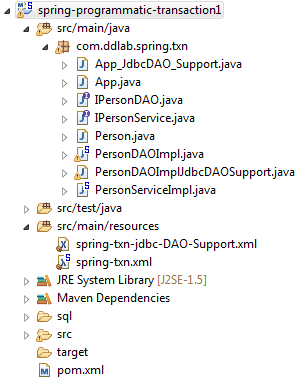
Programmatic Transaction Manager in Spring

**Using PlatformTransactionManager**

**Project Structure**



**Maven Configuration (pom.xml)**

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>spring-programmatic-transaction1</groupId>

<artifactId>spring-programmatic-transaction1</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>spring-programmatic-transaction1</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<spring.version>4.0.5.RELEASE</spring.version>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-beans</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>1.2.14</version>

</dependency>

<dependency>

<groupId>commons-dbcp</groupId>

<artifactId>commons-dbcp</artifactId>

<version>1.2.2</version>

</dependency>

<!-- For actual application, we use MYSQL -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>5.1.30</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.11</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

**Spring Configuration (spring-txn.xml)**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:aop=*"http://www.springframework.org/schema/aop"*

xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:jdbc=*"http://www.springframework.org/schema/jdbc"* xmlns:tx=*"http://www.springframework.org/schema/tx"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.0.xsd*

*http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-4.0.xsd*

*http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.0.xsd*

*http://www.springframework.org/schema/jdbc http://www.springframework.org/schema/jdbc/spring-jdbc-4.0.xsd*

*http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-4.0.xsd"*>

**<context:component-scan base-package=*"com.ddlab.spring.txn"*/>**

**<bean id=*"dataSource"* class=*"org.springframework.jdbc.datasource.DriverManagerDataSource"*>**

**<property name=*"driverClassName"* value=*"com.mysql.jdbc.Driver"*></property>**

**<property name=*"url"* value=*"jdbc:mysql://localhost:3306/test"*></property>**

**<property name=*"username"* value=*"deba"*></property>**

**<property name=*"password"* value=*"deba"*></property>**

**</bean>**

<!-- Initialization for TransactionManager -->

<bean id=*"transactionManager"* class=*"org.springframework.jdbc.datasource.DataSourceTransactionManager"*>

<property name=*"dataSource"* ref=*"dataSource"*></property>

</bean>

<bean id=*"jdbcTemplate"* class=*"org.springframework.jdbc.core.JdbcTemplate"*>

<property name=*"dataSource"* ref=*"dataSource"*></property>

</bean>

<bean id=*"personDAO"* class=*"com.ddlab.spring.txn.PersonDAOImpl"*>

<property name=*"txnManager"* ref = *"transactionManager"*/>

<property name=*"jdbcTemplate"* ref = *"jdbcTemplate"*/>

</bean>

<bean id=*"personService"* class=*"com.ddlab.spring.txn.PersonServiceImpl"*/>

</beans>

**Entity Layer**

**Person.java**

**package** com.ddlab.rnd.spring.jdbc;

**public** **class** Person {

**private** **int** id;

**private** String firstName;

**private** String lastName;

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getFirstName() {

**return** firstName;

}

**public** **void** setFirstName(String firstName) {

**this**.firstName = firstName;

}

**public** String getLastName() {

**return** lastName;

}

**public** **void** setLastName(String lastName) {

**this**.lastName = lastName;

}

}

**DAO or Repository layer( interface and implementation)**

**IPersonDAO.java**

**package** com.ddlab.rnd.spring.jdbc;

**import** java.util.List;

**public** **interface** IPersonDAO {

**public** **void** addPerson(Person person);

**public** **void** updatePerson(Person person);

**public** **void** deletePerson(Person person);

**public** List<Person> getAllPersons();

**public** Person getPersonById(**int** id);

}

**PersonDAOImpl.java**

**package** com.ddlab.spring.txn;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.util.List;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.jdbc.core.RowMapper;

