|  |
| --- |
| 北京航空航天大学 |
| 测试规格说明书 |
| **Spring Framwork** |
|  |
| |  |  | | --- | --- | | 王昕 SY1506415 | 王旭辰 SY1506413 | | 林森 SY1506416 | 李勃 SY1506402 | |
|  |

**版本变更历史**

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 提交日期 | 编制人 | 说明 |
| 1.0 | 2016/5/11 | 王昕 | 测试需求规格说明书 |
| 2.0 | 2016/5/19 | 王昕 | 测试需求初评审和复评审修改版 |
| 3.0 | 2016/5/22 | 王昕 | 课堂复评审修改版 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

[1 范围 1](#_Toc451717333)

[1.1 标识 1](#_Toc451717334)

[1.2 编写目的 1](#_Toc451717335)

[术语和缩略语表 1](#_Toc451717336)

[1.3 测试职责 1](#_Toc451717337)

[1.3.1 职责 1](#_Toc451717338)

[1.3.2 职责划分 2](#_Toc451717339)

[1.4 文档概述 2](#_Toc451717340)

[2测试计划 2](#_Toc451717341)

[2.1测试目标 2](#_Toc451717342)

[2.3测试工具 2](#_Toc451717343)

[2.3.1Juint插件 2](#_Toc451717344)

[2.3.2Juint应用实例 4](#_Toc451717345)

[3 测试规范及对照表 5](#_Toc451717346)

[3.1 编写测试计划 5](#_Toc451717347)

[3.2 设计测试用例 6](#_Toc451717348)

[3.3 测试用例与需求用例对照表 6](#_Toc451717349)

[4.容器的基本功能 7](#_Toc451717350)

[4.1容器的初始化 7](#_Toc451717351)

[4.2验证转换器 8](#_Toc451717352)

[4.3配置文件读取 9](#_Toc451717353)

[4.4解析标签读取功能 10](#_Toc451717354)

[4.5获取组件信息 10](#_Toc451717355)

[4.6获取组件方法信息 12](#_Toc451717356)

[4.7获得组件参数信息的验证 12](#_Toc451717357)

[4.8注册组件的功能 13](#_Toc451717358)

[4.9组件信息错误处理 14](#_Toc451717359)

[4.10验证依赖关系是否正确处理 15](#_Toc451717360)

[4.11依赖关系下组件信息获取验证 16](#_Toc451717361)

[5容器的扩展功能 17](#_Toc451717362)

[5.1解析配置文件 17](#_Toc451717363)

[6非功能性测试 18](#_Toc451717364)

[6.1容错性测试 18](#_Toc451717365)

[7参考文献 19](#_Toc451717366)

# 1 范围

## 1.1 标识

Spring版本号：Spring-Framework 3.2

模 块 名 称：管理容器，注解定义Java组件类，根据注解类生成XML配置文件。

测试环境：Eclipse ,Windows7,Junit4,jdk1.8

测试规格版本：V3.0

## 1.2 编写目的

本文档主要明确了本次测试的主要内容和测试小组各成员的分工，并对测试用例进行了初步的设计。测试技术和策略等问题不在本文档描述范围内。

## 术语和缩略语表

|  |  |
| --- | --- |
| **缩写、术语及符号** | **解 释** |
| 单元测试 | 是指对软件中的最小可测试单元进行检查和验证 |
| 集成测试 | 在单元测试的基础上，将所有模块按照设计要求（如根据结构图）组装成为子系统或系统，进行集成测试 |
| Juint | JUnit是一个Java语言的单元测试框架 |

## 1.3 测试职责

### 1.3.1 职责

测试是软件开发过程中的重要组成部分，肩负着如下责任：

* 编写合理的测试计划，并与需求分析相对应；
* 编写覆盖率高的测试用例；
* 针对测试需求进行相关测试技术的研究；
* 认真仔细地实施测试工作，并提交测试报告；
* 撰写工作进度日志。

### 1.3.2 职责划分

|  |  |
| --- | --- |
| 成员 | 主要职责 |
| 王昕 | 整合组员的测试成果，编写测试需求规格说明书 |
| 王旭辰 | 容器基本模块测试用例编写 |
| 林森 | 容器扩展模块测试用例的编写 |
| 李勃 | 容器的基本模块和扩展模块的集成测试 |

## 1.4 文档概述

本文档主要说明了测试阶段的工作内容。首先叙述了本组对测试工作的分析过程，给出了测试用例的需求用例与对应的测试用例的对照表，对每个测试用例按照RUCM4test的标准进行了说明。最后，本文对“非功能性需求”的“服务器性能测试”从效率的角度做了详细描述，给出了详细的测试过程。

# 2测试计划

## 2.1测试目标

通过测试来发现Spring框架所能提供给开发人员使用的功能是否符合《软件需求规格说明书》，并且在测试的过程中要尽可能的发现软件在现有阶段存在的问题，确保在软件交付客户使用之后出现的问题在可接受的范围之内。测试过程的时间在实验可接受的范围之内。

