

Exercise 6.2 – Bank Application AOP

BankProject:

The last part of this lab is using AOP in our Bank application.
Start with the bank application project that was the result of the Spring DI lab.
Add the following functionality with AOP:

- Log all database access to the console
- Measure the time for all service level classes

```
1 package bank.dao;
2
3 import org.aspectj.lang.JoinPoint;
4
5 @Aspect
6 public class DaoLoggingAdvice {
7
8     //log all database access to the console
9     @After("execution(* bank.dao.*(..))")
10    public void log(JoinPoint joinpoint) {
11        System.out.println("-----Call was made to: " + joinpoint.getSignature().getName()
12            + " on " + joinpoint.getTarget().getClass());
13    }
14 }
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33 import org.aspectj.lang.ProceedingJoinPoint;
34
35 @Aspect
36 public class ServiceTimerAdvice {
37
38     //measure the time for all service level classes
39     @Around("execution(* bank.service.*(..))")
40    public Object time(ProceedingJoinPoint call) throws Throwable {
41        Stopwatch sw = new Stopwatch();
42        sw.start(call.getSignature().getName());
43        Object retVal = call.proceed();
44        sw.stop();
45
46        long totalTime=sw.getLastTaskTimeMillis();
47        System.out.println("-----Time to execute "+call.getSignature().getName()+" = "+totaltime+" ms");
48
49        return retVal;
50    }
51 }
52
53 }
```

```
Application.java DaoLoggingAdvice.java ServiceTimerAdvice.java springconfig.xml
3 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4 xmlns:aop="http://www.springframework.org/schema/aop"
5 xmlns:tx="http://www.springframework.org/schema/tx"
6 xsi:schemaLocation="
7     http://www.springframework.org/schema/beans
8     http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
9     http://www.springframework.org/schema/tx
10    http://www.springframework.org/schema/tx/spring-tx-3.0.xsd
11    http://www.springframework.org/schema/aop
12    http://www.springframework.org/schema/aop/spring-aop-3.0.xsd"
13
14 <aop:aspectj-autoproxy />
15 <bean id="accountService" class="bank.service.AccountService">
16     <property name="accountDAO" ref="testaccountDao" />
17 </bean>
18 <bean id="accountDao" class="bank.dao.AccountDAO"/>
19 <bean id="testaccountDao" class="bank.dao.TestAccountDAO"/>
20 <bean id="servicetimer" class="bank.service.ServiceTimerAdvice"/>
21 <bean id="daoLogger" class="bank.dao.DaoLoggingAdvice"/>
22 </beans>
23
```

Exercise 6.2 – Bank Application AOP

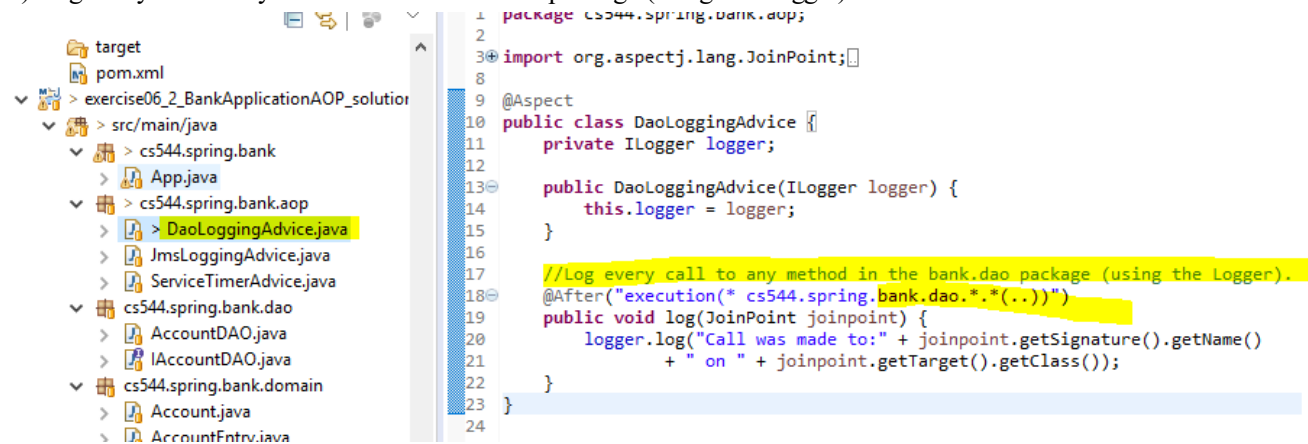
The Setup:

In this exercise, we will be extending the bank application to use AOP. Import exercise6.2. Be sure to update the pom.xml file dependencies for AOP.

The Exercise:

Use AOP to:

