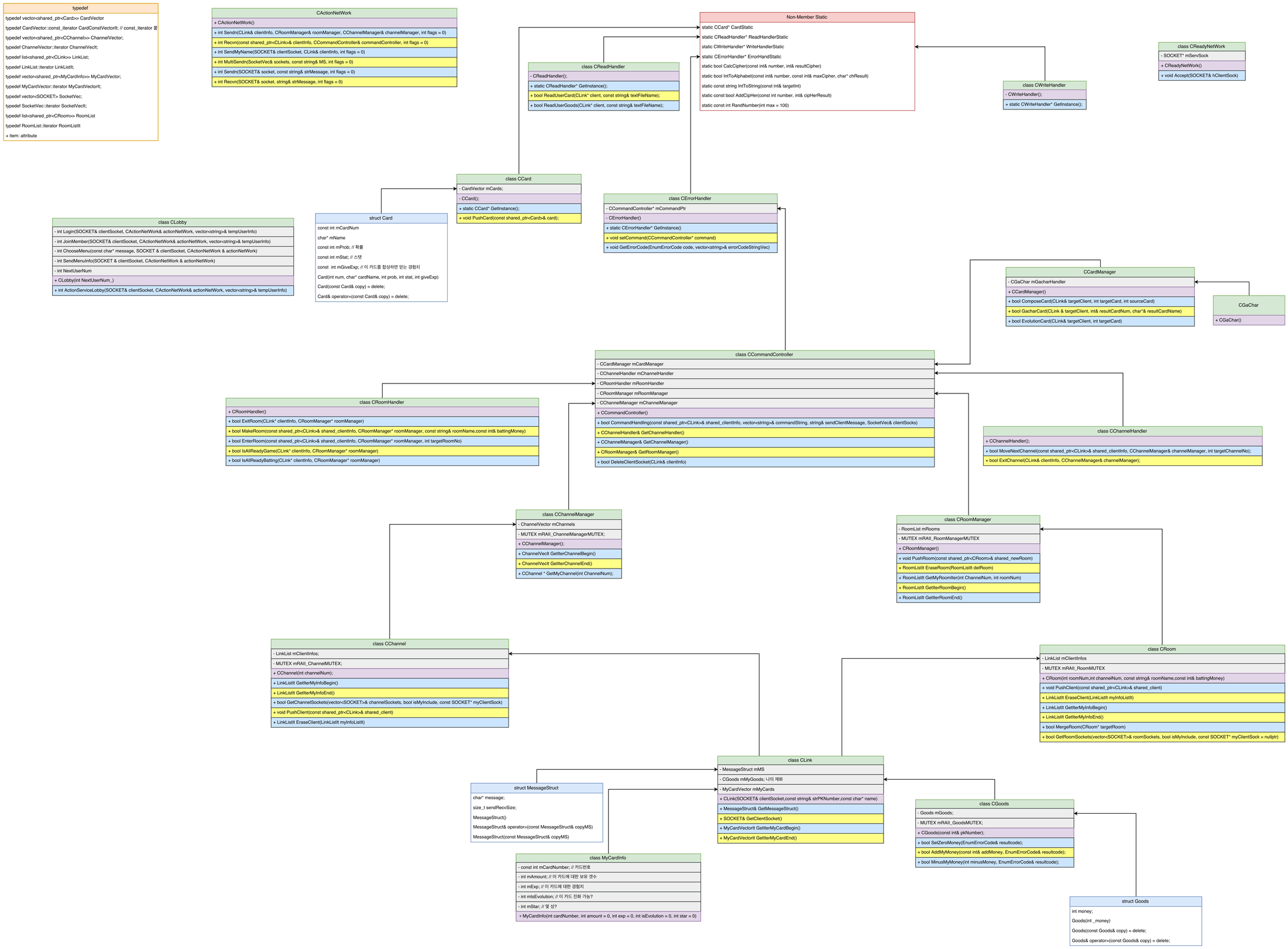
2017 08 06


typedef
typedef vector<shared_ptr<Card>> CardVector
typedef CardVector::const_iterator CardConstVectorIt; // const_iterator 붙
typedef vector<shared_ptr<CChannel>> ChannelVector;
typedef ChannelVector::iterator ChannelVecIt;
typedef list<shared_ptr<CLink>> LinkList;
typedef LinkList::iterator LinkListIt;
typedef vector<shared_ptr<MyCardInfo>> MyCardVector;
typedef MyCardVector::iterator MyCardVectorIt;
typedef vector<SOCKET> SocketVec;
typedef SocketVec::iterator SocketVecIt;
typedef list<shared_ptr<CRoom>> RoomList
typedef RoomList::iterator RoomListIt
+ item: attribute
CActionNetWork
+ CActionNetWork()
+ int Sendn(CLink& clientInfo, CRoomManager& roomManager, CChannelManager& channelManager, int flags = 0)
+ int Recvn(const shared_ptr<CLink>& clientInfo, CCommandController& commandController, int flags = 0)
+ int SendMyName(SOCKET& clientSocket, CLink& clientInfo, int flags = 0)
+ int MultiSendn(SocketVec& sockets, const string& MS, int flags = 0)
+ int Sendn(SOCKET& socket, const string& strMessage, int flags = 0)
+ int Recvn(SOCKET& socket, string& strMessage, int flags = 0)
Non-Member Static
static CCard* CardStatic
static CReadHandler* ReadHandlerStatic
static CWriteHandler* WriteHandlerStatic
static CErrorHandler* ErrorHandStatic
static bool CalcCipher(const int& number, int& resultCipher)
static bool IntToAlphabet(const int& number, const int& maxCipher, char* chResult)
static const string IntToString(const int& targetInt)
static const bool AddCipHer(const int number, int& cipHerResult)
static const int RandNumber(int max = 100)
class CReadHandler
- CReadHandler();
+ static CReadHandler* GetInstance();
+ bool ReadUserCard(CLink* client, const string& textFileName);
+ bool ReadUserGoods(CLink* client, const string& textFileName);
class CWriteHandler
- CWriteHandler();
+ static CWriteHandler* GetInstance();
class CReadyNetWork
- SOCKET* mServSock
+ CReadyNetWork()
+ void Accept(SOCKET& hClientSock)
class CCard
- CardVector mCards;
- CCard();
+ static CCard* GetInstance();
+ void PushCard(const shared_ptr<Card>& card);
class CErrorHandler
- CCommandController* mCommandPtr
- CErrorHandler()
+ static CErrorHandler* GetInstance()
+ void setCommand(CCommandController* command)
+ void GetErrorCode(EnumErrorCode code, vector<string>& errorCodeStringVec)
class CLobby
