Spring Framework
Module 6 – Поддержка ORM

Evgeniy Krivosheev
Vyacheslav Yakovenko
Last update: Feb, 2012

Содержание

- Обзор ORM модуля
- Обзор устаревшего API
- Преимущества в работе с ORM с использованием Spring
- JPA + Hibernate
- Инициализация JPA EntityManagerFactory;
- Пример запроса с использованием JPA EntityManagerFactory;

Spring :: Обзор ORM модуля

- org.springframework.orm
- org.springframework.orm.hibernate3
- org.springframework.orm.hibernate4
- org.springframework.orm.ibatis
- org.springframework.orm.jdo
- org.springframework.orm.jpa

Spring :: ORM - Обзор устаревшего API

- В Spring Framework v.2.* поддержка ORM осуществлялась на базе классов XxxTemplate:
  - @Deprecated JpaTemplate, @Deprecated JpaCallback<T>;
  - @Deprecated JdoTemplate, @Deprecated JdoCallback<T>; ;
  - org.springframework.orm.hibernate3.HibernateTemplate ;
- Трендом 3-й версии в Spring Framework является уход от XxxTemplate и возврат к поддержке максимально «нативного» API, конкретного ORM:
  - JPA:
    - LocalEntityManagerFactoryBean
    - LocalContainerEntityManagerFactoryBean
  - Hibernate:
    - org.springframework.orm.hibernate3.LocalSessionFactoryBean
    - org.springframework.orm.hibernate4.LocalSessionFactoryBean

Spring :: Преимущества в работе с ORM

LUXOFT TRAINING

- Тестирование ;
- Обработку исключений ;
- Управление ресурсами (DataSource, mappings) ;
- Управление транзакциями ;

Spring :: JPA + Hibernate

LUXOFT TRAINING

- Т.к. в настоящее время промышленным стандартом является JPA, а Hibernate (начиная с v.3.2) является его реализацией. В ходе тренинга мы рассмотрим именно этот вариант: Использование Hibernate 4 в качестве JPA 2.0 провайдера;
- Следует обратить внимание на то, что в следующих версиях Spring Framework JPA v.1.0 поддерживаться не будет;
- Кроме этого Spring Framework, начиная с v.3.0, не поддерживает версии Hibernate ниже 3.2;

Spring :: JPA + Hibernate

LUXOFT TRAINING

- Т.к. в настоящее время промышленным стандартом является JPA, а Hibernate (начиная с v.3.2) является его реализацией. В ходе тренинга мы рассмотрим именно этот вариант: Использование Hibernate 4 в качестве JPA 2.0 провайдера;
- Следует обратить внимание на то, что в следующих версиях Spring Framework JPA v.1.0 поддерживаться не будет;
- Кроме этого Spring Framework, начиная с v.3.0, не поддерживает версии Hibernate ниже 3.2;

Spring :: Инициализация JPA

В данный момент Spring поддерживает три варианта инициализации JPA EntityManagerFactory:

- Получение EntityManagerFactory из JNDI;
- Используя LocalEntityManagerFactoryBean:
  - Не требуется, обязательный с точки зрения стандарта JPA, persistence.xml;
  - Используется в небольших приложениях, прототипах и с целью организовать тестирование;
- LocalContainerEntityManagerFactoryBean – фабрика предоставляющая максимум возможностей:
  - Поддерживает множественные persistence unit-ы;
  - Конфигурируется под различные сервера приложений (WebLogic, OC4J, GlassFish, Tomcat, Resin, JBoss)

Spring :: Инициализация JPA

Получение EntityManagerFactory из JNDI:

```
<jee:jndi-lookup id="myEmf"
      jndi-name="persistence/myPersistenceUnit"/>
```

Spring :: Инициализация JPA

Используя LocalEntityManagerFactoryBean:

```
<bean id="myEmf"
   class="org.springframework.orm.jpa.LocalEntityManagerFactoryBean">
   <property name="persistenceUnitName"   value="myPersistenceUnit"/>
</bean>
```

Spring :: Инициализация JPA

LocalContainerEntityManagerFactoryBean – фабрика предоставляющая максимум возможностей:

```
<bean id="myEmf"
   class="org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean">
   <property name="dataSource" ref="dataSource"/>
   <property name="loadTimeWeaver">
        <bean
   class="org.springframework.instrument.classloading.InstrumentationLoadTimeWeaver"/>
   </property>
   <property name="persistenceUnitName" value="persistenceUnitName" />
</bean>
```

Spring :: Инициализация JPA

В случае использования Hibernate 4, в качестве JPA провайдера, понадобится дополнительная конфигурация.

application-context.xml:

```
<bean id="lcemf" class="org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean">
    <property name="loadTimeWeaver">
        <bean class="org.springframework.instrument.classloading.InstrumentationLoadTimeWeaver" />
    </property>
    <property name="dataSource" ref="dataSource"></property>
    <property name="persistenceUnitName" value="springframework.lab.orm.jpa" />
    <property name="persistenceProviderClass"     value="org.hibernate.ejb.HibernatePersistence"/>
</bean>

<bean id="countryDao" class="lab.dao.jpa.CountryJpaDaoImpl" />
```

META-INF/persistence.xml:

```
<persistence>
    <persistence-unit name="springframework.lab.orm.jpa">
        <class>lab.model.Country</class>
        <properties>
            <property name="hibernate.show_sql" value="true" />
            <property name="hibernate.hbm2ddl.auto" value="create" />
        </properties>
    </persistence-unit>
</persistence>
```

Spring :: JPA, Пример запроса

LUXOFT TRAINING

Пример связывания:

```
@Repository
public class CountryJpaDaoImpl {
   protected EntityManagerFactory emf;

   @PersistenceUnit
   public void setEntityManagerFactory(EntityManagerFactory emf) {
       this.emf = emf;
   }

   public List<Country> getAllCountries() {
       EntityManager em = emf.createEntityManager();
       return = em.createQuery("from Country", Country.class);
   }
}
```

Spring :: JPA, Пример запроса

Связывание происходит через метод аннотированный как **@PersistenceUnit**:

- Spring обращается к **LocalContainerEntityManagerFactoryBean**;
- Получает из него **EntityManagerFactory**;
- Используя механизм автосвязывания, выполняет инъекцию в имплементацию DAO;
- Имея в наличии экземпляр **EntityManagerFactory** обращаемся к нему для выполнения запросов.

Упражнения

№: 7 : «Использование ORM в Spring при работе с данными»

- 45 мин – самостоятельная работа;
- 15 мин – обсуждение;

Вопросы!?