Blade框架测试方案

文档版本控制

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 文档版本号 | 日期 | 作者 | 审核人 | 说明 |
| V1.0 | 2017/5/17 | 刘晔 | B组 |  |
| 2017/5/18 | 胡明昊 | B组 |  |
| 2017/5/18 | 穆鹏飞 | B组 |  |
| 2017/5/18 | 汪晓燕 | B组 |  |
| V1.1 | 2017/5/23 | 汪晓燕 | B组 |  |

目录

[1 项目简介部分 3](#_Toc482909774)

[1.1 文档编写目的 3](#_Toc482909775)

[1.2 测试项目背景描述 3](#_Toc482909776)

[1.3 测试工作内容和范围 4](#_Toc482909777)

[2 测试文档 5](#_Toc482909778)

[2.1 测试所需参考文档 5](#_Toc482909779)

[2.2 测试需提交文档 5](#_Toc482909780)

[3 测试安排和计划 6](#_Toc482909781)

[3.1 测试难点和重点 6](#_Toc482909782)

[3.1.1测试重点 6](#_Toc482909783)

[3.2 项目整体计划 6](#_Toc482909784)

[3.3 测试资源安排 7](#_Toc482909785)

[3.3.1人员分工 7](#_Toc482909786)

[3.3.2测试环境安排和使用 7](#_Toc482909787)

[3.3.3测试所需工具 8](#_Toc482909788)

[4 风险预估和应对 8](#_Toc482909789)

[5 功能测试方案 9](#_Toc482909790)

[5.1 case开发和管理的规范 9](#_Toc482909791)

[5.2 测试需求分析和策略制定 11](#_Toc482909792)

[5.2.1分功能测试需求分析 11](#_Toc482909793)

[5.2.2测试工具需求 18](#_Toc482909794)

[6 性能测试方案 18](#_Toc482909795)

[6.1 性能测试工具需求 18](#_Toc482909796)

[6.2 大规模的并发请求 19](#_Toc482909797)

[6.2.1场景概述 19](#_Toc482909798)

[6.2.2执行策略设计 19](#_Toc482909799)

[6.2.3测试数据需求 19](#_Toc482909800)

[6.2.4性能测试结果分析方法和预期 20](#_Toc482909801)

[6.3 频繁的数据操作 20](#_Toc482909802)

[6.4 路由转发的处理速度 20](#_Toc482909803)

# 项目简介部分

## **文档编写目的**

本实验小组制定的Blade框架的这一“测试方案”文档有助于实现以下目标：

1. 确定现有项目的信息和应测试的软件构件。
2. 列出推荐的测试需求（高级需求)。
3. 推荐可采用的测试策略，并对这些策略加以说明。
4. 确定所需的资源，并对测试的工作量进行估计。
5. 列出测试项目的可交付元素

## 测试项目背景描述

Blade是一款简洁易用的JavaWeb框架，它抽取了Spring的核心功能并重新实现。Blade在简洁和兼容两者之间选择了简洁，摒弃了繁复的配置，选择了Java 1.8，以及内嵌的服务器和数据库。它提供了IOC容器、MVC架构支持、模板引擎以及注解功能，并基于Maven进行管理。

**Blade的主要功能如下：**

1. IOC容器管理：提供实现创建，查找，注入的机制。
2. 配置管理：提供web应用的一般配置，如编码，错误页面处理及网站资源的组织等。
3. 数据库操作：提供访问数据库的一般接口，方便对数据的CRUD操作。
4. 拦截请求：在请求被处理之前，对请求进行过滤。
5. 路由转发：将请求与特定的业务控制等进行绑定。

