**Spring AOP**

1. **Create a logging aspect. Create a pointcut to log all methods Of services .**

**aop-spring-config.xml**

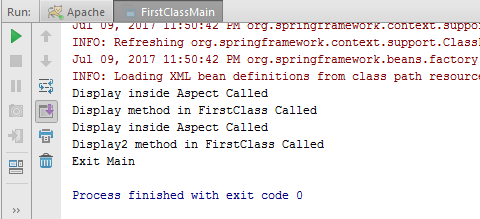
*<?***xml version="1.0" encoding="UTF-8"***?>*<**beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:aop="http://www.springframework.org/schema/aop"  
 xmlns:context="http://www.springframework.org/schema/context"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/aop  
 http://www.springframework.org/schema/aop/spring-aop.xsd  
 http://www.springframework.org/schema/context  
 http://www.springframework.org/schema/context/spring-context.xsd"**>  
  
<**context:component-scan base-package="com.spring.component"**/>  
  
<**aop:aspectj-autoproxy**/>  
  
<**bean class="com.spring.aspect.LoggingAspect"**></**bean**>  
</**beans**>

**LoggingAspect**

**package** com.spring.aspect;  
  
  
**import** org.aspectj.lang.annotation.Aspect;  
**import** org.aspectj.lang.annotation.Before;  
  
@Aspect  
  
**public class** LoggingAspect {  
 @Before(**"execution(\* \*())"**)  
 **void** display(){  
 System.***out***.println(**"Display inside Aspect Called"**);  
 }  
  
}

**FirstClass.java**

**package** com.spring.component;  
  
**import** org.springframework.stereotype.Controller;  
  
@Controller  
**public class** FirstClass {  
 **void** display(){  
 System.***out***.println(**"Display method in FirstClass Called"**);  
 }  
  
 **void** display2(){  
 System.***out***.println(**"Display2 method in FirstClass Called"**);  
 }  
}



**2) Add a logging statement before and after the method ends**

**LoggingAspect.java**

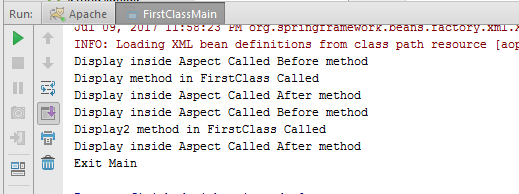
**package** com.spring.aspect;  
  
  
**import** org.aspectj.lang.annotation.After;  
**import** org.aspectj.lang.annotation.Aspect;  
**import** org.aspectj.lang.annotation.Before;  
  
@Aspect  
  
**public class** LoggingAspect {  
 @Before(**"execution(\* \*())"**)  
 **void** displayBefore(){  
 System.***out***.println(**"Display inside Aspect Called Before method"**);  
 }  
  
 @After(**"execution(\* \*())"**)  
 **void** displayAfter(){  
 System.***out***.println(**"Display inside Aspect Called After method"**);  
 }  
  
}

**FirstClass.java**

**package** com.spring.component;  
  
**import** org.springframework.stereotype.Controller;  
  
@Controller  
**public class** FirstClass {  
 **void** display(){  
 System.***out***.println(**"Display method in FirstClass Called"**);  
 }  
  
 **void** display2(){  
 System.***out***.println(**"Display2 method in FirstClass Called"**);  
 }  
}

**FirstClassMain.java**

**package** com.spring.component;  
  
**import** org.springframework.context.ApplicationContext;  
**import** org.springframework.context.support.ClassPathXmlApplicationContext;  
  
  
**public class** FirstClassMain {  
 **public static void** main(String[] args) {  
 ApplicationContext applicationContext = **new** ClassPathXmlApplicationContext(**"aop-spring-config.xml"**);  
 FirstClass firstClass = applicationContext.getBean(FirstClass.**class**);  
 firstClass.display();  
 firstClass.display2();  
 System.***out***.println(**"Exit Main"**);  
 }  
}



**3) Log all the methods which throw IOException**

**LoggingAspect.java**

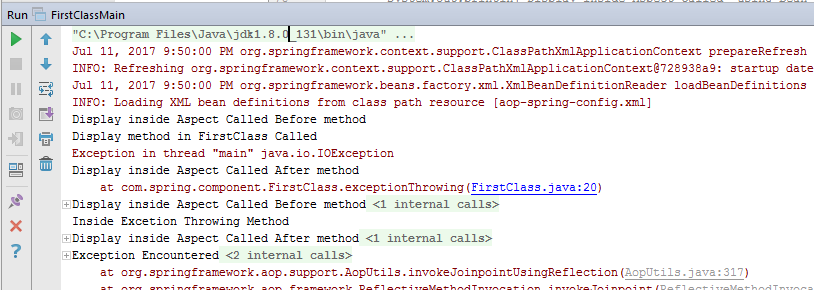
**package** com.spring.aspect;  
  
  
**import** org.aspectj.lang.annotation.After;  
**import** org.aspectj.lang.annotation.AfterThrowing;  
**import** org.aspectj.lang.annotation.Aspect;  
**import** org.aspectj.lang.annotation.Before;  
  
@Aspect  
  
**public class** LoggingAspect {  
 @Before(**"execution(\* \*())"**)  
 **void** displayBefore(){  
 System.***out***.println(**"Display inside Aspect Called Before method"**);  
 }  
  
 @After(**"execution(\* \*())"**)  
 **void** displayAfter(){  
 System.***out***.println(**"Display inside Aspect Called After method"**);  
 }  
  
