TransferMonneyLockApi application

Requirements:

1) Transfers should be specified by providing: accountFrom id, accountTo id, amount to transfer between account

2) The amount to transfer between account

3) It should not be possible for an account to end up with negative balance (we do not support overdrafts!)

4) Whenever a transfer is made, a notification should be sent to both account holders, with a message containing id of the other account and

amount transferred.

5) Use notificationService interface

6) This feature should be implemented in a thread-safe manner.

7) Solution should never deadlock, should never result in corrupted account state,

8) Should work efficiently for multiple transfers happening at the same time

Description

TransferConcurencyProject -s Spring boot application with was build by using concurrency model

for making possible process a big amount of payments in parallel manner.

Such functionality contains in TransferService interface with corresponding methods:

transferMoneyLock .

Such interface can be called from TransferController by using following url :

-"/v1/transfers/process/{accountIdFrom}/{accountIdTo}/{amount}",

where:

“accountIdFrom” – account from withdraw money,

“accountIdTo” – account to deposit money,

“amount” – account of transfer

Before testing application need to create accounts and add balance.

Such functionality is providing by interface AccountsService, which can be called from AccountsController by ising following urls:

For create account:

-"/v1/accounts/createAccount/{accountId}"

For deposit account:

- "/v1/ accounts /{accountId}/{amount}/balance/add"

where:

“accountId” – account name f,

“amount” – amount of deposit,

Application contains tests with can simulate multithreading behavior of "users" by executing a number of threads

with "withdraw" and "deposit" operations.

Simple run application

1.Download jar "java-transfer-multithreading.jar" from root directory of github project .

2.Open "Command Line" and go to directory where "java-transfer-multithreading.jar" located .

3.Execute in "Command Line" following command: "java -jar java-transfer-multithreading.jar" .

4.Open any browser and put following commands:

- view all accounts: http://localhost:18080/v1/accounts/all
- create account with name "1": http://localhost:18080/v1/accounts/createAccount/1
- create account with name "2": http://localhost:18080/v1/accounts/createAccount/2
- add balance to account "1" : http://localhost:18080/v1/accounts/1/20/balance/add
- add balance to account "2" : http://localhost:18080/v1/accounts/2/20/balance/add
- make transfer from account "1" to "account "2" for amount "20": http://localhost:18080/v1/transfers/process/1/2/20
- clear all accounts if exists: http://localhost:18080/v1/accounts/clear
- withdraw balance from account "2": http://localhost:18080/v1/accounts/2/10/balance/withdraw

5.For exit application just close "Command Line"

What need to add

Add possibility to read from config files and set to ConcurentHashMap constructor following properties: “InitialCapacity”,”loadFactor”,”concurencyLevel” for classes: AccountsRepositoryInMemory and TransferLocksRepositoryInMemory.

Make thread-safe methods “creditBalanceAccount” and “debitBalanceAccount” in `AccountsServiceImpl`.