**Blade的性能指标如下：**

1. 数据操作的快慢或者效率
2. 支持并发的请求数目的规模。

**Blade的一般指标如下：**

1. 框架的耦合度程度。
2. 框架的灵活度分析。

## 测试工作内容和范围

一般的，测试评审可能的阶段有评审、测试设计、单元测试、冒烟测试、手工测试、回归测试、自动化测试、性能测试、交叉自由测试等。

根据Blade的特点，选取评审、测试设计、单元测试、手工测试、自动化测试、性能测试等阶段。

考虑到Java语言层次对注解的支持以及拦截器作为IOC管理的一部分，在Blade的测试方案中不涉及拦截请求的功能的测试。

考虑到对框架的安全测试涉及到诸多内容，既要考虑框架本身设计上的问题，也要考虑第三方开发包可能存在的漏洞，当然，Java语言本身也不能保证绝对安全。

在测试方案中决定不进行Blade安全方面的测试，为此，做出了下述的假设。

不存在对框架的恶意攻击，一切都是在正常环境、正常操作下对框架及其应用的测试。

# 测试文档

## 测试所需参考文档

下表列出了制定和实施该测试方案时所需要使用的相关文档，并标明了各文档的可用性：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 文档[具体的文档名称和列表（版本/日期）] | 已创建或可用 | 已被接收或已经过复审 | 作者 | 备注 |
| 软件需求分析说明书（v1.7） | 是☑否□ | 是□否☑ | B组全体 |  |
| 类图 | 是☑否□ | 是☑否□ | B组全体 |  |
| 用例图 | 是☑否□ | 是☑否□ | B组全体 |  |
| 用例场景 | 是☑否□ | 是☑否□ | B组全体 |  |
| 单元测试设计方案 | 是☑否□ | 是□否☑ | B组各成员 |  |
| 性能测试方案 | 是☑否□ | 是□否☑ | B组全体 |  |
| 静态评审方案 | 是□否☑ | 是□否☑ | B组全体 |  |
| 测试工具参考文档 | 是□否☑ | 是□否☑ | B组全体 |  |

## 测试需提交文档

下表列出了制定和实施该测试方案时测试所需要提交的相关文档，并标明了各文档的可用性：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 文档[具体的文档名称和列表（版本/日期）] | 已创建或可用 | 已被接收或已经过复审 | 作者 | 备注 |
| 测试计划 | 是☑否□ | 是□否☑ | 穆鹏飞 |  |
| 单元测试设计（单元测试报告） | 是☑否□ | 是□否☑ | B组各成员 |  |
| 性能测试设计（性能测试报告） | 是☑否□ | 是□否☑ | B组全体 |  |
| 静态评审报告 | 是□否☑ | 是□否☑ | B组全体 |  |
| 项目总结 | 是□否☑ | 是□否☑ | B组全体 |  |
| 缺陷分析和测试设计补充 | 是□否☑ | 是□否☑ | B组全体 |  |
| 项目投入和时间数据汇总 | 是□否☑ | 是□否☑ | 穆鹏飞 |  |

# 测试安排和计划

## 测试难点和重点

考虑到web框架所提供的基本功能就是提高web应用的开发效率及web应用本身的健壮性。因此，大规模请求下的框架状态，数据操作的效率，以及框架的耦合度程度分析比较就显得比较重要了。

### 测试重点

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 重点项 | 重要性说明 | 备注 |
| 1 | 大规模并发的  用户请求 | 作为web框架，应当能够处理大量的并发用户请求，并且能够提供好的用户体验。 | 需要考虑自动化工具支持。 |
| 2 | 数据操作的效率 | 作为web框架，系统要能够支持快速的数据库操作 | 可以考虑使用单元测试方法 |
| 3 | 耦合度分析 | 作为web框架，系统既要提高开发的效率，也应注重框架的性能。 | 考虑使用代码静态审查方法，人工检查。 |

## 项目整体计划

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 项目阶段 | 时间段 | 参与人员 | 测试工作内容安排 | 产出 | 备注 |
| 讨论阶段 |  |  | 参与分析讨论 |  |  |
| 需求评审阶段 |  |  | 了解项目背景资料  反馈评审问题  参与需求评审  确认评审结论  初步评估测试计划 | 评审批注反馈  初步测试计划 |  |
| 详细设计及实施阶段 |  |  | 进行测试点拆分  确定测试方案  撰写测试case和相关关键字  准备测试数据  自动化生成case | 测试框架  测试点拆分文档  关键字列表  Case书写规范  测试case文档  自动化case  测试工具和程序 |  |
| 全面测试 |  |  | 执行手工测试  执行自动化case  性能测试  完善自动化case | 手工测试结论  部分关键字  完善或新补充的自动化case  性能测试结果  自动化case结果 |  |
| 项目总结阶段 |  |  | 相关总结；  资料汇总； |  |  |