## 2.3测试工具

### 2.3.1Juint插件

JUnit是由 Erich Gamma 和 Kent Beck 编写的一个回归测试框架（regression testing framework）,供Java开发人员编写单元测试之用。

　　Junit测试是程序员测试，即所谓白盒测试，因为程序员知道被测试的软件如何（How）完成功能和完成什么样（What）的功能。 本质上是一套框架，即开发者制定了一套条条框框，遵循这此条条框框要求编写测试代码，如继承某个类，实现某个接口，就可以用Junit进行自动测试了。 由于Junit相对独立于所编写的代码，可以测试代码的编写可以先于实现代码的编写，XP 中推崇的 test first design的实现有了现成的手段：用Junit写测试代码，写实现代码，运行测试，测试失败，修改实现代码，再运行测试，直到测试成功。以后对代码的修改和优化，运行测试成功，则修改成功。 Java 下的 team 开发，采用 cvs(版本控制) + ant(项目管理) + junit(集成测试) 的模式时，通过对ant的配置，可以很简单地实现测试自动化。 对不同性质的被测对象，如Class，Jsp，Servlet，Ejb等，Junit有不同的使用技巧，以后慢慢地分别讲叙。以下以Class测试为例讲解，除非特殊说明。junit中的assert方法全部放在Assert类中，总结一下junit类中assert方法的分类。

1.assertTrue/False([String message,]boolean condition);

用来查看变量是是否为false或true，如果assertFalse()查看的变量的值是false则测试成功，如果是true则失败，assertTrue()与之相反；

2.fail([String message,]);

直接用来抛出错误。

3.assertEquals([String message,]Object expected,Object actual);

判断是否相等，可以指定输出错误信息。

第一个参数是期望值，第二个参数是实际的值。

这个方法对各个变量有多种实现

4.assertNotNull/Null([String message,]Object obj);

判读一个对象是否非空(非空)。

5.assertSame/NotSame([String message,]Object expected,Object actual);

判断两个对象是否指向同一个对象。看内存地址。

6.ailNotSame/failNotEquals(String message, Object expected, Object actual)

当不指向同一个内存地址或者不相等的时候，输出错误信息。

注意信息是必须的，而且这个输出是格式化过的。

### 2.3.2Juint应用实例

最简单的范例如下：

1、创建一个TestCase的子类

package junitfaq;

import java.util.\*;

import junit.framework.\*;

public class SimpleTest extends TestCase {

public SimpleTest(String name) {

super(name);

}

2、写一个测试方法断言期望的结果

public void testEmptyCollection() {

Collection collection = new ArrayList();

assertTrue(collection.isEmpty());

}

注意：JUnit推荐的做法是以test作为待测试的方法的开头，这样这些方法可以被自动找到并被测试。

3、写一个suite()方法，它会使用反射动态的创建一个包含所有的testXxxx方法的测试套件

public static Test suite() {

return new TestSuite(SimpleTest.class);

}

4、写一个main()方法以文本运行器的方式方便的运行测试

public static void main(String args[]) {

junit.textui.TestRunner.run(suite());

}

}

5、运行测试

以文本方式运行：

java junitfaq.SimpleTest

通过的测试结果是：

Time: 0

OK (1 tests)

Time上的小点表示测试个数，如果测试通过则显示OK。否则在小点的后边标上F，表示该测试失败。

每次的测试结果都应该是OK的，这样才能说明测试是成功的，如果不成功就要马上根据提示信息进行修正了。

如果JUnit报告了测试没有成功，它会区分失败（failures）和错误（errors）。失败是你的代码中的assert方法失败引起的；而错误则是代码异常引起的，例如ArrayIndexOutOfBoundsException。

以图形方式运行：

java junit.swingui.TestRunner junitfaq.SimpleTest

通过的测试结果在图形界面的绿色条部分。

以上是最简单的测试样例，在实际的测试中我们测试某个类的功能是常常需要执行一些共同的操作，完成以后需要销毁所占用的资源（例如网络连接、数据库连接，关闭打开的文件等），TestCase类给我们提供了setUp方法和tearDown方法，setUp方法的内容在测试你编写的TestCase子类的每个testXxxx方法之前都会运行，而tearDown方法的内容在每个testXxxx方法结束以后都会执行。这个既共享了初始化代码，又消除了各个测试代码之间可能产生的相互影响。

