**大学新生入学管理系统**

**单元测试报告（初稿）**

|  |  |
| --- | --- |
| 文件信息 | |
| 文件状态 | [ ]草稿 [√]正在修改 [ ]正式发布 |
| 文件标识 | 20171110-DYCSBGCG |
| 当前版本 | 1.0 |
| 作者 | |
| 姓名 | 李达飞 |
| 学号 | 2014011424 |
| 团队 | Supreme |
| 完成日期 | 2017/11/10 |
| 版本更新信息 | 初稿 |

**单元测试用例及描述举例**

这里对IStudentDao接口中的方法进行测试。

目前IStudentDao中有如下几种方法：

/\*\*

\* 通过学生的学号获得学生的邮箱

\*/

1. String getEmailbyNum(String num)

/\*\*

\* 通过班级号获取该班的所有学生

\*/

2. List<Student> getStudentsbyClassNum(String classNum)

/\*\*

\* 获取所有学生的信息

\*/

3. List<Student> getAll()

/\*\*

\* 通过学号获取学生的信息

\*/

4. Student getStudentInfobyNum(String num)

Student类为domain包中的javaBean，对应数据库中的sa\_student表。

单元测试采用Spring+JUnit4测试框架，下面对其过程进行说明。

如下是一段单元测试的代码：

**package** com.bistu.supreme.test;

**import** org.junit.Test;

**import** org.junit.runner.RunWith;

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

**import** org.springframework.test.context.ContextConfiguration;

**import** org.springframework.test.context.junit4.SpringJUnit4ClassRunner;

**import** com.bistu.supreme.dao.IStudentDao;

/\*\*

\* Spring测试框架

\* \*/

@RunWith(SpringJUnit4ClassRunner.**class**)

/\*\*

\* 加载 beans配置文件

\* \*/

@ContextConfiguration("/beans.xml")

**public** **class** TestStudent {

@Autowired

**private** IStudentDao studentDao;

@Test

**public** **void** Test() {

System.***out***.println(studentDao.getStudentInfobyNum("4").getStdBirth());

}

}

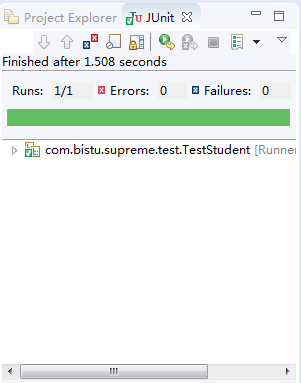
*注解：@RunWith(SpringJUnit4ClassRunner.class) //使用Spring的测试框架*

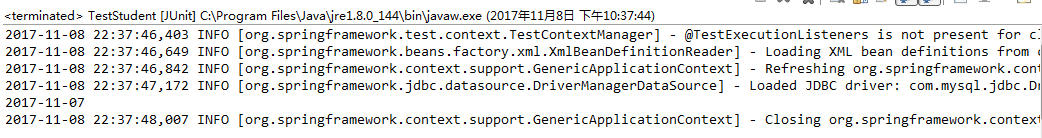
*@ContextConfiguration(“/beans.xml”) //加载Spring配置 文件*

*@Autowired //将beans.xml 中的接口相关bean注入到接口中（相当于new）*

*@Test //JUnit4测试框架，将方法作为测试用例，判断结果与预期结果是否一致*

通过单元测试，可以判断方法是否符合对应的功能需求的期望结果。例如，上面的单元测试测试的是getStudentInfobyNum()方法，输入值为字符串”4”（学号），结果如下图所示：





查询数据库中对应的数据可知，测试用例通过。当然，对于方法的测试在选取测试用例时，不仅要考虑正常的情况，还要考虑异常情况的处理。下面给出对IStudentDao接口中各方法的几个测试用例及预期结果，以及实际输出：

方法1：getEmailbyNum(String num)

|  |  |
| --- | --- |
| **用例描述** | getEmailbyNum方法测试用例 |
| **用例1** | |
| **输入** | num = “4” |
| **预期输出** | 学号为4的学生的邮箱 |
| **实际输出** | “verify@163.com”（与数据库中相同） |
| **结果** | 通过 |
| **用例2** | |
| **输入** | num=”50” |
| **预期输出** | “empty” |
| **实际输出** | “empty” |
| **结果** | 通过 |
| **用例3** | |
| **输入** | num=4（断开与数据库主机的连接） |
| **预期输出** | 处理SQLException |
| **实际输出** | “exception” |
| **结果** | 通过 |

方法2：getStudentsbyClassNum(String classNum)

|  |  |
| --- | --- |
| **用例描述** | getStudentsbyClassNum方法测试用例 |
| **用例1** | |
| **输入** | classNum=”软工1701” |
| **预期输出** | 所有软工1701班的学生信息 |
| **实际输出** | num[] = {“4”,”5”} |
| **结果** | 通过 |
| **用例2** | |
| **输入** | classNum=”123456” |
| **预期输出** | null |
| **实际输出** | null |
| **结果** | 通过 |
| **用例3** | |
| **输入** | classNum=”软工1701”（断开与数据库主机的连接） |
| **预期输出** | 处理SQLException |
| **实际输出** | list.get(0).getStdNum().equals( “-1”) |
| **结果** | 通过 |

方法3：getAll()

|  |  |
| --- | --- |
| **用例描述** | getAll方法测试用例 |
| **用例1** | |
| **输入** | 清空数据库中的所有学生 |
| **预期输出** | null |
| **实际输出** | null |
| **结果** | 通过 |
| **用例2** | |
| **输入** | 无 |
| **预期输出** | 所有学生的信息 |
| **实际输出** | num[]={“4”,”5”,”student”,”student1”,”student2”,”student3”} |
| **结果** | 通过 |
| **用例3** | |
| **输入** | 断开与数据库主机的连接 |
| **预期输出** | 处理SQLException |
| **实际输出** | list.get(0).getStdNum().equals(“-1”) |
| **结果** | 通过 |

方法4：getStudentInfobyNum(String num)

|  |  |
| --- | --- |
| **用例描述** | getStudentInfobyNum方法测试用例 |
| **用例1** | |
| **输入** | num=”4” |
| **预期输出** | 学号为4的学生的个人信息 |
| **实际输出** | name.equals(”student”)（与数据库中一致） |
| **结果** | 通过 |
| **用例2** | |
| **输入** | num=”50” |
| **预期输出** | 不存在该学生 |
| **实际输出** | student == null |
| **结果** | 通过 |
| **用例3** | |
| **输入** | num=4（断开与数据库主机的连接） |
| **预期输出** | 处理SQLException |
| **实际输出** | student.getStdNum().equals(“-1”) |
| **结果** | 通过 |