## 测试资源安排

### 人员分工

下表列出了在此测试中的人员配备方面所作的各种暂时假定。

|  |  |
| --- | --- |
| 人员 | 主要职责或注释 |
| 刘晔 | 1. 分析测试需求 2. 制定测试方案测试全程参与 3. 完成框架的耦合度静态评审分析 4. 完成框架的并发请求的性能测试 |
| 胡明昊 | 1. 测试全程参与 2. 完成IOC功能模块的单元测试 3. 完成改进部分的单元测试 |
| 穆鹏飞 | 1. 测试全程参与 2. 制定测试计划 3. 完成数据操作功能模块的单元测试及其性能测试。 |
| 汪晓燕 | 1. 负责管理文档资料、case、程序、工具 2. 测试全程参与 3. 完成路由转发功能的单元测试及其性能测试 4. 测试分析和测试报告 |

### 测试环境安排和使用

下表列出了测试的系统环境

|  |
| --- |
| 硬件环境（服务器、网络、虚拟机等需求） |
| 局域网 |
| 普通PC机（i5处理器，8G内存以上） |
| 软件环境（相关操作系统、软件及环境配置等） |
| Windows 8以上或者Ubuntu 16.04 |
| Blade框架或其上开发的应用 |
| Chrome浏览器（或者firefox，IE等） |
| IdeaJ集成开发环境 |
| Java8虚拟机环境 |

### 测试所需工具

下表列出了在此项目的使用工具方面所作的各种假定。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 工具 | 获取和访问地址 | 用途 | 支持人员 | 使用阶段 | 使用时间 | 备注 |
| Word |  | 撰写方案、case |  | 测试准备 |  |  |
| Microsoft Project |  | 撰写测试计划 |  | 测试准备 |  |  |
| Git | bhsei/17TeamB\_blade.git | 代码、文档、工具管理  测试报告  测试数据 |  | 测试准备  测试执行  测试总结 |  |  |
| 模拟大规模并发请求工具（待定） | Windows或者Ubuntu | 性能压力测试 |  | 性能测试 |  |  |

# 风险预估和应对

下表列出了在此项目的测试工作所存在的各种风险的假定，需要考虑项目测试过程中可能发生的具体事务，分别分析并加以应对，然后体现在测试计划中。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 风险类型 | 风险  责任方 | 风险  内容 | 相应处理优先级 | 可能发生的阶段 | 可能发生的时间段 | 应对所需资源 | 应对  措施 | 备注 |
| 时间计划 |  |  |  |  |  |  | 合理计划  及时调整 |  |
| 资源协调 |  |  |  |  |  |  | 充分估计  预留buffer  及时调整 |  |
| 插入事务 |  |  |  |  |  |  | 预留buffer  及时调整 |  |
| 任务超预期 |  |  |  |  |  |  | 及时调整 |  |

**时间计划：**关键milestone无法匹配的延期风险。诸如测试项目存在deadline、计划受到客观条件限制、非己方责任导致地被动延期等等；

**资源协调：**包括所需资源不能如期到位，或者资源质量低于预期等风险。比如测试任务工作量与任务难度难以匹配的风险。

**插入事务：**包括临时插入高优先级的事务，打乱原有计划等风险。

**任务超预期：**实际执行时的工作复杂程度、结果的质量同预期不符所带来的风险。属于不可预期的风险，只能待出现时及时合理地调整。

风险分为可预期的和不可预期的，对于可预期的风险，可以要求资源，制定提前的应对措施。但是对于不可预期的风险，只能待出现时，充分考虑各方因素，及时调整。所以，对于可预期的风险，需要的能力是充分预估，对于不可预期的风险，需要的是及时察觉并调整应对。

