Team Polandball

Banking System Phase IV - Backend Plan

Course Code: CSCI 3060/SOFE 3980U - Software Quality Assurance

Course Instructor: Dr. Jeremy Bradbury

Members:

Dominick Mancini - **100517944**

Scott McLean - **100379538**

Janahan Nirmalan - **100459567**

Tutorial CRN: **73386 - Tuesday, 2:10-3:30**

Report Due Date: **March 18th, 2016**

UML Diagram

The UML diagram for the backend has been enclosed in the project (backendUML.pdf).

Classes and Methods

CLASS	METHODS
Account	+ Account(String accountLine) : void + Account(String acctNum, String acctName) : void + getName() : String + getNumber() : String + getStatus() : char + getBalance() : double + getNumTransactions() : int + getPlan() : char + increaseBalance(double addAmt) : void + decreaseBalance(double subAmt) : void + isValid(double subAmt) : boolean + incrementTransactionCount() : void + setBalance(double newBalance) : void + newAcctAsMasterString() : String + newAcctAsCurrentString() : String + currentAcctAsString() : String + setSurcharge(char plan) : void + getSurcharge() : double + setPlan(char plan) : void + setStatus(char newStatus) : void
TransactionProcessor	+ withdrawal(Transaction t, Account a) : boolean + transfer(Transaction t, Account a) : boolean + paybill(Transaction t, Account a) : boolean + deposit(Transaction t, Account a) : boolean + create(Transaction t, ArrayList<Account> a) : boolean + delete(Transaction t, Account a, ArrayList<Account> allAccts) : boolean + disable(Transaction t, Account a) : boolean + changeplan(Transaction t, Account a) : boolean + enable(Transaction t, Account a) : boolean
Transaction	+ Transaction(String transactionRecord) : void + getTransactionNumber() : String + getName() : String + getAccountNumber() : String + getMisc() : String + getFundsInvolved() : double + transAsString() : String
Main	+ main(String[] args) : void
Utilities	+ getIndividualTransactionFiles() : File[] + mergeFiles(File[] files, String name) : File + tryDelete(Path pathToFile) : void + padSpaceRight(String s, int n) : String + padSpaceLeft(String s, int n) : String + padZeroesLeft(String num, int size) : String + getHighestAcctNum(ArrayList<Account> arrayOfAccounts) : int + createNewAccountNumber(ArrayList<Account> existingAccounts) : String + getAccountByNumber(String acctNum, ArrayList<Account> arrayOfAccounts) : Account + processTransaction(Transaction t, ArrayList<Account> accounts) : boolean

The backend is divided into five main files: Account, Main, Transaction, TransactionProcessor, and Utilities. These are also the five classes which are used for the backend. The main class is the driver for the backend. The Account and Transaction classes instantiate objects. These objects represent a record in the accounts file and a transaction in the transactions file respectively.

The TransactionProcessor class is a static class which is used by the Utilities class. The TransactionProcessor class contains the front-end methods, and will modify the state of account objects, and is called by the Utilities class.

The Utilities class is a static class which is called from both the main class and the TransactionProcessor class. The Utilities class contains methods which process Transactions and Accounts objects (such as padding data with whitespace).

The Transaction class instantiates Transaction objects, which hold the Transaction record data. It contains only accessor methods to return the data stored in the Transaction object.

The Main class takes one argument which is the path to the master accounts file. It will write out a new transaction log file, error log file, new_master_accounts, and current_accounts file.

Method Descriptions

Class	Method	Description
Account	**+ Account(String accountLine) : void**	**Constructor at initialization**
	+ Account(String acctNum, String acctName) : void	**Constructor for create command**
	+ getName() : String	**returns name of holder**
	+ getNumber() : String	**returns number of**

		holder
	+ getStatus() : char	returns status of account
	+ getBalance() : double	returns balance of account
	+ getNumTransactions() : int	returns number of account transactions
	+ getPlan() : char	returns plan of account
	+ increaseBalance(double addAmt) : void	Increases account balance
	+ decreaseBalance(double subAmt) : void	Decreases account balance
	+ isValid(double subAmt) : boolean	checks account validity
	+ incrementTransactionCount() : void	increments number of transactions on account
	+ setBalance(double newBalance) : void	sets new balance
	+ newAcctAsMasterString() : String	returns master account string
	+ newAcctAsCurrentString() : String	returns current account string
	+ currentAcctAsString() : String	returns current account string (old accounts file)
	+ setSurcharge(char plan) : void	returns surcharge of account
	+ getSurcharge() : double	sets surcharge of account
	+ setPlan(char plan) : void	sets plan of account

	+ setStatus(char newStatus) : void	sets whether account is active or disabled
TransactionProcessor	+ withdrawal(Transaction t, Account a) : boolean	Conducts withdrawal
	+ transfer(Transaction t, Account a) : boolean	Conducts transfer
	+ paybill(Transaction t, Account a) : boolean	Conducts paybill
	+ deposit(Transaction t, Account a) : boolean	Conducts deposit
	+ create(Transaction t, ArrayList<Account> a) : boolean	Conducts account creation
	+ delete(Transaction t, Account a, ArrayList<Account> allAccts) : boolean	Conducts account deletion
	+ disable(Transaction t, Account a) : boolean	Disables an account
	+ changeplan(Transaction t, Account a) : boolean	Changes an account plan
	+ enable(Transaction t, Account a) : boolean	Enables account
Transaction	+ Transaction(String transactionRecord) : void	Constructor for transaction object
	+ getTransactionNumber() : String	Returns transaction number
	+ getName() : String	Returns the holder name of the transaction

	+ getAccountNumber() : String	Gets the account number the transaction pertains to
	+ getMisc() : String	Get miscellaneous info about the transaction
	+ getFundsInvolved() : double	Gets transaction funds involved
	+ transAsString() : String	Returns the transaction as a string
Main	+ main(String[] args) : void	Initializes and drives the entire program until termination
Utilities	+ getIndividualTransactionFiles() : File[]	Returns the individual transaction files in an array
	+ mergeFiles(File[] files, String name) : File	Merges the individual transaction files into one file which it returns
	+ tryDelete(Path pathToFile) : void	Attempt to delete a file with file path pathToFile

	+ padSpaceRight(String s, int n) : String	**Add n spaces onto the right side of string s**
	+ padSpaceLeft(String s, int n) : String	**Add n spaces onto the left side of string s**
	+ padZeroesLeft(String num, int size) : String	**Add size zeroes onto the left side of string num**
	+ getHighestAcctNum(ArrayList<Account> arrayOfAccounts) : int	**Gets the highest valued account number in the list of all accounts**
	+ createNewAccountNumber(ArrayList<Account> existingAccounts) : String	**Creates a new (unused) account number**
	+ getAccountByNumber(String acctNum, ArrayList<Account> arrayOfAccounts) : Account	**Gets an account by its number**
	+ processTransaction(Transaction t, ArrayList<Account> accounts) : boolean	**Processes a specified transaction**