**import** org.springframework.transaction.PlatformTransactionManager;

**import** org.springframework.transaction.TransactionDefinition;

**import** org.springframework.transaction.TransactionStatus;

**import** org.springframework.transaction.support.DefaultTransactionDefinition;

**public** **class** PersonDAOImpl **implements** IPersonDAO {

**private JdbcTemplate jdbcTemplate;**

**private PlatformTransactionManager txnManager;**

**public** **void** setJdbcTemplate(JdbcTemplate jdbcTemplate) {

**this**.jdbcTemplate = jdbcTemplate;

}

**public** **void** setTxnManager(PlatformTransactionManager txnManager) {

**this**.txnManager = txnManager;

}

**public** **void** addPerson(Person person) {

DefaultTransactionDefinition txnDefn = **new** DefaultTransactionDefinition();

//You can set propagation behaviour like this

// txnDefn.setPropagationBehavior(DefaultTransactionDefinition.PROPAGATION\_REQUIRES\_NEW);

TransactionStatus status = txnManager.getTransaction(txnDefn);

**try** {

String insertQuery = "insert into person (FIRSTNAME,LASTNAME) values(?,?)";

Object[] params = **new** Object[] {person.getFirstName(), person.getLastName() };

jdbcTemplate.update(insertQuery, params);

txnManager.commit(status);

} **catch** (Exception e) {

txnManager.rollback(status);

}

}

**public** **void** updatePerson(Person person) {

TransactionDefinition txnDefn = **new** DefaultTransactionDefinition();

TransactionStatus status = txnManager.getTransaction(txnDefn);

**try** {

String queryString = "UPDATE person SET FIRSTNAME = ? WHERE id = ?";

Object[] params = **new** Object[] {person.getFirstName(), person.getId() };

**jdbcTemplate.update(queryString, params);**

**txnManager.commit(status);**

} **catch** (Exception e) {

**txnManager.rollback(status);**

}

}

**public** **void** deletePerson(Person person) {

TransactionDefinition txnDefn = **new** DefaultTransactionDefinition();

TransactionStatus status = txnManager.getTransaction(txnDefn);

**try** {

String queryString = "DELETE from person WHERE id = ?";

Object[] params = **new** Object[] {person.getId() };

**jdbcTemplate.update(queryString, params);**

**txnManager.commit(status);**

} **catch** (Exception e) {

txnManager.rollback(status);

}

}

**public** List<Person> getAllPersons() {

List<Person> personList = **null**;

**try** {

String queryString = "select \* from person";

**personList = jdbcTemplate.query(queryString, new PersonMapper() );**

} **catch** (Exception e) {

e.printStackTrace();

}

**return** personList;

}

**public** Person getPersonById(**int** id) {

String queryString = "select \* from person where id = ?";

Object[] params = **new** Object[]{id};

Person p = jdbcTemplate.queryForObject(queryString, params , **new** PersonMapper() );

**return** p;

}

//RowMapper implementation

**private** **class** PersonMapper **implements** RowMapper {

**public** Person mapRow(ResultSet rs, **int** rowNum) **throws** SQLException {

Person p = **new** Person();

p.setId(rs.getInt("ID"));

p.setFirstName(rs.getString("FIRSTNAME"));

p.setLastName(rs.getString("LASTNAME"));

**return** p;

}

}

}

**Service Layer ( interface and implementation )**

**IPersonService.java**

**package** com.ddlab.rnd.spring.jdbc;

**import** java.util.List;

**public** **interface** IPersonService {

**public** **void** createPerson( Person person );

**public** **void** updatePerson( Person person );

**public** **void** deletePerson( Person person );

**public** List<Person> getAllPersons();

**public** Person getPersonById( **int** id );

}

**PersonServiceImpl.java**

**package** com.ddlab.rnd.spring.jdbc;

**import** java.util.List;

**import** org.springframework.beans.factory.annotation.Autowired;

**public** **class** PersonServiceImpl **implements** IPersonService {

@Autowired

**private** IPersonDAO personDAO;

**public** **void** createPerson(Person person) {

personDAO.addPerson(person);

}

**public** **void** updatePerson(Person person) {

personDAO.updatePerson(person);

}

**public** **void** deletePerson(Person person) {

personDAO.deletePerson(person);

}

**public** List<Person> getAllPersons() {

**return** personDAO.getAllPersons();

}

**public** Person getPersonById(**int** id) {

**return** personDAO.getPersonById(id);

}

}

**Test Class (App.java)**

**package** com.ddlab.spring.txn;

**import** java.util.List;

**import** org.springframework.context.ApplicationContext;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** App {

**public** **static** **void** main(String[] args) {

Person person = **new** Person();