# 功能测试方案

## case开发和管理的规范

为了规范测试过程，制定本测试过程所使用的测试用例示例模板如下所示。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 项目/软件 | BLADE | 程序版本 | 0.1 |  | | |
| 功能模块名 | Login | 编制人 | xxx |
| 用例编号 | BLADE\_Login\_1 | 编制时间 | 2017.5.12 |
| 相关的用例 | 无 |  |  |
| 功能特性 | 用户身份验证 |  |  |
| 测试目的 | 验证是否输入合法的信息，允许合法登陆，阻止非法登陆 |  |  |
| 预置条件 | 无 | 特殊规程说明 | 如数据库访问权限 |
| 参考信息 | 需求说明中关于“登陆”的说明 |  |  |
| 测试数据 | 用户名=yiyh 密码=1 | | | | | |
| 操作步骤 | 操作描述 | 数 据 | 期望结果 | 实际结果 | 实际结果 | 测试状态（P/F） |
| 1 | 输入用户名称，按“登陆”按钮。 | 用户名=yiyh，密码为空 | 显示警告信息“请输入用户名和密码！” |  |  |  |
| 2 | 输入密码，按“登陆”按钮。 | 用户名为空，密码=1 | 显示警告信息“请输入用户名和密码！” |  |  |  |
| 3 | 输入用户名和密码，按“登陆”按钮。 | 用户名=yiyh，密码=2 | 显示警告信息“请输入用户名和密码！” |  |  |  |
| 4 | 输入用户名和密码，按“登陆”按钮。 | 用户名=xxx，密码=1 | 显示警告信息“请输入用户名和密码！” |  |  |  |
| ..... | ..... | ..... | ..... | ..... | ..... | ..... |
| 测试人员 |  | 开发人员 |  |  | 项目负责人 |  |

## 测试需求分析和策略制定

### 分功能测试需求分析

根据测试框架中的各个部分，进行测试需求分析，确定测试内容和测试方法。

#### 路由转发功能模块

1. 主要功能描述

由于请求响应模块中设计的流程比较多，这里选取其中重要的部分即路由转发功能模块，测试其路由转发模块是否正常的工作。

路由转发模块主要用于进行资源请求与控制逻辑的匹配，以求获得正确的业务处理返回结果。

由于用户请求的不确定性，模块既需要提供处理正确请求的路由转发，也要能对一些错误的请求进行适当处理。

1. 测试点分析

|  |  |  |  |
| --- | --- | --- | --- |
| 测试点 | 测试方法类型 | 测试方法详述 | 备注 |
| A正确请求的路由转发是否成功 | 手工测试  自动化测试  自动化辅助测试 | 准备测试数据，根据测试数据编写需要测试的应用，手工或者自动化生成测试用例。 |  |
| B 无效请求的路由转发处理 | 手工测试  自动化测试  自动化辅助测试 | 准备测试数据，根据测试数据编写需要测试的应用，手工或者自动化生成测试用例。 |  |
|  |  |  |  |

【注：各个测试方法类型解释如下。】

**手工测试：**采用人工操作，并人工观察确认测试结果的测试方法。如无特别的创新方法，诸如数据准备和场景描述策略等，此方法可以一笔带过。

**自动化测试：**使用提前准备好的自动化case完全无人工干预的测试。该方法如果需要特别的工具、关键字开发，需要注明。

**自动化辅助测试：**使用工具，将测试的部分过程，比如结果保存（抓图）、数据上传、结果验证等用程序自动化实现，但是部分过程还需要人工验证的测试。该方法可以提高部分效率，但是或许需要人工去分析严重结果。

* **IOC管理模块**
  + - 1. 主要功能描述

IOC模块使用依赖翻转技术为开发人员提供统一的对象生成和管理。在开发期间能够通过IOC容器初始化对象，而不需要人为通过代码初始化对象、管理对象生命周期，在一定程度上简化了开发人员的工作，将开发人员的关注点从语言特性上分离，能够更加集中于业务逻辑。

* + - 1. 测试点分析

在IOC用例中，开发人员主要借助配置扫描包，注册对象，注入值三个步骤解决项目中类依赖的问题。但事实上在Blade框架中，三个步骤被封装在一个Init方法中，因此本处的测试一个单元测试将对应IOC用例RUCM的多个步骤。根据分析，得到如下测试内容：