- int Login(SOCKET& clientSocket, CActionNetWork& actionNetWork, vector<string>& tempUserInfo)
- int JoinMember(SOCKET& clientSocket, CActionNetWork& actionNetWork, vector<string>& tempUserInfo)
- int ChooseMenu(const char* message, SOCKET & clientSocket, CActionNetWork & actionNetWork)
- int SendMenuInfo(SOCKET & clientSocket, CActionNetWork & actionNetWork)
- int NextUserNum
+ CLobby(int NextUserNum_)
+ int ActionServiceLobby(SOCKET& clientSocket, CActionNetWork& actionNetWork, vector<string>& tempUserInfo)
struct Card
const int mCardNum
char* mName
const int mProb; // 확률
const int mStat; // 스탯
const int mGiveExp; // 이 카드를 합성하면 얻는 경험치
Card(int num, char* cardName, int prob, int stat, int giveExp)
Card(const Card& copy) = delete;
Card& operator=(const Card& copy) = delete;
CCardManager
- CGaChar mGacharHandler
+ CCardManager()
+ bool ComposeCard(CLink& targetClient, int targetCard, int sourceCard)
+ bool GacharCard(CLink & targetClient, int& resultCardNum, char*& resultCardName)
+ bool EvolutionCard(CLink& targetClient, int targetCard)
CGaChar
+ CGaChar()
class CCommandController
- CCardManager mCardManager
- CChannelHandler mChannelHandler
- CRoomHandler mRoomHandler
- CRoomManager mRoomManager
- CChannelManager mChannelManager
+ CCommandController()
+ bool CommandHandling(const shared_ptr<CLink>& shared_clientInfo, vector<string>& commandString, string& sendClientMessage, SocketVec& clientSocks)
+ CChannelHandler& GetChannelHandler()
+ CChannelManager& GetChannelManager()
+ CRoomManager& GetRoomManager()
+ bool DeleteClientSocket(CLink& clientInfo)
class CRoomHandler
+ CRoomHandler()
+ bool ExitRoom(CLink* clientInfo, CRoomManager* roomManager)
+ bool MakeRoom(const shared_ptr<CLink>& shared_clientInfo, CRoomManager* roomManager, const string& roomName,const int& battingMoney)
+ bool EnterRoom(const shared_ptr<CLink>& shared_clientInfo, CRoomManager* roomManager, int targetRoomNo)
+ bool IsAllReadyGame(CLink* clientInfo, CRoomManager* roomManager)
+ bool IsAllReadyBatting(CLink* clientInfo, CRoomManager* roomManager)
class CChannelHandler
+ CChannelHandler();
+ bool MoveNextChannel(const shared_ptr<CLink>& shared_clientInfo, CChannelManager& channelManager, int targetChannelNo);
+ bool ExitChannel(CLink& clientInfo, CChannelManager& channelManager);
class CChannelManager
- ChannelVector mChannels
- MUTEX mRAII_ChannelManagerMUTEX;
