<<Java Package>>
items

<<Java Class>>
Ⓒ Item
items

- 🟥 ownerID: Integer
- 🟥 itemName: String
- 🟥 description: String
- 🟥 itemID: Integer
- 🟥 approved: boolean
- 🟥 price: Double

- 🟢 Item(Integer,String,String)
- 🟢 getOwnerID():Integer
- 🟢 getItemName():String
- 🟢 getDescription():String
- 🟢 getItemID():Integer
- 🟢 setItemID(Integer):void
- 🟢 isApproved():boolean
- 🟢 approve():void
- 🟢 setApproved(boolean):void
- 🟢 getPrice():Double
- 🟢 setPrice(Double):void
- 🟢 toString():String

<<Java Class>>
Ⓒ ItemInventory
items

- 🟢 ItemInventory(List<Item>)
- 🟢 getItemWithID(Integer):Item
- 🟢 getApprovedItems():List<Item>
- 🟢 getPendingItems():List<Item>
- 🟢 addToPending(Item):void
- 🟢 addToApproved(Item):void
- 🟢 removeFromApproved(Item):void
- 🟢 removeFromPending(Item):void
- 🟢 getOwnerID(Integer):Integer

-pendingItems
-approvedItems
0..*
0..*

<<Java Class>>
UserEnforcer
accounts

- tradeLimit: int
- incompleteLimit: int
- overBorrowLimit: int
- flaggedUsers: List<Integer>
- frozenUsers: List<Integer>
- unfrozenRequestIDs: List<Integer>

- UserEnforcer(UserManager,int,int,int)
- updateLimits(Map<String,Integer>):void
- flagUser(UserAccount):void
- unFlagUser(UserAccount):void
- getFlaggedUsers():List<Integer>
- getFrozenUsers():List<Integer>
- freezeUser(UserAccount):void
- unfreezeUser(UserAccount):void
- resetNumTrades(UserManager):void
- isFlagged(UserAccount):boolean
- getUnfreezeRequestIDs():List<Integer>
- requestUnfrozen(Integer):boolean
- addToUnfreezeRequests(Integer):void
- buildLists(UserManager):void

<<Java Class>>
Authenticator
accounts

- Authenticator()
- passwordHash(String):String
- comparePassword(String,String):boolean

<<Java Class>>
Account
accounts

- userID: Integer
- password: String

- Account(String)
- getUserID():Integer
- setUserID(Integer):void

<<Java Class>>
AdminAccount
accounts

- AdminAccount(String)

<<Java Class>>
UserAccount
accounts

- onVacation: boolean
- wishlist: List<Item>
- inventory: List<Item>
- overBorrowed: int
- numTrade: int
- incompleteTrade: int
- isFrozen: boolean
- totalFunds: Double
- city: String
- mode: String
- unfreezeRequested: boolean

- UserAccount(String)
- setMode(String):void
- getWishlist():List<Item>
- setWishlist(List<Item>):void
- addToWishlist(Item):void
- removeFromWishlist(Item):void
- removeFromWishlist(Integer):void
- addToInventory(Item):void
- removeFromInventory(Item):void
- removeFromInventory(Integer):void
- getInventory():List<Item>
- setInventory(List<Item>):void
- ownsItem(Item):boolean
- wantsItem(Item):boolean
- getOverBorrowed():int
- setOverBorrowed(int):void
- getNumTrade():int
- setNumTrade(int):void
- getIncompleteTrade():int
- setIncompleteTrade(int):void
- isFrozen():boolean
- setFrozen(Boolean):void
- getTotalFunds():Double
- setTotalFunds(Double):void
- getCity():String
- setCity(String):void
- userOnVacation(UserAccount):void
- userNotOnVacation(UserAccount):void
- isOnVacation():boolean
- setOnVacation(boolean):void
- setUnfreezeRequested(boolean):void
- getUnfreezeRequested():boolean

<<Java Class>>
UserManager
accounts

- UserManager()
- UserManager(Map<Integer,UserAccount>)
- getAllUsers():List<UserAccount>
- addUser(UserAccount):void
- removeUser(UserAccount):void
- getUser(Integer):UserAccount
- getAllUserOnVacation():Map<Integer,UserAccount>
- removeUserOnVacation(UserAccount):void
- incrementNumTrade(Integer):void
- changeOverborrowed(Integer,int):void
- getHomeCity(Integer):String
- generateDemoUserID():Integer

-userManager
0..1

+allUsers
allUserOnVacation
0..*

<<Java Class>> TradeLogManager
trades

- ◾ statusDictionary: Map<String,List<Trade>>
- ◾ userDictionary: Map<Integer,List<Trade>>