# 3 测试规范及对照表

## 3.1 编写测试计划

在进行软件测试前，周密的计划和合理的安排是必不可少的。测试计划的制定是要建立在充分理解需求的基础上的，是测试的起始步骤和重要环

|  |  |
| --- | --- |
| 过程要点 | 详细说明 |
| 启动条件 | 需求分析文档终稿的完成； |
| 工作内容 | 根据项目的需求分析文档，测试文档中应该至少包括以下关键内容：   * 根据已有的需求分析文档终稿，明确相应的测试用例。具体设计各大模块的测试用例，设计完成后进行评审工作。 * 根据评审后的测试用例编写测试代码 * 部署环境，具体测试各模块功能，记录测试数据及结果。 * 结合需求文档终稿，对测试结果进行分析。 * 完成整个测试需求文档，由评审组完成最终评审。   测试计划编写完毕后，必须提交给项目组全体成员，并由项目组组中各成员联合评审。 |
| 退出标准 | * 测试获得预期成果。 * 测试由评审组评审通过. |
| 责任人 | 王旭辰，林森，李勃 |

## 3.2 设计测试用例

一个测试用例描述了针对某个目标对程序进行测试所采用的一组实际输入、程序执行条件、测试步骤和预期的输出，以核实某个程序或其中的特定路径是否满足特定需求。在需求分析文档确立基线以后，需编写项目的测试用例，具体的任务和责任人如下：

|  |  |
| --- | --- |
| 过程要点 | 详细说明 |
| 输入条件 | 测试需求明确，测试计划明确 |
| 工作内容 | 根据每一步测试计划编写全部的测试用例 |
| 退出标准 | 测试用例需要覆盖大部分测试需求 |
| 责任人 | 全体组员 |

## 3.3 测试用例与需求用例对照表

本次测试中，小组挑选了各模块中核心及具有代表性需求用例进行测试用例的设计。测试用例与需求用例的对应表如下：

|  |  |  |
| --- | --- | --- |
| 模块 | 需求用例 | 测试用例 |
| 容器的基本模块 | 容器初始化 | 容器工厂初始化运行测试 |
| 验证转换器 | 转换器运行结果验证测试 |
| 配置文件读取 | 配置文件读取测试 |
| 解析标签读取功能 | 解析标签读取功能测试 |
| 获取组件信息 | 获取组件信息测试 |
| 获取组件信息 | 获取组件方法信息 |
| 获取组件参数信息 |
| 组成组件功能 |
| 组件错误信息处理 |
| 验证依赖关系 | 验证依赖关系测试 |
| 依赖关系下的组件信息获取 | 依赖关系下组件信息获取验证 |
| 容器的扩展模块 | 解析配置文件 | 解析配置文件测试 |
| 非功能性测试 | 具有一定的容错性 | 容错性测试 |

这是我们组已经测试的测试用例和需求用例的对应，没有与需求用例一一对应。原因是有一些测试用例不好测试。在此做相关的说明。

# 4.容器的基本功能

## 4.1容器的初始化

|  |  |  |
| --- | --- | --- |
| **Test Case Specification** | | |
| Name | 容器工厂初始化运行测试 | |
| Brief Description | 验证容器的工厂初始化功能是否成功 | |
| Precondition | 容器初始化开始 | |
| Tester | 测试员 王旭辰 | |
| Dependency | None | |
| Test Setup | Name | 工厂初始化 |
| Description | 打开所设置的单元测试函数，运行单元测试函数 |
| Basic Flow  (Test Setup) | Steps | |
| 1 | 测试员启动操作系统中的eclipse； |
| 2 | 测试员利用单元测试testfactory.java进行测试； |
| Postcondition  (Test Oracle) | 工厂初始化测试成功 |
| Basic Flow  (Test Sequence) | Steps | |
| 1 | 测试员为工厂初始化的测试设置一个测试函数，用来测试初始化过程中转换器存储区的正确运行 |
| 2 | 测试员运行测试函数； |
| 3 | 测试员测试默认Handler转换器能否成功创建 |
|  | 系统VALIDATES THAT默认Handler转换器成功创建 |
| 4 | 测试员测试命名的Handler转换器能否覆盖刚才创建的默认转换器 |
| 5 | 利用程序插桩技术，VALIDATES THAT创建成功后输出语句； |
| 6 | 测试员利用程序插桩技术测试缓存区，VERIFY THAT创建成功； |
| 7 | 测试员利用程序插桩技术测试服务定义信息的存储区，VERIFY THAT创建成功； |
| Postcondition  (Test Oracle) | 在工厂初始化测试成功 |
| Specific Alternative Flows  (Test Sequence) | RFS 4-7 | |
| 1 | 使用默认的Handler转换器继续完成创建工作 |
| 2 | RESUME STEP 5 |
| Postcondition  (Test Sequence) | 在工厂初始化测试成功 |

