**新光人寿**

**程序开发规范及标准白皮书**

**V1.1**

Sinosoft

**中科软科技股份有限公司**

**2017年11月13日**

**关于本文件**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 主 题 | | 程序开发规范及标准白皮书 | | | | |
| 说　　明 | | 主要描述LIS7系统程序开发规范及标准的文档 | | | | |
| 适用对象 | | 基于LIS7框架基础进行项目开发的决策者和程序师 | | | | |
| 修 订 历 史 | | | | | | |
| 版 本 | 章 节 | | 类 型 | 日 期 | 作 者 | 说 明 |
| V0.1 |  | | C | 2017-09-10 | 赵雪峰 | 新增 |
| V0.2 |  | | U | 2017-09-15 | 徐云鹏 | 修订及定稿 |
| V1.0 |  | | U | 2017-09-27 | 徐云鹏 | 修订及定稿 |
| V1.1 |  | | U | 2017-11-10 | 韩雪 | 修订及定稿 |
|  |  | |  |  |  |  |

说明：类型－新增（C）、修改（U）、删除（D）；

**评审记录**

|  |  |  |  |
| --- | --- | --- | --- |
| **评 审 记 录** | | | |
| **角 色** | **签 名** | **日 期** | **说 明** |
| 新光工程师 | 纪智腾 | 2017-09-22 | 审查v0.2 |
| PM | 刘昊 | 2017-10-6 | 审查v1.0 |
| PM | 刘昊 | 2017-11-13 | 审查v1.1 |

**目 录**

[1前言 4](#_Toc498507729)

[1.1 文档说明 4](#_Toc498507730)

[1.2 文档结构 4](#_Toc498507731)

[2 项目程序结构说明 6](#_Toc498507732)

[2.1 项目结构修改 6](#_Toc498507733)

[2.2 本项目程序结构 7](#_Toc498507734)

[2.3 本地SQLJ的Jar包版本问题 8](#_Toc498507735)

[3 建档说明 10](#_Toc498507736)

[3.1 建档路径 10](#_Toc498507737)

[3.2 命名规则 11](#_Toc498507738)

[3.3 建档方法 11](#_Toc498507739)

[4 UI、BLF、BL三层结构说明 12](#_Toc498507740)

[4.1 UI 13](#_Toc498507741)

[4.1.1 功能概述 13](#_Toc498507742)

[4.1.2 实作 13](#_Toc498507743)

[4.1.3 完整UI档实例 14](#_Toc498507744)

[4.2 BLF 14](#_Toc498507745)

[4.2.1 功能概述 14](#_Toc498507746)

[4.2.2 实作 15](#_Toc498507747)

[4.2.3 完整BLF档实例 15](#_Toc498507748)

[4.3 BL 15](#_Toc498507749)

[4.3.1 功能概述 15](#_Toc498507750)

[4.3.2 实作 16](#_Toc498507751)

[4.3.3 完整BL档实例 17](#_Toc498507752)

[5 存储过程的调用 18](#_Toc498507753)

[5.1 Controller中的操作 18](#_Toc498507754)

[5.2 在UI中的逻辑 20](#_Toc498507755)

[5.3 在BLF中的逻辑 21](#_Toc498507756)

[5.4 在BL中的逻辑 22](#_Toc498507757)

[6 实体工具类别 23](#_Toc498507758)

[7 存储过程的工具类别 24](#_Toc498507759)

[SQLj相关档案、指令产生工具 24](#_Toc498507760)

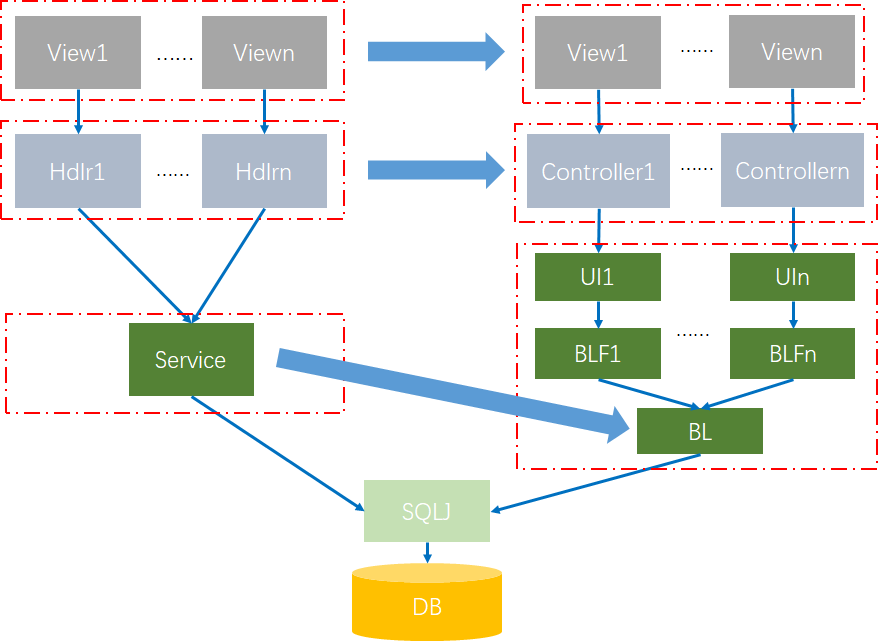
[8 Webservice 26](#_Toc498507761)

