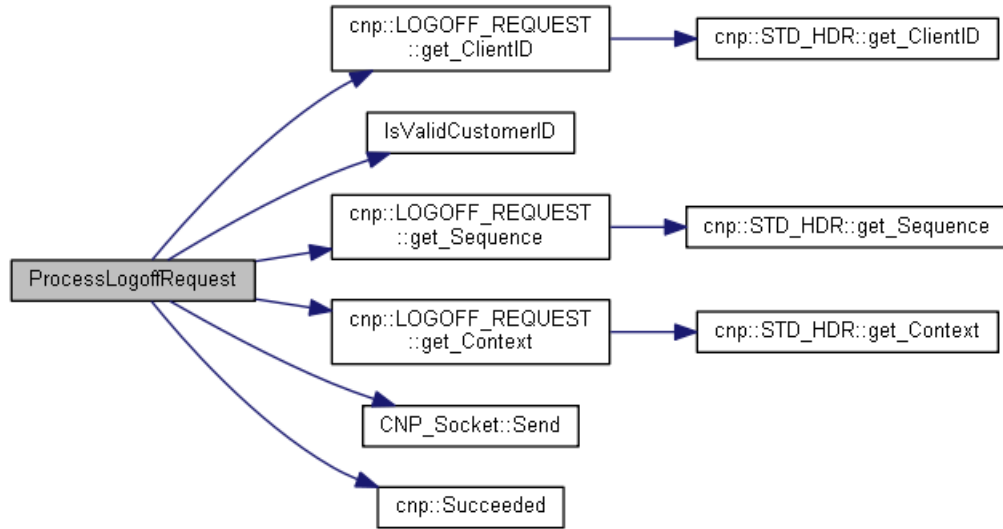
bool ProcessLogonRequest (const void * *pMsg*, size_t *cbMsgLen*)

Here is the call graph for this function:



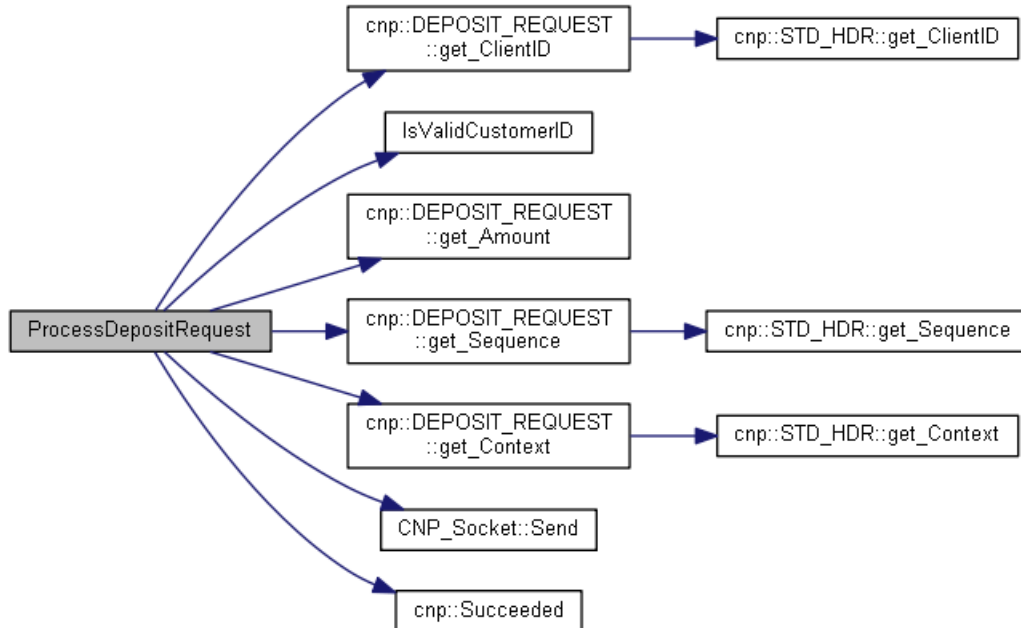
bool ProcessLogoffRequest (const void * *pMsg*, size_t *cbMsgLen*)

Here is the call graph for this function:



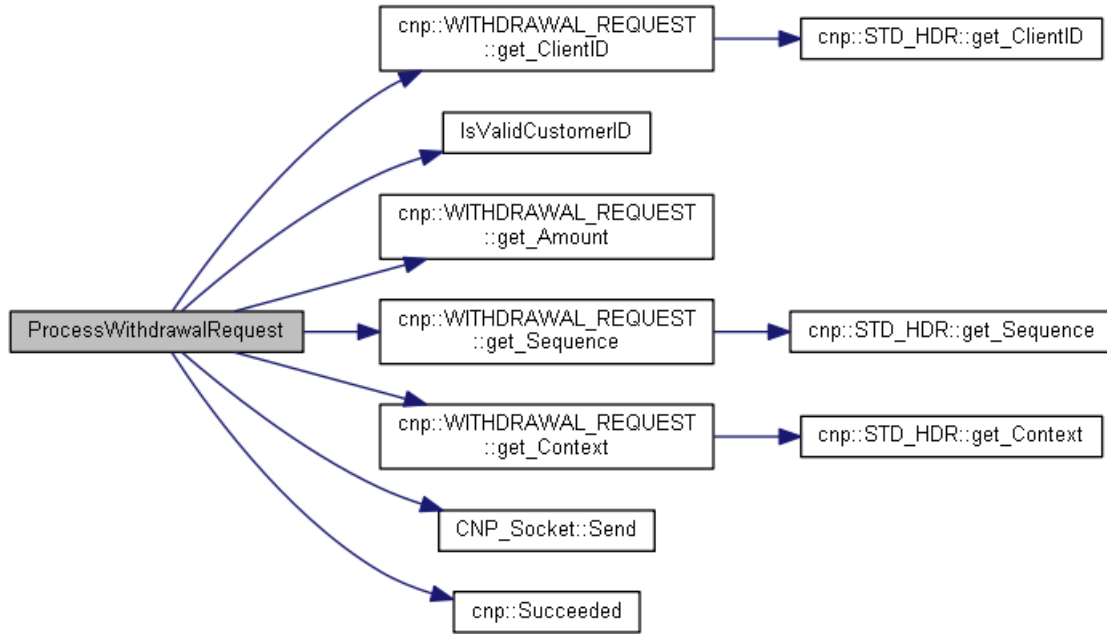
bool ProcessDepositRequest (const void * *pMsg*, size_t *cbMsgLen*)

Here is the call graph for this function:



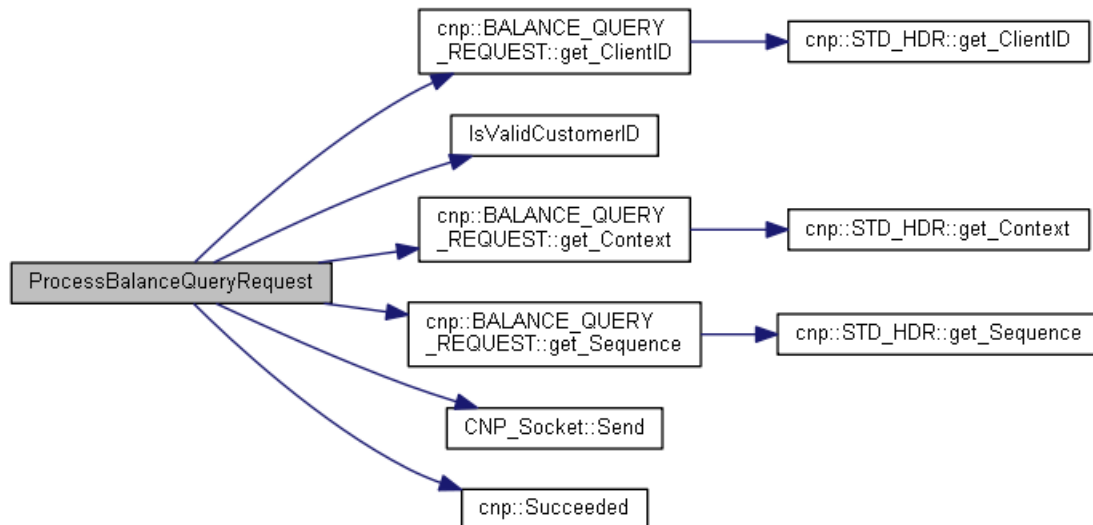
bool ProcessWithdrawalRequest (const void * *pMsg*, size_t *cbMsgLen*)

Here is the call graph for this function:



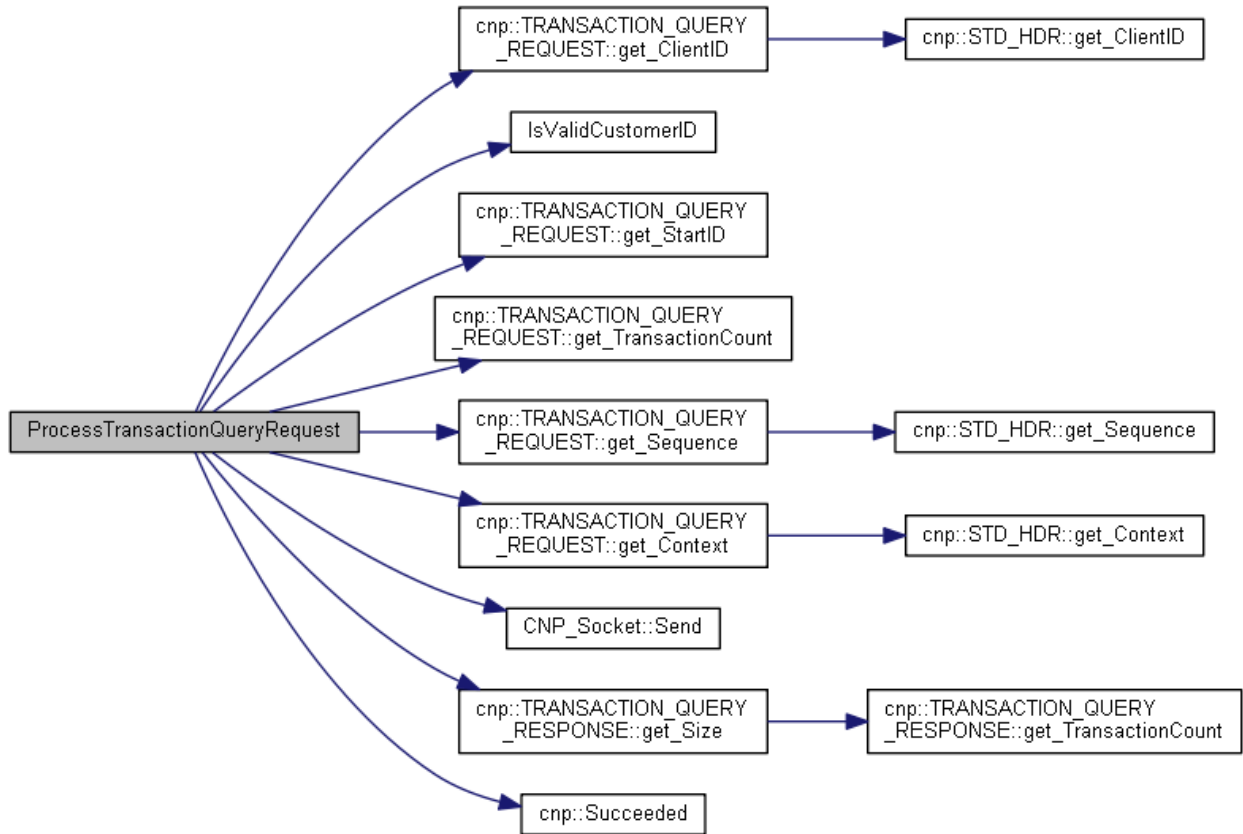
bool ProcessBalanceQueryRequest (const void * *pMsg*, size_t *cbMsgLen*)

Here is the call graph for this function:



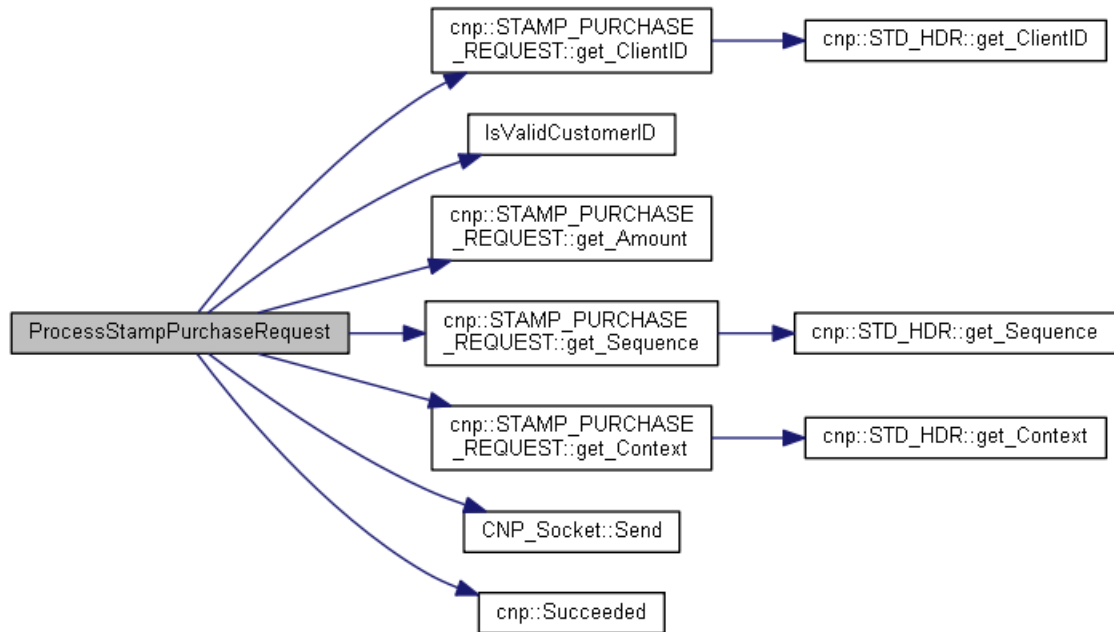
bool ProcessTransactionQueryRequest (const void * *pMsg*, size_t *cbMsgLen*)

Here is the call graph for this function:



bool ProcessStampPurchaseRequest (const void * *pMsg*, size_t *cbMsgLen*)

Here is the call graph for this function:



bool ProcessDisconnect ([cnp::WORD](#) wClientID)

Variable Documentation

[SessionMap_t](#) g_SessionInfo

std::mutex g_SessionMutex

[AccountMap_t](#) g_AccountInfo

std::mutex g_AccountMutex

[TransactionMap_t](#) g_TransactionInfo

std::mutex g_TransactionMutex

Server/CNP_Messaging.h File Reference

Message processing function prototypes.