 @AfterThrowing(pointcut = **"execution(\* \*())"**,throwing = **"exception"**)  
 **void** displayException(Exception exception){  
 System.***out***.println(**"Exception Encountered"**);  
 }  
  
}

**FirstClass.java**

**package** com.spring.component;  
  
**import** org.springframework.stereotype.Controller;  
  
**import** java.io.IOException;  
  
@Controller  
**public class** FirstClass {  
 **void** display(){  
 System.***out***.println(**"Display method in FirstClass Called"**);  
 }  
  
 **void** exceptionThrowing() **throws** IOException {  
 System.***out***.println(**"Inside Excetion Throwing Method"**);  
 **throw new** IOException();  
 }  
}

**FirstClassMain.java**

**package** com.spring.component;  
  
**import** org.springframework.context.ApplicationContext;  
**import** org.springframework.context.support.ClassPathXmlApplicationContext;  
  
**import** java.io.IOException;  
  
  
**public class** FirstClassMain {  
 **public static void** main(String[] args) **throws** IOException {  
 ApplicationContext applicationContext = **new** ClassPathXmlApplicationContext(**"aop-spring-config.xml"**);  
 FirstClass firstClass = applicationContext.getBean(FirstClass.**class**);  
 firstClass.display();  
 firstClass.exceptionThrowing();  
 System.***out***.println(**"Exit Main"**);  
 }  
}

 **4) Log all the methods annotated with @Deprecated Annotation**

FirstClassMain.java

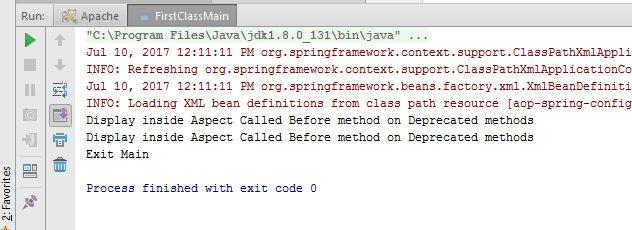
**package** com.spring.component;  
  
**import** org.springframework.context.ApplicationContext;  
**import** org.springframework.context.support.ClassPathXmlApplicationContext;  
  
**import** java.io.IOException;  
  
  
**public class** FirstClassMain {  
 **public static void** main(String[] args) **throws** IOException {  
 ApplicationContext applicationContext = **new** ClassPathXmlApplicationContext(**"aop-spring-config.xml"**);  
 FirstClass firstClass = applicationContext.getBean(FirstClass.**class**);  
 firstClass.display();  
 firstClass.display2();System.***out***.println(**"Exit Main"**);  
 }  
}

**FirstClass.java**

**package** com.spring.component;  
  
**import** org.springframework.stereotype.Controller;  
  
**import** java.io.IOException;  
  
@Controller  
**public class** FirstClass {  
 @Deprecated  
 **public void** display(){  
 System.***out***.println(**"Display method in FirstClass Called"**);  
 }  
@Deprecated  
 **public void** display2(){  
 System.***out***.println(**"Display2 method in FirstClass Called"**);  
 }  
}

**LoggingAspect.java**

**package** com.spring.aspect;  
**import** org.aspectj.lang.annotation.\*;  
  
@Aspect  
**public class** LoggingAspect {@Around(**"@annotation(Deprecated)"**)  
 **void** displayBeforeAnnotation(){  
 System.***out***.println(**"Display inside Aspect Called Before method on Deprecated methods"**);  
 }  
}



**5) Apply execution, within, args, this and bean expressions on the before and after advice of service bean.**

**LoggingAspect.java**

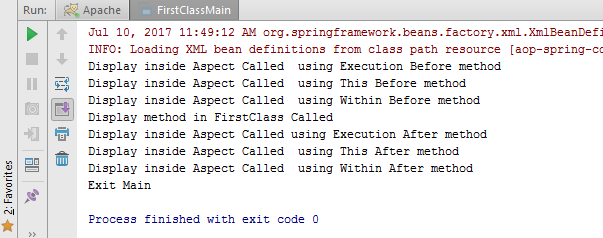
**package** com.spring.aspect;  
**import** org.aspectj.lang.annotation.\*;  
  
@Aspect  
**public class** LoggingAspect {@Before(**"execution(void display())"**)  
 **void** displayBefore(){  
 System.***out***.println(**"Display inside Aspect Called using Execution Before method"**);  
 }  
  
 @After(**"execution(void display())"**)  
 **void** displayAfter(){  
 System.***out***.println(**"Display inside Aspect Called using Execution After method"**);  
 }  
  
 @Before(**"within(com.spring.component.\*))"**)  
 **void** displayBeforeWithin(){  
 System.***out***.println(**"Display inside Aspect Called using Within Before method"**);  
 }  
  
 @After(**"within(com.spring.component.\*))"**)  
 **void** displayAfterWithin(){  
 System.***out***.println(**"Display inside Aspect Called using Within After method"**);  
 }  
  
 @Before(**"this(com.spring.component.FirstClass))"**)  
 **void** displayBeforeThis(){  
 System.***out***.println(**"Display inside Aspect Called using This Before method"**);  
 }  
  
 @After(**"within(com.spring.component.FirstClass))"**)  
 **void** displayAfterThis(){  
 System.***out***.println(**"Display inside Aspect Called using This After method"**);  
 }  
  
