«interface» MoneyService

+buyMoney(Order orderData): boolean
+sellMoney(Order orderData): boolean
+printSiteReport(String destination): void
+shutDownService(String destination): void
~getCurrencyMap(): Map<String, Currency>
~getAvailableAmount(String currencyCode): Optional<Double>

MoneyServiceAPP

-logger: Logger
-fh: FileHandler

+main(String[] args): void
+clientMenu(ExchangeSite theSite): void
+mainMenuCLI(ExchangeSite theSite): void
-createOrder(ExchangeSite theSite, Scanner keyboard): void
+test(): Consumer<LocalDate>

Transaction

-serialVersionUID = 1L: long
-id: int
-timeStamp: LocalDateTime
-currencyCode: String
-amount: int
-mode: TransactionMode
-uniqueId = 0: int

+Transaction(Order order): ctor
+Transaction(Order order, LocalDateTime ref): ctor
+getUniqueId(): int
+setUniqueId(int newuniqueId): void
+getId(): int
+getTimeStamp(): LocalDateTime
+getCurrencyCode(): String
+getAmount(): int
+getMode(): TransactionMode
+toString(): String

TransactionMode

TransactionMode: Enum

+TransactionMode(ctor)

Order

-TransactionType: TransactionMode
-value: int
-currencyCode: String

+Order(int value, String currencyCode, TransactionMode Transaction): ctor
+getValue(): int
+getTransactionType(): TransactionMode
+getCurrencyCode(): String
+toString(): String
+generateDailyOrder(List<ExchangeRate> tempCurrencies, int amount): List<Order>

ExchangeSite

+name: String
+transactionList = new ArrayList(): List<Transaction>
-backupReport: Report
-theBox: MoneyBox
-currencyMap: Map<String, Currency>
-rates: List<ExchangeRate>
-logger: Logger

+ExchangeSite(String Name): ctor
+ExchangeSite(String Name, LocalDateTime TimeStamp): ctor
+startTheDay(): void
+calculatePrice(String currencyCode, int amount, TransactionMode transactionType): int
+buyMoney(Order orderData): boolean
+sellMoney(Order orderData): boolean
+printSiteReport(String fileFormat): void
+shutDownService(String fileFormat): void
+getCurrencyMap(): Map<String, Currency>
+getAvailableAmount(String currencyCode): Optional<Double>
+completeOrder(Order orderData): Order
+addOrderToQueue(Order d): List<Order>
+processOrderQueue(List<Order> orderList): List<Order>
+getTransactionList(): List<Transaction>
+getRates(): List<ExchangeRate>
+setName(String name): void

ExchangeRate

-day: LocalDate
-currencyName: String
-exchangeRate: Float

+ExchangeRate(LocalDate day, int scale, String currency, Float price): ctor
+getLocalDate(): LocalDate
+getName(): String
+getExchangeRate(): Float
+setExchangeRate(Float rate): boolean
+toString(): String

Config

-siteName = "default": String
-logName = "MoneyServiceLog": String
-logFormat = "text": String
-logLevel = "OFF": String
-MIN_AMMOUNT = 50: int
-MAX_AMMOUNT = 50000: int
-buyRateConfig = 0.995f: float
-sellRateConfig = 1.005f: float
-password = char[]
+exchangeRateList = new ArrayList<ExchangeRate>(): List<ExchangeRate>
-logger: Logger

+readConfigFile(String filename): boolean
+setRatesInCurrency(List<ExchangeRate> currencyList, Map<String, Currency> currencyMap): void
+setTheRates(): List<ExchangeRate>
+fillTheMoneyBox(MoneyBox theBox, Map<String, Currency> currencyMap): MoneyBox
+controlPwd(String temp): boolean
~setPassword(String password): void
+getLogName(): String
+getExchangeRateList(): List<ExchangeRate>
+setMIN_AMMOUNT(int MIN_AMMOUNT): boolean
+setMAX_AMMOUNT(int MAX_AMMOUNT): boolean
+getMIN_AMMOUNT(): int
+getMAX_AMMOUNT(): int
+getBuyRateConfig(): float
+setBuyRateConfig(float buyRateConfig): void
+getSellRateConfig(): float
+setSellRateConfig(float sellRateConfig): void
+getLogFormat(): String
+getLogLevel(): String
+setLogName(String logName): void
+setLogFormat(String logFormat): void
+setLogLevel(Level logLevel): void
+setLogger(Logger logger): void
+getSiteName(): String
+setSiteName(String siteName): void

MoneyBox

-serialVersionUID = 1L: long
-logger: Logger
-currencyMap: Map<String, Currency>

+MoneyBox(Map<String, Currency> currencyMap): ctor
+getCurrencyMap(): Map<String, Currency>
+setCurrencyMap(Map<String, Currency> newCurrencyMap): void
+getSerialversionuid(): long
+addNewCurrency(double amount, String currencyName, Float rate): boolean
+denominationControl(String currencyName, double amount): int

Currency

-serialVersionUID = 1L: long
-totalValue: double
~buyRate: Float
~sellRate: Float

+Currency(int totalValue, Float buyRate, Float sellRate): ctor
+getBuyRate(): Float
+setBuyRate(Float buyRate): void
+getSellRate(): Float
+setSellRate(Float sellRate): void
+getTotalValue(): Double
+setTotalValue(int valueToChange): void
+toString(): String

MoneyServiceIO

+projectConfigFilename = "ProjectConfig_" + LocalDate.now().toString() + ".txt": String
+currencyConfigFilename = "CurrencyConfig_" + LocalDate.now().toString() + ".txt": String
+folderPaths = new ArrayList(): List<File>
~serializedDailyTransactionFilename = "DailyTransactions.ser": String
~textFormattedDailyTransactions = "DailyTransactions.txt": String
+referenceCurrency: String
+refDate: LocalDate
-LDT = LocalDateTime.now(): LocalDateTime

+changeDate(LocalDate date): void
+getReferenceCurrency(): String
+setRefDate(LocalDate refDate): void
+parseProjectConfig(List<String> listToBeParsed): Map<String, Double>
+parseCurrencyConfig(List<String> listToBeParsed): List<ExchangeRate>
+readTextFiles(String filename): List<String>
+saveSerializedDailyTransactions(List<Transaction> listToBeSaved, String filename): boolean
+saveTxtMoneyBox(Map<String, Currency> listToBeSaved, String filename): boolean
+saveSerializedReport(Report r): boolean
+readSerializedDailyTransactionList(String filename): List<Transaction>
+saveDailyTransactionListAsText(List<Transaction> dailyList, String filename): boolean
+setFolderPath(String path): void
+printPathList(): void
+getPathName(String folder): String

Report

-id: long
~day: LocalDateTime
~dailyTransactions: List<Transaction>
+uniqueId = 0: int
+uniqueFilename: String

+Report(LocalDateTime day, List<Transaction> dailyTransactions): ctor
+getDay(): LocalDateTime
+setDay(LocalDateTime inputDay): void
+getDailyTransaction(): List<Transaction>
+setDailyTransaction(List<Transaction> newDailyTransactions): void
+getId(): long
+getUniqueFileName(): String
+toString(): String