1. 扫描注解后初始化的对象与预期相同
2. 空路径提示错误
3. IOC容器能够向指定的对象中注入依赖的值。
4. 部分测试用例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 项目/软件 | BLADE | 程序版本 | 0.1 | | |
| 功能模块名 | IOC | 编制人 | 胡明昊 | | |
| 用例编号 | U1 | 编制时间 | 2017.5.17 | | |
| 相关的用例 | IOC | | | | |
| 功能特性 | IOC管理用户创建的对象的生命周期，简化用户开发 | | | | |
| 测试目的 | 测试IOC能够正常管理用户的对象 | | | | |
| 预置条件 | 无 | | | | |
| 参考信息 | IOC对开发代码扫描，对所有的类进行初始化，并向包含@Inject的对象中注入初始化后的值。 | | | | |
| 测试用例 | 操作描述 | 数 据 | 期望结果 | 实际结果 | 测试状态（P/F） |
| 1 | 1. 初始化加载器  2. 指定默认包路径  3. 调用扫描方法扫描注解 | 测试项目Test，类com.blade.ioc.loader.IocAnnotationLoaderTest，默认包为"org.hmh" | IOC容器中包含的Bean与测试项目中的类一一对应 | IOC容器中包含的Bean鱼测试项目中的类均一一对应 | P |
| 2 | 1. 初始化加载器  2. 指定默认包路径  3. 调用扫描方法扫描注解 | 测试项目Test，类com.blade.ioc.loader.IocAnnotationLoaderTest，默认包为"" | IOC提示默认“包路径为空”异常 | IOC提示默认“包路径为空”异常 | P |
| 3 | 1. 初始化加载器  2. 手动创建HelloController对象  3. 确认HelloController中的HelloService为空且IOC中不存在HelloService  2. 生成HelloService对象  4. 向IOC容器中添加该对象。  5. 调用注入方法向HelloController对象注入HelloService  5. 检查Hello | 测试项目Test，类com.blade.ioc.FieldInjectorTest | 1. （3）中HelloController的HelloService为空  2. 注入后不为空，与放入IOC容器中值相同 | 1. （3）中HelloController的HelloService为空  2. 注入后不为空，与放入IOC容器中值相同 | P |
| 测试人员 | 胡明昊 | 开发人员 | 王爵 | 项目负责人 | 穆鹏飞 |

#### 配置模块

* + - 1. 主要功能描述
      2. 测试点分析
      3. 部分测试用例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 项目/软件 | BLADE | 程序版本 | 0.1 | | |
| 功能模块名 | 配置 | 编制人 | 穆鹏飞 | | |
| 用例编号 | U2 | 编制时间 | 2017.5.17 | | |
| 相关的用例 | 配置 | | | | |
| 功能特性 | 对程序所需的资源以及内容更进行配置 | | | | |
| 测试目的 | 验证配置的资源信息在程序中是否可以正确获取到 | | | | |
| 预置条件 | 无 | | | | |
| 参考信息 | 无 | | | | |
| 测试用例 | 操作描述 | 数 据 | 期望结果 | 实际结果 | 测试状态（P/F） |
| 1 | 1.开发人员在基本配置类中配置Web服务器端口 | 正常运行的测试项目代码configureTest01 | 1.可以通过配置好的端口访问编写好的程序。  2.不可通过未配置的端口访问编写好的程序。 |  |  |
| 2 | 1.开发人员在基本配置类中配置是否是开发者模式 | 正常运行的测试项目代码configureTest02 | 1.当在配置文件中开启开发者模式时，测试程序应该支持开发者模式所具有的功能。  2. .当在配置文件中未开启开发者模式时，测试程序应该不支持开发者模式所具有的功能。 |  |  |
| 3 | 1.开发人员设置一个静态资源目录，存放在资源目录下 | 正常运行的含有资源文件的正常代码configureTest03 | 1.静态资源文件能正确的放到配置文件配置的静态资源目录下。 |  |  |
| 4 | 1.开发人员在出错界面类中设置出现404的统一界面。  2.开发人员在出错界面类中设置出现500的统一界面。 | 含有404,500出错信息的代码configureTest04 | 1.对于正常运行的代码，不会出现404与500界面。  2.对于有404错误的代码，则应跳转到在配置类中404界面中。 |  |  |
| 5. | 开发人员在配置类中获取配置信息 | 含有配置信息的测试代码configureTest05 | 1.在配置类中能正确的访问到测试类中的配置信息 |  |  |
| 测试人员 | 穆鹏飞 | 开发人员 | 王爵 | 项目负责人 | 穆鹏飞 |