Functions

- [cnp::WORD ProcessConnectRequest](#) (const void *pMsg, size_t cbLen, [CNP Socket](#) *pSocket)
 - bool [ProcessBalanceQueryRequest](#) (const void *pMsg, size_t cbLen)
 - bool [ProcessCreateAccountRequest](#) (const void *pMsg, size_t cbLen)
 - bool [ProcessDepositRequest](#) (const void *pMsg, size_t cbLen)
 - bool [ProcessLogoffRequest](#) (const void *pMsg, size_t cbLen)
 - bool [ProcessLogonRequest](#) (const void *pMsg, size_t cbLen)
 - bool [ProcessStampPurchaseRequest](#) (const void *pMsg, size_t cbLen)
 - bool [ProcessTransactionQueryRequest](#) (const void *pMsg, size_t cbLen)
 - bool [ProcessWithdrawalRequest](#) (const void *pMsg, size_t cbLen)
 - bool [ProcessDisconnect](#) ([cnp::WORD](#) wClientID)
-

Detailed Description

Author:

Mark L. Short

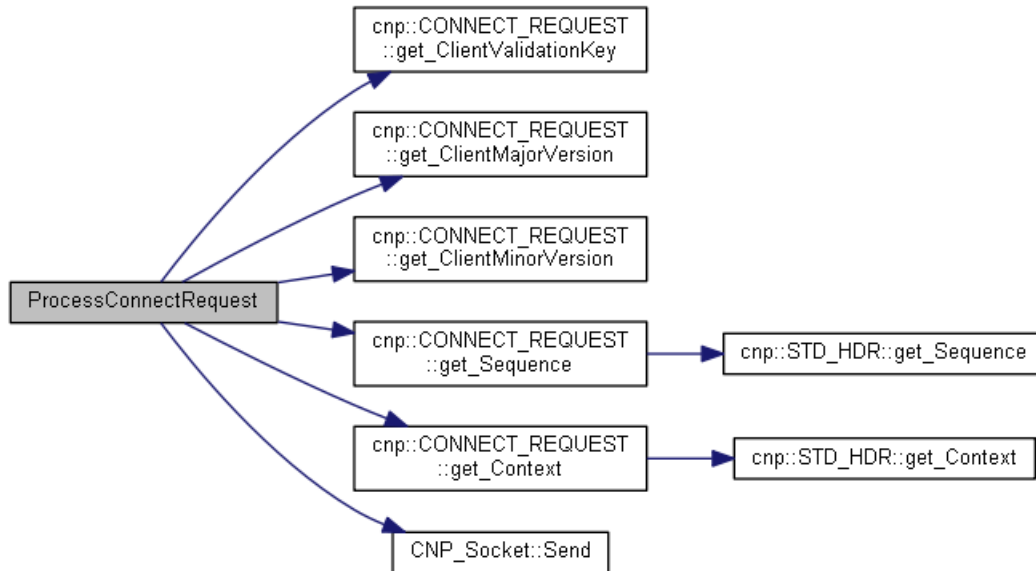
Date:

April 10, 2015

Function Documentation

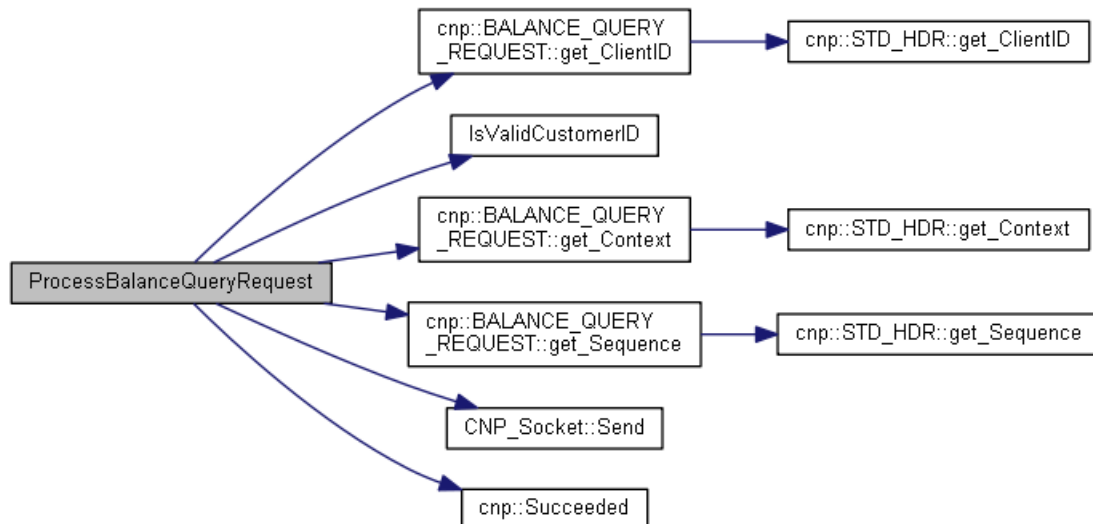
[cnp::WORD](#) ProcessConnectRequest (const void * *pMsg*, size_t *cbLen*, [CNP_Socket](#) * *pSocket*)

Here is the call graph for this function:



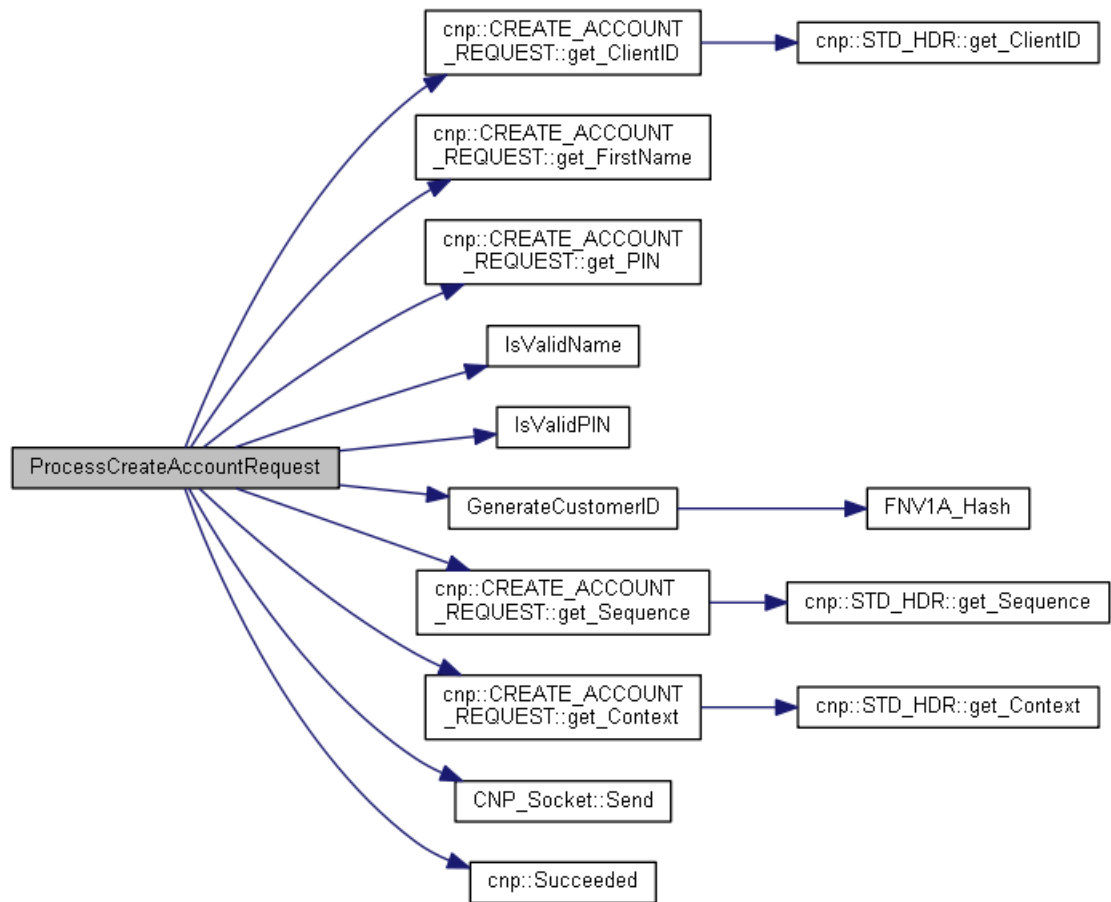
bool ProcessBalanceQueryRequest (const void * *pMsg*, size_t *cbLen*)

Here is the call graph for this function:



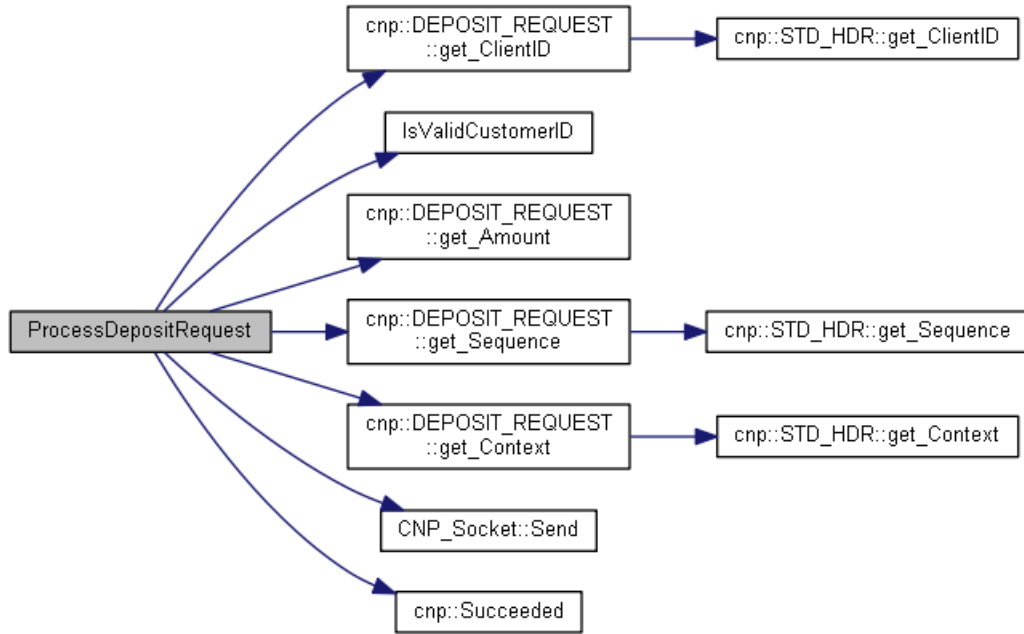
bool ProcessCreateAccountRequest (const void * *pMsg*, size_t *cbLen*)

Here is the call graph for this function:



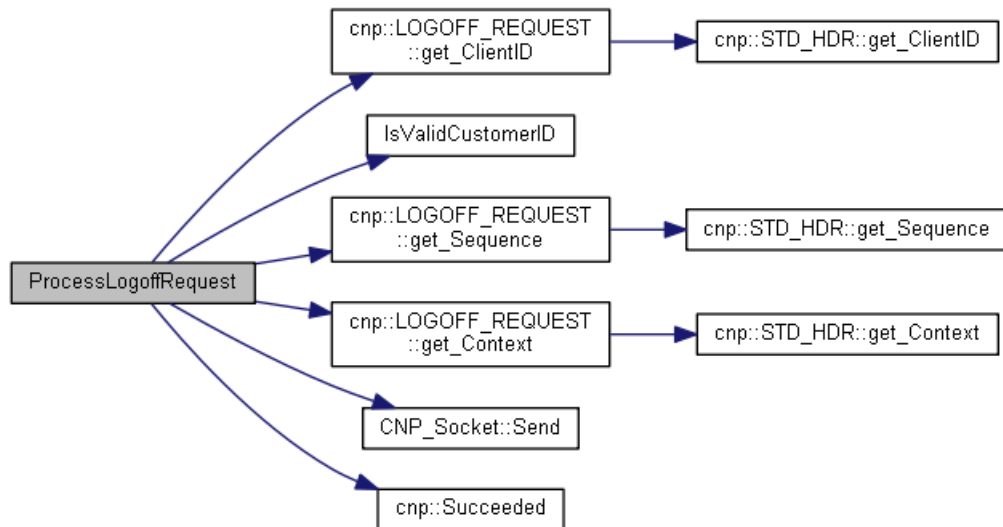
bool ProcessDepositRequest (const void * *pMsg*, size_t *cbLen*)