- ● TradeLogManager()
- ● TradeLogManager(TradeLog)
- ◾ buildStatusDictionary(TradeLog):Map<String,List<Trade>>
- ◾ getStringListMap():Map<String,List<Trade>>
- ◾ buildUserDictionary(TradeLog):Map<Integer,List<Trade>>
- ● getUserTrades(Integer):List<Trade>
- ● getUserTradesByStatus(Integer,String):List<Trade>
- ● addTrade(Integer,Trade):void
- ● removeTrade(Trade):void
- ● updateStatus(Trade,String):void
- ● getTradeWithID(Integer):Trade
- ◾ findCommonHelper(Map<Integer,Integer>):List<Integer>
- ● getTopTradingPartners(UserAccount):List<Integer>
- ◾ User1IdHelper(Map<Integer,Integer>,List<Integer>):void
- ◾ User2IdHelper(Map<Integer,Integer>,Integer):void

<<Java Class>> TradeLog
trades

-tradeLog
0..1

- ● TradeLog()
- ● TradeLog(Map<Integer,Trade>)
- ● getTradeHistory():Map<Integer,Trade>
- ● addTrade(Integer,Trade):void
- ● removeTrade(Trade):void

<<Java Class>> TradeManager
trades

-tradeLogManager
0..1

- ● TradeManager(TradeLogManager)
- ● updateStatus(Trade,String):void
- ● createTrade(int,Map<Integer,List<Integer>>,boolean):Trade
- ● acceptTrade(Trade):void
- ● declineTrade(Trade):void
- ● confirmTrade(Trade):boolean
- ● completeTrade(Trade):void
- ● getRecentTrades(UserAccount):List<Trade>
- ● incompleteTrade(Trade):void

<<Java Class>> Trade
trades

- ◾ tradeID: Integer
- ◾ isPermanent: boolean
- ◾ status: String
- ◾ receiverMap: Map<Integer,List<Integer>>
- ◾ acceptedMap: Map<Integer,Integer>
- ◾ requesterID: Integer

-tradeHistory
0..*

- ● Trade(Integer,Map<Integer,List<Integer>>)
- ● getTradeID():Integer
- ● setTradeID(Integer):void
- ● getIsPermanent():boolean
- ● setIsPermanent(boolean):void
- ● getStatus():String
- ● setStatus(String):void
- ● getReceiverMap():Map<Integer,List<Integer>>
- ● getRequesterID():Integer
- ● getMeeting():Meeting
- ● setMeeting(Meeting):void
- ● getTraderIDs():List<Integer>
- ● getAllItems():List<Integer>
- ● getReceivedItems(Integer):List<Integer>
- ● getReceiver(Integer):Integer
- ● toString():String
- ● acceptTrade(Integer):void
- ● acceptedByAllUsers():boolean
- ● acceptedByReceivers():boolean

<<Java Class>> Meeting
trades

- ◾ lastEditorID: Integer
- ◾ numEdits: Integer
- ◾ time: LocalDateTime
- ◾ location: String
- ◾ status: String

-meeting
0..1

- ● Meeting(int,LocalDateTime,String,String)
- ● getLastEditorID():int
- ● setLastEditorID(Integer):void
- ● getNumEdits():Integer
- ● setNumEdits(Integer):void
- ● increaseNumEdits():void
- ● getLocation():String
- ● setLocation(String):void
- ● getTime():LocalDateTime
- ● setTime(LocalDateTime):void
- ● getStatus():String
- ● setStatus(String):void
- ● toString():String

<<Java Package>>
database

<<Java Class>>
DatabaseDriver
database

- database: String

- DatabaseDriver(String)
- connectOrCreateDataBase():Connection
- initialize(DatabaseColumnsInterface,Connection):Connection
- addTables(Map<String,List<String>>,Connection):boolean
- tableParseHelper(Map<String,List<String>>,Statement):void

<<Java Class>>
DataDeleter
database

- database: String

- DataDeleter(String)
- removeItemFromWishlist(Connection,int,int,boolean):void
- removeTradedItems(Connection,int):void
- removeData(int,int,int):void
- removeItemsTraded(Connection,int):void
- removeTrade(Connection,int):void
- removeItemFromInventory(Connection,int,int):void

<<Java Class>>
TablesBuilder
database

- tablesList: Map<String,List<String>>

- TablesBuilder()
- setUserColumns():void
- setAccountColumns():void
- setTradeColumns():void
- setPasswordColumns():void
- setItemsColumns():void
- setItemTradedColumns():void
- setWishlistColumns():void
- createTablesMap():void
- addTableMapToDatabase():Map<String,List<String>>