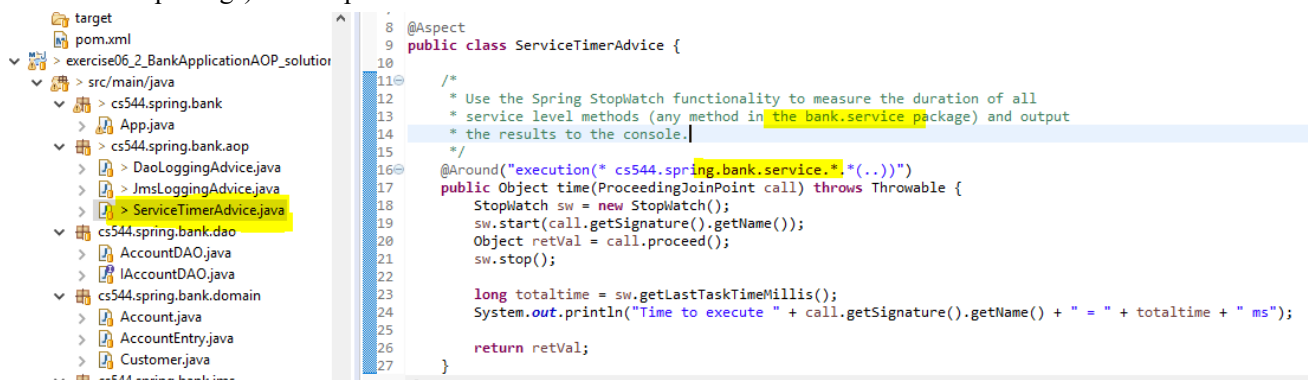
a) Log every call to any method in the bank.dao package (using the Logger).



The screenshot shows an IDE with a project structure on the left and a code editor on the right. The project structure includes a 'target' directory, a 'pom.xml' file, and a 'src/main/java' directory. The 'src/main/java' directory contains several packages: 'cs544.spring.bank', 'cs544.spring.bank.aop', 'cs544.spring.bank.dao', and 'cs544.spring.bank.domain'. The 'cs544.spring.bank.aop' package contains 'App.java', 'DaoLoggingAdvice.java', 'JmsLoggingAdvice.java', and 'ServiceTimerAdvice.java'. The 'cs544.spring.bank.dao' package contains 'AccountDAO.java', 'IAccountDAO.java', and 'AccountEntry.java'. The 'cs544.spring.bank.domain' package contains 'Account.java' and 'AccountEntry.java'. The code editor shows the 'DaoLoggingAdvice.java' file with the following code:

```
1 package cs544.spring.bank.aop;
2
3 import org.aspectj.lang.JoinPoint;
4
5 @Aspect
6 public class DaoLoggingAdvice {
7     private ILogger logger;
8
9     public DaoLoggingAdvice(ILogger logger) {
10         this.logger = logger;
11     }
12
13     //Log every call to any method in the bank.dao package (using the Logger).
14     @After("execution(* cs544.spring.bank.dao.*(..))")
15     public void log(JoinPoint joinpoint) {
16         logger.log("Call was made to:" + joinpoint.getSignature().getName()
17             + " on " + joinpoint.getTarget().getClass());
18     }
19 }
```

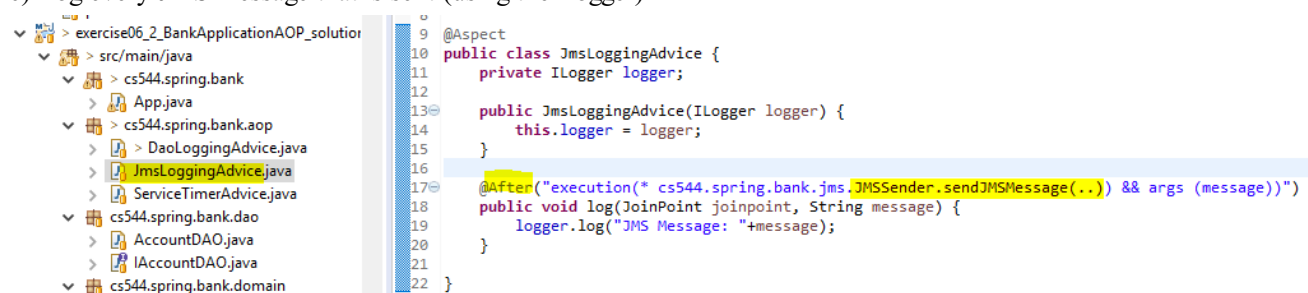
b) Use the Spring Stopwatch functionality to measure the duration of all service level methods (any method in the bank.service package) and output the results to the console.



The screenshot shows an IDE with a project structure on the left and a code editor on the right. The project structure is the same as in the previous screenshot. The code editor shows the 'ServiceTimerAdvice.java' file with the following code:

```
8 @Aspect
9 public class ServiceTimerAdvice {
10
11     /*
12     * Use the Spring Stopwatch functionality to measure the duration of all
13     * service level methods (any method in the bank.service package) and output
14     * the results to the console.
15     */
16     @Around("execution(* cs544.spring.bank.service.*(..))")
17     public Object time(ProceedingJoinPoint call) throws Throwable {
18         Stopwatch sw = new Stopwatch();
19         sw.start(call.getSignature().getName());
20         Object retVal = call.proceed();
21         sw.stop();
22
23         long totaltime = sw.getLastTaskTimeMillis();
24         System.out.println("Time to execute " + call.getSignature().getName() + " = " + totaltime + " ms");
25
26         return retVal;
27     }
28 }
```

c) Log every JMS message that is sent (using the Logger)



The screenshot shows an IDE with a project structure on the left and a code editor on the right. The project structure is the same as in the previous screenshots. The code editor shows the 'JmsLoggingAdvice.java' file with the following code:

```
9 @Aspect
10 public class JmsLoggingAdvice {
11     private ILogger logger;
12
13     public JmsLoggingAdvice(ILogger logger) {
14         this.logger = logger;
15     }
16
17     @After("execution(* cs544.spring.bank.jms.JMSsender.sendJMSMessage(..) && args (message))")
18     public void log(JoinPoint joinpoint, String message) {
19         logger.log("JMS Message: " + message);
20     }
21 }
```

d) Remove all the other logger calls from AccountService (doing so will make it easier to see whether your advice is running or not).