Here is the call graph for this function:



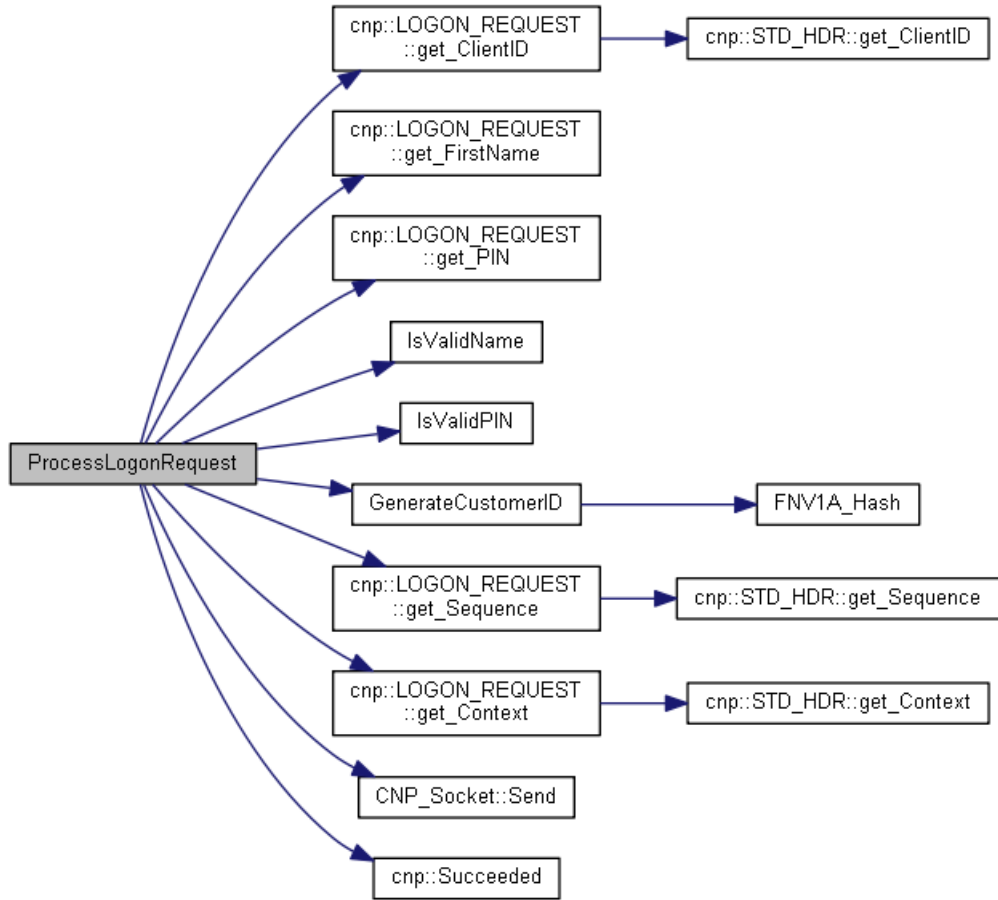
bool ProcessLogoffRequest (const void * *pMsg*, size_t *cbLen*)

Here is the call graph for this function:



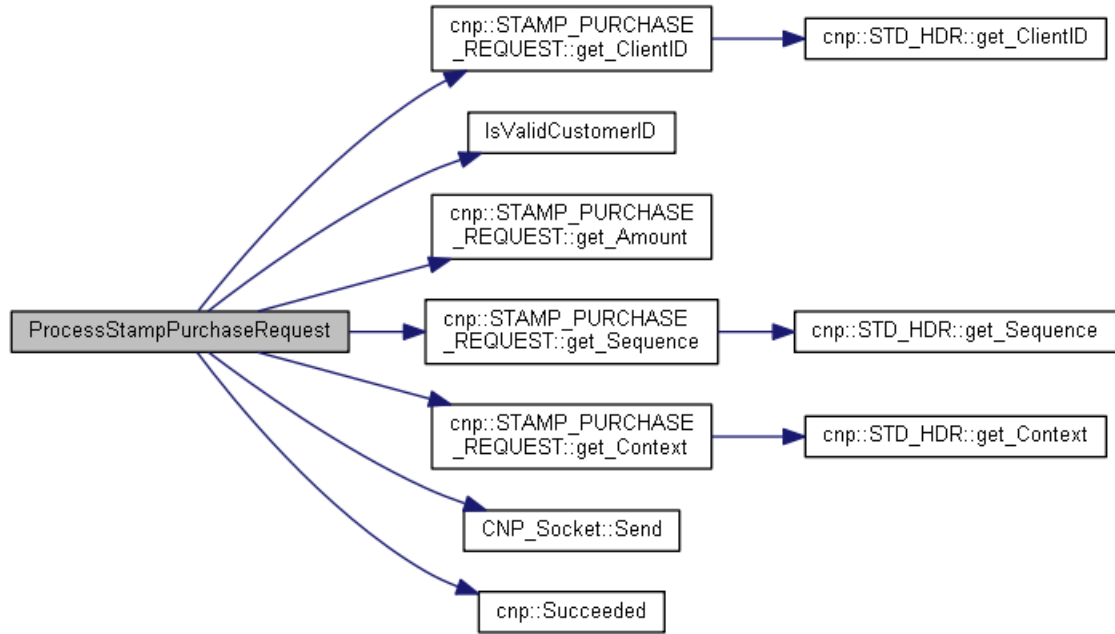
bool ProcessLogonRequest (const void * *pMsg*, size_t *cbLen*)

Here is the call graph for this function:



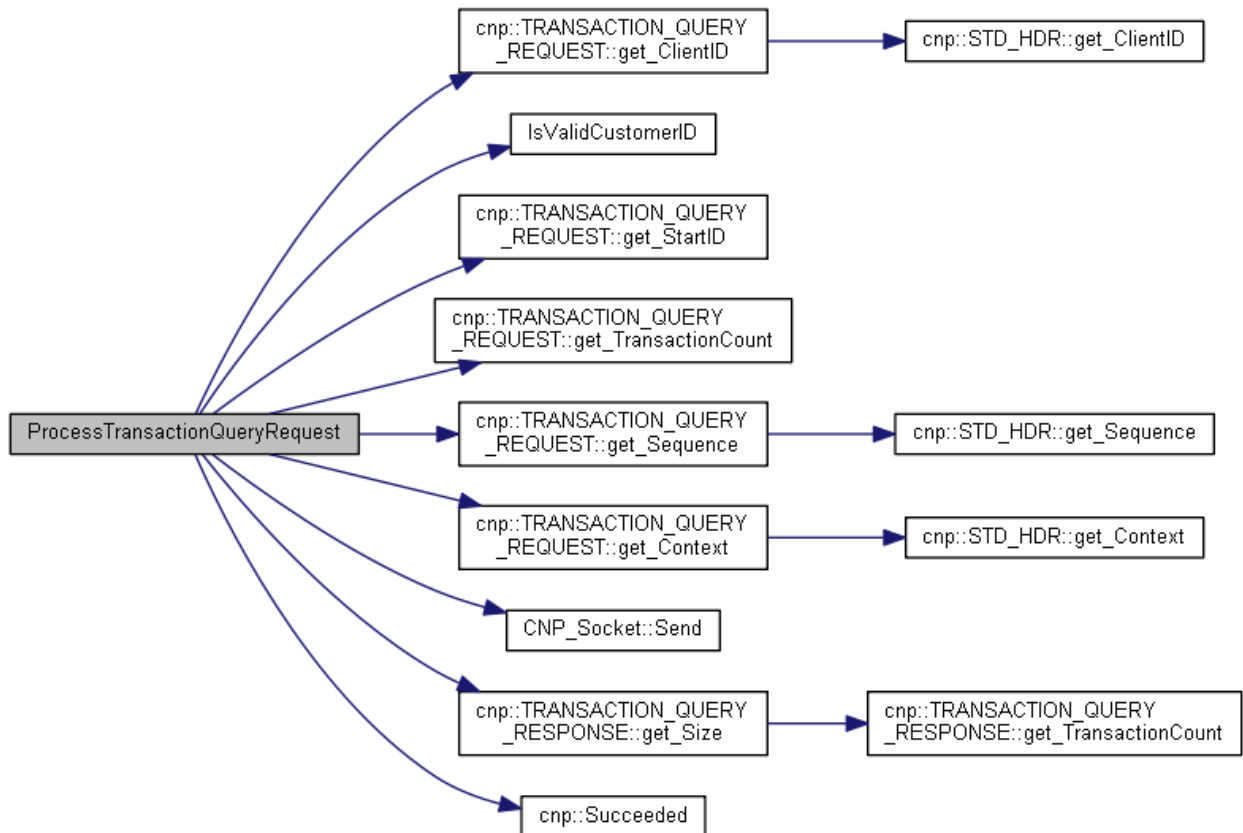
bool ProcessStampPurchaseRequest (const void * *pMsg*, size_t *cbLen*)

Here is the call graph for this function:



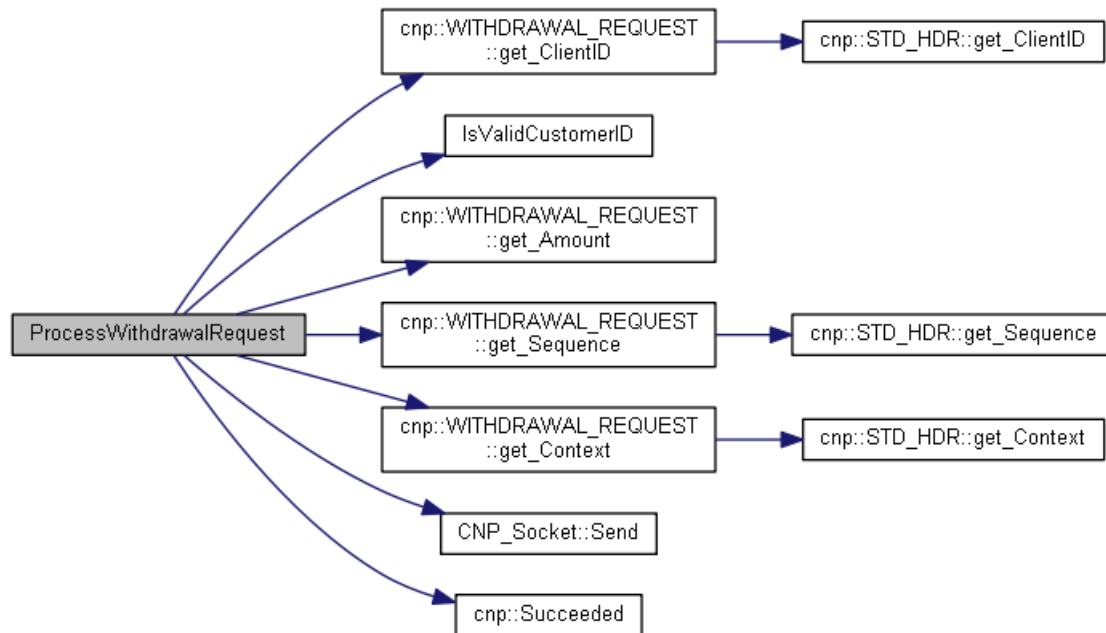
bool ProcessTransactionQueryRequest (const void * *pMsg*, size_t *cbLen*)

Here is the call graph for this function:



bool ProcessWithdrawalRequest (const void * *pMsg*, size_t *cbLen*)

Here is the call graph for this function:

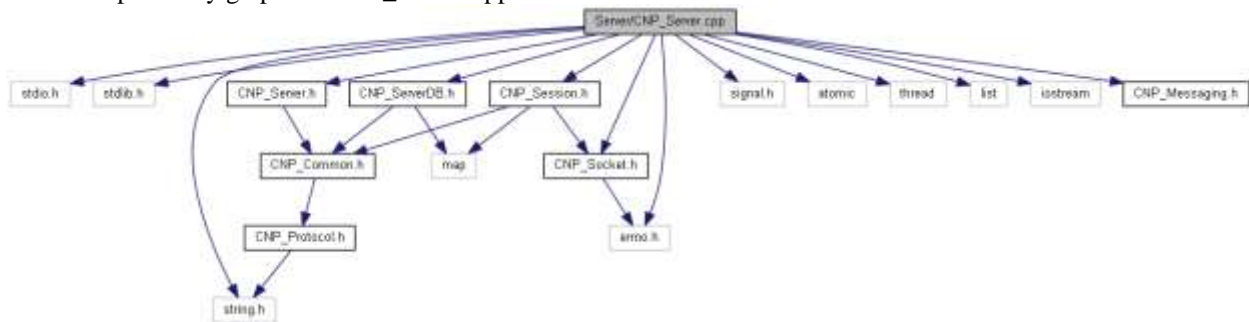


bool ProcessDisconnect ([cnp::WORD](#) *wClientID*)

Server/CNP_Server.cpp File Reference

Server Main.