 @Before(**"args(Integer)"**)  
 **void** displayBeforeArgs(){  
 System.***out***.println(**"Display inside Aspect Called using Args Before method"**);  
 }  
  
 @After(**"args(Integer)"**)  
 **void** displayAfterArgs(){  
 System.***out***.println(**"Display inside Aspect Called using Args After method"**);  
 }  
  
 @Before(**"bean(SecondClass)"**)  
 **void** displayBeforeBean(){  
 System.***out***.println(**"Display inside Aspect Called using Bean Before method"**);  
 }  
  
 @After(**"bean(SecondClass)"**)  
 **void** displayAfterBean(){  
 System.***out***.println(**"Display inside Aspect Called using Bean After method"**);  
 }  
  
}

**FirstClass.java**

**package** com.spring.component;  
  
**import** org.springframework.stereotype.Controller;  
  
**import** java.io.IOException;  
  
@Controller  
**public class** FirstClass {  
 **public void** display(){  
 System.***out***.println(**"Display method in FirstClass Called"**);  
 }  
}

**FirstClassMain.java**

**package** com.spring.component;  
  
**import** org.springframework.context.ApplicationContext;  
**import** org.springframework.context.support.ClassPathXmlApplicationContext;  
  
**import** java.io.IOException;  
  
  
**public class** FirstClassMain {  
 **public static void** main(String[] args) **throws** IOException {  
 ApplicationContext applicationContext = **new** ClassPathXmlApplicationContext(**"aop-spring-config.xml"**);  
 FirstClass firstClass = applicationContext.getBean(FirstClass.**class**);  
 firstClass.display();  
System.***out***.println(**"Exit Main"**);  
 }  
}

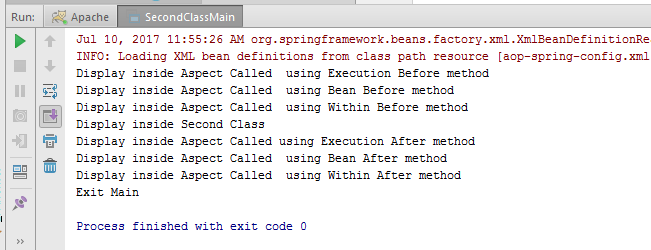


**Secondclass.java**

**package** com.spring.component;  
  
**public class** SecondClass {  
 **public void** display(){  
 System.***out***.println(**"Display inside Second Class"**);  
 }  
}

**SecondClassMain.java**

**package** com.spring.component;  
  
**import** org.springframework.context.ApplicationContext;  
**import** org.springframework.context.support.ClassPathXmlApplicationContext;  
  
**import** java.io.IOException;  
**public class** SecondClassMain {  
 **public static void** main(String[] args) **throws** IOException {  
 ApplicationContext applicationContext = **new** ClassPathXmlApplicationContext(**"aop-spring-config.xml"**);  
 SecondClass secondClass = applicationContext.getBean(SecondClass.**class**);  
 secondClass.display();  
 System.***out***.println(**"Exit Main"**);  
 }  
}



**6) Demonstrate the use of pointCut @Pointcut annotation and Reuse the expression created**

**LoggingAspect.java**

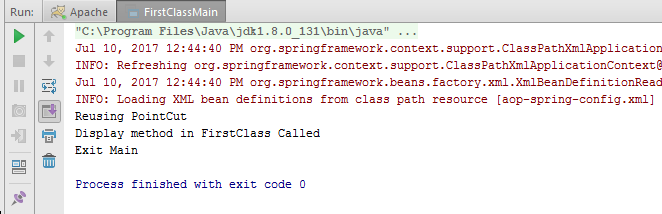
**package** com.spring.aspect;  
**import** org.aspectj.lang.annotation.\*;  
  
@Aspect  
**public class** LoggingAspect {@Before(**"displayPointCut()"**)  
 **void** displayBefore(){  
 System.***out***.println(**"Reusing PointCut"**);  
  
}  
  
@Pointcut(**"execution(void display())"**)  
**void** displayPointCut(){}  
  
}

**FirstClass.java**

**package** com.spring.component;  
  
**import** org.springframework.stereotype.Controller;  
  
**import** java.io.IOException;  
  
@Controller  
**public class** FirstClass {  
  
 **public void** display(){  
 System.***out***.println(**"Display method in FirstClass Called"**);  
 }  
}

**FirstClassMain.java**

**package** com.spring.component;  
  
**import** org.springframework.context.ApplicationContext;  
**import** org.springframework.context.support.ClassPathXmlApplicationContext;  
  
**import** java.io.IOException;  
  
  
**public class** FirstClassMain {  
 **public static void** main(String[] args) **throws** IOException {  
 ApplicationContext applicationContext = **new** ClassPathXmlApplicationContext(**"aop-spring-config.xml"**);  
 FirstClass firstClass = applicationContext.getBean(FirstClass.**class**);  
 firstClass.display();  
System.***out***.println(**"Exit Main"**);  
 }  
}



**7) Access the properties of the JoinPoint in before advice. Print Signature of method being called and its arguments**

**LoggingAspect.java**

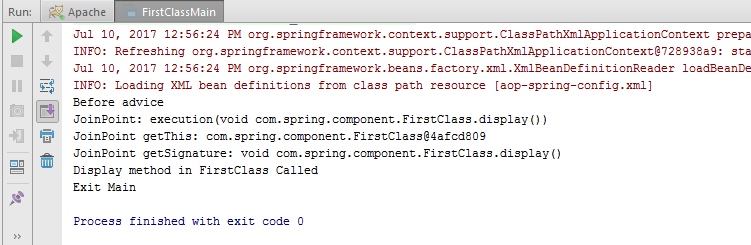
**package** com.spring.aspect;  
**import** org.aspectj.lang.JoinPoint;  
**import** org.aspectj.lang.annotation.\*;  
  
