Lesson 8: Spring and Hibernate

Exercise 7.1 – Spring Support

The Setup:

The goal of this exercise is to practice integrating Spring and Hibernate. The exercise will ask you to use combined Spring and Hibernate configuration, and use Spring's Transaction Support.

Start this exercise by importing the **exercise7_1** project and add the **Spring context dependency** to it and also add the following additional spring dependencies:

```
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-tx</artifactId>
<version>4.1.6.RELEASE</version>
</dependency>
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-orm</artifactId>
<version>4.1.6.RELEASE</version>
</dependency>`
```

The Exercise:

Although the code already uses Hibernate, it does not yet use Spring.

a.   Change the project to start using Spring, and configure Hibernate through Spring's com-
     bined configuration possibilities (you can remove HibernateUtil and the hiber-
     nate.cfg.xml).

b.   Add Spring Declarative Transaction Demarcation to the service layer, removing the pro-
     grammatic transaction demarcation used before.

c.   Also add declarative transaction demarcation to the data access objects. Although it
     doesn't change much in this application, it is good practice to specify transactional re-
     quirements for DAO objects.

bank.service

bank

Application

+main()

<<interface>>
IAccountService

+createAccount()
+deposit()
+withdraw()
+depositEuros()
+withdrawEuros()
+getAccount()
+getAllAccounts()
+transferFunds()
+printAccountStatements()

AccountService

+createAccount()
+deposit()
+withdraw()
+depostEuros()
+withdrawEuros()
+getAccount()
+getAllAccounts()
+transferFunds()
+printAccountStatements()

<<interface>>
ICurrencyConverter

+euroToDollars()
+dollarsToEuros()

CurrencyConverter

+euroToDollars()
+dollarsToEuros()

bank.domain

<<entity>>
Account

+accountnumber

+deposit()
+withdraw()
+getBalance()
+transferFunds()

<<entity>>
Customer

+name

<<entity>>
AccountEntry

+date
+amount
+description
+fromAccountNumber
+fromPersonName

bank.dao

<<interface>>
IAccountDAO

+saveAccount()
+updateAccount()
+badAccount()
+getAccounts()

AccountDAO

+saveAccount()
+updateAccount()
+loadAccount()
+getAccounts()

AccountDAOHibernate

+saveAccount()
+updateAccount()
+badAccount()
+getAccounts()

bank.service.aop

StopWatchAdvice

+methodTimer()

JMSLogAdvice

+logJMSMessage()

DAOLogAdvice

+logDAOCall()

bank.jms

<<interface>>
IJMSSender

+sendJMSMessage()

JMSSender

+sendJMSMessage()

bank.logging

<<interface>>
ILogger

+log()

Logger

+log()