表1 容器工厂初始化测试用例规格说明

## 4.2验证转换器

|  |  |  |
| --- | --- | --- |
| **Test Case Specification** | | |
| Name | Handler转换器运行结果验证测试 | |
| Brief Description | 验证容器的Handler转换器能否成功得到注解类中的 | |
| Precondition | 服务定义信息的存储区创建成功 | |
| Tester | 王旭辰 | |
| Dependency | None | |
| Test Setup | Name | Handler转换器 |
| Description | 利用Handler转换器获得注解相关的信息 |
| Basic Flow  (Test Setup) | Steps | |
| 1 | 测试员启动操作系统中的eclipse； |
| 2 | 测试员利用单元测试testhandler.java进行测试； |
| Postcondition  (Test Oracle) | 运行测试函数，等待测试函数执行 |
| Basic Flow  (Test Sequence) | Steps | |
| 1 | 测试员为Handler转换器的测试设置一个测试函数，用来测试注解信息能否正确获得 |
| 2 | 测试员运行测试函数； |
| 3 | 测试员利用程序插桩技术测试服务定义信息的存储区； |
| 4 | 测试员VALIDATES THAT创建成功； |
| 5 | 测试员通过类名称获得的包名并VALIDATES THAT与实际是否相符； |
| 6 | 测试员利用插桩方法VALIDATES THAT 包名正确； |
| 7 | 测试员通过插桩方法判断获得组件的名称类型是否正确，VALIDATES THAT 名称类型正确； |
| 8 | 测试员VALIDATES THAT 名称类型正确； |
| 9 | Do 测试员获得并验证方法与实际是否相符； |
| 10 | 测试员 VALIDATES THAT 名称类型正确； |
| 11 | UNTIL 注解的方法不存在 |
| 12 | Do 测试员获得并验证注解与实际是否相符； |
| 13 | 测试员 VALIDATES THAT 名称类型正确； |
| 14 | UNTIL 注解的参数不存在 |
| Postcondition  (Test Oracle) | Handler转换器运行成功 |
| Specific Alternative Flows  (Test Sequence) | RFS 4 | |
| 1 | 系统VALIDATES THAT服务定义信息的存储区创建失败 |
| Postcondition  (Test Sequence) | Handler转换器测试失败 |
| Specific Alternative Flows  (Test Sequence) | RFS 5 | |
| 1 | 测试员 VALIDATES THAT 包名错误 |
| 2 | RESUME STEP 5 |
| Postcondition  (Test Sequence) | Handler转换器测试失败 |
| Specific Alternative Flows  (Test Sequence) | RFS 7 | |
| 1 | 测试员 VALIDATES THAT 方法名错误 |
| 2 | RESUME STEP 7 |
| Postcondition  (Test Sequence) | Handler转换器测试失败 |

表2 容器Handler转换器测试用例规格说明

## 4.3配置文件读取

|  |  |  |
| --- | --- | --- |
| **Test Case Specification** | | |
| Name | 配置文件读取功能测试 | |
| Brief Description | 验证配置文件读取功能是否完善 | |
| Precondition | 配置文件编写完成且放在指定位置 | |
| Tester | 李勃 | |
| Dependency | None | |
| Test Setup | Name | 读取功能测试准备 |
| Description | 运行读取功能测试代码的准备工作 |
| Basic Flow  (Test Setup) | Steps | |
| 1 | 测试员将配置文件置于配置文件指定文件夹下； |
| 2 | 测试员启动操作系统中的eclipse并导入测试代码至工程中； |
| Postcondition  (Test Oracle) | 代码导入成功，无报错； |
| Basic Flow  (Test Sequence) | Steps | |
| 1 | 测试员编写一个测试bean，与配置文件相吻合； |
| 2 | 测试员实例化一个解析器，使用解析器读取配置文件 |
| 3 | 测试员运行测试代码，检查配置文件是否被成功读取； |
| Postcondition  (Test Oracle) | 打印配置文件被成功读取的语句 |
| Specific Alternative Flows  (Test Sequence) | RFS 3 | |
| 1 | 打印未成功读取消息 |
| 2 | 检查对应的读取功能的代码实现 |
| 3 | 修改对应部分代码 |
| 4 | RESUME STEP 3 |
| Postcondition  (Test Sequence) | 打印配置文件被成功读取的语句 |

表3 配置文件读取测试

## 4.4解析标签读取功能

|  |  |  |
| --- | --- | --- |
| **Test Case Specification** | | |
| Name | 解析标签读取功能测试 | |
| Brief Description | 验证解析标签读取功能是否完善 | |
| Precondition |  | |
| Tester | 李勃 | |
| Dependency | None | |
| Test Setup | Name | 解析功能测试准备 |
| Description | 运行解析功能测试代码的准备工作 |
| Basic Flow  (Test Setup) | Steps | |
| 1 | 测试员将配置文件置于测试代码相同的文件夹下； |
| 2 | 测试员启动操作系统中的eclipse并导入测试代码至工程中； |
| Postcondition  (Test Oracle) | 代码导入成功，无报错； |
| Basic Flow  (Test Sequence) | Steps | |
| 1 | 测试员编写一个测试bean，与配置文件相吻合，其中包含各种定义的标签； |
| 2 | 测试员实例化一个解析器，使用解析器读取配置文件 |
| 3 | 测试员使用容器实例化一个步骤1编写的测试bean |
| 4 | 测试员打印该测试bean的基本类型属性值 |
| 5 | 测试员运行测试代码，检查各显示属性值与配置文件中定义是否相同 |
| Postcondition  (Test Oracle) | 各显示属性值与配置文件中定义相同 |
| Specific Alternative Flows  (Test Sequence) | RFS 5 | |
| 1 | 各显示属性值与配置文件中定义不相同 |
| 2 | 检查不相同属性对应的解析功能的代码实现 |
| 3 | 修改对应部分代码 |
| 4 | RESUME STEP 8 |
| Postcondition  (Test Sequence) | 各显示属性值与配置文件中定义相同 |

