

AccountType
credit
line
chequing
saving
moneyMarket
jointChequing
USDCheking
AccountType(String, int)
getType(String)
AccountType
constructAccount(String[], Customer)
Account
newAccount(String, String)
Account
addAccount(Customer, int[])
Account
name
String
index
int

Accounts
Accounts
Accounts()
getAccounts()
Accounts
addAccount(Account)
void
constructAccount(String[], Customer)
void
accountFinder(String)
Account
updateTransaction()
void
depositAll()
void
rewriteWholeTransactions()
void
loadInterests()
void
accountsList
List<Account>

Transaction
money
double
receivingAccount
Account
sendingAccount
Account
transactionTime
Date
Transaction(double, Account, Account, Date)
Transaction(double, Account, boolean, Date)
undo()
void
getMoney()
double
toString()
String
show()
String

	FinancialProduct	
	◦ period	int
	◦ principle	double
	◦ dateOfMature	Date
	◦ owner	Customer
	◦ productType	FinancialProductType
	◦ FinancialProduct(double, int, String)	
	◦ makePayment()	void
	◦ writeFile()	void
	◦ reWriteWholeFile()	void
	◦ maturityDate	Date
	◦ nextDateOfPayment	Date
	◦ dateOfStart	Date

	LoanProduct	
	◦ loanRateSaver	LoanRateSaver
	◦ monthlyPayment	double
	◦ LoanProduct(double, int, String)	
	◦ makePayment()	void
	◦ setPaymentDate(Date)	Date
	◦ getCurrentTerm()	int
	◦ writeFile()	void
	◦ toString()	String
	◦ outstandingBalance	double

	Bond	
	◦ bondRateSaver	BondRateSaver
	◦ Bond(double, int, String)	
	◦ makePayment()	void
	◦ setPaymentDate(Date)	Date
	◦ writeFile()	void
	◦ toString()	String

	GIC	
	◦ gicRateSaver	GICRateSaver
	◦ GIC(double, int, String)	
	◦ makePayment()	void
	◦ writeFile()	void
	◦ toString()	String

	RateSaver	
	◦ computePayment(double)	double

	BondRateSaver	
	◦ semiInterest	double
	◦ semiCoupon	double
	◦ period	int
	◦ BondRateSaver(int)	
	◦ computePayment(double)	double
	◦ computePrice(double)	double

	GICRateSaver	
	◦ interest	double
	◦ period	int
	◦ GICRateSaver(int)	
	◦ computePayment(double)	double

	LoanRateSaver	
	◦ interest	double
	◦ term	int
	◦ monthlyPayment	double
	◦ LoanRateSaver(int)	
	◦ getInterest()	double
	◦ setTerm(int)	void
	◦ computePayment(double)	double
	◦ computePrincipleRepaid(int, double)	double



	FinancialProductType	
	bond	
	gic	
	loan	
	◦ FinancialProductType(String, int)	
	◦ getType(String)	FinancialProductType
	◦ newFinancialProduct(double, int, String)	FinancialProduct
	◦ addFinancialProduct(Customer, double, int)	void
	◦ name	String
	◦ index	int



Users		
customers	List<Customer>	
financialAdvisers	List<FinancialAdviser>	
Users	Users	
Users()		
getUsers()	Users	
constructManager()	void	
constructUser(ArrayList<String>)	Customer	
constructFinancialUser(ArrayList<String>)	FinancialUser	
constructHelper(ArrayList<String>, Customer)	Customer	
addCustomer(Customer)	void	
addFinancialAdviser(FinancialAdviser)	void	
addFinancialProduct(FinancialProduct)	void	
loadCustomer()	void	
loadFinancialAdviser()	void	
loadFinancialProduct(String)	void	
loadGICProduct(double, int, String, Date)	void	
loadBondProduct(double, int, String, Date, Date)	void	
loadLoanProduct(double, int, String, Date, Date, double)	void	
customerFinder(String)	Customer	
financialAdviserFinder(String)	FinancialAdviser	
loadFile()	void	
customers	ArrayList<Customer>	
financialProducts	List<FinancialProduct>	
manager	Manager	
financialAdvisers	ArrayList<FinancialAdviser>	

ATM		
cash5	int	
cash10	int	
cash20	int	
cash50	int	
total	int	
date	Date	
format	SimpleDateFormat	
ATM	ATM	
ATM()		
getATM()	ATM	
checkATMBalance()	boolean	
displayAlert()	String	
writeFile()	void	
withdrawCash(int)	int[]	
withdraw(int)	void	
deposit(int, int, int, int)	void	
loadFile()	void	
CADtoUSDRate	double	



Simulator		
simulator	Simulator	
Simulator()		
getSimulator()	Simulator	
constructManager(String, String)	void	
setSystemDate()	void	
initialize()	void	
loginPass(String, String)	Person	
LoginPassHelper(Person, String)	Person	
updateDate()	void	
recordInterests()	void	