#### 数据库操作模块

1. 主要功能描述
2. 测试点分析
3. 部分测试用例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 项目/软件 | BLADE | 程序版本 | 0.1 | | |
| 功能模块名 | 数据增加 | 编制人 | 穆鹏飞 | | |
| 用例编号 | U3 | 编制时间 | 2017.5.17 | | |
| 相关的用例 | 数据库增加 | | | | |
| 功能特性 | 在数据库中增加需要添加的信息 | | | | |
| 测试目的 | 验证数据库增加模块是否能完成在数据库中正确的增加操作 | | | | |
| 预置条件 | 无 | | | | |
| 参考信息 | 无 | | | | |
| 测试用例 | 操作描述 | 数 据 | 期望结果 | 实际结果 | 测试状态（P/F） |
| 1 | 1.开发人员的正常插入操作 | 数据增加测试代码  AddTest01 | 1.对于正常的数据增加操作，应成功在数据库中插入一条内容。 |  |  |
| 2 | 1.数据库的异常插入操作 | 插入时含有异常数据的增加测试代码  AddTest02 | 1.对于数据库表中不允许非空的数据，插入时该列数据为空，则不能执行插入操作。  2.插入的主键重复时，不能执行插入操作。  3.插入时数据类型与数据库数据表中的数据类型不一致时，将不能执行插入操作 |  |  |
| 测试人员 | 穆鹏飞 | 开发人员 | 王爵 | 项目负责人 | 穆鹏飞 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 项目/软件 | BLADE | 程序版本 | 0.1 | | |
| 功能模块名 | 数据库删除 | 编制人 | 穆鹏飞 | | |
| 用例编号 | U3 | 编制时间 | 2017.5.17 | | |
| 相关的用例 | 数据库删除 | | | | |
| 功能特性 | 在数据库中删除需要删除的数据 | | | | |
| 测试目的 | 验证数据库删除模块是否能正确完成数据的删除功能 | | | | |
| 预置条件 | 无 | | | | |
| 参考信息 | 无 | | | | |
| 测试用例 | 操作描述 | 数 据 | 期望结果 | 实际结果 | 测试状态（P/F） |
| 1 | 1.开发人员的正常删除操作 | 数据删除测试代码  DeleteTest01 | 1.对于正常的数据删除操作，应成功在数据库中删除一条内容。 |  |  |
| 2 | 1.数据库的异常删除操作 | 删除的数据不存在AddTest02 | 1.要删除的数据在数据库中不存在，删除失败。 |  |  |
| 测试人员 | 穆鹏飞 | 开发人员 | 王爵 | 项目负责人 | 穆鹏飞 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 项目/软件 | BLADE | 程序版本 | 0.1 | | |
| 功能模块名 | 数据库查询 | 编制人 | 穆鹏飞 | | |
| 用例编号 | U4 | 编制时间 | 2017.5.17 | | |
| 相关的用例 | 数据库查询 | | | | |
| 功能特性 | 在数据库中查询需要查看的数据 | | | | |
| 测试目的 | 验证数据库查询模块是否能正确完成数据的查询功能 | | | | |
| 预置条件 | 无 | | | | |
| 参考信息 | 无 | | | | |
| 测试用例 | 操作描述 | 数 据 | 期望结果 | 实际结果 | 测试状态（P/F） |
| 1 | 1.开发人员的普通查询操作 | 数据查询测试代码  SelectTest01 | 1.对于正常的数据查询操作，应成功查询出用户需要查询的内容。 |  |  |
| 2 | 1.数据分页查询 | 数据分页查询测试代码SelectTest02 | 1.根据分页要求，正确的展示分页查询的内容。 |  |  |
| 测试人员 | 穆鹏飞 | 开发人员 | 王爵 | 项目负责人 | 穆鹏飞 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 项目/软件 | BLADE | 程序版本 | 0.1 | | |
| 功能模块名 | 数据修改 | 编制人 | 穆鹏飞 | | |
| 用例编号 | U5 | 编制时间 | 2017.5.17 | | |
| 相关的用例 | 数据库修改 | | | | |
| 功能特性 | 开发人员可以横据要求在数据库中修改需要修改的内容 | | | | |
| 测试目的 | 验证数据库修改模块是否能完成在数据库中正确的修改操作 | | | | |
| 预置条件 | 无 | | | | |
| 参考信息 | 无 | | | | |
| 测试用例 | 操作描述 | 数 据 | 期望结果 | 实际结果 | 测试状态（P/F） |
| 1 | 1.开发人员的正常修改操作 | 数据修改测试代码  UpdateTest01 | 1.对于正常的数据修改操作，应成功修改数据库中的一条内容。 |  |  |
| 2 | 1.数据库的异常修改操作 | 插入时含有异常数据的数据修改测试代码  UpdateTest02 | 1.对于数据库表中不允许非空的数据，修改后该列数据为空，则不能执行修改操作。  2.修改后的主键重复时，不能执行修改操作。  3.修改后数据类型与数据库数据表中的数据类型不一致时，将不能执行修改操作 |  |  |
| 测试人员 | 穆鹏飞 | 开发人员 | 王爵 | 项目负责人 | 穆鹏飞 |