<<Java Class>>
DataReader
database

- database: String

- DataReader(String)
- getMeetingInfo(int):Map<String,String>
- getIsAdmin(int):boolean
- getReceivingUsers(int):Map<Integer,List<Integer>>
- getOwnerIDOfItem(int):int
- getOwnerFunds(int):double
- getItemPrice(int):double
- getIsFrozen(int):boolean
- getIsFlagged(int):boolean
- getTradeThresholds():Map<String,Integer>
- findPassword(int):String
- getAdminAccount(int):Map<String,Object>
- getUser(int):Map<String,Object>
- getIsOnVacation(int):boolean
- getItem(int):Map<String,Object>
- getUserWishlist(int):List<Integer>
- getUserInventory(int):List<Integer>
- getTrade(int):Map<String,Object>
- getAllUserIDs():List<Integer>
- getAllTradeIDs():List<Integer>
- getAllAdminIDs():List<Integer>
- getOriginalOwnership(Connection,int):Map<Integer,List<Integer>>
- getRecieversMap(Connection,int):Map<Integer,List<Integer>>
- getTradedItems(int):Map<String,Object>
- addData(ResultSet):Map<String,Object>
- closeResources(Connection,ResultSet,Statement):void
- closeResources(ResultSet,Statement):void

<<Java Class>>
DataInserter
database

- database: String

- DataInserter(String)
- addItemToUserWishlist(int,int):void
- insertTrade(Map<Integer,List<Integer>>,Map<String,Object>):int
- insertItem(Map<String,Object>,boolean):int
- insertUser(String,Map<String,Object>):int
- insertAdmin(String):int
- writeTradeThresholds(Map<String,Integer>):void
- storePassword(Connection,int,String):void
- insertAccountInDB(Connection,int):int
- insertUserInDB(Connection,int,Map<String,Object>):int
- insertItemInDB(Connection,Map<String,Object>,int):int
- insertTradedItemsInDB(Connection,Map<Integer,List<Integer>>,int):void
- createTradeStatement(Connection,String,Map<String,Object>):PreparedStatement
- insertTradeInDB(Connection,Map<String,Object>):int

<<Java Class>>
DataUpdater
database

- database: String

- DataUpdater(String)
- updateTradeMap(int,int,int,int):void
- updateTrade(int,int,String):void
- checkTradeCompleted(Connection,int,int,String):void
- approveItem(int):void
- updateNumTrades(int,int):void
- changeFreezeStatus(int):void
- updateUserFund(Connection,int,double):void
- updateItemPrice(int,double):void
- updateUserFunds(int,Double):void
- checkColumnUpdated(int):void
- setExpectedReturnData(Connection,int):void
- getColumns(ResultSet):Map<Integer,String>
- updateUserTradeInfo(Connection,int,int):void
- updateItemInventories(Connection,int,int):void
- updateOwnerID(Connection,int,int):void
- getSecondMeetingMap(Map<Integer,List<Integer>>,Map<Integer,List<Integer>>):Map<Integer,List<Integer>>
- updateItemsTradedSecondMeeting(Connection,int,Map<Integer,List<Integer>>,Map<Integer,List<Integer>>):void
- updateUserFundsAfterTrade(Connection,int,int,int):void
- changeVacationStatus(int):void
- changeUnfreezeRequest(int,int):void
- completeTrade(Connection,int,int):void
- incrementUserIncomplete(Connection,int):void
- incompleteTrade(Connection,int):void
- closeResources(ResultSet,Statement):void
- closeResources(Connection,PreparedStatement,ResultSet):void
- closeResources(PreparedStatement,ResultSet,Statement):void
- closeResources(Connection,PreparedStatement):void
- closeResultSet(ResultSet):void
- closeConnection(Connection):void
- closeStatement(Statement):void
- closePreparedStatement(PreparedStatement):void

<<Java Interface>>
DatabaseColumnsInterface
database

- tablesList: Map<String,List<String>>

- createTablesMap():void
- addTableMapToDatabase():Map<String,List<String>>

<<Java Class>>
UserManagerBuilder
controllers

- dataReader: DataReader
- mode: String

- UserManagerBuilder(String)
- createItem(Map<String,Object>):Item
- getWishlist(Integer):List<Item>
- getInventory(Integer):List<Item>
- userObjHelper(Map<String,Object>,Integer,UserAccount):void
- createUserObject(Map<String,Object>):UserAccount
- getUserAccount(int):UserAccount
- createUserManager():UserManager