# 1前言

* 1. 文档说明

本文档编写的目的是爲了使开发人员，通过本文档可以顺利的进行开发工作。本文档着重介绍后台开发，介绍开发人员在使用LIS7架构开发的过程中，可能需要用到的方法。

如下图所示，本文档主要说明后台，即UI、BLF、BL的开发规范。其中，view和controller的部分爲前台。



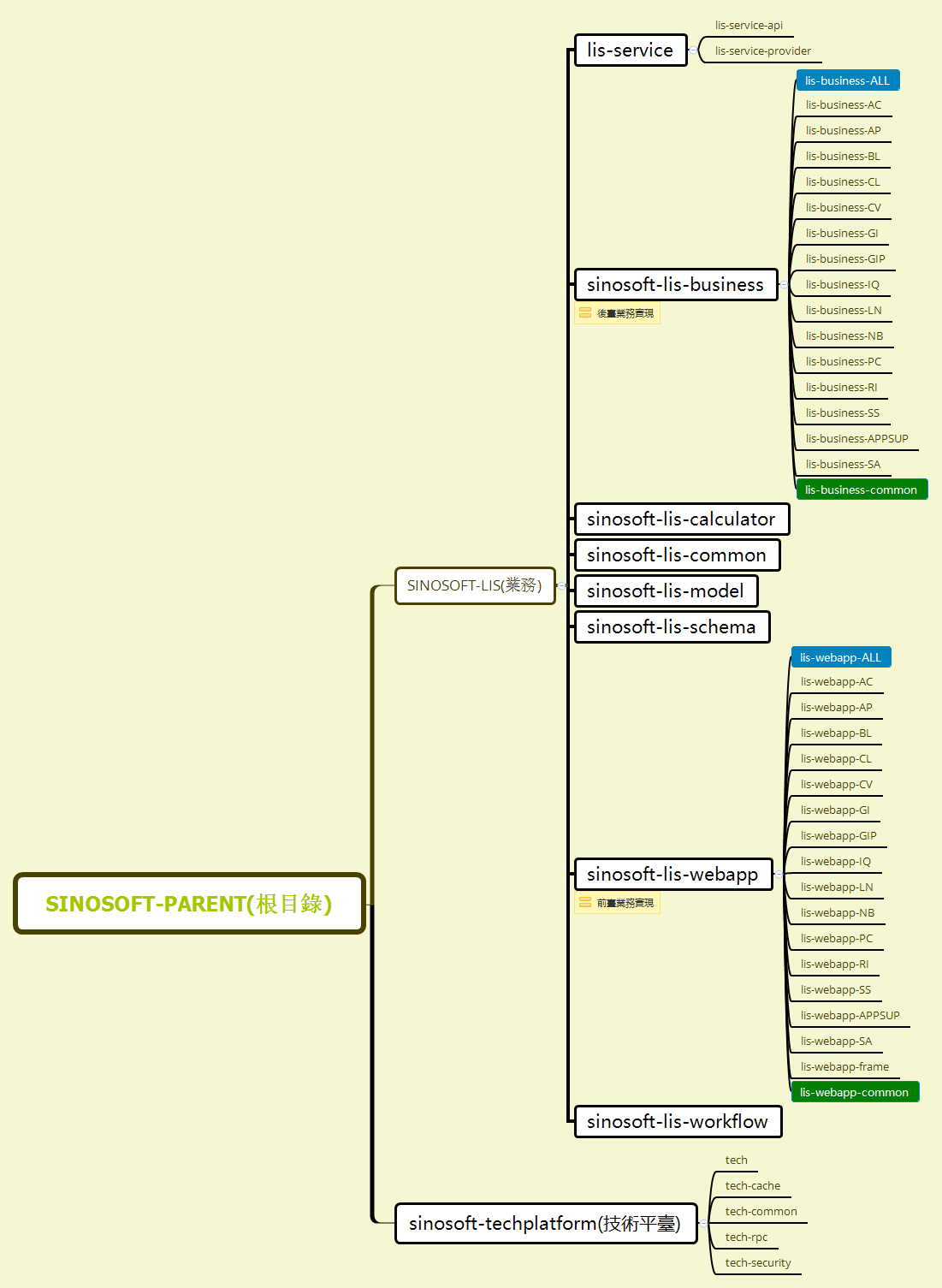
## 文档结构

* 项目程序结构说明
* 建档说明
* UI BLF BL三层结构使用
* 存储过程的调用
* 实体类别工具
* 存储过程生成工具
* Webservice配置说明