表4解析标签读取功能测试

## 4.5获取组件信息

|  |  |  |
| --- | --- | --- |
| **Test Case Specification** | | |
| Name | 获得工厂内组件名称信息的验证测试 | |
| Brief Description | 将数据存入工厂后，从工厂提取信息 | |
| Precondition | 工厂的初始化以及handler转换器的测试成功 | |
| Tester | 王旭辰 | |
| Dependency | None | |
| Test Setup | Name | 获得工厂内信息 |
| Description | 利用初始化后的工厂获得组件名称信息 |
| Basic Flow  (Test Setup) | Steps | |
| 1 | 测试员启动操作系统中的eclipse； |
| 2 | 测试员利用单元测试testdata.java进行测试； |
| Postcondition  (Test Oracle) | 运行测试函数，等待测试函数执行 |
| Basic Flow  (Test Sequence) | Steps | |
| 1 | 测试员为工厂数据获得的测试设置一个测试函数，用来测试组件信息能否正确获得 |
| 2 | 测试员运行测试函数； |
| 3 | 测试员利用程序插桩技术测试工厂初始化成功，并且handler转换器成功获得组件信息，并存储； |
| 4 | 测试员VALIDATES THAT工厂初始化成功并且handler转换器运行成功； |
| 5 | 测试员从服务信息存储区中获得要找的组件信息 |
| 6 | 测试员VALIDATES THAT 组件信息在缓存中； |
| 7 | 测试员判断从缓存中获得的组件名称是否正确 |
| Postcondition  (Test Oracle) | 获得了所寻找组件的相关信息 |
| Specific Alternative Flows  (Test Sequence1) | RFS 3 | |
| 1 | 系统VALIDATES THAT初始化失败或者handler转换器获取信息失败 |
| Postcondition  (Test Sequence) | 工厂中获得组件信息测试失败 |
| Specific Alternative Flows  (Test Sequence2) | RFS 5 | |
| 1 | 测试员 VALIDATES THAT 组件信息不在缓存中 |
| 2 | 测试员判断根据服务信息创建的组件是否成功创建 |
| 3 | 测试员 VALIDATES THAT 组件信息成功创建 |
| 4 | 测试员将组件信息存入缓存中是否成功 |
| 5 | 测试员 VALIDATES THAT 存入缓存成功 |
| 6 | 测试员判断从缓存中获得的组件名称是否正确 |
| Postcondition  (Test Sequence) | 获得了所寻找组件的相关信息 |
| Specific Alternative Flows  (Test Sequence) | RFS Test Sequence2-2 | |
| 1 | 测试员 VALIDATES THAT 创建失败 |
| 2 | RESUME Test Sequence2 STEP 2 |
| Postcondition  (Test Sequence) | 工厂中获得组件信息测试失败 |
| Specific Alternative Flows  (Test Sequence) | RFS Test Sequence2-4 | |
| 1 | 测试员 VALIDATES THAT 存入缓存失败 |
| 2 | RESUME Test Sequence2 STEP 4 |
| Postcondition  (Test Sequence) | 工厂中获得组件信息测试失败 |

表5 容器工厂获取组件名称测试用例规格说明

## 4.6获取组件方法信息

|  |  |  |
| --- | --- | --- |
| **Test Case Specification** | | |
| Name | 获得工厂内组件方法信息的验证测试 | |
| Brief Description | 将数据存入工厂后，从工厂提取信息 | |
| Precondition | 工厂的初始化以及handler转换器的测试成功 | |
| Tester | 测试员 王旭辰 | |
| Dependency | None | |
| Test Setup | Name | 获得工厂内信息 |
| Description | 利用初始化后的工厂获得组件名称信息 |
| Basic Flow  (Test Setup) | Steps | |
| 1 | 测试员启动操作系统中的eclipse； |
| 2 | 测试员利用单元测试testdata2.java进行测试； |
| Postcondition  (Test Oracle) | 获得了所寻找组件的方法信息 |
| Basic Flow  (Test Sequence) | Steps | |
| 1 | 测试员为工厂数据获得的测试设置一个测试函数，用来测试组件信息能否正确获得 |
| 2 | 测试员运行测试函数； |
| 3 | 测试员利用程序插桩技术测试工厂初始化成功，并且handler转换器成功获得组件信息，并存储； |
| 4 | 测试员VERIFY THAT工厂初始化成功并且handler转换器运行成功； |
| 5 | 测试员VERIFY THAT 组件信息在缓存中； |
| 6 | 测试员判断从缓存中获得的组件方法是否正确 |
| Postcondition  (Test Oracle) | 获得了所寻找组件的方法信息 |
| Specific Alternative Flows  (Test Sequence1) | RFS 3 | |
| 1 | 系统VALIDATES THAT初始化失败或者handler转换器获取信息失败 |
| Postcondition  (Test Sequence) | 工厂中获得组件方法信息测试失败 |