Include dependency graph for `CNP_Server.cpp`:



Classes

- struct [THREAD_INFO](#)

Functions

- void [ClientThreadHandler](#) (void *pData)
 - void [TerminateHandler](#) (int iSignal)
 - int [main](#) (int argc, char *argv[])
-

Detailed Description

Author:

Mark L. Short

Date:

April 10, 2015

Function Documentation

void ClientThreadHandler (void * *pData*)

Here is the call graph for this function:



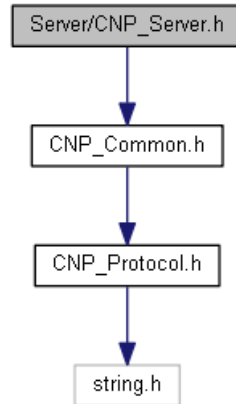
```
void TerminateHandler (int iSignal)
```

```
int main (int argc, char * argv[])
```

Here is the call graph for this function:

Server/CNP_Server.h File Reference

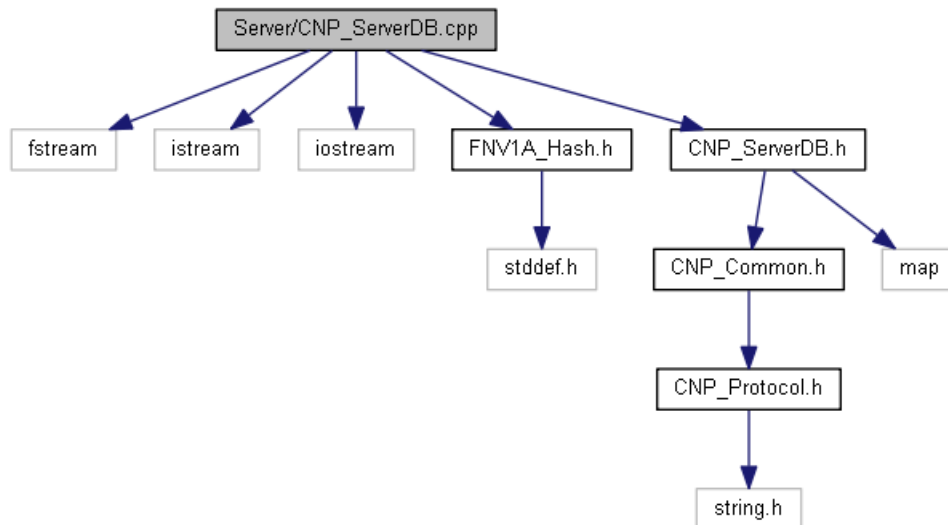
Include dependency graph for CNP_Server.h:



Server/CNP_ServerDB.cpp File Reference

Server DB persistence implementation.

Include dependency graph for CNP_ServerDB.cpp:



Functions

- [cnp::QWORD GenerateCustomerID](#) (const char *szFirstName, size_t cbLen, [cnp::WORD](#) wPIN)
- [template<class _MapType > size_t LoadServerDB](#) (const char *szFileName, _MapType &Container)

Generic template function for loading STL maps.

- template<class _MapType > size_t [SaveServerDB](#) (const char *szFileName, const _MapType &Container)
Generic template function for persisting STL maps.
- size_t [LoadServerDB](#) (void)
- size_t [SaveServerDB](#) (void)

Variables

- const char [g_szAccountDBFileName](#) [] = "../Data//AccountDB.Dat"
File name of server [ACCOUNT INFO](#) table store.
- const char [g_szTransactDBFileName](#) [] = "../Data//TransactDB.Dat"
File name of server [TRANSACTION INFO](#) table store.
- [AccountMap_t g_AccountInfo](#)
- [TransactionMap_t g_TransactionInfo](#)

Detailed Description

Author:

Mark L. Short

Date:

April 10, 2015

April 25, 2015 updated code comments

Function Documentation

[cnp::QWORD](#) [GenerateCustomerID](#) (const char * [szFirstName](#), size_t [cbLen](#), [cnp::WORD](#) [wPIN](#))

Generate a unique customer ID from a given name + PIN combination

Parameters:

in	szFirstName	address of NULL terminated string containing customer's first name
in	cbLen	count of bytes (cb) length of first name string
in	wPIN	customer's PIN number

Return values:

cnp::QWORD	containing unique 64bit customer ID
----------------------------	-------------------------------------

Here is the call graph for this function:



template<class _MapType > size_t [LoadServerDB](#) (const char * [szFileName](#), _MapType & [Container](#))

[LoadServerDB](#) provides a generic function to use for different map type containers.

Precondition:

- _MapType is a std::map<_KeyType, _MappedType> collection
- _MappedType implements the get_PrimaryKey() method

Parameters:

in	<i>szFileName</i>	address of the NULL terminated string that contains the name of the file to open
in	<i>Container</i>	a reference to the std::map container instance to insert loaded file records into

Return values:

<i>size_t</i>	containing the number of records actually loaded
---------------	--

template<class _MapType > size_t SaveServerDB (const char * *szFileName*, const _MapType & *Container*)

SaveServerDB provides a generic function to use for different map type containers.

Precondition:

_MapType is a std::map<_KeyType, _MappedType> collection

Parameters:

in	<i>szFileName</i>	address of the NULL terminated string that contains the name of the file to open for saving
in	<i>Container</i>	a reference to the std::map container instance to iterate for persistence

Return values:

<i>size_t</i>	containing the number of records actually saved
---------------	---

size_t LoadServerDB (void)

Loads into runtime memory the server database records from the persisted store

Return values:

<i>size_t</i>	containing the number of records loaded
---------------	---

Here is the call graph for this function:

**size_t SaveServerDB (void)**

Saves the current server database records to persisted store

Return values:

<i>size_t</i>	contain the number of records saved to file
---------------	---

Here is the call graph for this function:



Variable Documentation

`const char g_szAccountDBFileName[] = "../Data/AccountDB.Dat"`

`const char g_szTransactDBFileName[] = "../Data/TransactDB.Dat"`

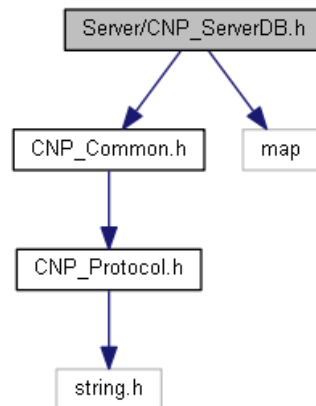
[AccountMap_t](#) g_AccountInfo

[TransactionMap_t](#) g_TransactionInfo

Server/CNP_ServerDB.h File Reference

[ACCOUNT_INFO](#) & [TRANSACTION_INFO](#) struct definitions.

Include dependency graph for CNP_ServerDB.h:



Classes

- struct [ACCOUNT_INFO](#)
- struct [TRANSACTION_INFO](#)

Typedefs

- typedef std::map< [ACCOUNT_INFO::key_type](#), [ACCOUNT_INFO](#) > [AccountMap_t](#)
- typedef std::map< [TRANSACTION_INFO::key_type](#), [TRANSACTION_INFO](#) > [TransactionMap_t](#)

Functions

- [cnpr::QWORD GenerateCustomerID](#) (const char *szFirstName, size_t cbLen, [cnpr::WORD](#) wPIN)
- size_t [LoadServerDB](#) (void)
- size_t [SaveServerDB](#) (void)

Detailed Description

Author:

Mark L. Short

Date:

April 10, 2015 original date

April 25, 2015 comments added

Typedef Documentation

`typedef std::map<ACCOUNT_INFO::key_type, ACCOUNT_INFO> AccountMap_t`

`typedef std::map<TRANSACTION_INFO::key_type, TRANSACTION_INFO> TransactionMap_t`

Function Documentation

[cnp::QWORD](#) [GenerateCustomerID](#) (const char * *szFirstName*, size_t *cbLen*, [cnp::WORD](#) *wPIN*)

Generate a unique customer ID from a given name + PIN combination

Parameters:

in	<i>szFirstName</i>	address of NULL terminated string containing customer's first name
in	<i>cbLen</i>	count of bytes (cb) length of first name string
in	<i>wPIN</i>	customer's PIN number

Return values:

cnp::QWORD	containing unique 64bit customer ID
----------------------------	-------------------------------------

Here is the call graph for this function:

**size_t LoadServerDB (void)**

Loads into runtime memory the server database records from the persisted store

Return values:

<i>size_t</i>	containing the number of records loaded
---------------	---

Here is the call graph for this function:

**size_t SaveServerDB (void)**

Saves the current server database records to persisted store

Return values:

<i>size_t</i>	contain the number of records saved to file
---------------	---

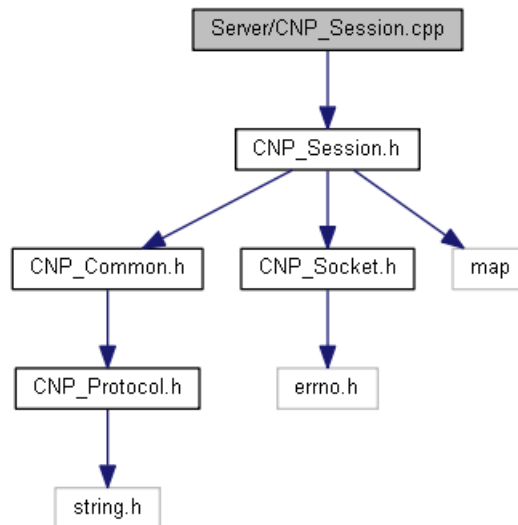
Here is the call graph for this function:



Server/CNP_Session.cpp File Reference

SessionMap_t Global Instance.

Include dependency graph for CNP_Session.cpp:



Variables

- [SessionMap_t g_SessionInfo](#)
Global SessionMap_t instance.

Detailed Description

Author:

Mark L. Short

Date:

April 10, 2015

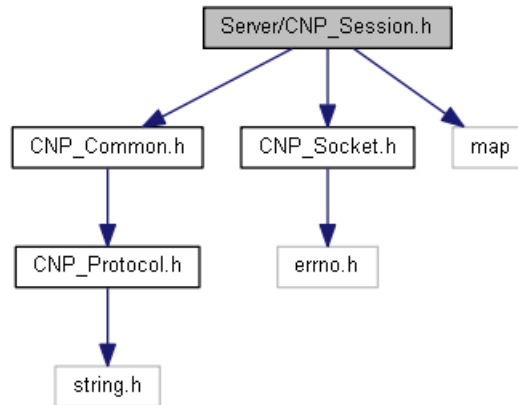
Variable Documentation

[SessionMap_t g_SessionInfo](#)

Server/CNP_Session.h File Reference

[SESSION_INFO](#) struct definition.

Include dependency graph for CNP_Session.h:



Classes

- struct [SESSION_INFO](#)

Typedefs

- typedef std::map<[SESSION_INFO::key_type](#), [SESSION_INFO](#)> [SessionMap_t](#)

Enumerations

- enum [SESSION_STATE](#) { [SS_INVALID](#) = 0, [SS_CONNECTED](#), [SS_ACCOUNT_CREATED](#), [SS_LOGGED_ON](#), [SS_LOGGED_OFF](#), [SS_DISCONNECTING](#) }

Detailed Description

Author:

Mark L. Short

Date:

April 10, 2015

April 25, 2015 updated comments

Typedef Documentation

typedef std::map<[SESSION_INFO::key_type](#), [SESSION_INFO](#)> [SessionMap_t](#)

Enumeration Type Documentation

enum [SESSION_STATE](#)

A basic enumeration of allowable session states

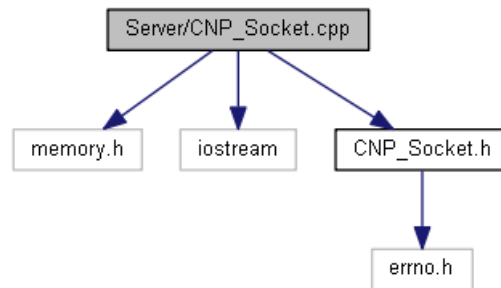
Enumerator

SS_INVALID
SS_CONNECTED
SS_ACCOUNT_CREATED
SS_LOGGED_ON
SS_LOGGED_OFF
SS_DISCONNECTING

Server/CNP_Socket.cpp File Reference

Server [CNP_Socket](#) class implementation.

Include dependency graph for CNP_Socket.cpp:



Detailed Description

Author:

Mark L. Short

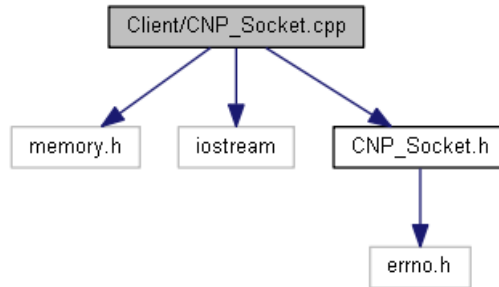
Date:

April 10, 2015

Client/CNP_Socket.cpp File Reference

Client [CNP_Socket](#) class implementation.

Include dependency graph for CNP_Socket.cpp:



Detailed Description

Author:

Mark L. Short

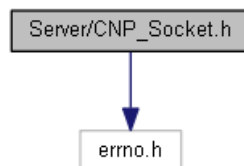
Date:

April 10, 2015

Server/CNP_Socket.h File Reference

Server [CNP_Socket](#) class interface.

Include dependency graph for CNP_Socket.h:



Classes

- class [CNP_Socket](#)

Detailed Description

[CNP_Socket](#) class provides the basic TCP socket functionality. It supports both Windows & Linux platforms*

Author:

Mark L. Short

Date:

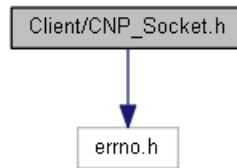
April 10, 2015

April 25, 2015

Client/CNP_Socket.h File Reference

Client [CNP_Socket](#) class interface.

