**Spring with hibernate**

Hibernate :

* Object/Relational mapping tool

The core interfaces of Hibernate framework are:

* Configuration
* SessionFactory
* Session
* Query
* Criteria
* Transaction

Session.save - > we can save if there is no primary key is defined in the table.

Session.saveorUpdate -> we can save data even if primary key is definend in table.

**HQL simplifies query writing process while in case of JDBC we need to write query**.

**Spring-config.xml**

<bean id="dataSource"

class="org.springframework.jdbc.datasource.DriverManagerDataSource">

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

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

<property name="username" value="root"/>

<property name="password" value="syntel123$"/>

</bean>

**Log4j.properties**

### direct log messages to stdout ###

log4j.appender.stdout=org.apache.log4j.ConsoleAppender

log4j.appender.stdout.Target=System.out

log4j.appender.stdout.layout=org.apache.log4j.PatternLayout

log4j.appender.stdout.layout.ConversionPattern=%d{ABSOLUTE} %5p %c{1}:%L - %m%n

### set log levels - for more verbose logging change 'info' to 'debug' ###

log4j.rootLogger=error, stdout

**Bean**

@Entity

@Table(name = "PERSON")

public class Person {

private Integer id;

private String name;

private String email;

@Id

@GeneratedValue

@Column(name = "ID")

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

**PersonDAO**

public void insert(Person person) {

Session session = getSessionFactory().getCurrentSession();

session.beginTransaction();

session.save(person);

session.getTransaction().commit();

}

public List<Person> selectAll() {

Session session = getSessionFactory().getCurrentSession();

session.beginTransaction();

Criteria criteria = session.createCriteria(Person.class);

List<Person> persons = (List<Person>) criteria.list();

session.getTransaction().commit();

return persons;

}

**Services**

@Component

public class PersonService {

private PersonDao personDao;

public PersonDao getPersonDao() {

return personDao;

}

@Autowired

public void setPersonDao(PersonDao personDao) {

this.personDao = personDao;

}

public void addPerson(Person person) {

getPersonDao().insert(person);

}

public List<Person> fetchAllPersons() {

return getPersonDao().selectAll();

}

}

**Main Class**

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class TestSpringHibernateSessionFactoryAnnotation {

public static void main(String[] args) {

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\* BEGINNING PROGRAM \*\*\*\*\*\*\*\*\*\*\*\*\*\*");

ApplicationContext context = new ClassPathXmlApplicationContext("spring-config.xml");

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

Person person = new Person();

person.setName("Alba");

person.setEmail("alba@gmail.com");

personService.addPerson(person);

System.out.println("Person : " + person + " added successfully");

List<Person> persons = personService.fetchAllPersons();

System.out.println("The list of all persons = " + persons);

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\* ENDING PROGRAM \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

}