# 目录

[1 引言 1](#_Toc477470664)

[1.1 背景说明 1](#_Toc477470665)

[1.2 参考资料 1](#_Toc477470666)

[1.3 术语和缩略语 1](#_Toc477470667)

[2 软件总体概述 1](#_Toc477470668)

[2.1 目标 1](#_Toc477470669)

[2.2 系统模型 1](#_Toc477470670)

[2.3 假设和约束 1](#_Toc477470671)

[3 详细需求 1](#_Toc477470672)

[3.1 功能需求 1](#_Toc477470673)

[3.2 性能需求 1](#_Toc477470674)

[3.3 输入和输出 2](#_Toc477470675)

[3.4 数据库特性 2](#_Toc477470676)

[3.5 故障处理 2](#_Toc477470677)

[4 环境需求 2](#_Toc477470678)

[4.1 设备环境 2](#_Toc477470679)

[4.2 支持软件环境 2](#_Toc477470680)

[4.3 接口 2](#_Toc477470681)

[4.4 安全和保密 2](#_Toc477470682)

# 软件需求说明书

## 1 引言

### 1.1 编写目的

以Blade开源项目源码和资料作为基础，分析Blade框架的设计需求。结合软件工程综合实验相关要求、UML建模技术及源码注释生成需求模型。输出产物为该软件需求规格说明书，指导后续测试实验和改进实验。

### 1.2 系统概述

90年代末期，JavaWeb技术开始应用于服务器、网站开发。Sun公司制定了J2EE标准，借助编程语言Java快速的发展并流行起来。J2EE的广泛实现是在1999年和2000年开始的，它的出现带来了诸如事务管理之类的核心中间层概念的标准化，但是在实践中并没有获得绝对的成功。因为J2EE的开发效率，开发难度和实际的性能都令人失望。而Spring框架出现的初衷就是为了解决类似的这些问题，提供了一整套的JavaWeb支持，包括安全、事务、数据库等操作的简化。但是Spring是企业级的，所以更关注于需求，扩展了越来越多的功能，衍生出许多其他的分支项目，导致框架越来越大，也越来越杂，甚至演化出Spring Boot用于简化Spring自身的配置。因此选择Spring如此庞大、驳杂的框架在短期内进行分析是非常困难和不明智的。

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在简洁和兼容两者上选择了简洁，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。

它的主要特点如下：

* + 基本零配置
  + 轻量级。不依赖于更多的库，摆脱SSH的臃肿，模块化设计，使用起来更轻便
  + Restful风格的路由接口
  + 单jar运行易于部署

因此，本次实验选择Blade框架作为分析的目标，并撰写此需求说明书。

### 1.3 参考资料

//TODO

### 1.4 术语和缩略语

在Blade框架中，涉及了多项Java语言和计算机网络的术语，较为核心的概念术语及其解释如下所示：

1）MVC：Model View Controller，即模型—视图—控制器的缩写，一种软件设计典范。使用业务逻辑、数据、界面分离的方法组织代码，将业务逻辑聚集于一个部件。在改进和个性化定制界面及用户交互的同时，不需要重新改写业务逻辑。

2）IOC：Inversion of control的缩写，意为控制反转，一种重要的面向对象编程的法则。它能指导我们如何设计出松耦合、更优良的程序。传统应用程序都是由开发人员在类内部主动创建依赖对象，从而导致类与类之间高耦合，难于测试；有了IOC容器后，把创建和查找依赖对象的控制权交给了容器，由容器进行注入组合对象。对象间互相不知道对方的存在，而统一由容器进行管理，利于功能复用。更重要的是使得程序的整个体系结构变得非常灵活，耦合性低。将设计好的对象交给容器控制，而不是传统的在对象内部直接控制。

3）DI：Dependency Injection，即“依赖注入”： 组件之间依赖关系由容器在运行期决定，形象的说，即由容器动态的将某个依赖关系注入到组件之中。依赖注入的目的并非为软件系统带来更多功能，而是为了提升组件重用的频率，并为系统搭建一个灵活、可扩展的平台。通过依赖注入机制，我们只需要通过简单的配置，而无需任何代码就可指定目标需要的资源，完成自身的业务逻辑，而不需要关心具体的资源来自何处，由谁实现。