Include dependency graph for CNP_Socket.h:



Classes

- class [CNP_Socket](#)

Detailed Description

[CNP_Socket](#) class provides the basic TCP socket functionality. It supports both Windows & Linux platforms.

Author:

Mark L. Short

Date:

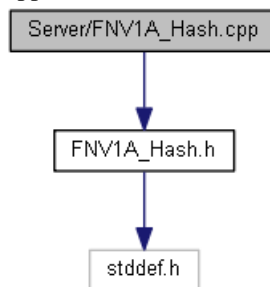
April 10, 2015

April 25, 2015 updated code comments

Server/FNV1A_Hash.cpp File Reference

FNV1A Hash function implementation.

Include dependency graph for FNV1A_Hash.cpp:



Typedefs

- typedef unsigned short [WORD](#)
- typedef unsigned long [DWORD](#)

Functions

- size_t [FNV1A_Hash](#) (const char *pKey, size_t cbLen)

Detailed Description

Author:

Mark L. Short

Date:

April 10, 2015

April 25, 2015 corrected variable naming that previously implied Key parameter was null-terminated, which it is not required.

See also:

http://en.wikipedia.org/wiki/Fowler%E2%80%93Noll%E2%80%93Vo_hash_function

Typedef Documentation

typedef unsigned short [WORD](#)

typedef unsigned long [DWORD](#)

Function Documentation

size_t FNV1A_Hash (const char * *pKey*, size_t *cbLen*)

Performs a hash computation against a series of bytes. The series may contain embedded NULL characters and does not require a NULL-terminated string as the target.

Parameters:

in	<i>pKey</i>	address of sequence of bytes
in	<i>cbLen</i>	count of byte (cb) length of sequence

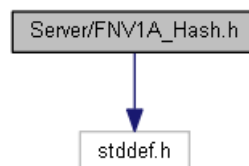
Return values:

<i>size_t</i>	containing results of hash
---------------	----------------------------

Server/FNV1A_Hash.h File Reference

FN1A Hash function prototype.

Include dependency graph for FNV1A_Hash.h:



Functions

- size_t [FNV1A_Hash](#) (const char *pKey, size_t cbLen)
A FNV1a hash implementation.

Detailed Description

Author:

Mark L. Short

Date:

April 10, 2015

Cite: The function implementation was based off of the FNV1a hash algorithm, that has been released to public domain.

See also:

http://en.wikipedia.org/wiki/Fowler%E2%80%93Noll%E2%80%93Vo_hash_function

Function Documentation

size_t FNV1A_Hash (const char * pKey, size_t cbLen)

Performs a hash computation against a series of bytes. The series may contain embedded NULL characters and does not require a NULL-terminated string as the target.

Parameters:

in	pKey	address of sequence of bytes
in	cbLen	count of byte (cb) length of sequence

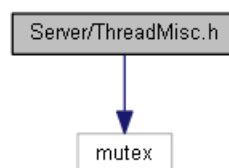
Return values:

size_t	containing results of hash
--------	----------------------------

Server/ThreadMisc.h File Reference

Interface for the Thread Related Classes/Templates.

Include dependency graph for ThreadMisc.h:



Classes

- class [TLock< Ty >](#)
- "Stack-based" Template class [TAutoLock< Ty >](#)

"Stack-based" Template Typedefs

- typedef [TLock](#)< std::recursive_mutex > [CLock](#)
- typedef [TAutoLock](#)< std::recursive_mutex > [CAutoLok](#)
- typedef std::recursive_mutex [CAutoCriticalSection](#)

Detailed Description

Author:

Mark L. Short

Date:

February 1, 2006

April 15, 2015 ported to LINUX

Typedef Documentation

typedef [TLock](#)<std::recursive_mutex> [CLock](#)

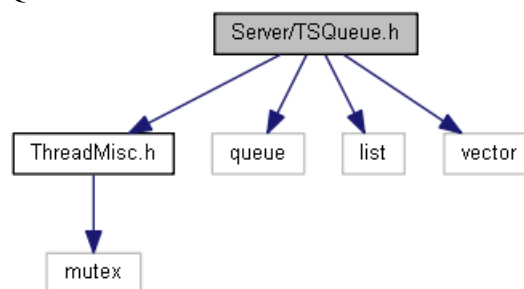
typedef [TAutoLock](#)<std::recursive_mutex> [CAutoLok](#)

typedef std::recursive_mutex [CAutoCriticalSection](#)

Server/TSQueue.h File Reference

interface for the [TTSTQueue](#) class.

Include dependency graph for TSQueue.h:



Classes

- class [TTSTQueue](#)< [_Ty](#) >

Detailed Description

Thread Safe Queue with Associated Critical Section.

Author:

Mark L. Short

Date:

February 1, 2006

Index

- _AutoLock
 - TTSQueue, 81
- _BALANCE_QUERY_REQUEST
 - cnp::prim::_BALANCE_QUERY_REQUEST, 13
- _BALANCE_QUERY_RESPONSE
 - cnp::prim::_BALANCE_QUERY_RESPONSE, 13
- _Base
 - ACCOUNT_INFO, 29
- _CONNECT_REQUEST
 - cnp::prim::_CONNECT_REQUEST, 15
- _CONNECT_RESPONSE
 - cnp::prim::_CONNECT_RESPONSE, 16
- _CREATE_ACCOUNT_REQUEST
 - cnp::prim::_CREATE_ACCOUNT_REQUEST, 17
- _CREATE_ACCOUNT_RESPONSE
 - cnp::prim::_CREATE_ACCOUNT_RESPONSE, 19
- _DEPOSIT_REQUEST
 - cnp::prim::_DEPOSIT_REQUEST, 20
- _DEPOSIT_RESPONSE
 - cnp::prim::_DEPOSIT_RESPONSE, 20
- _Lock
 - TTSQueue, 81
- _LOGOFF_REQUEST
 - cnp::prim::_LOGOFF_REQUEST, 21
- _LOGOFF_RESPONSE
 - cnp::prim::_LOGOFF_RESPONSE, 21
- _LOGON_REQUEST
 - cnp::prim::_LOGON_REQUEST, 22
- _LOGON_RESPONSE
 - cnp::prim::_LOGON_RESPONSE, 23
- _Myt
 - TTSQueue, 81
- _STAMP_PURCHASE_REQUEST
 - cnp::prim::_STAMP_PURCHASE_REQUEST, 24
- _STAMP_PURCHASE_RESPONSE
 - cnp::prim::_STAMP_PURCHASE_RESPONSE, 24
- _TRANSACTION_QUERY_REQUEST
 - cnp::prim::_TRANSACTION_QUERY_REQUEST, 25
- _TRANSACTION_QUERY_RESPONSE
 - cnp::prim::_TRANSACTION_QUERY_RESPONSE, 26
- _WITHDRAWAL_REQUEST
 - cnp::prim::_WITHDRAWAL_REQUEST, 27
- _WITHDRAWAL_RESPONSE
 - cnp::prim::_WITHDRAWAL_RESPONSE, 28
- ~CNP_Socket
 - CNP_Socket, 35, 36
- ~TAutoLock
 - TAutoLock, 69
- ~THREAD_INFO
 - THREAD_INFO, 70
- ~TTSQueue
 - TTSQueue, 81
- Accept
 - CNP_Socket, 36, 38
- ACCOUNT_INFO, 28
 - _Base, 29
 - ACCOUNT_INFO, 29
 - decr_Balance, 30
 - get_Balance, 30
 - get_CustomerID, 30
 - get_PrimaryKey, 29
 - incr_Balance, 30
 - key_type, 29
 - m_dwBalance, 30
 - m_qwCustomerID, 30
 - set_Balance, 30
- AccountMap_t
 - CNP_ServerDB.h, 121
- BALANCE_QUERY_REQUEST
 - cnp::BALANCE_QUERY_REQUEST, 31
- BALANCE_QUERY_RESPONSE
 - cnp::BALANCE_QUERY_RESPONSE, 33
- CASE_CERTYPE
 - CNP_Client.cpp, 88
- CAutoCriticalSection
 - ThreadMisc.h, 129
- CAutoLok
 - ThreadMisc.h, 129
- CER_ACCOUNT_EXISTS
 - cnp, 11
- CER_ACCOUNT_NOT_FOUND
 - cnp, 11
- CER_AUTHENTICATION_FAILED

- cnp, 11
- CER_CLIENT_NOT_LOGGEDON
 - cnp, 11
- CER_DRAWER_BLOCKED
 - cnp, 11
- CER_ERROR
 - cnp, 11
- CER_INSUFFICIENT_FUNDS
 - cnp, 11
- CER_INVALID_ARGUMENTS
 - cnp, 11
- CER_INVALID_CLIENT_ID
 - cnp, 11
- CER_INVALID_NAME_PIN
 - cnp, 11
- CER_SUCCESS
 - cnp, 11
- CER_TYPE
 - cnp, 11
- CER_UNSUPPORTED_PROTOCOL
 - cnp, 11
- CerTypeToString
 - CNP_Client.cpp, 88
- CFC_ACCOUNT
 - Message TypeDefs, 8
- CFC_CONNECT
 - Message TypeDefs, 8
- CFC_CREDENTIALS
 - Message TypeDefs, 8
- CFC_FUNCTIONAL
 - Message TypeDefs, 8
- CFC_TYPE
 - Message TypeDefs, 7
- CFC_UNDEFINED
 - Message TypeDefs, 8
- Client Messages, 6
- Client/CNP_Client.cpp, 87
- Client/CNP_Client.h, 95
- Client/CNP_Socket.cpp, 124
- Client/CNP_Socket.h, 126
- ClientThreadHandler
 - CNP_Server.cpp, 113
- CLock
 - ThreadMisc.h, 129
- Close
 - CNP_Socket, 36, 37
- CMS_INVALID
 - cnp, 11
- CMS_REQUEST
 - cnp, 11
- CMS_RESPONSE
 - cnp, 11
- CMT_BALANCE_QUERY
 - cnp, 11
- CMT_CONNECT
 - cnp, 10
- CMT_CREATE_ACCOUNT
 - cnp, 10
- CMT_DEPOSIT
 - cnp, 10
- CMT_INVALID
 - cnp, 10
- CMT_LOGOFF
 - cnp, 10
- CMT_LOGON
 - cnp, 10
- CMT_PURCHASE_STAMPS
 - cnp, 11
- CMT_TRANSACTION_QUERY
 - cnp, 11
- CMT_WITHDRAWAL
 - cnp, 11
- cnp, 8
 - CER_ACCOUNT_EXISTS, 11
 - CER_ACCOUNT_NOT_FOUND, 11
 - CER_AUTHENTICATION_FAILED, 11
 - CER_CLIENT_NOT_LOGGEDON, 11
 - CER_DRAWER_BLOCKED, 11
 - CER_ERROR, 11
 - CER_INSUFFICIENT_FUNDS, 11
 - CER_INVALID_ARGUMENTS, 11
 - CER_INVALID_CLIENT_ID, 11
 - CER_INVALID_NAME_PIN, 11
 - CER_SUCCESS, 11
 - CER_TYPE, 11
 - CER_UNSUPPORTED_PROTOCOL, 11
 - CMS_INVALID, 11
 - CMS_REQUEST, 11
 - CMS_RESPONSE, 11
 - CMT_BALANCE_QUERY, 11
 - CMT_CONNECT, 10
 - CMT_CREATE_ACCOUNT, 10
 - CMT_DEPOSIT, 10
 - CMT_INVALID, 10
 - CMT_LOGOFF, 10
 - CMT_LOGON, 10
 - CMT_PURCHASE_STAMPS, 11
 - CMT_TRANSACTION_QUERY, 11
 - CMT_WITHDRAWAL, 11
 - CNP_MSG_SUBTYPE, 11
 - CNP_MSG_TYPE, 10
 - DWORD, 10
 - g_dwValidationKey, 12
 - g_wMajorVersion, 11
 - g_wMinorVersion, 12
 - INVALID_CLIENT_ID, 12
 - INVALID_PIN, 12
 - MAX_NAME_LEN, 12
 - QWORD, 10
 - Succeeded, 11
 - WORD, 10
- CNP Protocol Messages, 5