@Aspect  
**public class** LoggingAspect {  
 @Before(**"execution(void display())"**)  
 **void** displayBefore(JoinPoint joinPoint) {  
 System.***out***.println(**"Before advice"**);  
 System.***out***.println(**"JoinPoint: "**+joinPoint);  
 System.***out***.println(**"JoinPoint getThis: "**+joinPoint.getThis());  
 System.***out***.println(**"JoinPoint getSignature: "**+joinPoint.getSignature());  
 }  
}

**FirstClass.java**

**package** com.spring.component;  
  
**import** org.springframework.stereotype.Controller;  
  
**import** java.io.IOException;  
  
@Controller  
**public class** FirstClass {  
  
 **public void** display(){  
 System.***out***.println(**"Display method in FirstClass Called"**);  
 }  
}

**FirstClassMain.java**

**package** com.spring.component;  
  
**import** org.springframework.context.ApplicationContext;  
**import** org.springframework.context.support.ClassPathXmlApplicationContext;  
  
**import** java.io.IOException;  
  
  
**public class** FirstClassMain {  
 **public static void** main(String[] args) **throws** IOException {  
 ApplicationContext applicationContext = **new** ClassPathXmlApplicationContext(**"aop-spring-config.xml"**);  
 FirstClass firstClass = applicationContext.getBean(FirstClass.**class**);  
 firstClass.display();  
System.***out***.println(**"Exit Main"**);  
 }  
}



**Spring Transaction**

1. **Create User Account table with name,balance fields**

**Transaction-config.xml**

<**beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:tx="http://www.springframework.org/schema/tx"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/tx  
 http://www.springframework.org/schema/tx/spring-tx.xsd"**>  
  
 <**tx:annotation-driven transaction-manager="transactionManager"**/>  
  
 <**bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource"**>  
 <**property name="url" value="jdbc:mysql://localhost:3306/springDemo"**/>  
 <**property name="driverClassName" value="com.mysql.jdbc.Driver"**/>  
 <**property name="username" value="root"**/>  
 <**property name="password" value="password"**></**property**>  
 </**bean**>  
  
 <**bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate"**>  
 <**property name="dataSource" ref="dataSource"**/>  
 </**bean**>  
  
 <**bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager"**>  
 <**property name="dataSource" ref="dataSource"**/>  
 </**bean**>  
  
 <**bean id="accountService" class="com.spring.transaction.AccountService"**>  
 <**property name="jdbcTemplate" ref="jdbcTemplate"**/>  
 <**property name="dataSource" ref="dataSource"**/>  
 </**bean**>  
  
 </**beans**>

**UserAccount.java**

**package** com.spring.transaction;  
  
**public class** UserAccount {  
  
 String **name**;  
 **int balance**;  
  
 **public** String getName() {  
 **return name**;  
 }  
  
 **public void** setName(String name) {  
 **this**.**name** = name;  
 }  
  
 **public int** getBalance() {  
 **return balance**;  
 }  
  
 **public void** setBalance(**int** balance) {  
 **this**.**balance** = balance;  
 }  
}

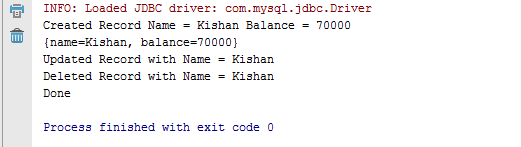
**2) Create CRUD operation in Account Service (Add/Get/Delete/Update amount in User Account table)**

**AccountService.java**

**package** com.spring.transaction;  
  
**import** org.springframework.jdbc.core.JdbcTemplate;  
**import** org.springframework.transaction.annotation.Isolation;  
**import** org.springframework.transaction.annotation.Transactional;  
  
**import** javax.sql.DataSource;  
  
*/\*\*  
 \* Created by krishan on 7/10/2017.  
 \*/***public class** AccountService  
{  
 **private** JdbcTemplate **jdbcTemplate**;  
 **private** DataSource **dataSource**;  
  
 **public** JdbcTemplate getJdbcTemplate() {  
 **return jdbcTemplate**;  
 }  
  
 **public void** setJdbcTemplate(JdbcTemplate jdbcTemplate) {  
 **this**.**jdbcTemplate** = jdbcTemplate;  
 }  
  
 **public** DataSource getDataSource() {  
 **return dataSource**;  
 }  
  
 **public void** setDataSource(DataSource dataSource) {  
 **this**.**dataSource** = dataSource;  
 }  
  
 @Transactional(isolation = Isolation.***READ\_UNCOMMITTED***,rollbackFor = RuntimeException.**class**)  
 **public void** create(String name, Integer balance) {  
 String SQL = **"insert into UserAccount (name, balance) values (?, ?)"**;  
 **jdbcTemplate**.update( SQL, **new** Object[]{name,balance});  
 System.***out***.println(**"Created Record Name = "** + name + **" Balance = "** + balance);  
  