+ CChannelManager();
+ ChannelVecIt GetIterChannelBegin()
+ ChannelVecIt GetIterChannelEnd()
+ CChannel * GetMyChannel(int ChannelNum);
class CRoomManager
- RoomList mRooms
- MUTEX mRAII_RoomManagerMUTEX
+ CRoomManager()
+ void PushRoom(const shared_ptr<CRoom>& shared_newRoom)
+ RoomListIt EraseRoom(RoomListIt delRoom)
+ RoomListIt GetMyRoomIter(int ChannelNum, int roomNum)
+ RoomListIt GetIterRoomBegin()
+ RoomListIt GetIterRoomEnd()
class CChannel
- LinkList mClientInfos;
- MUTEX mRAII_ChannelMUTEX;
+ CChannel(int channelNum);
+ LinkListIt GetIterMyInfoBegin()
+ LinkListIt GetIterMyInfoEnd()
+ bool GetChannelSockets(vector<SOCKET>& channelSockets, bool isMyInclude, const SOCKET* myClientSock)
+ void PushClient(const shared_ptr<CLink>& shared_client)
+ LinkListIt EraseClient(LinkListIt myInfoListIt)
class CRoom
- LinkList mClientInfos
- MUTEX mRAII_RoomMUTEX
+ CRoom(int roomNum,int channelNum, const string& roomName,const int& battingMoney)
+ void PushClient(const shared_ptr<CLink>& shared_client)
+ LinkListIt EraseClient(LinkListIt myInfoListIt)
+ LinkListIt GetIterMyInfoBegin()
+ LinkListIt GetIterMyInfoEnd()
+ bool MergeRoom(CRoom* targetRoom)
+ bool GetRoomSockets(vector<SOCKET>& roomSockets, bool isMyInclude, const SOCKET* myClientSock = nullptr)
class CLink
- MessageStruct mMS
- CGoods mMyGoods; 나의 재화
- MyCardVector mMyCards
+ CLink(SOCKET& clientSocket,const string& strPKNumber,const char* name)
+ MessageStruct& GetMessageStruct()
+ SOCKET& GetClientSocket()
+ MyCardVectorIt GetIterMyCardBegin()
+ MyCardVectorIt GetIterMyCardEnd()
struct MessageStruct
char* message;
size_t sendRecvSize;
MessageStruct()
MessageStruct& operator=(const MessageStruct& copyMS)
MessageStruct(const MessageStruct& copyMS)
class CGoods
- Goods mGoods;
- MUTEX mRAII_GoodsMUTEX;
+ CGoods(const int& pkNumber);
+ bool SetZeroMoney(EnumErrorCode& resultcode);
+ bool AddMyMoney(const int& addMoney, EnumErrorCode& resultcode);
+ bool MinusMyMoney(int minusMoney, EnumErrorCode& resultcode);
class MyCardInfo
- const int mCardNumber; // 카드번호
- int mAmount; // 이 카드에 대한 보유 갯수
- int mExp; // 이 카드에 대한 경험치
- int mIsEvolution; // 이 카드 진화 가능?
- int mStar; // 몇 성?
+ MyCardInfo(int cardNumber, int amount = 0, int exp = 0, int isEvolution = 0, int star = 0)
struct Goods
int money;
Goods(int _money)
Goods(const Goods& copy) = delete;
Goods& operator=(const Goods& copy) = delete;