//No need to set ID as in MySQL, it is auto increment

person.setFirstName("Arzoo");

person.setLastName("Mishra");

ApplicationContext context = **null**;

**try** {

context = **new** ClassPathXmlApplicationContext("spring-txn.xml");

IPersonService personService = (IPersonService) context.getBean("personService");

//Create a person

personService.createPerson(person);

//Update a person

Person updatedPerson = **new** Person();

updatedPerson.setId(2);

updatedPerson.setFirstName("Bibhu");

personService.updatePerson(updatedPerson);

//Get all persons

List<Person> personList = personService.getAllPersons();

**for**( Person p : personList ) {

System.***out***.println(p.getId()+"----"+p.getFirstName()+"----"+p.getLastName());

}

//Get person by Id

Person p = personService.getPersonById(3);

System.***out***.println(p.getId()+"----"+p.getFirstName()+"----"+p.getLastName());

System.***out***.println("Saved Successfully...");

//Delete a person

Person deletedPerson = **new** Person();

deletedPerson.setId(1);

personService.deletePerson(deletedPerson);

} **catch** (Exception e) {

e.printStackTrace();

}

}

}

**How to use JDBCDAOSupport in the above caase**

**Spring Configuration (spring-txn-jdbc-DAO-support.xml)**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:aop=*"http://www.springframework.org/schema/aop"*

xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:jdbc=*"http://www.springframework.org/schema/jdbc"* xmlns:tx=*"http://www.springframework.org/schema/tx"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.0.xsd*

*http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-4.0.xsd*

*http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.0.xsd*

*http://www.springframework.org/schema/jdbc http://www.springframework.org/schema/jdbc/spring-jdbc-4.0.xsd*

*http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-4.0.xsd"*>

<context:component-scan base-package=*"com.ddlab.spring.txn"*/>

<bean id=*"dataSource"* class=*"org.springframework.jdbc.datasource.DriverManagerDataSource"*>

<property name=*"driverClassName"* value=*"com.mysql.jdbc.Driver"*></property>

<property name=*"url"* value=*"jdbc:mysql://localhost:3306/test"*></property>

<property name=*"username"* value=*"deba"*></property>

<property name=*"password"* value=*"deba"*></property>

</bean>

<!-- Initialization for TransactionManager -->

<bean id=*"transactionManager"* class=*"org.springframework.jdbc.datasource.DataSourceTransactionManager"*>

<property name=*"dataSource"* ref=*"dataSource"*></property>

</bean>

<bean id=*"personDAO"* class=*"com.ddlab.spring.txn.PersonDAOImplJdbcDAOSupport"*>

<property name=*"txnManager"* ref = *"transactionManager"*/>

<property name=*"dataSource"* ref=*"dataSource"*></property> <!-- Important to note -->

</bean>

<bean id=*"personService"* class=*"com.ddlab.spring.txn.PersonServiceImpl"*/>

</beans>

**PersonDAOImplJdbcDAOSupport.java**

**package** com.ddlab.spring.txn;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.util.List;

**import** org.springframework.jdbc.core.RowMapper;

**import** org.springframework.jdbc.core.support.JdbcDaoSupport;

**import** org.springframework.transaction.PlatformTransactionManager;

**import** org.springframework.transaction.TransactionDefinition;

**import** org.springframework.transaction.TransactionStatus;

**import** org.springframework.transaction.support.DefaultTransactionDefinition;

**public** **class** **PersonDAOImplJdbcDAOSupport** **extends** JdbcDaoSupport **implements** IPersonDAO {

**private** PlatformTransactionManager txnManager;

**public** **void** setTxnManager(PlatformTransactionManager txnManager) {

**this**.txnManager = txnManager;

}

**public** **void** addPerson(Person person) {

**TransactionDefinition txnDefn = new DefaultTransactionDefinition();**

**TransactionStatus status = txnManager.getTransaction(txnDefn);**

**try** {

String insertQuery = "insert into person (FIRSTNAME,LASTNAME) values(?,?)";

Object[] params = **new** Object[] {person.getFirstName(), person.getLastName() };

**getJdbcTemplate().update(insertQuery, params);**

**txnManager.commit(status);**

} **catch** (Exception e) {

txnManager.rollback(status);