 }  
  
 @Transactional(isolation = Isolation.***READ\_UNCOMMITTED***)  
 **public void** getAccount(String name) {  
 String SQL = **"select** *\** **from useraccount where name = ?"**;  
 System.***out***.println(**jdbcTemplate**.queryForMap(SQL, **new** Object[]{name}));  
 }  
  
  
 **public void** delete(String name){  
 String SQL = **"delete from useraccount where name = ?"**;  
 **jdbcTemplate**.update(SQL, name);  
 System.***out***.println(**"Deleted Record with Name = "** + name );  
 **return**;  
 }  
  
 @Transactional(isolation = Isolation.***READ\_UNCOMMITTED***,rollbackFor = RuntimeException.**class**)  
 **public void** update(String name, Integer balance){  
 String SQL = **"update useraccount set balance = ? where name = ?"**;  
 **jdbcTemplate**.update(SQL, balance,name);  
 System.***out***.println(**"Updated Record with Name = "** + name );  
 **return**;  
 }  
  
}

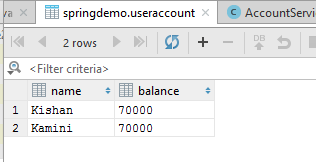
**TransactionMain.java**

**package** com.spring.transaction;  
  
**import** org.springframework.context.ApplicationContext;  
**import** org.springframework.context.support.ClassPathXmlApplicationContext;  
**public class** TransactionMain {  
 **public static void** main(String[] args) {  
 ApplicationContext applicationContext = **new** ClassPathXmlApplicationContext(**"transaction-config.xml"**);  
 AccountService accountService = applicationContext.getBean(AccountService.**class**);  
 accountService.create(**"Kishan"**,70000);  
 accountService.getAccount(**"Kishan"**);  
 accountService.update(**"Kishan"**,80000);  
 accountService.delete(**"Kishan"**);  
 System.***out***.println(**"Done"**);  
 }  
}



**3) Transfer amount from "user1" to "user2". Using Programmatic Transaction**

**Before Transfer**



**Transaction-config.xml**

<**beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:tx="http://www.springframework.org/schema/tx"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/tx  
 http://www.springframework.org/schema/tx/spring-tx.xsd"**>  
  
 <**tx:annotation-driven transaction-manager="transactionManager"**/>  
  
 <**bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource"**>  
 <**property name="url" value="jdbc:mysql://localhost:3306/springDemo"**/>  
 <**property name="driverClassName" value="com.mysql.jdbc.Driver"**/>  
 <**property name="username" value="root"**/>  
 <**property name="password" value="password"**></**property**>  
 </**bean**>  
  
 <**bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate"**>  
 <**property name="dataSource" ref="dataSource"**/>  
 </**bean**>  
  
 <**bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager"**>  
 <**property name="dataSource" ref="dataSource"**/>  
 </**bean**>  
  
 <**bean id="accountService" class="com.spring.transaction.AccountService"**>  
 <**property name="jdbcTemplate" ref="jdbcTemplate"**/>  
 <**property name="dataSource" ref="dataSource"**/>  
 </**bean**>  
  
 <**bean id="transferService" class="com.spring.transaction.TransferService"**>  
 <**property name="jdbcTemplate" ref="jdbcTemplate"**/>  
 <**property name="dataSource" ref="dataSource"**/>  
 <**property name="platformTransactionManager" ref="transactionManager"**/>  
 </**bean**>  
  
 </**beans**>

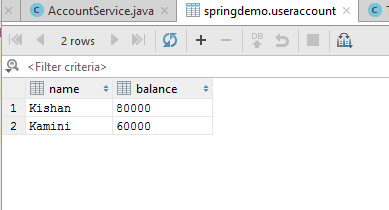
**TransferService.java**

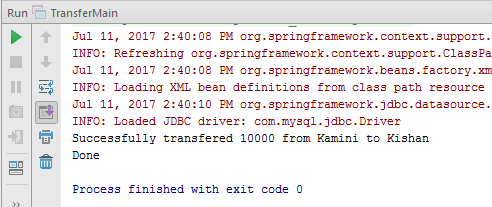
**package** com.spring.transaction;  
  
**import** org.springframework.dao.DataAccessException;  
**import** org.springframework.jdbc.core.JdbcTemplate;  
**import** org.springframework.transaction.PlatformTransactionManager;  
**import** org.springframework.transaction.TransactionDefinition;  
**import** org.springframework.transaction.TransactionStatus;  
**import** org.springframework.transaction.support.DefaultTransactionDefinition;  
  
**import** javax.sql.DataSource;  
  
**public class** TransferService {  
 **private** JdbcTemplate **jdbcTemplate**;  
 **private** DataSource **dataSource**;  
 **private** PlatformTransactionManager **platformTransactionManager**;  
  
 **public void** setPlatformTransactionManager(PlatformTransactionManager platformTransactionManager) {  
 **this**.**platformTransactionManager** = platformTransactionManager;  
 }  
  
 **public** PlatformTransactionManager getPlatformTransactionManager() {  
 **return platformTransactionManager**;  
 }  
  
 **public** JdbcTemplate getJdbcTemplate() {  
 **return jdbcTemplate**;  
 }  
  
 **public void** setJdbcTemplate(JdbcTemplate jdbcTemplate) {  
 **this**.**jdbcTemplate** = jdbcTemplate;  
 }  
  
 **public** DataSource getDataSource() {  
 **return dataSource**;  
 }  
  
 **public void** setDataSource(DataSource dataSource) {  
 **this**.**dataSource** = dataSource;  
 }  
  
 **public void** transfer(String name1, String name2, Integer balance) {  
 TransactionDefinition transactionDefinition = **new** DefaultTransactionDefinition();  
 TransactionStatus transactionStatus;  
 transactionStatus = **platformTransactionManager**.getTransaction(transactionDefinition);  
  