表6获取组件方法信息

## 4.7获得组件参数信息的验证

|  |  |  |
| --- | --- | --- |
| **Test Case Specification** | | |
| Name | 获得工厂内组件参数信息的验证测试 | |
| Brief Description | 将数据存入工厂后，从工厂提取信息 | |
| Precondition | 工厂的初始化以及handler转换器的测试成功 | |
| Tester | 测试员 王旭辰 | |
| Dependency | None | |
| Test Setup | Name | 获得工厂内信息 |
| Description | 利用初始化后的工厂获得组件参数信息 |
| Basic Flow  (Test Setup) | Steps | |
| 1 | 测试员启动操作系统中的eclipse； |
| 2 | 测试员利用单元测试testdata3.java进行测试； |
| Postcondition  (Test Oracle) | 获得了所寻找组件的参数信息 |
| Basic Flow  (Test Sequence) | Steps | |
| 1 | 测试员为工厂数据获得的测试设置一个测试函数，用来测试组件信息能否正确获得 |
| 2 | 测试员运行测试函数； |
| 3 | 测试员利用程序插桩技术测试工厂初始化成功，并且handler转换器成功获得组件信息，并存储； |
| 4 | 测试员VERIFY THAT工厂初始化成功并且handler转换器运行成功； |
| 5 | 测试员VERIFY THAT 组件信息在缓存中； |
| 6 | 测试员判断从缓存中获得的组件参数是否正确 |
| Postcondition  (Test Oracle) | 获得了所寻找组件的参数信息 |
| Specific Alternative Flows  (Test Sequence1) | RFS 3 | |
| 1 | 系统VALIDATES THAT初始化失败或者handler转换器获取信息失败 |
| Postcondition  (Test Sequence) | 工厂中获得组件参数信息测试失败 |

表7获得组件参数信息的验证

## 4.8注册组件的功能

|  |  |  |
| --- | --- | --- |
| **Test Case Specification** | | |
| Name | 注册组件功能测试 | |
| Brief Description | 验证组件注册功能是否完善 | |
| Precondition | 配置文件编写完成且放在指定位置 | |
| Tester | 王旭辰 | |
| Dependency | None | |
| Test Setup | Name | 注册组件功能测试准备 |
| Description | 运行注册组件功能测试代码的准备工作 |
| Basic Flow  (Test Setup) | Steps | |
| 1 | 测试员将配置文件置于配置文件指定文件夹下； |
| 2 | 测试员启动操作系统中的eclipse并导入测试代码至工程中； |
| Postcondition  (Test Oracle) | 代码导入成功，无报错； |
| Basic Flow  (Test Sequence) | Steps | |
| 1 | 测试员编写一个测试bean，与配置文件相吻合，配置文件中包含各种标签的使用； |
| 2 | 测试人员在注册组件相关模块插入插桩语句，记录各个组件注册的信息； |
| 3 | 测试员实例化一个解析器，使用解析器读取配置文件 |
| 4 | 测试员运行测试代码，打印插桩语句，并验证组件的注册情况与预期是否相同； |
| Postcondition  (Test Oracle) | 验证打印的组件注册信息与预想一致 |
| Specific Alternative Flows  (Test Sequence) | RFS 4 | |
| 1 | 验证打印的组件注册信息与预想不一致 |
| 2 | 检查对应的组件注册的代码实现 |
| 3 | 修改对应部分代码 |
| 4 | RESUME STEP 4 |
| Postcondition  (Test Sequence) | 验证打印的组件注册信息与预想一致 |

