**Delicacy\_Square**

**系统开发指南**

苏睿电商版权所有

www.surui.com

**版本信息**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **版本** | **时间** | **状态** | **简要描述** | **部门** | **更改人** | **批准人** |
| V1.0 | 2013-10-02 | N | 创建 |  | 刘从文 |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

注：状态可以为N-新建、A-增加、M-更改、D-删除。

**目录**

[1. 概述 1](#_Toc368483506)

[1.1. 文档目的 1](#_Toc368483507)

[1.2. 范围 1](#_Toc368483508)

[1.3. 规范定义原则 1](#_Toc368483509)

[1.4. 相关文档 2](#_Toc368483510)

[1.5. 参考 2](#_Toc368483511)

[2. 系统开发规范 2](#_Toc368483512)

[2.1. 展示层 2](#_Toc368483513)

[2.2. 应用层 2](#_Toc368483514)

[2.3. 业务逻辑服务层 3](#_Toc368483515)

[2.4. 缓存 4](#_Toc368483516)

[2.5. cookie 5](#_Toc368483517)

[2.6. 日志 5](#_Toc368483518)

[2.7. 异常 6](#_Toc368483519)

[2.8. 事务 7](#_Toc368483520)

# 概述

## 文档目的

用于指导开发和设计人员对Delicacy\_Square系统的设计和开发工作，确保所开发的代码具有统一良好的可读性，符合通常优化准则的语句结构和使用用法，具有良好的维护性。同时也为设计评审和代码评审提供统一评审依据。

通过学习本文档，能够对Delicacy\_Square系统模块有一个简单的认识了解，掌握开发的常用语法、使用技巧及注意事项。

## 范围

本文档适用于以下应用场景：

* 本文档适用于Delicacy\_Square项目；
* 本文档涉及规范在新开发和维护代码中必须遵守；

## 规范定义原则

本文档内容定义及更新应遵循以下原则：

* 简单性原则

文档只定义项目设计编码中应遵循的规则和建议，不作为技术提高资料，对一些规范的理解，可以查询相关的技术资料；

* 指导性原则：

文档中尽量给出建议的编写示例供开发人员；

## 相关文档

编码规范与范例.doc

## 参考

# 系统开发规范

## 展示层

参见编码规范与范例.doc -- web前端部分

## 应用层

### 概述

应用层为myit-protal和myit-admin两个工程，作为Web应用程序，提供页面展示和请求/响应的处理功能。

myit-protal对外部用户公开，作为其通过Web浏览器访问Delicacy\_Square系统的入口；myit-admin则提供系统及数据管理等功能。

### 包命名规则

myit-portal

|  |  |  |
| --- | --- | --- |
| **包名** | **说明** | **备注** |
| com.myit.portal.action | 前台控制层文件包 |  |
| com.myit.portal.bean | 前台javaBean文件包 |  |
| com.myit.portal.filter | 前台过滤器文件包 |  |
| com.myit.portal.servlet | 前台servlet文件包 |  |
| com.myit.portal.util | 前台工具集合文件包 |  |
| com.myit.portal.converter | 前台格式转换工具文件包 |  |

myit-admin

|  |  |  |
| --- | --- | --- |
| **包名** | **说明** | **备注** |
| com.myit.admin.action | 前台控制层文件包 |  |
| com.myit.admin.bean | 前台javaBean文件包 |  |
| com.myit.admin.filter | 前台过滤器文件包 |  |
| com.myit.admin.servlet | 前台servlet文件包 |  |
| com.myit.admin.util | 前台工具集合文件包 |  |
| com.myit.admin.converter | 前台格式转换工具文件包 |  |
| com.myit.admin.servlet | 前台servlet文件包 |  |
|  |  |  |