 **try** {  
 String SQL = **"update useraccount set balance = balance-? where name = ?"**;  
 **jdbcTemplate**.update(SQL, **new** Object[]{balance,name1});  
 String SQL2 = **"update useraccount set balance = balance+? where name = ?"**;  
 **jdbcTemplate**.update(SQL2,**new** Object[]{balance,name2});  
 **platformTransactionManager**.commit(transactionStatus);  
 System.***out***.println(**"Successfully transfered "** + balance + **" from "** + name1 + **" to "** + name2);  
 }  
 **catch** (DataAccessException exception){  
 **platformTransactionManager**.rollback(transactionStatus);  
 }  
  
 }  
}

**TransferMain.java**

**package** com.spring.transaction;  
  
**import** org.springframework.context.ApplicationContext;  
**import** org.springframework.context.support.ClassPathXmlApplicationContext;  
**public class** TransferMain {  
 **public static void** main(String[] args) {  
 ApplicationContext applicationContext = **new** ClassPathXmlApplicationContext(**"transaction-config.xml"**);  
 TransferService transferService = applicationContext.getBean(TransferService.**class**);  
 transferService.transfer(**"Kamini"**,**"Kishan"**,10000);  
 System.***out***.println(**"Done"**);  
 }  
}

**After Transfer**





**4) Transfer amount from "user1" to "user2". Using Declarative(Annotation Based) Transaction**

**Transaction-config.xml**

<**beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:tx="http://www.springframework.org/schema/tx"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/tx  
 http://www.springframework.org/schema/tx/spring-tx.xsd"**>  
  
 <**tx:annotation-driven transaction-manager="transactionManager"**/>  
  
 <**bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource"**>  
 <**property name="url" value="jdbc:mysql://localhost:3306/springDemo"**/>  
 <**property name="driverClassName" value="com.mysql.jdbc.Driver"**/>  
 <**property name="username" value="root"**/>  
 <**property name="password" value="password"**></**property**>  
 </**bean**>  
  
 <**bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate"**>  
 <**property name="dataSource" ref="dataSource"**/>  
 </**bean**>  
  
 <**bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager"**>  
 <**property name="dataSource" ref="dataSource"**/>  
 </**bean**><**bean id="decTransferService" class="com.spring.transaction.DecTransferService"**>  
 <**property name="jdbcTemplate" ref="jdbcTemplate"** />  
 </**bean**>  
  
 </**beans**>

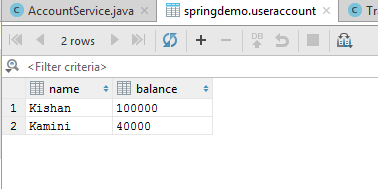
**DecTransferService.java**

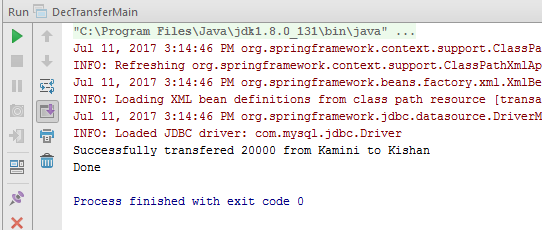
**package** com.spring.transaction;  
  
**import** org.springframework.dao.DataAccessException;  
**import** org.springframework.jdbc.core.JdbcTemplate;  
**import** org.springframework.transaction.TransactionDefinition;  
**import** org.springframework.transaction.TransactionStatus;  
**import** org.springframework.transaction.annotation.Transactional;  
**import** org.springframework.transaction.support.DefaultTransactionDefinition;  
  
**import** javax.sql.DataSource;  
  
**public class** DecTransferService {  
 **private** JdbcTemplate **jdbcTemplate**;  
 **private** DataSource **dataSource**;  
  
 **public** JdbcTemplate getJdbcTemplate() {  
 **return jdbcTemplate**;  
 }  
  
 **public void** setJdbcTemplate(JdbcTemplate jdbcTemplate) {  
 **this**.**jdbcTemplate** = jdbcTemplate;  
 }  
  
 **public** DataSource getDataSource() {  
 **return dataSource**;  
 }  
  
 **public void** setDataSource(DataSource dataSource) {  
 **this**.**dataSource** = dataSource;  
 }  
  
 @Transactional  
 **public void** transfer(String name1, String name2, Integer balance) {  
 **try** {  
 String SQL = **"update useraccount set balance = balance-? where name = ?"**;  
 **jdbcTemplate**.update(SQL, **new** Object[]{balance,name1});  
 String SQL2 = **"update useraccount set balance = balance+? where name = ?"**;  
 **jdbcTemplate**.update(SQL2,**new** Object[]{balance,name2});  
 System.***out***.println(**"Successfully transfered "** + balance + **" from "** + name1 + **" to "** + name2);  
 }  
 **catch** (DataAccessException exception){  
 System.***out***.println(**"Exception occured"**);  
 }  
  
 }  
  
  
  
