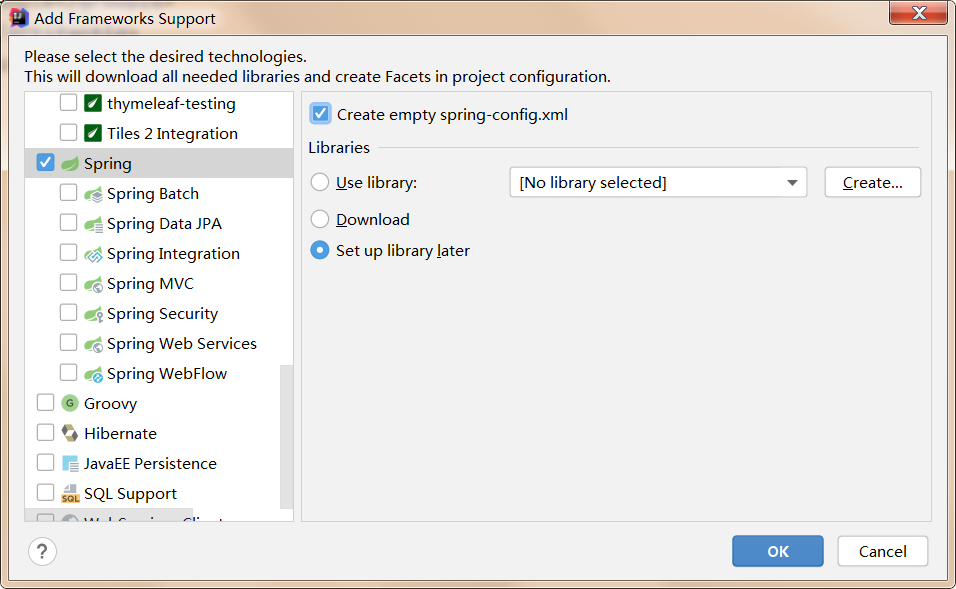
实验五 Spring

1. 实习要求
2. 掌握IOC及DI的概念
3. 掌握Spring的概念及功能
4. 掌握基于配置文件、注解的对象配置与依赖注入
5. 了解Spring AOP
6. 实习内容
7. Idea中配置Spring工程

首先创建一个空的Maven工程，在左侧工程结构中右单击工程名，在弹出菜单中选择“Add Framework Support...”，在弹出对话框的左侧列表中选择“Spring”，并在右侧中选择“Create empty Spring-config.xml”，及“Set up library late”，结果如下图：



移动包“java”下刚才创建的“Spring-config.xml”到“resources”中。

打开pom.xml文件，添加如下内容：

<packaging>jar</packaging>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>5.3.1</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-aspects</artifactId>

<version>5.3.1</version>

</dependency>

</dependencies>

1. Spring IOC

使用JDBC完成对表user完成增删改查。字段如下：

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 说明 |
| id | 整型 | 主键，自增 |
| user\_name | varchar(10) | 用户名 |
| passwd | varchar(10) | 口令 |

通过Spring管理相关的类及其之间的依赖关系。代码如下：

//用户实体类

package com.entity;

public class User {

private int id;

private String userName;

private String passwd;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getUserName() {

return userName;

}

public void setUserName(String userName) {

this.userName = userName;

}

public String getPasswd() {

return passwd;

}

public void setPasswd(String passwd) {

this.passwd = passwd;

}

@Override

public String toString() {

return "User{" +

"id=" + id +

", userName='" + userName + '\'' +

", passwd='" + passwd + '\'' +

'}';

}

}

//数据库连接

package com.model;

public class DBConnection {

private String driver;

private String url;

private String user;

private String passwd;

static Connection con;

public Connection getCon() throws ClassNotFoundException, SQLException {

if(con == null){

Class.forName(driver);

con = DriverManager.getConnection(url,user,passwd);

}

return con;

}

public String getDriver() {

return driver;

}

public void setDriver(String driver) {

this.driver = driver;

}

public String getUrl() {

return url;

}

public void setUrl(String url) {

this.url = url;

}

public String getUser() {

return user;

}

public void setUser(String user) {

this.user = user;

}

public String getPasswd() {

return passwd;

}

public void setPasswd(String passwd) {

this.passwd = passwd;

}

}

//数据库操作基类

package com.model;

@Component

public class DBBase {

@Resource

DBConnection connection;

public ResultSet selectAll(String sql) throws SQLException, ClassNotFoundException {

Connection con = connection.getCon();

Statement stat = con.createStatement();

ResultSet res = stat.executeQuery(sql);

return res;

}

}

//具体表操作类

package com.model;

@Service

public class UserService {

@Resource

DBBase dbBase;

static String SELECTSQL = "select \* from user";

public List<User> list() throws SQLException, ClassNotFoundException {

ResultSet rs = dbBase.selectAll(SELECTSQL);

List<User> users = new ArrayList<User>();

while (rs.next()){

User user = new User();

user.setId(rs.getInt("id"));

user.setUserName(rs.getString("user\_name"));

user.setPasswd(rs.getString("passwd"));

users.add(user);

}

return users;

}

}

//Spring配置文件——spring-config.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:context="http://www.springframework.org/schema/context"

xsi:schemaLocation="

http://www.springframework.org/schema/beans

http://www.springframework.org/schema/beans/spring-beans.xsd

http://www.springframework.org/schema/context

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

">

<!--扫描并注册bean-->

<context:component-scan base-package="com.model" />

<!--注册数据库连接类-->

<bean id="dbConnection" class="com.model.DBConnection">

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

<property name="url" value="jdbc:mysql://192.168.0.200/test?characterEncoding=utf8" />

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

<property name="passwd" value="123456" />

</bean>

</beans>

//测试类

public class Main {

public static void main(String[] argv) throws SQLException, ClassNotFoundException {

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

UserService userService = context.getBean(UserService.class);

System.out.println(userService.list());

}

}

完善UserService，完成对User的增删改查。

分析：对于“DBConnection”类的注册，能否使用注解进行注册；说明基于配置文件和注解注册的优缺点。

1. AOP

对1中所有的数据库操作方法记录操作日志。部分代码如下：

@Aspect

@Component

public class LogAop {

@Pointcut(value = "execution(\* com.model.\*Service.\*(..))")

public void cutPoint(){}

@After(value = "cutPoint()")

public void log(JoinPoint point){

System.out.println(point.getSignature() + "方法被执行.....");

}

}

功能完善：

建一张表用来记录所有的数据库操作。

**实习报告提交地址：www.zqldm.top:81**