- cnp::BALANCE_QUERY_REQUEST, 30
 - BALANCE_QUERY_REQUEST, 31
 - get_ClientID, 31
 - get_Context, 32
 - get_MsgType, 31
 - get_Sequence, 32
 - get_Size, 31
 - m_Hdr, 32
 - m_Request, 32
- cnp::BALANCE_QUERY_RESPONSE, 32
 - BALANCE_QUERY_RESPONSE, 33
 - get_Balance, 34
 - get_MsgType, 33
 - get_ResponseResult, 34
 - get_Size, 34
 - m_Hdr, 34
 - m_Response, 34
- cnp::CONNECT_REQUEST, 39
 - CONNECT_REQUEST, 40
 - get_ClientID, 41
 - get_ClientMajorVersion, 41
 - get_ClientMinorVersion, 41
 - get_ClientValidationKey, 41
 - get_Context, 41
 - get_MsgType, 40
 - get_Sequence, 41
 - get_Size, 40
 - m_Hdr, 41
 - m_Request, 41
- cnp::CONNECT_RESPONSE, 41
 - CONNECT_RESPONSE, 43
 - get_ClientID, 43
 - get_MsgType, 43
 - get_ResponseResult, 43
 - get_Size, 43
 - m_Hdr, 43
 - m_Response, 43
- cnp::CREATE_ACCOUNT_REQUEST, 43
 - CREATE_ACCOUNT_REQUEST, 45
 - get_ClientID, 45
 - get_Context, 46
 - get_DLNumber, 46
 - get_EmailAddress, 46
 - get_FirstName, 46
 - get_LastName, 46
 - get_MsgType, 45
 - get_PIN, 46
 - get_Sequence, 46
 - get_Size, 45
 - get_SSNumber, 46
 - m_Hdr, 46
 - m_Request, 46
- cnp::CREATE_ACCOUNT_RESPONSE, 46
 - CREATE_ACCOUNT_RESPONSE, 47
 - get_MsgType, 48
 - get_ResponseResult, 48
 - get_Size, 48
 - m_Hdr, 48
 - m_Response, 48
- cnp::DEPOSIT_REQUEST, 48
 - DEPOSIT_REQUEST, 49
 - get_Amount, 50
 - get_ClientID, 50
 - get_Context, 50
 - get_DepositType, 50
 - get_MsgType, 50
 - get_Sequence, 50
 - get_Size, 49
 - m_Hdr, 50
 - m_Request, 50
- cnp::DEPOSIT_RESPONSE, 51
 - DEPOSIT_RESPONSE, 52
 - get_MsgType, 52
 - get_ResponseResult, 52
 - get_Size, 52
 - m_Hdr, 52
 - m_Response, 52
- cnp::LOGOFF_REQUEST, 52
 - get_ClientID, 54
 - get_Context, 54
 - get_MsgType, 54
 - get_Sequence, 54
 - get_Size, 54
 - LOGOFF_REQUEST, 53
 - m_Hdr, 55
 - m_Request, 55
- cnp::LOGOFF_RESPONSE, 55
 - get_MsgType, 56
 - get_ResponseResult, 56
 - get_Size, 56
 - LOGOFF_RESPONSE, 56
 - m_Hdr, 56
 - m_Response, 56
- cnp::LOGON_REQUEST, 56
 - get_ClientID, 58
 - get_Context, 58
 - get_FirstName, 59
 - get_MsgType, 58
 - get_PIN, 59
 - get_Sequence, 58
 - get_Size, 58
 - LOGON_REQUEST, 57
 - m_Hdr, 59
 - m_Request, 59
- cnp::LOGON_RESPONSE, 59
 - get_MsgType, 60
 - get_ResponseResult, 60
 - get_Size, 60
 - LOGON_RESPONSE, 60
 - m_Hdr, 60
 - m_Response, 60
- cnp::prim, 12

cnp::prim::_BALANCE_QUERY_REQUEST, 13
 _BALANCE_QUERY_REQUEST, 13
 cnp::prim::_BALANCE_QUERY_RESPONSE, 13
 _BALANCE_QUERY_RESPONSE, 13
 get_Balance, 14
 m_dwBalance, 14
 m_dwResult, 14
 cnp::prim::_CONNECT_REQUEST, 14
 _CONNECT_REQUEST, 15
 m_dwValidationKey, 15
 m_wMajorVersion, 15
 m_wMinorVersion, 15
 cnp::prim::_CONNECT_RESPONSE, 15
 _CONNECT_RESPONSE, 16
 m_dwResult, 16
 m_wClientID, 16
 m_wMajorVersion, 16
 m_wMinorVersion, 16
 cnp::prim::_CREATE_ACCOUNT_REQUEST, 16
 _CREATE_ACCOUNT_REQUEST, 17
 m_dwDLNumber, 18
 m_dwSSNumber, 18
 m_szEmailAddress, 18
 m_szFirstName, 18
 m_szLastName, 18
 m_wPIN, 18
 set_EmailAddress, 18
 set_FirstName, 18
 set_LastName, 18
 cnp::prim::_CREATE_ACCOUNT_RESPONSE, 18
 _CREATE_ACCOUNT_RESPONSE, 19
 m_dwResult, 19
 cnp::prim::_DEPOSIT_REQUEST, 19
 _DEPOSIT_REQUEST, 20
 m_dwAmount, 20
 m_wType, 20
 cnp::prim::_DEPOSIT_RESPONSE, 20
 _DEPOSIT_RESPONSE, 20
 m_dwResult, 20
 cnp::prim::_LOGOFF_REQUEST, 21
 _LOGOFF_REQUEST, 21
 cnp::prim::_LOGOFF_RESPONSE, 21
 _LOGOFF_RESPONSE, 21
 m_dwResult, 21
 cnp::prim::_LOGON_REQUEST, 22
 _LOGON_REQUEST, 22
 m_szFirstName, 22
 m_wPIN, 22
 set_FirstName, 22
 cnp::prim::_LOGON_RESPONSE, 23
 _LOGON_RESPONSE, 23
 m_dwResult, 23
 cnp::prim::_STAMP_PURCHASE_REQUEST, 23
 _STAMP_PURCHASE_REQUEST, 24
 m_dwAmount, 24
 cnp::prim::_STAMP_PURCHASE_RESPONSE, 24
 _STAMP_PURCHASE_RESPONSE, 24
 m_dwResult, 24
 cnp::prim::_TRANSACTION_QUERY_REQUEST, 25
 _TRANSACTION_QUERY_REQUEST, 25
 m_dwStartID, 25
 m_wTransactionCount, 25
 cnp::prim::_TRANSACTION_QUERY_RESPONSE, 25
 _TRANSACTION_QUERY_RESPONSE, 26
 get_TransactionCount, 26
 m_dwResult, 26
 m_rgTransactions, 26
 m_wTransactionCount, 26
 cnp::prim::_WITHDRAWAL_REQUEST, 27
 _WITHDRAWAL_REQUEST, 27
 m_dwAmount, 27
 cnp::prim::_WITHDRAWAL_RESPONSE, 27
 _WITHDRAWAL_RESPONSE, 28
 m_dwResult, 28
 cnp::STAMP_PURCHASE_REQUEST, 62
 get_Amount, 64
 get_ClientID, 64
 get_Context, 64
 get_MsgType, 63
 get_Sequence, 64
 get_Size, 63
 m_Hdr, 64
 m_Request, 64
 STAMP_PURCHASE_REQUEST, 63
 cnp::STAMP_PURCHASE_RESPONSE, 64
 get_MsgType, 65
 get_ResponseResult, 66
 get_Size, 66
 m_Hdr, 66
 m_Response, 66
 STAMP_PURCHASE_RESPONSE, 65
 cnp::STD_HDR, 66
 get_ClientID, 67
 get_Context, 67
 get_MsgType, 67
 get_Sequence, 67
 m_dwContext, 68
 m_dwMsgType, 68
 m_dwSequence, 68
 m_wClientID, 68
 m_wDataLen, 68
 STD_HDR, 67
 cnp::TRANSACTION, 72
 get_Amount, 73
 get_DateTime, 73
 get_ID, 73
 get_Type, 73
 m_dwAmount, 73
 m_dwID, 73
 m_qwDateTime, 73

- m_wType, 73
- TRANSACTION, 73
- cnp::TRANSACTION_QUERY_REQUEST, 75
 - get_ClientID, 77
 - get_Context, 77
 - get_MsgType, 77
 - get_Sequence, 77
 - get_Size, 77
 - get_StartID, 78
 - get_TransactionCount, 78
 - m_Hdr, 78
 - m_Request, 78
 - TRANSACTION_QUERY_REQUEST, 76
- cnp::TRANSACTION_QUERY_RESPONSE, 78
 - get_MsgType, 79
 - get_ResponseResult, 80
 - get_Size, 79
 - get_TransactionCount, 80
 - m_Hdr, 80
 - m_Response, 80
 - TRANSACTION_QUERY_RESPONSE, 79
- cnp::WITHDRAWAL_REQUEST, 83
 - get_Amount, 85
 - get_ClientID, 84
 - get_Context, 85
 - get_MsgType, 84
 - get_Sequence, 85
 - get_Size, 84
 - m_Hdr, 85
 - m_Request, 85
 - WITHDRAWAL_REQUEST, 84
- cnp::WITHDRAWAL_RESPONSE, 85
 - get_MsgType, 86
 - get_ResponseResult, 86
 - get_Size, 86
 - m_Hdr, 87
 - m_Response, 87
 - WITHDRAWAL_RESPONSE, 86
- CNP_Client.cpp
 - CASE_CERTYPE, 88
 - CerTypeToString, 88
 - g_szBuffer, 95
 - main, 93
 - PrintBankMenu, 88
 - RawTimeToLocalTimeString, 88
 - SendBalance, 90
 - SendConnect, 88
 - SendCreateAccount, 88
 - SendDeposit, 89
 - SendLogIn, 89
 - SendLogOut, 89
 - SendStampPurchase, 92
 - SendTransaction, 91
 - SendWithdrawal, 90
 - TransTypeToString, 88
- CNP_Common.h
 - g_wServerMajorVersion, 99
 - g_wServerMinorVersion, 99
 - INVALID_BALANCE, 99
 - INVALID_CUSTOMER_ID, 99
 - IsValidCustomerID, 99
 - IsValidName, 99
 - IsValidPIN, 99
- CNP_Messaging.cpp
 - g_AccountInfo, 106
 - g_AccountMutex, 106
 - g_SessionInfo, 106
 - g_SessionMutex, 106
 - g_TransactionInfo, 106
 - g_TransactionMutex, 106
 - ProcessBalanceQueryRequest, 104
 - ProcessConnectRequest, 100
 - ProcessCreateAccountRequest, 101
 - ProcessDepositRequest, 103
 - ProcessDisconnect, 106
 - ProcessLogoffRequest, 102
 - ProcessLogonRequest, 102
 - ProcessStampPurchaseRequest, 105
 - ProcessTransactionQueryRequest, 104
 - ProcessWithdrawalRequest, 103
- CNP_Messaging.h
 - ProcessBalanceQueryRequest, 107
 - ProcessConnectRequest, 107
 - ProcessCreateAccountRequest, 107
 - ProcessDepositRequest, 108
 - ProcessDisconnect, 112
 - ProcessLogoffRequest, 109
 - ProcessLogonRequest, 109
 - ProcessStampPurchaseRequest, 110
 - ProcessTransactionQueryRequest, 111
 - ProcessWithdrawalRequest, 112
- CNP_MSG_SUBTYPE
 - cnp, 11
- CNP_MSG_TYPE
 - cnp, 10
- CNP_Protocol.h
 - COUNTOF, 98
 - MAKE_ERROR_RESULT, 98
 - MAKE_MSG_TYPE, 98
- CNP_Server.cpp
 - ClientThreadHandler, 113
 - main, 115
 - TerminateHandler, 115
- CNP_ServerDB.cpp
 - g_AccountInfo, 120
 - g_szAccountDBFileName, 120
 - g_szTransactDBFileName, 120
 - g_TransactionInfo, 120
 - GenerateCustomerID, 118
 - LoadServerDB, 118, 119
 - SaveServerDB, 119
- CNP_ServerDB.h

- AccountMap_t, 121
- GenerateCustomerID, 121
- LoadServerDB, 121
- SaveServerDB, 121
- TransactionMap_t, 121
- CNP_Session.cpp
 - g_SessionInfo, 122
- CNP_Session.h
 - SESSION_STATE, 123
 - SessionMap_t, 123
 - SS_ACCOUNT_CREATED, 124
 - SS_CONNECTED, 124
 - SS_DISCONNECTING, 124
 - SS_INVALID, 124
 - SS_LOGGED_OFF, 124
 - SS_LOGGED_ON, 124
- CNP_Socket, 34
 - ~CNP_Socket, 35, 36
 - Accept, 36, 38
 - Close, 36, 37
 - CNP_Socket, 35
 - Connect, 36, 37
 - Create, 36, 37
 - GetError, 37, 38
 - Listen, 36, 37
 - m_hSocket, 39
 - m_iError, 39
 - m_LocalAddr, 39
 - m_RemoteAddr, 39
 - m_wPort, 39
 - Receive, 36, 38
 - Send, 36, 38
 - SetBlocking, 37, 38
 - SetSocketOption, 36, 38
 - Shutdown, 37, 38
- Connect
 - CNP_Socket, 36, 37
- CONNECT_REQUEST
 - cnp::CONNECT_REQUEST, 40
- CONNECT_RESPONSE
 - cnp::CONNECT_RESPONSE, 43
- COUNTOF
 - CNP_Protocol.h, 98
- Create
 - CNP_Socket, 36, 37
- CREATE_ACCOUNT_REQUEST
 - cnp::CREATE_ACCOUNT_REQUEST, 45
- CREATE_ACCOUNT_RESPONSE
 - cnp::CREATE_ACCOUNT_RESPONSE, 47
- decr_Balance
 - ACCOUNT_INFO, 30
- DEPOSIT_REQUEST
 - cnp::DEPOSIT_REQUEST, 49
- DEPOSIT_RESPONSE
 - cnp::DEPOSIT_RESPONSE, 52
- DEPOSIT_TYPE
 - Message TypeDefs, 8
- DT_CASH
 - Message TypeDefs, 8
- DT_CHECK
 - Message TypeDefs, 8
- DT_INVALID
 - Message TypeDefs, 8
- DWORD
 - cnp, 10
 - FNV1A_Hash.cpp, 127
- FNV1A_Hash
 - FNV1A_Hash.cpp, 127
 - FNV1A_Hash.h, 128
- FNV1A_Hash.cpp
 - DWORD, 127
 - FNV1A_Hash, 127
 - WORD, 127
- FNV1A_Hash.h
 - FNV1A_Hash, 128
- Front
 - TTSQueue, 82
- g_AccountInfo
 - CNP_Messaging.cpp, 106
 - CNP_ServerDB.cpp, 120
- g_AccountMutex
 - CNP_Messaging.cpp, 106
- g_dwValidationKey
 - cnp, 12
- g_SessionInfo
 - CNP_Messaging.cpp, 106
 - CNP_Session.cpp, 122
- g_SessionMutex
 - CNP_Messaging.cpp, 106
- g_szAccountDBFileName
 - CNP_ServerDB.cpp, 120
- g_szBuffer
 - CNP_Client.cpp, 95
- g_szTransactDBFileName
 - CNP_ServerDB.cpp, 120
- g_TransactionInfo
 - CNP_Messaging.cpp, 106
 - CNP_ServerDB.cpp, 120
- g_TransactionMutex
 - CNP_Messaging.cpp, 106
- g_wMajorVersion
 - cnp, 11
- g_wMinorVersion
 - cnp, 12
- g_wServerMajorVersion
 - CNP_Common.h, 99
- g_wServerMinorVersion
 - CNP_Common.h, 99
- GenerateCustomerID
 - CNP_ServerDB.cpp, 118
 - CNP_ServerDB.h, 121
- get_Amount