}

**DecTransferMain.java**

**package** com.spring.transaction;  
  
**import** org.springframework.context.ApplicationContext;  
**import** org.springframework.context.support.ClassPathXmlApplicationContext;  
**public class** DecTransferMain {  
 **public static void** main(String[] args) {  
 ApplicationContext applicationContext = **new** ClassPathXmlApplicationContext(**"transaction-config.xml"**);  
 DecTransferService decTransferService = applicationContext.getBean(DecTransferService.**class**);  
 decTransferService.transfer(**"Kamini"**,**"Kishan"**,20000);  
 System.***out***.println(**"Done"**);  
 }  
}

**After Transfer**



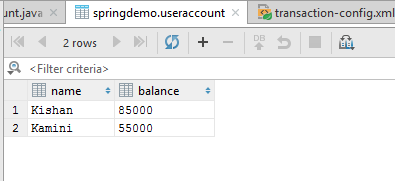


**5) Provide Appropriate Readonly attribute as per Method Behaviour.**

**package** com.spring.transaction;  
  
**import** org.springframework.jdbc.core.JdbcTemplate;  
**import** org.springframework.transaction.annotation.Isolation;  
**import** org.springframework.transaction.annotation.Transactional;  
  
**import** javax.sql.DataSource;  
**public class** AccountService  
{  
 **private** JdbcTemplate **jdbcTemplate**;  
 **private** DataSource **dataSource**;  
  
 **public** JdbcTemplate getJdbcTemplate() {  
 **return jdbcTemplate**;  
 }  
  
 **public void** setJdbcTemplate(JdbcTemplate jdbcTemplate) {  
 **this**.**jdbcTemplate** = jdbcTemplate;  
 }  
  
 **public** DataSource getDataSource() {  
 **return dataSource**;  
 }  
  
 **public void** setDataSource(DataSource dataSource) {  
 **this**.**dataSource** = dataSource;  
 }  
  
 @Transactional(isolation = Isolation.***READ\_UNCOMMITTED***,rollbackFor = RuntimeException.**class**)  
 **public void** create(String name, Integer balance) {  
 String SQL = **"insert into UserAccount (name, balance) values (?, ?)"**;  
 **jdbcTemplate**.update( SQL, **new** Object[]{name,balance});  
 System.***out***.println(**"Created Record Name = "** + name + **" Balance = "** + balance);  
  
 }  
  
 @Transactional(isolation = Isolation.***READ\_UNCOMMITTED***,**readOnly = true**)  
 **public void** getAccount(String name) {  
 String SQL = **"select** *\** **from useraccount where name = ?"**;  
 System.***out***.println(**jdbcTemplate**.queryForMap(SQL, **new** Object[]{name}));  
 }  
  
  
 **public void** delete(String name){  
 String SQL = **"delete from useraccount where name = ?"**;  
 **jdbcTemplate**.update(SQL, name);  
 System.***out***.println(**"Deleted Record with Name = "** + name );  
 **return**;  
 }  
  
 @Transactional(isolation = Isolation.***READ\_UNCOMMITTED***,rollbackFor = RuntimeException.**class**)  
 **public void** update(String name, Integer balance){  
 String SQL = **"update useraccount set balance = ? where name = ?"**;  
 **jdbcTemplate**.update(SQL, balance,name);  
 System.***out***.println(**"Updated Record with Name = "** + name );  
 **return**;  
 }  
  
}

**6) Create account\_transaction table with sender,receiver, balance\_transferred fields.  
Save values in this table on transfer amount. Even If any error occurs while saving values in transaction table,account table should get updated.**

**Before transaction**



**AccountTransaction.java**

**package** com.spring.transaction;  
  
 **import** org.springframework.jdbc.core.JdbcTemplate;  
 **import** org.springframework.transaction.annotation.Transactional;  
  
 **import** javax.sql.DataSource;  
  
**public class** AccountTransaction {  
 **private** JdbcTemplate **jdbcTemplate**;  
 **private** DataSource **dataSource**;  
  
 **public** JdbcTemplate getJdbcTemplate() {  
 **return jdbcTemplate**;  
 }  
  
 **public void** setJdbcTemplate(JdbcTemplate jdbcTemplate) {  
 **this**.**jdbcTemplate** = jdbcTemplate;  
 }  
  
 **public** DataSource getDataSource() {  
 **return dataSource**;  
 }  
  
 **public void** setDataSource(DataSource dataSource) {  
 **this**.**dataSource** = dataSource;  
 }  
  
 @Transactional(rollbackFor = Exception.**class**)  
 **public void** transfer(String name1, String name2, Integer balance) {  
  
 String SQL = **"update useraccount set balance = balance-? where name = ?"**;  
 **jdbcTemplate**.update(SQL, **new** Object[]{balance,name1});  
 String SQL2 = **"update useraccount set balance = balance+? where name = ?"**;  
 **jdbcTemplate**.update(SQL2,**new** Object[]{balance,name2});  
 System.***out***.println(**"Successfully transfered "** + balance + **" from "** + name1 + **" to "** + name2);  
 String SQL3 = **"insert into account\_transaction (sender,receiver,balance\_tranfer) values(?,?,?)"**;  
 **jdbcTemplate**.update(SQL3,**new** Object[]{name1,name2,balance});  
 System.***out***.println(**"Successfully inserted into account\_transaction"**);  
}  
  