* BladeTest测试模块
  + - 1. 主要功能描述

Blade提供了非常易用的IOC容器，但IOC容器给测试人员带来了极大的麻烦，因为IOC容器打破了原有的依赖关系。在不启动Blade的情况下进行单元测试，对象无法从IOC容器中获得依赖的对象因此无法正常运行大部分程序代码。在BladeTest的辅助下，测试开始之前会提供一个IOC容器用于提供类与类的依赖关系，保证了测试程序的正常运行。

* + - 1. 测试点分析

在提供测试环境的过程中，BladeTest组件主要抽取Blade核心的IOC加载模块，并提供默认搜索包使得IOC容器初始化成功。在此基础上，利用Java的反射机制，向测试程序中使用到的对象中注入依赖值。由于BladeTest采用适配器模式对Junit进行了封装，暴露的接口需要通过JUnit运行环境回调使用，因此仅能进行集成测试。

综上，对于BladeTest的测试的可做如下测试：

1. 测试清理IOC容器
2. 测试配置默认包，确认IOC容器初始化成功
3. 测试BladeTest能够向测试类中注入值
   * + 1. 部分测试用例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 项目/软件 | Blade | 程序版本 | 1.0 | | |
| 功能模块名 | BladeTest | 编制人 | 胡明昊 | | |
| 用例编号 | U6 | 编制时间 | 2017.5.23 | | |
| 相关的用例 | IOC | | | | |
| 功能特性 | 为测试程序提供模拟的IOC环境保证程序正常运行。 | | | | |
| 测试目的 | 发现BladeTest中可能存在的错误 | | | | |
| 预置条件 | 无 | | | | |
| 参考信息 | 使用注解@RunWith接入JUnit运行器，使用BladeConfiguration配置默认搜索路径。 | | | | |
| 测试用例 | 操作描述 | 数 据 | 期望结果 | 实际结果 | 测试状态（P/F） |
| 1 | 1. 配置默认包  2. 实例化IOCAdapter对象  3. 确认IOC容器中对象个数大于0  3. 通过IOCAdapter清理IOC容器  4. 确认IOC容器中的对象个数为0 | 测试项目BladeTest，类cn.hhchat.BladeTest.adapter.IOCAdapterTest，默认包为"org.hmh" | 1. (3)中IOC容器中对象个数大于0  2. 清理后IOC容器中对象个数为0 | 1. (3)中IOC容器中对象个数大于0  2. 清理后IOC容器中对象个数为0 | P |
| 2 | 1. 配置默认包  2. 实例化IOCAdapter对象  3. 确认IOCAdaptor对象包含预计的对象 | 测试项目BladeTest，类com.blade.ioc.loader.IocAnnotationLoaderTest，默认包为"org.hmh.controller" | IOC中的对象与预计的对象一一对应 | IOC中的对象与预计的对象一一对应 | P |
| 3 | 1. 配置运行器  2. 配置扫描包  3. 初始化加载器  4. 在测试类中声明需要注入的HelloController变量  5. 编写包含调用HelloController方法的测试  6. 运行测试程序 | 测试项目Test，类org.hmh.MainTest. TestMain，默认包为"org.hmh" | HelloController被调用的方法将正确的执行，输出"Say before in Controller" | HelloController被调用的方法将正确的执行，输出"Say before in Controller" | P |
| 4 | 1. 配置运行器  2. 配置扫描包  3. 初始化加载器  4. 在测试类中声明需要注入的HelloController变量  5. 编写包含调用HelloController方法的测试  6. 运行测试程序 | 测试项目Test，类org.hmh.MainTest. TestMain，默认包为"org.hmh.model" | 提示"字段HelloService没有可依赖的对象"，无法注入对象。 | 提示"字段HelloService没有可依赖的对象"，无法注入对象。 | P |
| 测试人员 | 胡明昊 | 开发人员 | 王爵 | 项目负责人 | 穆鹏飞 |