<<Java Class>>
LoginSignUpController
controllers

- LoginSignUpController()
- login(int,String):boolean
- demoAccountLogin(int,String):boolean
- isAdmin(int,String):boolean
- signUpHelper(String,String,Map<String,Object>):void
- signUpDemoHelper(String,String,String,UserManager,DataInserter,Map<String,Object>):int
- signUpUser(String,String,String,UserManager):Integer
- signUpDemoUser(String,String,String,UserManager):Integer
- signUpAdmin(String):Integer

<<Java Class>>
TradeLogBuilder
controllers

- mode: String
- dataReader: DataReader

- TradeLogBuilder(String)
- createFirstMeeting(Map<String,Object>):Meeting
- createSecondMeeting(Map<String,Object>):Meeting
- createMeeting(Map<String,Object>,List<String>):Meeting
- tradeModified(UserAccount,Trade):boolean
- createTrade(Map<String,Object>,DataReader):Trade
- buildTradeLog():TradeLog

<<Java Class>>
MainController
controllers

- currentUser: UserAccount
- tradeManager: TradeManager
- userManager: UserManager
- tradeLogManager: TradeLogManager
- userEnforcer: UserEnforcer
- itemInventory: ItemInventory
- dataReader: DataReader
- mode: String

- MainController(String,String)
- initAfterLogin(String,boolean):void
- creationHelper(String):void
- createTradeLogManager():TradeLogManager
- createUserManager():UserManager
- createItemInventory(UserManager):ItemInventory
- getMeetingEditLimit():int
- createTradeManager(TradeLogManager):TradeManager
- createUserEnforcer(UserManager):UserEnforcer
- firstTimeRunHelper(DatabaseDriver,Connection):void
- firstTimeRun():void
- getItemInventory():ItemInventory
- getUserManager():UserManager
- getTradeManager():TradeManager
- getAdminController():AdminController
- getAdminUndoController():AdminUndoable
- getUserEnforcer():UserEnforcer
- getTradeLogManager():TradeLogManager
- getCurrentUserAccount():UserAccount
- getTradeController():TradeController
- getMode():String
- getUserMenuController():UserMenuController

<<Java Class>>
UserMenuController
controllers

- dataDeleter: DataDeleter
- dataInserter: DataInserter
- dataReader: DataReader
- currentUser: UserAccount
- dataUpdater: DataUpdater
- mode: String

- UserMenuController(UserAccount,String)
- returnItemInfoMap(String,String,double):Map<String,Object>
- requestAddToMyInventory(String,String,double,ItemInventory):void
- removeItemHelper(Integer,Item,UserManager):void
- removeFromItemInv(Integer,ItemInventory,UserManager):void
- addToMyWishList(Integer,ItemInventory):void
- removeFromWishList(Integer,ItemInventory):void
- getRecentTrades(TradeManager):List<Trade>
- getFrequentTraders(TradeLogManager):List<Integer>
- requestToUnfreeze(UserEnforcer):boolean
- goOnVacation(UserAccount):void
- changeVacationStatus(Integer):void
- getVacationStatus(int):boolean
- getUserFunds(int):Double

<<Java Class>>
TradeController
controllers

- currentUser: UserAccount
- tradeManager: TradeManager
- tradeLogManager: TradeLogManager
- itemInventory: ItemInventory
- userManager: UserManager
- dataDeleter: DataDeleter
- dataInserter: DataInserter
- dataUpdater: DataUpdater
- dataReader: DataReader
- meetingEditLimit: int
- mode: String

- TradeController(UserAccount,TradeManager,TradeLogManager,ItemInventory,UserManager,int,String)
- requestOneWay(Integer,Integer,boolean):boolean
- requestTwoWay(Integer,Integer,Integer,boolean):boolean
- acceptTradeForCurrentUser(Integer,Trade):boolean
- declineTrade(Trade):boolean
- confirmTrade(Trade):boolean
- completeTrade(Trade):boolean
- incompleteTrade(Trade):boolean
- getTraders(Trade):List<UserAccount>
- getOneWayMap(Integer,Integer):Map<Integer,List<Integer>>
- getTwoWayMap(Integer,Integer,Integer):Map<Integer,List<Integer>>
- requestTrade(Integer,Map<Integer,List<Integer>>,boolean):boolean
- reqHelper(Integer,Map<Integer,List<Integer>>):void
- checkThreeWayTradeFunds(Map<Integer,List<Integer>>):boolean
- updateTradeStatusFile(Trade,String):void
- mapTradeData(Trade):Map<String,Object>
- getOwnerID(Integer):Integer
- getGivenItems(Trade,UserAccount):List<Integer>
- incrementUserVariables(Trade):void
- getItems(Trade):List<Item>
- validateItems(Trade):boolean
- validateUsers(Trade):boolean
- validateTrade(Trade):boolean
- deleteItems(Trade):boolean
- deleteItemsFromFile(Trade):boolean
- getMeetingController():MeetingController
- getTradePresenter():TradePresenter