}

}

**public** **void** updatePerson(Person person) {

**TransactionDefinition txnDefn = new DefaultTransactionDefinition();**

**TransactionStatus status = txnManager.getTransaction(txnDefn);**

**try** {

String queryString = "UPDATE person SET FIRSTNAME = ? WHERE id = ?";

Object[] params = **new** Object[] {person.getFirstName(), person.getId() };

**getJdbcTemplate().update(queryString, params);**

**txnManager.commit(status);**

} **catch** (Exception e) {

**txnManager.rollback(status);**

}

}

**public** **void** deletePerson(Person person) {

TransactionDefinition txnDefn = **new** DefaultTransactionDefinition();

TransactionStatus status = txnManager.getTransaction(txnDefn);

**try** {

String queryString = "DELETE from person WHERE id = ?";

Object[] params = **new** Object[] {person.getId() };

**getJdbcTemplate().update(queryString, params);**

**txnManager.commit(status);**

} **catch** (Exception e) {

**txnManager.rollback(status);**

}

}

**public** List<Person> getAllPersons() {

List<Person> personList = **null**;

**try** {

String queryString = "select \* from person";

personList = getJdbcTemplate().query(queryString, **new** PersonMapper() );

} **catch** (Exception e) {

e.printStackTrace();

}

**return** personList;

}

**public** Person getPersonById(**int** id) {

String queryString = "select \* from person where id = ?";

Object[] params = **new** Object[]{id};

Person p = getJdbcTemplate().queryForObject(queryString, params , **new** PersonMapper() );

**return** p;

}

//RowMapper implementation

**private** **class** PersonMapper **implements** RowMapper {

**public** Person mapRow(ResultSet rs, **int** rowNum) **throws** SQLException {

Person p = **new** Person();

p.setId(rs.getInt("ID"));

p.setFirstName(rs.getString("FIRSTNAME"));

p.setLastName(rs.getString("LASTNAME"));

**return** p;

}

}

}

**Using TransactionTemplate**

**Spring Configuration (spring-txn.xml)**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:aop=*"http://www.springframework.org/schema/aop"*

xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:jdbc=*"http://www.springframework.org/schema/jdbc"* xmlns:tx=*"http://www.springframework.org/schema/tx"*

xsi:schemaLocation=*"http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-4.0.xsd*

*http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-4.0.xsd*

*http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-4.0.xsd*

*http://www.springframework.org/schema/jdbc http://www.springframework.org/schema/jdbc/spring-jdbc-4.0.xsd*

*http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-4.0.xsd"*>

<context:component-scan base-package=*"com.ddlab.spring.txn"*/>

<bean id=*"dataSource"* class=*"org.springframework.jdbc.datasource.DriverManagerDataSource"*>

<property name=*"driverClassName"* value=*"com.mysql.jdbc.Driver"*></property>

<property name=*"url"* value=*"jdbc:mysql://localhost:3306/test"*></property>

<property name=*"username"* value=*"deba"*></property>

<property name=*"password"* value=*"deba"*></property>

</bean>

<!-- Initialization for TransactionManager -->

<bean id=*"transactionManager"* class=*"org.springframework.jdbc.datasource.DataSourceTransactionManager"*>

<property name=*"dataSource"* ref=*"dataSource"*></property>

</bean>

<bean id=*"transactionTemplate"* class=*"org.springframework.transaction.support.TransactionTemplate"*>

<property name=*"transactionManager"* ref=*"transactionManager"*/>

</bean>

<bean id=*"jdbcTemplate"* class=*"org.springframework.jdbc.core.JdbcTemplate"*>

<property name=*"dataSource"* ref=*"dataSource"*></property>

</bean>

<bean id=*"personDAO"* class=*"com.ddlab.rnd.spring.txn.PersonDAOImpl"*>

<property name=*"txnTemplate"* ref = *"transactionTemplate"*/>

<property name=*"jdbcTemplate"* ref = *"jdbcTemplate"*/>

</bean>

<bean id=*"personService"* class=*"com.ddlab.rnd.spring.txn.PersonServiceImpl"*/>