## ReadFile

  readFile(String) ArrayList<String>

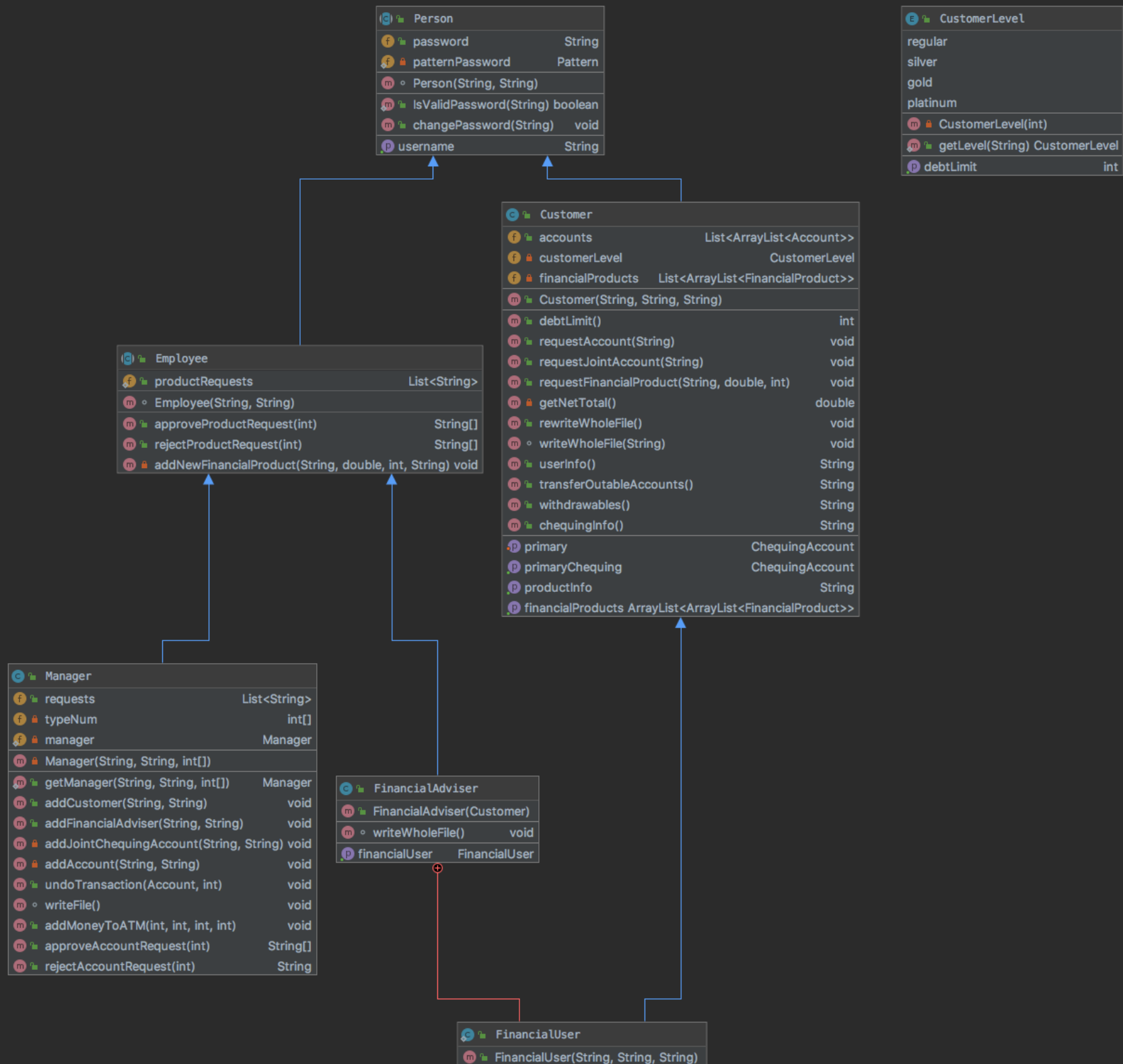
  readFolder(String) ArrayList<ArrayList<String>>

## WriteFile

  write(String, String) void

  clearInfo(String) void

Powered by yFiles



CustomerMenu		
user	Customer	
viewAccountsInfo(ActionEvent)	void	
payBill(ActionEvent)	void	
makeTransaction(ActionEvent)	void	
requestAccount(ActionEvent)	void	
withdraw(ActionEvent)	void	
requestProduct(ActionEvent)	void	
changePassword(ActionEvent)	void	
setPrimaryAccount(ActionEvent)	void	
deposit(ActionEvent)	void	
logout(ActionEvent)	void	

ManagerClass		
menuScene	Scene	
manager	Manager	
createCustomerOnClick(ActionEvent)	void	
viewAccountRequestsOnClick(ActionEvent)	void	
undoTransactionsOnClick(ActionEvent)	void	
logoutOnClick(ActionEvent)	void	
loadMoneyOnClick(ActionEvent)	void	
viewFPRequestsOnClick(ActionEvent)	void	
shutDownOnClick(ActionEvent)	void	
changePasswordOnClick(ActionEvent)	void	
createFAOnClick(ActionEvent)	void	