4）路由：在Blade中，路由是一个HTTP方法配对一个URL匹配模型，每一个路由可以应对一个处理方法。

5）拦截器：Blade中的拦截器用于接收请求时做额外操作，比如存储数据，校验数据，过滤请求等。

6）HTML：超文本标记语言。

7）RESTful架构：Representational State Transfer，一种软件架构风格，提供了一组设计原则和约束条件。它主要用于客户端和服务器交互类的软件。基于这个风格设计的软件可以更简洁，更有层次，更易于实现缓存等机制。

8）服务器：一个管理资源并为用户提供服务的计算机软件，用于接收用户请求并响应相应的数据给用户。

9）客户端：客户端（Client），是指与服务器相对应，为客户提供本地服务的程序。一般安装在普通的用户机上，需要与服务端互相配合运行。

10）JSON：JSON(JavaScript Object Notation, JS 对象标记) 是一种轻量级的数据交换格式。它基于 ECMAScript 规范的一个子集，采用完全独立于编程语言的文本格式来存储和表示数据。简洁和清晰的层次结构使得 JSON 成为理想的数据交换语言。 易于人阅读和编写，同时也易于机器解析和生成，并有效地提升网络传输效率。

11）JDBC: JDBC（Java Data Base Connectivity, java数据库连接）是一种用于执行SQL语句的Java API，可以为多种关系数据库提供统一访问，它由一组用Java语言编写的类和接口组成。JDBC提供了一种基准，据此可以构建更高级的工具和接口，使数据库开发人员能够编写数据库应用程序。

12）数据库：简单来说可视为电子化的文件柜—存储电子文件的处所，用户可以对文件中的数据运行新增、截取、更新、删除等操作

13）Java：Java是一种广泛使用的计算机编程语言，拥有跨平台、面向对象、泛型编程的特性，广泛应用于企业级Web应用开发和移动应用开发。

14）Java web：是用Java技术来解决相关web互联网领域的技术总和。web包括：web服务器和web客户端两部分。

15）Tomcat：Tomcat是由Apache软件基金会下属的Jakarta项目开发的一个Servlet容器，按照Sun Microsystems提供的技术规范，实现了对Servlet和JavaServer Page（JSP）的支持，并提供了作为Web服务器的一些特有功能，如Tomcat管理和控制平台、安全域管理和Tomcat阀等。由于Tomcat本身也内含了一个HTTP服务器，它也可以被视作一个单独的Web服务器。

16）HTTP协议：HTTP是一个客户端终端（用户）和服务器端（网站）请求和应答的标准（TCP）。

17）GET：常用的HTTP请求方法，从指定的资源请求数据。GET请求可被缓存，并且保留在浏览器历史记录中，所以GET请求不应在处理敏感数据时使用。

18）POST：另一种常用的HTTP请求方法，不仅可以请求数据还可以向指定的资源提交要被处理的数据。POST请求不会被缓存，并且不会保留在浏览器历史纪录中。

19）Request: Request对象是从客户端向服务器发出请求，包括用户提交的信息以及客户端的一些信息。客户端可通过HTML表单或在网页地址后面提供参数的方法提交数据，然后通过request对象的相关方法来获取这些数据。

20）Response: Response对象用于动态响应客户端请求，控制发送给用户的信息，并将动态生成响应。

## 2 软件总体概述

### 2.1 目标。

* 模块化(你可以选择使用哪些组件)，以完成需要的功能。
* 使开发出的框架是一款轻量级的，不依赖更多的库，摆脱SSH的臃肿，模块化设计，使用起来更轻便。
* 使框架具有Restful风格的路由接口。
* 不依赖与Orm，以 Active Record方式对数据库进行各种操作。
* 具有模板引擎支持。
* 在开发和发布时以非web的方式。