表8注册组件的功能测试

## 4.9组件信息错误处理

|  |  |  |
| --- | --- | --- |
| **Test Case Specification** | | |
| Name | 组件信息错误处理测试 | |
| Brief Description | 验证组件信息错误时能否正常退出 | |
| Precondition | 工厂的初始化以及handler转换器创建成功 | |
| Tester | 测试员 王旭辰 | |
| Dependency | None | |
| Test Setup | Name | 组件依赖关系获得组件信息处理 |
| Description | 打开所设置的单元测试函数，运行单元测试函数 |
| Basic Flow  (Test Setup) | Steps | |
| 1 | 测试员启动操作系统中的eclipse； |
| 2 | 测试员利用单元测试testerror.java进行测试； |
| Postcondition  (Test Oracle) | 组件信息错误处理正常退出 |
| Basic Flow  (Test Sequence) | Steps | |
| 1 | 测试员将组件注解信息修改为错误信息 |
| 2 | 测试员为Handler转换器的测试设置一个测试函数，用来测试注解信息能否正确获得 |
| 3 | 测试员运行测试函数； |
| 4 | 测试员利用程序插桩技术测试服务定义信息的存储区； |
| 5 | 测试员VERIFY THAT创建成功；进入Handler转换器 |
| 6 | 测试员VERIFY THAT Handler转换器不处理被修改的组件类 |
| 7 | 测试员VERIFY THAT Handler转换器退出 |
| Postcondition  (Test Oracle) | 组件信息错误处理正常退出 |
| Specific Alternative Flows  (Test Sequence1) | RFS 6 | |
| 1 | 测试员VERIFY THAT Handler转换器仍然处理错误组件类 |
| Postcondition  (Test Sequence) | 组件信息错误非正常退出 |

表9组件信息错误处理测试

## 4.10验证依赖关系是否正确处理

|  |  |  |
| --- | --- | --- |
| **Test Case Specification** | | |
| Name | 容器出现组件依赖关系handler转换器处理运行测试 | |
| Brief Description | 验证容器的工厂在出现依赖关系时能否正在运行 | |
| Precondition | None | |
| Tester | 王旭辰 | |
| Dependency | None | |
| Test Setup | Name | 组件依赖关系handler转换器处理 |
| Description | 打开所设置的单元测试函数，运行单元测试函数 |
| Basic Flow  (Test Setup) | Steps | |
| 1 | 测试员启动操作系统中的eclipse； |
| 2 | 测试员利用单元测试testdependence.java进行测试； |
| Postcondition  (Test Oracle) | 运行测试函数，等待测试函数执行 |
| Basic Flow  (Test Sequence) | Steps | |
| 1 | 测试员为容器的依赖测试设置一个测试函数，用来测试初始化过程中出现依赖关系时的正确运行 |
| 2 | 测试员运行测试函数； |
| 3 | 测试员测试在利用handler转换器获得组件信息时遇见依赖关系时，先处理依赖关系组件 |
| 4 | 测试员VALIDATES THAT依赖组件先被处理 |
| 5 | 测试员测试当依赖组件处理后，能否正确回到原组件 |
| 6 | 测试员VALIDATES THAT原组件被处理； |
| Postcondition  (Test Oracle) | 依赖关系组件handler转换器处理测试成功 |
| Specific Alternative Flows  (Test Sequence1) | RFS 3 | |
| 1 | 测试员VALIDATES THAT原组件被处理 |
| Postcondition  (Test Sequence) | 依赖关系组件handler转换器处理测试失败 |
| Specific Alternative Flows  (Test Sequence2) | RFS 5 | |
| 1 | 测试员VALIDATES THAT处理直接结束 |
| Postcondition  (Test Sequence) | 依赖关系组件handler转换器处理测试失败 |

表10 容器出现组件依赖关系handler转换器处理测试用例规格说明

## 4.11依赖关系下组件信息获取验证

|  |  |  |
| --- | --- | --- |
| **Test Case Specification** | | |
| Name | 容器出现组件依赖关系获得组件信息处理运行测试 | |
| Brief Description | 验证容器的工厂在出现依赖关系时能否正在运行 | |
| Precondition | None | |
| Tester | 王旭辰 | |
| Dependency | None | |
| Test Setup | Name | 组件依赖关系获得组件信息处理 |
| Description | 打开所设置的单元测试函数，运行单元测试函数 |
| Basic Flow  (Test Setup) | Steps | |
| 1 | 测试员启动操作系统中的eclipse； |
| 2 | 测试员利用单元测试testdependence2.java进行测试； |
| Postcondition  (Test Oracle) | 运行测试函数，等待测试函数执行 |
| Basic Flow  (Test Sequence) | Steps | |
| 1 | 测试员为容器的依赖测试设置一个测试函数，用来测试组件出现依赖关系时的获得组件信息的正确运行 |
| 2 | 测试员运行测试函数； |
| 3 | 测试员测试在利用工厂获得组件存储信息提供者时遇见依赖关系时，先处理依赖关系组件 |
| 4 | 测试员VALIDATES THAT依赖组件先被处理 |
| 5 | 测试员测试当依赖组件处理后，能否正确回到原组件 |
| 6 | 测试员VALIDATES THAT原组件被处理； |
| Postcondition  (Test Oracle) | 依赖关系组件获得组件信息处理测试成功 |
| Specific Alternative Flows  (Test Sequence1) | RFS 3 | |
| 1 | 测试员VALIDATES THAT原组件被处理 |
| Postcondition  (Test Sequence) | 依赖关系组件获得组件信息处理测试失败 |
| Specific Alternative Flows  (Test Sequence2) | RFS 5 | |
| 1 | 测试员VALIDATES THAT处理直接结束 |
| Postcondition  (Test Sequence) | 依赖关系组件获得组件信息处理测试失败 |