### 类规则

类名：{模块功能名}Action，

增加类的注解：

|  |
| --- |
| @Controller  @RequestMapping("url") |

作为controller的请求url；

声明需要用到的service，并加注解:

|  |
| --- |
| @Resource |

### 方法规则

增加方法的注解：

|  |
| --- |
| @RequestMapping("url") |

作为方法的请求位置；

方法为ajax调用：返回参数需要是String, 返回值为视图名称；

方法为非ajax：方法中需要有ModelMap，存放返回的参数，调用方法返回值为视图名称。

## 业务逻辑服务层

### 概述

服务层定义为myit-server和myit-intf工程，为应用层提供各种组合/原子服务的调用功能。其中接口定义在myit-intf，实现类在myit-server。

### 包命名规则

myit-server

|  |  |  |
| --- | --- | --- |
| **包名** | **说明** | **备注** |
| com.myit.server.service | 服务层文件包 |  |
| com.myit.server.util | 服务层工具类文件包 |  |

### 业务service

业务service按功能模块放在对应的包内。

service接口命名规则：{功能名}Service。

service实现类命名规则：{功能名}ServiceImpl。

实现类需要增加注解@Service；声明需要实现的接口Service；应用时添加加注解@Resource。

### 数据访问dao

|  |  |  |
| --- | --- | --- |
| **包名** | **说明** | **备注** |
| com.myit.server.dao | 数据访问层文件包 |  |
| com.myit.server.model | 数据实体模型文件包 |  |

数据库访问统一采用Hibernate接口实现。

实现类中需要访问数据库的，需要继承HibernateDaoSupport父类，注入SessionFactory的复写方法：

|  |
| --- |
| @Autowired  **public** **void** setSessionFactoryOverride(SessionFactory sessionFactory) {  **super**.setSessionFactory(sessionFactory);  } |

实现sessionFactory的初始化

## 缓存

### 概述

为了提高系统响应速率，增强用户体验，对于系统内一些常用的、变化较小的数据做系统缓存处理。

本系统主要使用Cache-Client组件来实现数据的缓存，数据以json串的形式缓存存储。

### 使用约定

本系统涉及到需要缓存的数据与对应键值的清单如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **序号** | **缓存数据** | **键值** | **有效期** | **备注** |
| 1 | 酒店详细信息 | com.suning.b2g.hotel.hotelDetail\_{hotelId} | 24hour | hotelId：酒店id |
| 2 | 酒店价格信息 | com.suning.b2g.hotel.hotelPrice\_{hotelId}\_{checkInDate}\_{checkOutDate} | 6hour | hotelId：酒店id  checkInDate：入住日期  checkOutDate：离店日期 |
| 3 | 酒店城市商圈行政区信息 | com.suning.b2g.hotel.cityArea\_{cityId} | 30day | cityId：城市id |
| 4 | 酒店城市分类信息 | com.suning.b2g.hotel.cityCatalog\_{groupId} | 30day | groupId：企业id |
| 5 | 酒店差旅政策 | com.suning.b2g.hotel.policy\_{groupId}\_{levelId} | 30day | groupId：企业id  levelId： 职级id |
|  |  |  |  |  |

## cookie

### 概述

企业差旅酒店系统，用户在进入到酒店详情页时，需要记录改酒店为用户浏览过的酒店。由于每个用户浏览酒店情况不同，且不需要做数据持久化保存，所有考虑采用javascript cookie来实现。

### 使用约定

本系统涉及到需要cookie的数据与对应键值的清单如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **序号** | **缓存数据** | **键值** | **有效期** | **备注** |
| 1 | 浏览过的酒店信息 | \_CK\_VIEWED\_HOTELS | 30day | 酒店详情页面追加；  酒店列表页面展示列表。 |
|  |  |  |  |  |

## 日志

### 概述

Delicacy\_Square系统项目的日志框架使用Logback，通过SpringAOP将关注的方法入参、返回值，以及抛出的异常作为日志并以文件形式保存在应用服务器对应目录。前台日志和后台日志分开存放。

### 记录原则

* 在一个方法的入口及返回前，记录对应的info级别日志，以跟踪系统方法调用的情况，形式如下：

LOGGER.info(“{方法名} IN”);