TransferMoney		
user	Customer	
accountsInfo	TextFlow	
outAccounts	ChoiceBox<String>	
inAccount	TextField	
transferAmount	TextField	
payeeName	TextField	
initialize(URL, ResourceBundle)	void	
setOutAccounts()	void	
transfer(ActionEvent)	void	
pay(ActionEvent)	void	
back(ActionEvent)	void	

ViewAccountRequests		
requestsDisplay	TextFlow	
requestNumberInput	ChoiceBox<Integer>	
requests	List<String>	
invalidChoose()	void	
acceptOnClick(ActionEvent)	void	
rejectOnClick(ActionEvent)	void	
backOnClick(ActionEvent)	void	
loadRequests()	void	
initialize(URL, ResourceBundle)	void	
displayRequests()	void	

FinancialAdviserMenu		
employee	FinancialAdviser	
viewFPRequestsOnClick(ActionEvent)	void	

Deposit		
transferAmount	TextField	
cash5	TextField	
cash20	TextField	
cash10	TextField	
cash50	TextField	
user	Customer	
initialize(URL, ResourceBundle)	void	
depositCheque(ActionEvent)	void	
depositCash(ActionEvent)	void	
back(ActionEvent)	void	

ATMInterface		
window	Stage	
user	Person	
main(String[])	void	
start(Stage)	void	
invalidNumberAlert()	void	
produceErrorAlert(String, String)	void	
produceConfirmationAlert(String)	void	
produceInformationAlert(String, String)	void	
playMusic(String)	void	

ViewFinancialProductRequests		
requestNumberInput	ChoiceBox<Integer>	
requestsDisplay	TextFlow	
requests	List<String>	
approveOnClick(ActionEvent)	void	
rejectOnClick(ActionEvent)	void	
backOnClick(ActionEvent)	void	
loadRequests()	void	
initialize(URL, ResourceBundle)	void	
displayRequests()	void	

ChangePassword		
oldPasswordInput	PasswordField	
newPasswordInput	PasswordField	
newPasswordAgainInput	PasswordField	
incorrectPasswordAlert()	void	
passwordNotMatchAlert()	void	
passwordNotStrongAlert()	void	
confirmOnClick(ActionEvent)	void	
backOnClick(ActionEvent)	void	

Withdraw		
user	Customer	
transferAmount	TextField	
withdrawInfo	TextFlow	
withdrawAccounts	ChoiceBox<String>	
initialize(URL, ResourceBundle)	void	
setWithdrawAccounts()	void	
withdrawMoney()	void	
back(ActionEvent)	void	

SetPrimaryAccount		
user	Customer	
chequingInfo	TextFlow	
chequingAccounts	ChoiceBox<String>	
initialize(URL, ResourceBundle)	void	
setChequingAccounts()	void	
back(ActionEvent)	void	
primary	ActionEvent	

LoadMoney		
bill5Input	TextField	
bill10Input	TextField	
bill20Input	TextField	
bill50Input	TextField	
loadMoneyOnClick(ActionEvent)	void	
backOnClick(ActionEvent)	void	
viewAlertOnClick(ActionEvent)	void	

LoginClass		
dateDisplay	TextFlow	
usernameInput	TextField	
passwordInput	PasswordField	
loginButtonOnClick(ActionEvent)	void	
loginFailedAlert()	void	
initialize(URL, ResourceBundle)	void	

ApplyProduct		
user	Customer	
productType	ChoiceBox<String>	
initialize(URL, ResourceBundle)	void	
setProductType()	void	
applyProduct(ActionEvent)	void	
back(ActionEvent)	void	

ApplyAccount		
user	Customer	
accountType	ChoiceBox<String>	
initialize(URL, ResourceBundle)	void	
setAccountType()	void	
applyAccount(ActionEvent)	void	
back(ActionEvent)	void	

SwitchScene		
switchScene	SwitchScene	
SwitchScene()		
getSwitchScene()	SwitchScene	
switchScene(String)	void	
backToMenu()	void	

UndoTransactions		
accountNumberInput	TextField	
numberOfTransactionInput	ChoiceBox<Integer>	
backOnClick(ActionEvent)	void	
confirmOnClick(ActionEvent)	void	
initialize(URL, ResourceBundle)	void	

FACreation		
usernameInput	TextField	
levelInput	ChoiceBox<String>	
FACreationConfirmOnClick(ActionEvent)	void	
FACreationBackOnClick(ActionEvent)	void	
initialize(URL, ResourceBundle)	void	

CustomerCreation		
levelInput	ChoiceBox<String>	
customerUsernameInput	TextField	
initialize(URL, ResourceBundle)	void	
backOnClick(ActionEvent)	void	
createCustomerOnClick(ActionEvent)	void	

CustomerInfoDisplay		
information	TextFlow	
productInfo	TextFlow	
initialize(URL, ResourceBundle)	void	
back(ActionEvent)	void	

InitialInterface		
dateInput	DatePicker	
usernameInput	TextField	
passwordInput	PasswordField	
RegisterOnClick(ActionEvent)	void	