### 测试工具需求

BladeTest：blade框架单元测试工具。

Junit: Java单元测试框架。

# 性能测试方案

## 性能测试工具需求

http\_load web：框架压力测试工具。【其他】

## 大规模的并发请求

### 场景概述

在网站开发过程中，随着应用市场的扩大，用户访问量的增加，有必要在网站的开始设计阶段，考虑使用具有能处理高并发特性的框架进行后续的开发实现。

主要的关注点是框架能否提供较为多的轻量级线程以及页面缓存等技术进行用户请求的处理，以支持大规模的并发请求。

大规模并发请求的一般场景生成：



### 执行策略设计

|  |  |  |
| --- | --- | --- |
| 性能场景 | 执行策略（并发数、时长） | 备注 |
| 发送请求，等待响应 | 10用户并发，发送请求，等待响应，重复操作10分钟，退出。 | 得到不同并发数下系统的性能指标  对系统的容量做出估计  列出测试的数据指标项有哪些，值在什么区间内 |
| 50用户并发，，发送请求，等待响应，重复操作10分钟，退出。 |
| 100用户并发，发送请求，等待响应，重复操作15分钟 |

### 测试数据需求

根据Blade框架编写web应用，仅仅提供首页服务，测试输入为普通的网站首页url地址。

### 性能测试结果分析方法和预期

测试结果采用自动化辅助人工分析方法，预期Web框架应当支持每秒10000次的并发访问请求，页面响应时间应不超过10s。

## 路由转发的处理性能

### 场景概述

路由转发模块主要用于进行资源请求与控制逻辑的匹配，以求获得正确的业务处理返回结果。网站应用中，路由转发功能非常重要，在很多的应用框架中，路由转发大多是依赖于应用其所运行的web Server。比如Spring 对tomcat的依赖。作为一个集成了server的框架，一方面我们需要关注Blade对路由转发处理的正确性，另一方面，我们也需要关注Blade对于路由转发功能的性能表现上，所以有必要对路由转发模块，既进行单元测试，也对其进行性能测试。

### 执行策略设计

|  |  |  |
| --- | --- | --- |
| 性能场景 | 执行策略（并发数、时长） | 备注 |
| 发送请求，路由转发处理 | 每秒10并发请求，发送请求，监测路由转发处理。重复操作10分钟 | 目标是得到路由转发处理的峰值吞吐量/s |
| 每秒50并发请求，发送请求，监测路由转发处理。重复操作10分钟 |
| 每秒100并发请求，发送请求，监测路由转发处理。重复操作10分钟 |

### 测试数据需求

使用BladeTest测试框架测试Blade的路由转发性能，需要准备好测试代码，既有请求部分，也有路由控制处理部分的代码。

### 测试目标

获得Blade框架路由转发模块的峰值性能，即路由转发请求的吞吐量/s。