<<Java Class>>
AdminController
controllers

- userManager: UserManager
- itemInventory: ItemInventory
- userEnforcer: UserEnforcer
- mode: String
- dataUpdater: DataUpdater
- dataReader: DataReader
- dataDeleter: DataDeleter
- dataInserter: DataInserter
- mainFrame: Component

- AdminController(UserAccount,UserManager,ItemInventory,UserEnforcer,String)
- limitsHelper(String,String,String,Map<String,Integer>):void
- changeAllLimits(String,String,String,String):boolean
- displayLimits():String
- getItemInventory():ItemInventory
- approveItem(Item):boolean
- denyItem(Item):Boolean
- getPendingItems():boolean
- changeFreezeStatus(UserAccount):void
- checkFreezeStatus(UserAccount):boolean
- getFrozenUsers():List<UserAccount>
- checkFlaggedStatus(UserAccount):boolean
- getUserName(UserAccount):String
- getOverBorrowedNum(UserAccount):Integer
- freezeUser(UserAccount):void
- unfreezeUser(UserAccount):void
- resetWeeklyTrades():void
- getUnfreezeRequests():List<Integer>

<<Java Class>>
MeetingController
controllers

- dataUpdater: DataUpdater
- dataReader: DataReader
- mode: String

- MeetingController(String)
- allMeetingsComplete(Trade):boolean
- firstMeetingComplete(Integer):boolean
- secondMeetingComplete(Integer):boolean
- editMeeting(int,Integer,Meeting,String,LocalDateTime):void
- storeReturnMeeting(Integer,Meeting,String):void
- getMeeting(int):Meeting
- getReturnMeeting(int):Meeting
- acceptMeeting(Integer,Meeting):boolean
- getTradePermanent(Integer):Boolean
- completeFirstMeeting(Integer,Integer):boolean
- completeReturnMeeting(Integer,Integer):boolean
- incompleteMeeting(Trade):void
- mapSecondMeeting(Meeting):Map<String,Object>
- getTime(int):String

<<Java Class>>
TradePresenter
controllers

- TradePresenter()
- getAvailableItems(UserAccount,ItemInventory,UserManager):List<Item>
- getItemToRequest(Integer,Integer,ItemInventory):Item
- getItemsToOffer(UserAccount,Integer,ItemInventory,UserManager):List<Item>
- getHomeCityUsers(UserAccount,UserManager):List<UserAccount>
- thirdUserHelper(UserAccount,UserAccount,Map<Integer,List<Integer>>,UserAccount,Item,List<Item>):void
- findThreeWayTradeMap(UserAccount,UserAccount,Integer,ItemInventory,UserManager):Map<Integer,List<Integer>>
- formatThreeWayTrade(Integer,Map<Integer,List<Integer>>,ItemInventory):String
- getStatusTrades(List<Integer,String,TradeLogManager>):List<Trade>
- getStatusTrade(Integer,Integer,String,TradeLogManager):Trade
- getFundsChange(Integer,Trade,ItemInventory):double
- getFundsChange(Integer,Map<Integer,List<Integer>>,ItemInventory):double

<<Java Class>>
AdminUndoController
controllers

- NUMRECENTTRADES: int

- AdminUndoController(UserAccount,UserManager,ItemInventory,UserEnforcer,String)
- undoLastTrade(TradeLogManager):boolean
- undoRecentTrades(TradeLogManager):boolean
- undoRecentTradeHelper(int,List<Integer>,TradeLogManager):boolean
- undoGivenTrade(int,TradeLogManager):boolean
- undoRecentTrade(int,TradeLogManager):boolean
- removeItemFromInventory(Integer):boolean
- removeItemFromWishlist(Integer,Integer):boolean

<<Java Interface>>
AdminUndoable
controllers

- undoLastTrade(TradeLogManager):boolean
- undoRecentTrades(TradeLogManager):boolean
- undoGivenTrade(int,TradeLogManager):boolean
- removeItemFromWishlist(Integer,Integer):boolean
- removeItemFromInventory(int):boolean

-tradeController 0..1
-adminController 0..1
-userMenuController 0..1
-tradePresenter 0..1
-meetingController 0..1
-adminUndoController 0..1