</beans>

**Entity Layer**

**Person.java**

**package** com.ddlab.rnd.spring.jdbc;

**public** **class** Person {

**private** **int** id;

**private** String firstName;

**private** String lastName;

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getFirstName() {

**return** firstName;

}

**public** **void** setFirstName(String firstName) {

**this**.firstName = firstName;

}

**public** String getLastName() {

**return** lastName;

}

**public** **void** setLastName(String lastName) {

**this**.lastName = lastName;

}

}

**DAO or Repository layer( interface and implementation)**

**IPersonDAO.java**

**package** com.ddlab.rnd.spring.jdbc;

**import** java.util.List;

**public** **interface** IPersonDAO {

**public** **void** addPerson(Person person);

**public** **void** updatePerson(Person person);

**public** **void** deletePerson(Person person);

**public** List<Person> getAllPersons();

**public** Person getPersonById(**int** id);

}

**PersonDAOImpl.java**

**package** com.ddlab.rnd.spring.txn;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** java.util.List;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.jdbc.core.RowMapper;

**import** org.springframework.transaction.support.TransactionTemplate;

**public** **class** PersonDAOImpl **implements** IPersonDAO {

**private JdbcTemplate jdbcTemplate;**

**private TransactionTemplate txnTemplate;**

**public** **void** setJdbcTemplate(JdbcTemplate jdbcTemplate) {

**this**.jdbcTemplate = jdbcTemplate;

}

**public** **void** setTxnTemplate(TransactionTemplate txnTemplate) {

**this**.txnTemplate = txnTemplate;

}

**public** **void** addPerson(Person person) {

String insertQuery = "insert into person (FIRSTNAME,LASTNAME) values(?,?)";

Object[] params = **new** Object[] {person.getFirstName(), person.getLastName() };

**NoResultTransactionCallbackImpl txnCallbackImpl = new NoResultTransactionCallbackImpl(insertQuery, params, jdbcTemplate);**

txnTemplate.execute(txnCallbackImpl);

}

**public** **void** updatePerson(Person person) {

String queryString = "UPDATE person SET FIRSTNAME = ? WHERE id = ?";

Object[] params = **new** Object[] {person.getFirstName(), person.getId() };

**NoResultTransactionCallbackImpl txnCallbackImpl = new NoResultTransactionCallbackImpl(queryString, params, jdbcTemplate);**

txnTemplate.execute(txnCallbackImpl);

}

**public** **void** deletePerson(Person person) {

String queryString = "DELETE from person WHERE id = ?";

Object[] params = **new** Object[] {person.getId() };

**NoResultTransactionCallbackImpl txnCallbackImpl = new NoResultTransactionCallbackImpl(queryString, params, jdbcTemplate);**

txnTemplate.execute(txnCallbackImpl);

}

**public** List<Person> getAllPersons() {

List<Person> personList = **null**;

**try** {

String queryString = "select \* from person";

personList = jdbcTemplate.query(queryString, **new** PersonMapper() );

} **catch** (Exception e) {

e.printStackTrace();

}

**return** personList;

}

**public** Person getPersonById(**int** id) {

String queryString = "select \* from person where id = ?";

Object[] params = **new** Object[]{id};

**WithResultTransactionCallbackImpl resultTxnCallback = new WithResultTransactionCallbackImpl(queryString, params, jdbcTemplate);**

**return** txnTemplate.execute(resultTxnCallback);

}

//RowMapper implementation

**private** **class** PersonMapper **implements** RowMapper {

**public** Person mapRow(ResultSet rs, **int** rowNum) **throws** SQLException {

Person p = **new** Person();

p.setId(rs.getInt("ID"));

p.setFirstName(rs.getString("FIRSTNAME"));

p.setLastName(rs.getString("LASTNAME"));

**return** p;

}

}

}

**NoResultTransactionCallbackImpl.java**

**package** com.ddlab.rnd.spring.txn;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.transaction.TransactionStatus;

**import** org.springframework.transaction.support.TransactionCallbackWithoutResult;

**public** **class** NoResultTransactionCallbackImpl **extends** TransactionCallbackWithoutResult {

**private** String queryString;

**private** Object[] params;

private JdbcTemplate jdbcTemplate;