 }

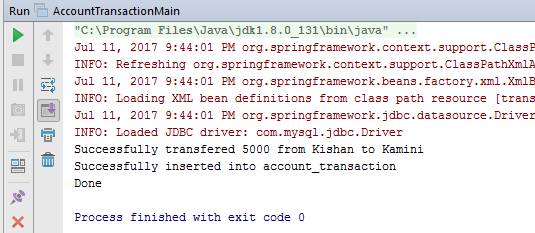
**AccountTransactionMain.java**

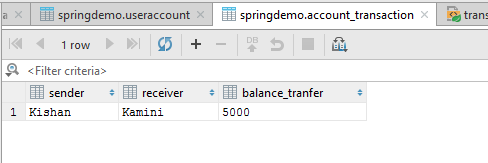
**package** com.spring.transaction;  
  
**import** org.springframework.context.ApplicationContext;  
**import** org.springframework.context.support.ClassPathXmlApplicationContext;  
  
**public class** AccountTransactionMain {  
 **public static void** main(String[] args) {  
 ApplicationContext applicationContext = **new** ClassPathXmlApplicationContext(**"transaction-config.xml"**);  
 AccountTransaction accountTransaction = applicationContext.getBean(AccountTransaction.**class**);  
 accountTransaction.transfer(**"Kishan"**,**"Kamini"**,5000);  
 System.***out***.println(**"Done"**);  
 }  
}

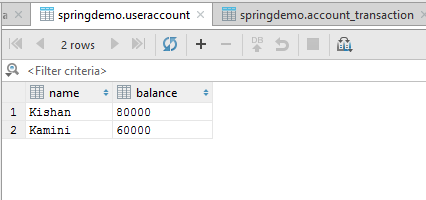
**transaction-config.xml**

<**beans xmlns="http://www.springframework.org/schema/beans"  
 xmlns:tx="http://www.springframework.org/schema/tx"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.springframework.org/schema/beans  
 http://www.springframework.org/schema/beans/spring-beans.xsd  
 http://www.springframework.org/schema/tx  
 http://www.springframework.org/schema/tx/spring-tx.xsd"**>  
  
 <**tx:annotation-driven transaction-manager="transactionManager"**/>  
  
 <**bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource"**>  
 <**property name="url" value="jdbc:mysql://localhost:3306/springDemo"**/>  
 <**property name="driverClassName" value="com.mysql.jdbc.Driver"**/>  
 <**property name="username" value="root"**/>  
 <**property name="password" value="password"**></**property**>  
 </**bean**>  
  
 <**bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate"**>  
 <**property name="dataSource" ref="dataSource"**/>  
 </**bean**>  
  
 <**bean id="transactionManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager"**>  
 <**property name="dataSource" ref="dataSource"**/>  
 </**bean**>  
  
  
 <**bean id="decTransferService" class="com.spring.transaction.DecTransferService"**>  
 <**property name="jdbcTemplate" ref="jdbcTemplate"** />  
 </**bean**>  
  
 <**bean id="accountTransaction" class="com.spring.transaction.AccountTransaction"**>  
 <**property name="jdbcTemplate" ref="jdbcTemplate"** />  
 </**bean**>  
  
</**beans**>

**After transaction**







**If exception occurred**

**AccountTransaction.java**

**package** com.spring.transaction;  
  
 **import** org.springframework.jdbc.core.JdbcTemplate;  
 **import** org.springframework.transaction.annotation.Transactional;  
  
 **import** javax.sql.DataSource;  
  
**public class** AccountTransaction {  
 **private** JdbcTemplate **jdbcTemplate**;  
 **private** DataSource **dataSource**;  
  
 **public** JdbcTemplate getJdbcTemplate() {  
 **return jdbcTemplate**;  
 }  
  
 **public void** setJdbcTemplate(JdbcTemplate jdbcTemplate) {  
 **this**.**jdbcTemplate** = jdbcTemplate;  
 }  
  
 **public** DataSource getDataSource() {  
 **return dataSource**;  
 }  
  
 **public void** setDataSource(DataSource dataSource) {  
 **this**.**dataSource** = dataSource;  
 }  
  
 @Transactional(rollbackFor = Exception.**class**)  
 **public void** transfer(String name1, String name2, Integer balance) {  
  
 String SQL = **"update useraccount set balance = balance-? where name = ?"**;  
 **jdbcTemplate**.update(SQL, **new** Object[]{balance,name1});  
 String SQL2 = **"update useraccount set balance = balance+? where name = ?"**;  
 **jdbcTemplate**.update(SQL2,**new** Object[]{balance,name2});  
 System.***out***.println(**"Successfully transfered "** + balance + **" from "** + name1 + **" to "** + name2);  
 String SQL3 = **"insert into account\_transaction (sender,receiver,balance\_tranfer) values(?,?,?)"**;  
 **jdbcTemplate**.update(SQL3,**new** Object[]{name1,name2,balance});  
 System.***out***.println(**"Successfully inserted into account\_transaction"**);  
 **throw new** RuntimeException();  
  
  
 }  
  
 }

**AccountTransactionMain.java**

**package** com.spring.transaction;  
  
**import** org.springframework.context.ApplicationContext;  
**import** org.springframework.context.support.ClassPathXmlApplicationContext;  
  
**public class** AccountTransactionMain {  
 **public static void** main(String[] args) {  
 ApplicationContext applicationContext = **new** ClassPathXmlApplicationContext(**"transaction-config.xml"**);  
 AccountTransaction accountTransaction = applicationContext.getBean(AccountTransaction.**class**);  
 accountTransaction.transfer(**"Kishan"**,**"Kamini"**,5000);  
 System.***out***.println(**"Done"**);  
 }  
}