LOGGER.info(“{方法名} OUT”);

* 方法存在入参，则在记录入口日志后，需要对传入的业务参数记录debug级别日志，以定位方法调用时传入的入参情况，形式如下：

LOGGER.debug(“Params: {参数名1} = ” + {参数1} + “, {参数名2} = ” + {参数名2} ... );

* 方法存在出参，则在记录出口日志前，需要对返回的参数记录debug级别日志，以定位方法调用后返回的出参情况，形式如下：

LOGGER.debug(“Return: {出参名} = ” + {出参} );

* 调用其他接口/方法，则在调用之前，需要对应调用的接口/方法记录info级别日志，以跟踪系统方法调用的情况；如果接口/方法存在返回值，则在调用之后，需要对返回值记录info级别日志，形式如下：

LOGGER.info(“Invoke {方法名}”);

LOGGER.info(“ {变量名} = ” + {变量});

## 异常

### 概述

采用unchecked异常传递机制。底层抛出的checked异常被转化为unchecked异常，各层的public方法无需显式声明会抛出的异常定义；

异常体系中将异常划分为系统异常和业务异常两大类。系统异常产生于系统框架运行时的错误，例如资源读取错误、与其它系统的通信异常及数据转换错误等；业务异常产生于业务逻辑处理过程中发生的错误；

异常发生后的日志打印统一由系统框架实现，异常错误信息的显示由开发人员根据具体需要进行处理；

采用国际化信息(i18n)支持，所有异常信息定义在消息文件中，通过特定的方法根据消息ID和参数组合错误信息。

注：参见《异常规范.docx》

### 底层处理机制

将在底层捕获到的异常转化为unchecked异常抛出。捕捉到的异常根据业务异常类型或系统类型，抛出对应的异常（参见com.suning.vgs.common.exception.hotel下的类说明），异常信息需要在message配置文件中配置:

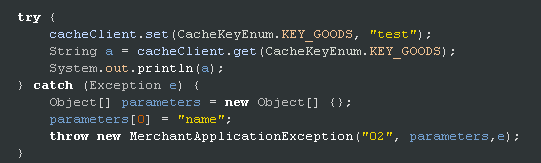
\vgs-b2g-comm\src\main\resources\i18n\messages.properties

\vgs-b2g-comm\src\main\resources\i18n\messages\_en.properties

\vgs-b2g-comm\src\main\resources\i18n\messages\_zh\_CN.properties

支持国际化和参数化。

使用范例：



其中“02“为message配置文件中的id

配置文件样式：

|  |
| --- |
| 01=发生系统错误，请联系客服。  02=您不能操作该订单，请联系客服。  ... |

### 页面请求异常处理方法

非AJAX提交：产生的异常由系统提供默认的异常处理：统一跳转到异常处理页面。

AJAX提交：开发人员需在ajax调用返回之前，捕捉到对应的异常，并转换成ajax发起端可处理的数据返回。

### 数据库访问异常处理方法

所有异常在数据访问层捕获并处理，记录相应日志后，抛出DalException异常，由调用捕获做处理。

### JOB/WebService请求异常处理方法

所有异常都应该在controller层捕获并处理，不再抛出。

## 事务

### 概述

采用Spring配置使用JTA进行事务管理，通过JAVAEE容器进行控制。

### 原则及使用方法

涉及到对多张表写操作的业务逻辑或一般的批量操作，相应的代码实现必须添加事务控制。

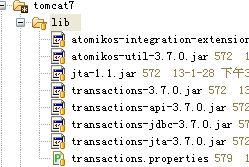
开发者在需要事务控制的方法添加注解：@Transactional即可。

***方法位于应用层***：该方法必须调用组合服务层或原子服务层来实现事务处理、无需@Transactional 注解

***方法位于数据服务层***：根据业务需求判断是否需要添加@Transactional注解；

### Tomcat中使用

在tomcat的安装目录中添加atomikos和JTA的JAR包和配置文件：



参见/vgs-b2g-pom/readme.txt