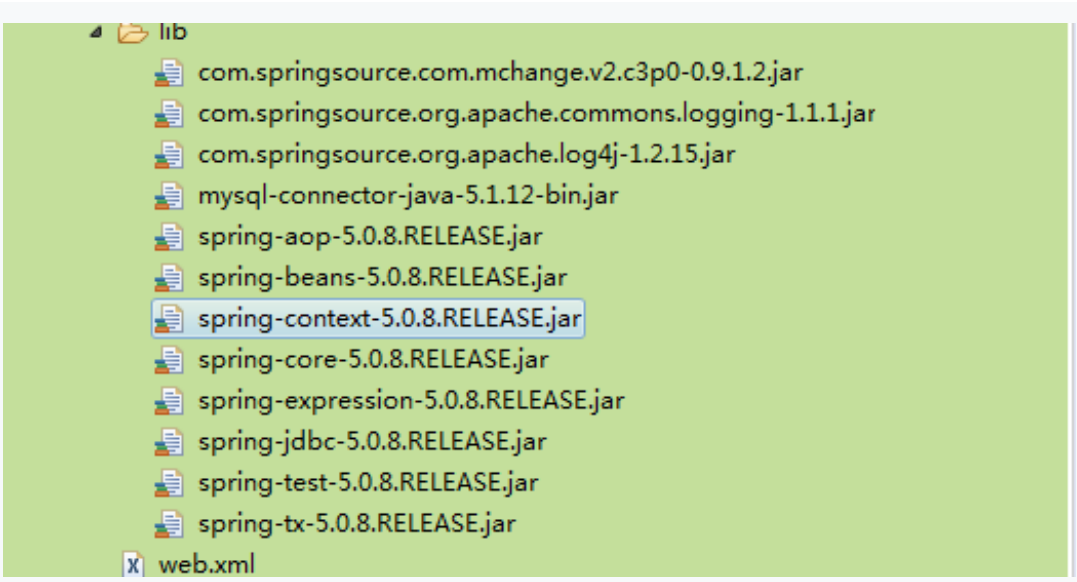
# 16 Spring_整合jdbc-JBCTemplate模板

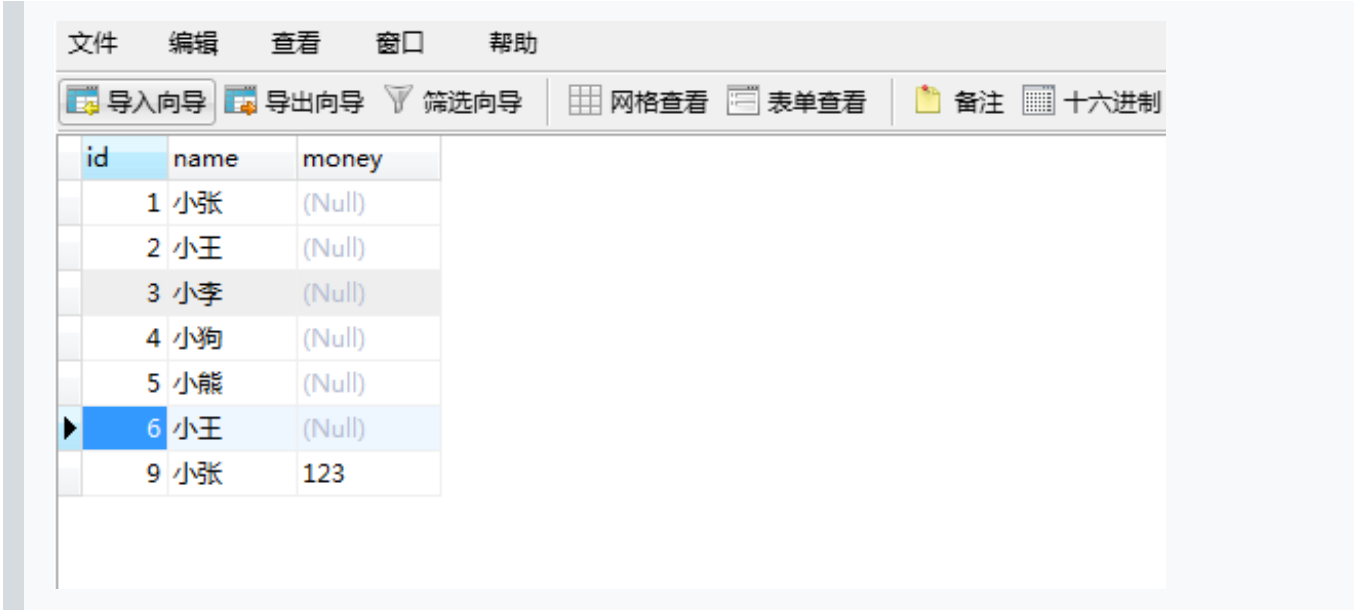
`Spring`

## 一：导入JAR包




## 二：JBCTemplate模板对象

### 1、数据库表




| id | name | money |
|---|---|---|
| 1 | 小张 | (Null) |
| 2 | 小王 | (Null) |
| 3 | 小李 | (Null) |
| 4 | 小狗 | (Null) |
| 5 | 小熊 | (Null) |
| 6 | 小王 | (Null) |
| 9 | 小张 | 123 |

## 2、JDBCTemplate模版

```
@Test
    public void getVoid() throws PropertyVetoException{
        //1.准备连接池
        ComboPooledDataSource dataSource = new ComboPooledDataSource();
        dataSource.setDriverClass("com.mysql.jdbc.Driver");

dataSource.setJdbcUrl("jdbc:mysql://172.16.10.164:3306/hibernates?useU
nicode=true&amp;characterEncoding=utf-8");
        dataSource.setUser("root");
        dataSource.setPassword("root");
        //2.创建连接池
         JdbcTemplate jdbcTemplate = new JdbcTemplate();
         jdbcTemplate.setDataSource(dataSource);

        //3、编写SQL,并执行
         String sql="INSERT INTO users values (null,'小张',234)";
         jdbcTemplate.update(sql);
```

# 三:JDBC模版API

## 1、实体类

```
public class Users {
  private  Integer  id;
  private String  name;
  private  float  money;
  //对应的set get方法
  }
```

## 2、接口

```
public interface IUsersDao {
    /**
     * 添加
     */
    void save(Users us)throws Exception;
    /**
     * 删除
     */
    void delete(Integer id)throws Exception;
    /**
     * 修改
     */
    void  update(Users us) throws Exception;
    /**
     * 查询返回对象
     */
    Users getById(Integer id)throws Exception;
    /**
     * 查询返回总记录数
     */
    int  getTotalCount()throws Exception;
    /**
     * 查询返回集合对象
     */
    List<Users> getAll()throws Exception;
}
```

## 3、接口实现类

```
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.List;
```

```java
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.RowMapper;
import com.spring.Dao.IUsersDao;
import com.spring.pojo.Users;

public class UsersDaoImpl implements IUsersDao {
    //必须要有get set 方法方便Spring注入