- cnpp::DEPOSIT_REQUEST, 50
- cnpp::STAMP_PURCHASE_REQUEST, 64
- cnpp::TRANSACTION, 73
- cnpp::WITHDRAWAL_REQUEST, 85
- get_Balance
 - ACCOUNT_INFO, 30
 - cnpp::BALANCE_QUERY_RESPONSE, 34
 - cnpp::prim::BALANCE_QUERY_RESPONSE, 14
- get_ClientID
 - cnpp::BALANCE_QUERY_REQUEST, 31
 - cnpp::CONNECT_REQUEST, 41
 - cnpp::CONNECT_RESPONSE, 43
 - cnpp::CREATE_ACCOUNT_REQUEST, 45
 - cnpp::DEPOSIT_REQUEST, 50
 - cnpp::LOGOFF_REQUEST, 54
 - cnpp::LOGON_REQUEST, 58
 - cnpp::STAMP_PURCHASE_REQUEST, 64
 - cnpp::STD_HDR, 67
 - cnpp::TRANSACTION_QUERY_REQUEST, 77
 - cnpp::WITHDRAWAL_REQUEST, 84
 - SESSION_INFO, 62
- get_ClientMajorVersion
 - cnpp::CONNECT_REQUEST, 41
- get_ClientMinorVersion
 - cnpp::CONNECT_REQUEST, 41
- get_ClientValidationKey
 - cnpp::CONNECT_REQUEST, 41
- get_Context
 - cnpp::BALANCE_QUERY_REQUEST, 32
 - cnpp::CONNECT_REQUEST, 41
 - cnpp::CREATE_ACCOUNT_REQUEST, 46
 - cnpp::DEPOSIT_REQUEST, 50
 - cnpp::LOGOFF_REQUEST, 54
 - cnpp::LOGON_REQUEST, 58
 - cnpp::STAMP_PURCHASE_REQUEST, 64
 - cnpp::STD_HDR, 67
 - cnpp::TRANSACTION_QUERY_REQUEST, 77
 - cnpp::WITHDRAWAL_REQUEST, 85
- get_CustomerID
 - ACCOUNT_INFO, 30
 - SESSION_INFO, 62
 - TRANSACTION_INFO, 75
- get_DateTime
 - cnpp::TRANSACTION, 73
- get_DepositType
 - cnpp::DEPOSIT_REQUEST, 50
- get_DLNumber
 - cnpp::CREATE_ACCOUNT_REQUEST, 46
- get_EmailAddress
 - cnpp::CREATE_ACCOUNT_REQUEST, 46
- get_FirstName
 - cnpp::CREATE_ACCOUNT_REQUEST, 46
 - cnpp::LOGON_REQUEST, 59
- get_ID
 - cnpp::TRANSACTION, 73
- get_LastName
 - cnpp::CREATE_ACCOUNT_REQUEST, 46
- get_MsgType
 - cnpp::BALANCE_QUERY_REQUEST, 31
 - cnpp::BALANCE_QUERY_RESPONSE, 33
 - cnpp::CONNECT_REQUEST, 40
 - cnpp::CONNECT_RESPONSE, 43
 - cnpp::CREATE_ACCOUNT_REQUEST, 45
 - cnpp::CREATE_ACCOUNT_RESPONSE, 48
 - cnpp::DEPOSIT_REQUEST, 50
 - cnpp::DEPOSIT_RESPONSE, 52
 - cnpp::LOGOFF_REQUEST, 54
 - cnpp::LOGOFF_RESPONSE, 56
 - cnpp::LOGON_REQUEST, 58
 - cnpp::LOGON_RESPONSE, 60
 - cnpp::STAMP_PURCHASE_REQUEST, 63
 - cnpp::STAMP_PURCHASE_RESPONSE, 65
 - cnpp::STD_HDR, 67
 - cnpp::TRANSACTION_QUERY_REQUEST, 77
 - cnpp::TRANSACTION_QUERY_RESPONSE, 79
 - cnpp::WITHDRAWAL_REQUEST, 84
 - cnpp::WITHDRAWAL_RESPONSE, 86
- get_PIN
 - cnpp::CREATE_ACCOUNT_REQUEST, 46
 - cnpp::LOGON_REQUEST, 59
- get_PrimaryKey
 - ACCOUNT_INFO, 29
 - TRANSACTION_INFO, 75
- get_ResponseResult
 - cnpp::BALANCE_QUERY_RESPONSE, 34
 - cnpp::CONNECT_RESPONSE, 43
 - cnpp::CREATE_ACCOUNT_RESPONSE, 48
 - cnpp::DEPOSIT_RESPONSE, 52
 - cnpp::LOGOFF_RESPONSE, 56
 - cnpp::LOGON_RESPONSE, 60
 - cnpp::STAMP_PURCHASE_RESPONSE, 66
 - cnpp::TRANSACTION_QUERY_RESPONSE, 80
 - cnpp::WITHDRAWAL_RESPONSE, 86
- get_Sequence
 - cnpp::BALANCE_QUERY_REQUEST, 32
 - cnpp::CONNECT_REQUEST, 41
 - cnpp::CREATE_ACCOUNT_REQUEST, 46
 - cnpp::DEPOSIT_REQUEST, 50
 - cnpp::LOGOFF_REQUEST, 54
 - cnpp::LOGON_REQUEST, 58
 - cnpp::STAMP_PURCHASE_REQUEST, 64
 - cnpp::STD_HDR, 67
 - cnpp::TRANSACTION_QUERY_REQUEST, 77
 - cnpp::WITHDRAWAL_REQUEST, 85
- get_Size
 - cnpp::BALANCE_QUERY_REQUEST, 31
 - cnpp::BALANCE_QUERY_RESPONSE, 34
 - cnpp::CONNECT_REQUEST, 40
 - cnpp::CONNECT_RESPONSE, 43
 - cnpp::CREATE_ACCOUNT_REQUEST, 45
 - cnpp::CREATE_ACCOUNT_RESPONSE, 48

- cnp::DEPOSIT_REQUEST, 49
- cnp::DEPOSIT_RESPONSE, 52
- cnp::LOGOFF_REQUEST, 54
- cnp::LOGOFF_RESPONSE, 56
- cnp::LOGON_REQUEST, 58
- cnp::LOGON_RESPONSE, 60
- cnp::STAMP_PURCHASE_REQUEST, 63
- cnp::STAMP_PURCHASE_RESPONSE, 66
- cnp::TRANSACTION_QUERY_REQUEST, 77
- cnp::TRANSACTION_QUERY_RESPONSE, 79
- cnp::WITHDRAWAL_REQUEST, 84
- cnp::WITHDRAWAL_RESPONSE, 86
- get_SSNumber
 - cnp::CREATE_ACCOUNT_REQUEST, 46
- get_StartID
 - cnp::TRANSACTION_QUERY_REQUEST, 78
- get_State
 - SESSION_INFO, 62
- get_TransactionCount
 - cnp::prim::_TRANSACTION_QUERY_RESPONSE, 26
 - cnp::TRANSACTION_QUERY_REQUEST, 78
 - cnp::TRANSACTION_QUERY_RESPONSE, 80
- get_Type
 - cnp::TRANSACTION, 73
- GetError
 - CNP_Socket, 37, 38
- GetItems
 - TTSQueue, 82
- Include/CNP_Protocol.h, 95
- incr_Balance
 - ACCOUNT_INFO, 30
- Init
 - TTSQueue, 82
- INVALID_BALANCE
 - CNP_Common.h, 99
- INVALID_CLIENT_ID
 - cnp, 12
- INVALID_CUSTOMER_ID
 - CNP_Common.h, 99
- INVALID_PIN
 - cnp, 12
- IsEmpty
 - TTSQueue, 83
- IsValidCustomerID
 - CNP_Common.h, 99
- IsValidName
 - CNP_Common.h, 99
- IsValidPIN
 - CNP_Common.h, 99
- items_type
 - TTSQueue, 81
- key_type
 - ACCOUNT_INFO, 29
 - SESSION_INFO, 61
 - TRANSACTION_INFO, 74

- Listen
 - CNP_Socket, 36, 37
- LoadServerDB
 - CNP_ServerDB.cpp, 118, 119
 - CNP_ServerDB.h, 121
- Lock
 - TLock, 72
 - TTSQueue, 82
- LOGOFF_REQUEST
 - cnp::LOGOFF_REQUEST, 53
- LOGOFF_RESPONSE
 - cnp::LOGOFF_RESPONSE, 56
- LOGON_REQUEST
 - cnp::LOGON_REQUEST, 57
- LOGON_RESPONSE
 - cnp::LOGON_RESPONSE, 60
- m_bTerminate
 - THREAD_INFO, 70
- m_cs
 - TTSQueue, 83
- m_dwAmount
 - cnp::prim::_DEPOSIT_REQUEST, 20
 - cnp::prim::_STAMP_PURCHASE_REQUEST, 24
 - cnp::prim::_WITHDRAWAL_REQUEST, 27
 - cnp::TRANSACTION, 73
- m_dwBalance
 - ACCOUNT_INFO, 30
 - cnp::prim::_BALANCE_QUERY_RESPONSE, 14
- m_dwContext
 - cnp::STD_HDR, 68
- m_dwDLNumber
 - cnp::prim::_CREATE_ACCOUNT_REQUEST, 18
- m_dwID
 - cnp::TRANSACTION, 73
- m_dwMsgType
 - cnp::STD_HDR, 68
- m_dwResult
 - cnp::prim::_BALANCE_QUERY_RESPONSE, 14
 - cnp::prim::_CONNECT_RESPONSE, 16
 - cnp::prim::_CREATE_ACCOUNT_RESPONSE, 19
 - cnp::prim::_DEPOSIT_RESPONSE, 20
 - cnp::prim::_LOGOFF_RESPONSE, 21
 - cnp::prim::_LOGON_RESPONSE, 23
 - cnp::prim::_STAMP_PURCHASE_RESPONSE, 24
 - cnp::prim::_TRANSACTION_QUERY_RESPONSE, 26
 - cnp::prim::_WITHDRAWAL_RESPONSE, 28
- m_dwSequence
 - cnp::STD_HDR, 68
- m_dwSSNumber

```

    cnp::prim::_CREATE_ACCOUNT_REQUEST,
    18
m_dwStartID
    cnp::prim::_TRANSACTION_QUERY_REQUEST,
    25
m_dwValidationKey
    cnp::prim::_CONNECT_REQUEST, 15
m_Hdr
    cnp::BALANCE_QUERY_REQUEST, 32
    cnp::BALANCE_QUERY_RESPONSE, 34
    cnp::CONNECT_REQUEST, 41
    cnp::CONNECT_RESPONSE, 43
    cnp::CREATE_ACCOUNT_REQUEST, 46
    cnp::CREATE_ACCOUNT_RESPONSE, 48
    cnp::DEPOSIT_REQUEST, 50
    cnp::DEPOSIT_RESPONSE, 52
    cnp::LOGOFF_REQUEST, 55
    cnp::LOGOFF_RESPONSE, 56
    cnp::LOGON_REQUEST, 59
    cnp::LOGON_RESPONSE, 60
    cnp::STAMP_PURCHASE_REQUEST, 64
    cnp::STAMP_PURCHASE_RESPONSE, 66
    cnp::TRANSACTION_QUERY_REQUEST, 78
    cnp::TRANSACTION_QUERY_RESPONSE, 80
    cnp::WITHDRAWAL_REQUEST, 85
    cnp::WITHDRAWAL_RESPONSE, 87
m_hSocket
    CNP_Socket, 39
m_iError
    CNP_Socket, 39
m_LocalAddr
    CNP_Socket, 39
m_pCS
    TLock, 72
m_pSocket
    SESSION_INFO, 62
    THREAD_INFO, 70
m_pThread
    THREAD_INFO, 70
m_que
    TTSQueue, 83
m_qwCustomerID
    ACCOUNT_INFO, 30
    SESSION_INFO, 62
    TRANSACTION_INFO, 75
m_qwDateTime
    cnp::TRANSACTION, 73
m_RemoteAddr
    CNP_Socket, 39
m_Request
    cnp::BALANCE_QUERY_REQUEST, 32
    cnp::CONNECT_REQUEST, 41
    cnp::CREATE_ACCOUNT_REQUEST, 46
    cnp::DEPOSIT_REQUEST, 50
    cnp::LOGOFF_REQUEST, 55
    cnp::LOGON_REQUEST, 59
    cnp::STAMP_PURCHASE_REQUEST, 64
    cnp::TRANSACTION_QUERY_REQUEST, 78
    cnp::WITHDRAWAL_REQUEST, 85
m_Response
    cnp::BALANCE_QUERY_RESPONSE, 34
    cnp::CONNECT_RESPONSE, 43
    cnp::CREATE_ACCOUNT_RESPONSE, 48
    cnp::DEPOSIT_RESPONSE, 52
    cnp::LOGOFF_RESPONSE, 56
    cnp::LOGON_RESPONSE, 60
    cnp::STAMP_PURCHASE_RESPONSE, 66
    cnp::TRANSACTION_QUERY_RESPONSE, 80
    cnp::WITHDRAWAL_RESPONSE, 87
m_rgTransactions
    cnp::prim::_TRANSACTION_QUERY_RESPONSE,
    26
m_szEmailAddress
    cnp::prim::_CREATE_ACCOUNT_REQUEST,
    18
m_szFirstName
    cnp::prim::_CREATE_ACCOUNT_REQUEST,
    18
    cnp::prim::_LOGON_REQUEST, 22
m_szLastName
    cnp::prim::_CREATE_ACCOUNT_REQUEST,
    18
m_wClientID
    cnp::prim::_CONNECT_RESPONSE, 16
    cnp::STD_HDR, 68
    SESSION_INFO, 62
m_wDataLen
    cnp::STD_HDR, 68
m_wMajorVersion
    cnp::prim::_CONNECT_REQUEST, 15
    cnp::prim::_CONNECT_RESPONSE, 16
m_wMinorVersion
    cnp::prim::_CONNECT_REQUEST, 15
    cnp::prim::_CONNECT_RESPONSE, 16
m_wPIN
    cnp::prim::_CREATE_ACCOUNT_REQUEST,
    18
    cnp::prim::_LOGON_REQUEST, 22
m_wPort
    CNP_Socket, 39
m_wState
    SESSION_INFO, 62
m_wTransactionCount
    cnp::prim::_TRANSACTION_QUERY_REQUEST,
    25
    cnp::prim::_TRANSACTION_QUERY_RESPONSE,
    26
m_wType
    cnp::prim::_DEPOSIT_REQUEST, 20
    cnp::TRANSACTION, 73
main
    CNP_Client.cpp, 93