**public** NoResultTransactionCallbackImpl( String queryString ,Object[] params , JdbcTemplate jdbcTemplate ) {

**this**.queryString = queryString;

**this**.params = params;

**this**.jdbcTemplate = jdbcTemplate;

}

@Override

**protected** **void** doInTransactionWithoutResult(TransactionStatus txnStatus) {

**try** {

**jdbcTemplate.update(queryString, params);**

// throw new NullPointerException("Some unwanted exception");

} **catch** (Exception e) {

txnStatus.setRollbackOnly();

}

}

}

**WithResultTransactionCallbackImpl.java**

**package** com.ddlab.rnd.spring.txn;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

**import** org.springframework.jdbc.core.JdbcTemplate;

**import** org.springframework.jdbc.core.RowMapper;

**import** org.springframework.transaction.TransactionStatus;

**import** org.springframework.transaction.support.TransactionCallback;

**public** **class** WithResultTransactionCallbackImpl **implements** TransactionCallback<Person> {

**private** String queryString;

**private** Object[] params;

**private** JdbcTemplate jdbcTemplate;

**public** WithResultTransactionCallbackImpl( String queryString ,Object[] params , JdbcTemplate jdbcTemplate ) {

**this**.queryString = queryString;

**this**.params = params;

**this**.jdbcTemplate = jdbcTemplate;

}

**public** Person doInTransaction( TransactionStatus txnStatus ) {

Person p = jdbcTemplate.queryForObject(queryString, params , **new** PersonMapper() );

**return** p;

}

**private** **class** PersonMapper **implements** RowMapper { **//RowMapper implementation**

**public** Person mapRow(ResultSet rs, **int** rowNum) **throws** SQLException {

Person p = **new** Person();

p.setId(rs.getInt("ID"));

p.setFirstName(rs.getString("FIRSTNAME"));

p.setLastName(rs.getString("LASTNAME"));

**return** p;

}

}

}

**Service Layer ( interface and implementation )**

**IPersonService.java**

**package** com.ddlab.rnd.spring.jdbc;

**import** java.util.List;

**public** **interface** IPersonService {

**public** **void** createPerson( Person person );

**public** **void** updatePerson( Person person );

**public** **void** deletePerson( Person person );

**public** List<Person> getAllPersons();

**public** Person getPersonById( **int** id );

}

**PersonServiceImpl.java**

**package** com.ddlab.rnd.spring.jdbc;

**import** java.util.List;

**import** org.springframework.beans.factory.annotation.Autowired;

**public** **class** PersonServiceImpl **implements** IPersonService {

@Autowired

**private** IPersonDAO personDAO;

**public** **void** createPerson(Person person) {

personDAO.addPerson(person);

}

**public** **void** updatePerson(Person person) {

personDAO.updatePerson(person);

}

**public** **void** deletePerson(Person person) {

personDAO.deletePerson(person);

}

**public** List<Person> getAllPersons() {

**return** personDAO.getAllPersons();

}

**public** Person getPersonById(**int** id) {

**return** personDAO.getPersonById(id);

}

}

**Test Class (App.java)**

**package** com.ddlab.rnd.spring.txn;

**import** java.util.List;

**import** org.springframework.context.ApplicationContext;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** App {

**public** **static** **void** main(String[] args) {

Person person = **new** Person();

//No need to set ID as in MySQL, it is auto increment

person.setFirstName("Lilu");

person.setLastName("Mishra");

ApplicationContext context = **null**;

**try** {

context = **new** ClassPathXmlApplicationContext("spring-txn.xml");

IPersonService personService = (IPersonService) context.getBean("personService");

//Create a person

personService.createPerson(person);

//Update a person

// Person updatedPerson = new Person();

// updatedPerson.setId(2);

// updatedPerson.setFirstName("Bibhu");

// personService.updatePerson(updatedPerson);

//

// //Get all persons

// List<Person> personList = personService.getAllPersons();

// for( Person p : personList ) {

// System.out.println(p.getId()+"----"+p.getFirstName()+"----"+p.getLastName());

// }

//

//Get person by Id

Person p = personService.getPersonById(5);

System.***out***.println(p.getId()+"----"+p.getFirstName()+"----"+p.getLastName());

System.***out***.println("Saved Successfully...");

// //Delete a person

// Person deletedPerson = new Person();

// deletedPerson.setId(1);

// personService.deletePerson(deletedPerson);

} **catch** (Exception e) {

e.printStackTrace();

}

}

}