    private JdbcTemplate jt;

    @Override
    public void save(Users us) throws Exception {
        String sql = "INSERT INTO users values (null,?,?)";
        jt.update(sql, us.getName(), us.getMoney());
    }

    @Override
    public void delete(Integer id) throws Exception {
        String sql = "delete from users where id=?";
        jt.update(sql, id);
    }

    @Override
    public void update(Users us) throws Exception {
        String sql = "update  users set name=?,money=? where id=?";
        jt.update(sql, us.getName(), us.getMoney(), us.getId());
    }

    @Override
    public Users getById(Integer id) throws Exception {
        String sql = "SELECT * FROM users where id=?";
        return jt.queryForObject(sql, new RowMapper<Users>() {
            @Override
            public Users mapRow(ResultSet arg0, int arg1) throws SQLExc
eption {
                Users users = new Users();
                users.setId(arg0.getInt("id"));
                users.setName(arg0.getString("name"));
                users.setMoney(arg0.getFloat("money"));
                return users;
            }
        }, id);
    }

    @Override
    public int getTotalCount() throws Exception {
```

```
48.         String sql = "SELECT count(*) FROM users ";
49.         Integer in = jt.queryForObject(sql, Integer.class);
50.         return in;
51.     }
52. 
53.     @Override
54.     public List<Users> getAll() throws Exception {
55.         String sql = "SELECT * FROM users ";
56.         return jt.query(sql, new RowMapper<Users>() {
57.             @Override
58.             public Users mapRow(ResultSet arg0, int arg1) throws SQLExc
    eption {
59.                 Users users = new Users();
60.                 users.setId(arg0.getInt("id"));
61.                 users.setName(arg0.getString("name"));
62.                 users.setMoney(arg0.getFloat("money"));
63.                 return users;
64.             }
65. 
66.         });
67.     }
68. }
```