```

CNP_Server.cpp, 115
 MAKE_ERROR_RESULT
 CNP_Protocol.h, 98
 MAKE_MSG_TYPE
 CNP_Protocol.h, 98
 MAX_NAME_LEN
 cnp, 12
 Message TypeDefs, 6
 CFC_ACCOUNT, 8
 CFC_CONNECT, 8
 CFC_CREDENTIALS, 8
 CFC_FUNCTIONAL, 8
 CFC_TYPE, 7
 CFC_UNDEFINED, 8
 DEPOSIT_TYPE, 8
 DT_CASH, 8
 DT_CHECK, 8
 DT_INVALID, 8
 MSG_TYPE, 7
 MT_BALANCE_QUERY_REQUEST, 7
 MT_BALANCE_QUERY_RESPONSE, 7
 MT_CONNECT_REQUEST, 7
 MT_CONNECT_RESPONSE, 7
 MT_CREATE_ACCOUNT_REQUEST, 7
 MT_CREATE_ACCOUNT_RESPONSE, 7
 MT_DEPOSIT_REQUEST, 7
 MT_DEPOSIT_RESPONSE, 7
 MT_INVALID, 7
 MT_LOGOFF_REQUEST, 7
 MT_LOGOFF_RESPONSE, 7
 MT_LOGON_REQUEST, 7
 MT_LOGON_RESPONSE, 7
 MT_PURCHASE_STAMPS_REQUEST, 7
 MT_PURCHASE_STAMPS_RESPONSE, 7
 MT_TRANSACTION_QUERY_REQUEST, 7
 MT_TRANSACTION_QUERY_RESPONSE, 7
 MT_WITHDRAWAL_REQUEST, 7
 MT_WITHDRAWAL_RESPONSE, 7
 TRANSACTION_TYPE, 8
 TT_DEPOSIT, 8
 TT_INVALID, 8
 TT_STAMP_PURCHASE, 8
 TT_WITHDRAWAL, 8
 MSG_TYPE
 Message TypeDefs, 7
 MT_BALANCE_QUERY_REQUEST
 Message TypeDefs, 7
 MT_BALANCE_QUERY_RESPONSE
 Message TypeDefs, 7
 MT_CONNECT_REQUEST
 Message TypeDefs, 7
 MT_CONNECT_RESPONSE
 Message TypeDefs, 7
 MT_CREATE_ACCOUNT_REQUEST
 Message TypeDefs, 7
 MT_CREATE_ACCOUNT_RESPONSE
 Message TypeDefs, 7
 Message TypeDefs, 7
 MT_DEPOSIT_REQUEST
 Message TypeDefs, 7
 MT_DEPOSIT_RESPONSE
 Message TypeDefs, 7
 MT_INVALID
 Message TypeDefs, 7
 MT_LOGOFF_REQUEST
 Message TypeDefs, 7
 MT_LOGOFF_RESPONSE
 Message TypeDefs, 7
 MT_LOGON_REQUEST
 Message TypeDefs, 7
 MT_LOGON_RESPONSE
 Message TypeDefs, 7
 MT_PURCHASE_STAMPS_REQUEST
 Message TypeDefs, 7
 MT_PURCHASE_STAMPS_RESPONSE
 Message TypeDefs, 7
 MT_TRANSACTION_QUERY_REQUEST
 Message TypeDefs, 7
 MT_TRANSACTION_QUERY_RESPONSE
 Message TypeDefs, 7
 MT_WITHDRAWAL_REQUEST
 Message TypeDefs, 7
 MT_WITHDRAWAL_RESPONSE
 Message TypeDefs, 7
 operator=
 THREAD_INFO, 70
 TTSQueue, 82
 Pop
 TTSQueue, 82
 PopFront
 TTSQueue, 82
 PrintBankMenu
 CNP_Client.cpp, 88
 ProcessBalanceQueryRequest
 CNP_Messaging.cpp, 104
 CNP_Messaging.h, 107
 ProcessConnectRequest
 CNP_Messaging.cpp, 100
 CNP_Messaging.h, 107
 ProcessCreateAccountRequest
 CNP_Messaging.cpp, 101
 CNP_Messaging.h, 107
 ProcessDepositRequest
 CNP_Messaging.cpp, 103
 CNP_Messaging.h, 108
 ProcessDisconnect
 CNP_Messaging.cpp, 106
 CNP_Messaging.h, 112
 ProcessLogoffRequest
 CNP_Messaging.cpp, 102
 CNP_Messaging.h, 109
 ProcessLogonRequest
 CNP_Messaging.cpp, 102

- CNP_Messaging.h, 109
- ProcessStampPurchaseRequest
 - CNP_Messaging.cpp, 105
 - CNP_Messaging.h, 110
- ProcessTransactionQueryRequest
 - CNP_Messaging.cpp, 104
 - CNP_Messaging.h, 111
- ProcessWithdrawalRequest
 - CNP_Messaging.cpp, 103
 - CNP_Messaging.h, 112
- Push
 - TTSQueue, 82
- que_type
 - TTSQueue, 81
- QWORD
 - cnp, 10
- RawTimeToLocalTimeString
 - CNP_Client.cpp, 88
- Receive
 - CNP_Socket, 36, 38
- SaveServerDB
 - CNP_ServerDB.cpp, 119
 - CNP_ServerDB.h, 121
- Send
 - CNP_Socket, 36, 38
- SendBalance
 - CNP_Client.cpp, 90
- SendConnect
 - CNP_Client.cpp, 88
- SendCreateAccount
 - CNP_Client.cpp, 88
- SendDeposit
 - CNP_Client.cpp, 89
- SendLogIn
 - CNP_Client.cpp, 89
- SendLogOut
 - CNP_Client.cpp, 89
- SendStampPurchase
 - CNP_Client.cpp, 92
- SendTransaction
 - CNP_Client.cpp, 91
- SendWithdrawal
 - CNP_Client.cpp, 90
- Server Messages, 5
- Server/CNP_Common.h, 98
- Server/CNP_Messaging.cpp, 99
- Server/CNP_Messaging.h, 106
- Server/CNP_Server.cpp, 112
- Server/CNP_Server.h, 117
- Server/CNP_ServerDB.cpp, 117
- Server/CNP_ServerDB.h, 120
- Server/CNP_Session.cpp, 122
- Server/CNP_Session.h, 123
- Server/CNP_Socket.cpp, 124
- Server/CNP_Socket.h, 125
- Server/FNV1A_Hash.cpp, 126
- Server/FNV1A_Hash.h, 127
- Server/ThreadMisc.h, 128
- Server/TSQueue.h, 129
- SESSION_INFO, 61
 - get_ClientID, 62
 - get_CustomerID, 62
 - get_State, 62
 - key_type, 61
 - m_pSocket, 62
 - m_qwCustomerID, 62
 - m_wClientID, 62
 - m_wState, 62
 - SESSION_INFO, 61
 - set_ClientID, 62
 - set_CustomerID, 62
 - set_State, 62
- SESSION_STATE
 - CNP_Session.h, 123
- SessionMap_t
 - CNP_Session.h, 123
- set_Balance
 - ACCOUNT_INFO, 30
- set_ClientID
 - SESSION_INFO, 62
- set_CustomerID
 - SESSION_INFO, 62
- set_EmailAddress
 - cnp::prim::_CREATE_ACCOUNT_REQUEST, 18
- set_FirstName
 - cnp::prim::_CREATE_ACCOUNT_REQUEST, 18
 - cnp::prim::_LOGON_REQUEST, 22
- set_LastName
 - cnp::prim::_CREATE_ACCOUNT_REQUEST, 18
- set_State
 - SESSION_INFO, 62
- SetBlocking
 - CNP_Socket, 37, 38
- SetLock
 - TLock, 72
- SetSocketOption
 - CNP_Socket, 36, 38
- Shutdown
 - CNP_Socket, 37, 38
- Size
 - TTSQueue, 83
- size_type
 - TTSQueue, 81
- SS_ACCOUNT_CREATED
 - CNP_Session.h, 124
- SS_CONNECTED
 - CNP_Session.h, 124
- SS_DISCONNECTING
 - CNP_Session.h, 124

SS_INVALID	cnp::TRANSACTION_QUERY_RESPONSE, 79
CNP_Session.h, 124	TRANSACTION_TYPE
SS_LOGGED_OFF	Message TypeDefs, 8
CNP_Session.h, 124	TransactionMap_t
SS_LOGGED_ON	CNP_ServerDB.h, 121
CNP_Session.h, 124	TransTypeToString
STAMP_PURCHASE_REQUEST	CNP_Client.cpp, 88
cnp::STAMP_PURCHASE_REQUEST, 63	TT_DEPOSIT
STAMP_PURCHASE_RESPONSE	Message TypeDefs, 8
cnp::STAMP_PURCHASE_RESPONSE, 65	TT_INVALID
STD_HDR	Message TypeDefs, 8
cnp::STD_HDR, 67	TT_STAMP_PURCHASE
Succeeded	Message TypeDefs, 8
cnp, 11	TT_WITHDRAWAL
TAutoLock	Message TypeDefs, 8
~TAutoLock, 69	TTSQueue
TAutoLock, 69	_AutoLock, 81
TAutoLock<_Ty>, 68	_Lock, 81
TerminateHandler	_Myt, 81
CNP_Server.cpp, 115	~TTSQueue, 81
THREAD_INFO, 69	Front, 82
~THREAD_INFO, 70	GetItems, 82
m_bTerminate, 70	Init, 82
m_pSocket, 70	IsEmpty, 83
m_pThread, 70	items_type, 81
operator=, 70	Lock, 82
THREAD_INFO, 70	m_cs, 83
ThreadMisc.h	m_que, 83
CAutoCriticalSection, 129	operator=, 82
CAutoLok, 129	Pop, 82
CLock, 129	PopFront, 82
TLock	Push, 82
Lock, 72	que_type, 81
m_pCS, 72	Size, 83
SetLock, 72	size_type, 81
TLock, 71	TTSQueue, 81
Unlock, 72	Unlock, 82
TLock<_Ty>, 70	TTSTQueue<_Ty>, 80
TRANSACTION	Unlock
cnp::TRANSACTION, 73	TLock, 72
TRANSACTION_INFO, 74	TTSQueue, 82
get_CustomerID, 75	WITHDRAWAL_REQUEST
get_PrimaryKey, 75	cnp::WITHDRAWAL_REQUEST, 84
key_type, 74	WITHDRAWAL_RESPONSE
m_qwCustomerID, 75	cnp::WITHDRAWAL_RESPONSE, 86
TRANSACTION_INFO, 75	WORD
TRANSACTION_QUERY_REQUEST	cnp, 10
cnp::TRANSACTION_QUERY_REQUEST, 76	FNV1A_Hash.cpp, 127
TRANSACTION_QUERY_RESPONSE	