### 2.2 系统模型

图示说明该软件的所有功能及其相互关系和数据传递情况

### 2.3 假设和约束

说明影响该软件开发和运行环境的某些假设和约束，论述影响系统能力的若干限制。

## 3 详细需求

### 3.1 功能需求

对系统中每一个功能要用图示或文字详细描述

### 3.2 性能需求

定量的描述该软件系统应满足的具体性能需求

### 3.3 输入和输出

### 3.4 数据库特性

Bladed框架的数据库有以下特性：

1. 语法简介，代码量少。
2. 除了日志接口，不依赖第三方框架。
3. DSL风格，在程序书写过程中，类似链式风格。
4. 内置连接池，支持与其他连接池共用。

### 3.5 故障处理

HTTP 400 - 请求无效  
HTTP 401.1 - 未授权：登录失败  
HTTP 401.2 - 未授权：服务器配置问题导致登录失败  
HTTP 401.3 - ACL 禁止访问资源  
HTTP 401.4 - 未授权：授权被筛选器拒绝  
HTTP 401.5 - 未授权：ISAPI 或 CGI 授权失败  
HTTP 403 - 禁止访问  
HTTP 403 - 对 Internet 服务管理器 (HTML) 的访问仅限于 Localhost  
HTTP 403.1 禁止访问：禁止可执行访问  
HTTP 403.2 - 禁止访问：禁止读访问  
HTTP 403.3 - 禁止访问：禁止写访问  
HTTP 403.4 - 禁止访问：要求 SSL  
HTTP 403.5 - 禁止访问：要求 SSL 128  
HTTP 403.6 - 禁止访问：IP 地址被拒绝  
HTTP 403.7 - 禁止访问：要求客户证书  
HTTP 403.8 - 禁止访问：禁止站点访问  
HTTP 403.9 - 禁止访问：连接的用户过多  
HTTP 403.10 - 禁止访问：配置无效  
HTTP 403.11 - 禁止访问：密码更改  
HTTP 403.12 - 禁止访问：映射器拒绝访问  
HTTP 403.13 - 禁止访问：客户证书已被吊销  
HTTP 403.15 - 禁止访问：客户访问许可过多  
HTTP 403.16 - 禁止访问：客户证书不可信或者无效  
HTTP 403.17 - 禁止访问：客户证书已经到期或者尚未生效  
HTTP 404.1 - 无法找到 Web 站点  
HTTP 404 - 无法找到文件  
HTTP 405 - 资源被禁止  
HTTP 406 - 无法接受  
HTTP 407 - 要求代理身份验证  
HTTP 410 - 永远不可用  
HTTP 412 - 先决条件失败  
HTTP 414 - 请求 - URI 太长

出现以上错误时，则调用框架内置的400错误界面，并显示，同时提示出错信息。

HTTP 500 - 内部服务器错误  
HTTP 500.100 - 内部服务器错误 - ASP 错误  
HTTP 500-11 服务器关闭  
HTTP 500-12 应用程序重新启动  
HTTP 500-13 - 服务器太忙  
HTTP 500-14 - 应用程序无效  
HTTP 500-15 - 不允许请求 global.asa

出现以上错误，则调用框架内置的500错误界面，并显示，同时提示错误信息。

Error 501 - 未实现

出现以上错误，调用框架内置的501错误界面，并显示，同时提示错误信息。

HTTP 502 - 网关错误

出现以上错误时，调用框架内置的502界面，并显示，同时提示错误信息。

## 4 环境需求

### 设备环境

操作系统：Microsoft Windows winxp以上；

硬件：Intel i3 CPU 2.30GHz 2.30GHz ，2.00GB的内存。

### 支持软件环境

JDK1.6以上

Maven安装配置：

<dependencies>

<dependency>

<groupId>com.bladejava</groupId>

<artifactId>blade-core</artifactId>

<version>1.7.2-beta</version>

</dependency>

<dependency>

<groupId>com.bladejava</groupId>

<artifactId>blade-embed-jetty</artifactId>

<version>0.1.3</version>

</dependency>

</dependencies>

### 4.3 接口

### 4.4 安全和保密

1.在对数据库进行操作的时候，如若遇到突发的意外情况，如网络通信故障，突然断电等情况，要保证对数据库操作的ACID属性。

2.对于数据库的各种操作而言，要防止非法用户进行sql注入，危害内部数据。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 编号 | 中文名称 | 英文名称 | 父类 | 说明 |
|  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 中文名称 | 英文名称 | 类型 | 说明 |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 中文名称 | 英文名称 | 参数说明 | 返回值类型 | 说明 |
|  |  |  |  |  |