# 四:Dao配置到spring容器

```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <beans xmlns="http://www.springframework.org/schema/beans"
3.      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4.      xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans.xsd">
5.  <!--1.将连接池放入Spring容器中-->
6.  <bean name="dataSource"
    class="com.mchange.v2.c3p0.ComboPooledDataSource">
7.    <property name="driverClass" value="com.mysql.jdbc.Driver">
    </property>
8.    <property name="jdbcUrl"
    value="jdbc:mysql://172.16.10.164:3306/hibernates?
    useUnicode=true&amp;characterEncoding=utf-8"></property>
9.    <property name="user" value="root"></property>
10.   <property name="password" value="root"></property>
11. </bean>
12. <!--2、将JDBCTemplate对象放入容器中  -->
13. <bean name="jdbcTemplate"
```

```
    class="org.springframework.jdbc.core.JdbcTemplate">
14.   <property name="dataSource" ref="dataSource"></property>
15. </bean>
16. <!--3、将IUsesDao放入Spring容器中  -->
17. <bean name="userDao" class="com.spring.Dao.imp.UsersDaoImpl">
18.  <property name="jt" ref="jdbcTemplate"></property>
19. </bean>
20. </beans>
```

# 五：测试

```
1.  @RunWith(SpringJUnit4ClassRunner.class)
2.  @ContextConfiguration("classpath:applicationContext.xml")
3.  public class UserDemo {
4.
5.        @Resource(name="userDao")
6.        private IUsersDao  usersDao;
7.
8.      @Test
9.      public void getVoid01() throws Exception{
10.         Users users = new Users();
11.         users.setName("毛猫");
12.         users.setMoney(23.2f);
13.         usersDao.save(users);
14.     }
15.     }}
```

# 六、扩展:

## 1、JdbcDaoSupport

> 1、可以继承JdbcDaoSupport，根据连接池创建JDBC模版
> 2、就不用需要准备JDBC模版了，直接从父类获取即可

```
1.  public class UsersDaoImpl extends JdbcDaoSupport implements IUsersDao
    {
2.
```

```
    @Override
    public void save(Users us) throws Exception {
        String sql = "INSERT INTO users values (null,?,?)";
        super.getJdbcTemplate().update(sql, us.getName(), us.getMoney()
);
    }
//其他则相同即可 super.getJdbcTemplate().XXXXX
}
```

## 2、配置

> 1、dao直接注入到数据源

```
<beans xmlns="http://www.springframework.org/schema/beans"
    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">
<!--1.将连接池放入Spring容器中-->
<bean name="dataSource"
class="com.mchange.v2.c3p0.ComboPooledDataSource">
  <property name="driverClass" value="com.mysql.jdbc.Driver">
</property>
  <property name="jdbcUrl"
value="jdbc:mysql://172.16.10.164:3306/hibernates?
useUnicode=true&amp;characterEncoding=utf-8"></property>
  <property name="user" value="root"></property>
  <property name="password" value="root"></property>
</bean>
<!--3、将IUsesDao放入Spring容器中  -->
<bean name="userDao" class="com.spring.Dao.imp.UsersDaoImpl">
 <!-- <property name="jt" ref="jdbcTemplate"></property> -->
 <property name="dataSource" ref="dataSource"></property>
</bean>
</beans>
```

# 七、引入外部Properties文件信息:

## 1、properties文件

```
jdbc.driverClass=com.mysql.jdbc.Driver
jdbc.jdbcUrl=jdbc:mysql://192.168.43.173:3306/hibernates?useUnicode=true&amp;characterEncoding=utf-8
jdbc.user=root
jdbc.password=root
```

## 2、XML配置文件

> 1、开启冥名空间:xmlns:context="http://www.springframework.org/schema/context"

```
<!--指定读取properties文件  -->
<context:property-placeholder location="classpath:db.porperties"/>
<!--1.将连接池放入Spring容器中-->
<bean name="dataSource" class="com.mchange.v2.c3p0.ComboPooledDataSource">
  <property name="driverClass" value="${jdbc.driverClass}"></property>
  //获取其他properties信息同样式${}
</bean>
```