表11容器出现组件依赖关系handler转换器处理测试用例规格说明

# 5容器的扩展功能

## 5.1解析配置文件

|  |  |  |
| --- | --- | --- |
| **Test Case Specification** | | |
| Name | 解析配置文件功能测试 | |
| Brief Description | 验证解析配置文件功能是否完善 | |
| Precondition | None | |
| Tester | 林森 | |
| Dependency | None | |
| Test Setup | Name | 解析功能测试准备 |
| Description | 运行解析功能测试代码的准备工作 |
| Basic Flow  (Test Setup) | Steps | |
| 1 | 测试员将配置文件置于测试代码相同的文件夹下； |
| 2 | 测试员启动操作系统中的eclipse并导入测试代码至工程中 |
| Postcondition  (Test Oracle) | 代码导入成功，无报错 |
| Basic Flow  (Test Sequence) | Steps | |
| 1 | 测试员编写一个测试bean，该测试bean与配置文件对应（包含了基本类型属性，容器类属性，内部bean引用，外部bean引用，带参数的构造函数，不带参数的构造函数； |
| 2 | 测试员实例化一个解析器，使用解析器读取配置文件 |
| 3 | 测试员使用容器实例化一个步骤1编写的测试bean |
|  | 测试员打印该测试bean的基本类型属性值 |
| 4 | 测试员打印该测试bean的容器类属性的值 |
| 5 | 测试员用容器实例化该测试bean的容器类，并打印其  属性的值 |
| 6 | 测试员用容器实例化该测试bean的外部引用类 |
| 7 | 测试员用容器实例化该测试bean的内部引用类 |
| 8 | 测试员运行测试代码，检查各显示属性值与配置文件中定义是否相同 |
| Postcondition  (Test Oracle) | 各显示属性值与配置文件中定义相同 |
| Specific Alternative Flows  (Test Sequence) | RFS 8 | |
| 1 | 各显示属性值与配置文件中定义不相同 |
| 2 | 检查不相同属性对应的解析功能的代码实现 |
| 3 | 修改对应部分代码 |
| 4 | RESUME STEP 8 |
| Postcondition  (Test Sequence) | 各显示属性值与配置文件中定义不相同 |

表12 解析配置文件

# 6非功能性测试

## 6.1容错性测试

|  |  |  |
| --- | --- | --- |
| **Test Case Specification** | | |
| Name | 容错性测试 | |
| Brief Description | 测试异常情况下程序能够回滚 | |
| Precondition | 代码块中有会使程序终止的代码块 | |
| Tester | 王昕 | |
| Dependency | None | |
| Test Setup | Name | 打开Eclipse |
| Description | 把代码输入到Eclipse中 |
| Basic Flow  (Test Setup) | Steps | |
| 1 | 创建工程项目 |
| 2 | 配置Spring 运行环境 |
| Postcondition  (Test Oracle) | Spring 运行环境搭建完成，能编写Spring 代码 |
| Basic Flow  (Test Sequence) | Steps | |
| 1 | 测试员进入Eclipse |
| 2 | 测试员编写Junit测试类 |
| 3 | 创建Junit运行环境 |
| 4 | 在测试类中添加除数为0的情况 |
| 5 | 执行测试 |
| 6 | 测试员 VALIDATE THAT 程序能够运行 |
| 7 | 测试员 VALIDATE THAT 有错误的代码没有测 |
| 8 | 测试员VALIDATE THAT程序回滚到了原始状态 |
| 9 | 测试员查看Juint测试转态 |
| Postcondition  (Test Oracle) | 程序正常执行 |
| Specific Alternative  Flows  (Test Sequence) | RFS 1 | |
| 1 | 程序不能执行 |
| Postcondition  (Test Oracle) | 测试对程序产生了影响，数据出现不一致状态 |
| RFS 2 | |
| 1 | Juint状态条红色 |
| Postcondition  (Test Oracle) | 测试前与测试后数据出现不一致现象 |

表13容错性测试

# 7参考文献

[1] http://junit.org/junit4/

[2] http://huihai.iteye.com/blog/1986568

[3] 肖丁、吴建林等. 软件工程模型与方法[M]. 北京邮电大学出版社. 2008-03.

[4] http://jingyan.baidu.com/article/fcb5aff7e1c277edaa4a71ee.html

[5] 蔡建平. 软件测试实验指导教程[M]. 清华大学出版社. 2009-11.

[6] 《Spring-framework 软件需求规格说明书V2.0》