1. 项目程序结构说明
   1. 项目结构修改

针对本项目，LIS程序结构修改为13线后，项目结构为，每一线有对应的前后台项目，每一线的前台都依赖lis-webapp-common且都依赖在lis-webapp-ALL中；每一线的后台都依赖lis-business-common且都依赖在lis-business-ALL中。lis-webapp-ALL将lis-business-ALL依赖进去，达到前后台一起的目的。SA模块为系统管理是必备模块。Webapp-frame为前台公共的页面。

下图中，sinosoft-lis-business是后台UI、BLF、BL档放置的地方，sinosoft-lis-webapp是前台jsp，js，json，controller档放置的地方。



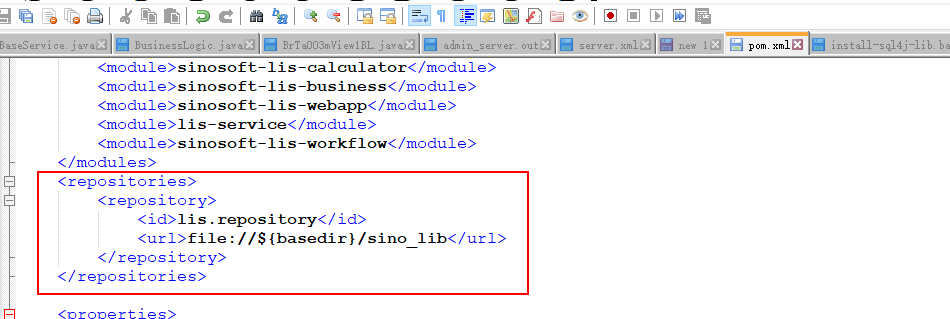
* 1. 本项目程序结构

本项目程序结构沿用LIS7原有程序结构不变。

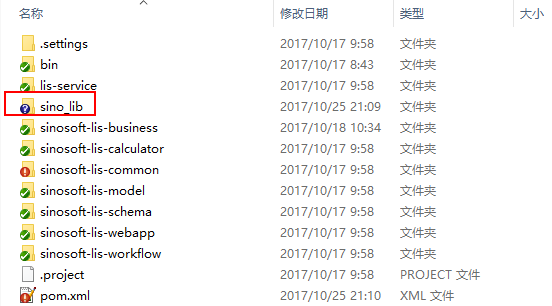
* 1. 本地SQLJ的Jar包版本问题

在以后的开发中，如果有开发人员修改存储过程重新生成jar包供自己开发时，不能使用nexus上的jar包版本。针对这种情况，采用本地库的方式管理个性化的sqlj的jar包。管理方法如下：

1. Pom档中增加指向程序目录的本地仓库



1. 程序目录中增加sino\_lib资料夹



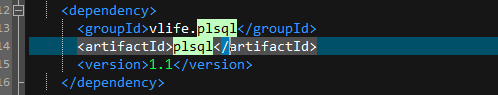
sino\_lib目录下，按照如下结构存放plsql-1.0-dev.jar。在1.0档加下放置jar

包。

即：sino\_lib\oracle\sqlj\sqlj\1.0-dev文件夹下。

1. 开发人员本地如果需要使用自己的jar包，在sino\_lib下，增加版本号，新建对应版本号的资料夹，并把jar包放里面，同时修改pom.xml的sqlj的版本号。即放置plsql-1.1-dev.jar到1.1-dev文件夹中

即：sino\_lib\oracle\sqlj\sqlj\1.1-dev



1. 建档说明

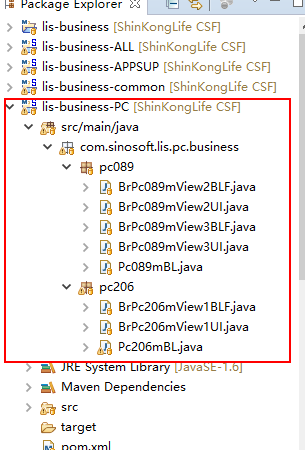
每个页面的都对应有后台的逻辑文档。LIS7中后台的逻辑文档有三种，分别是UI、BLF、BL。UI、BLF是保留的原有LIS7的功能，BLF提供访问BL的逻辑。而真正的逻辑功能都是在BL中进行实现的，在LIS7现行系统中，每个m对应同一个BL，并通过各个view不同的BLF进行指定访问。

具体逻辑在第三章介绍。

* 1. 建档路径

UI、BLF、BL都属于java档，需要写在LIS7的后台，即lis-business中。如lis-business-PC等package下。其路径如下图所示：

其中，com\sinosoft\lis是固定的，pc是某类M的简称，business是说明文件在后台，pc089是当前档所属M的分支。



* 1. 命名规则

UI、BLF、BL档有自己的命名规则。

其中，每个UI、BLF对应一个view页面，如3.1图中所示，BrPc089mView2.jsp页面（页面部分会在页面设计规范及标准白皮书.docx中说明）对应的UI和BLF分别是BrPc089mView2UI.java和BrPc089mView2BLF.java。

每个M对应一个BL，命名方式爲当前M的加上BL的形式，如上图中Pc089MBL.java。当前M的多个view对应一个BL。

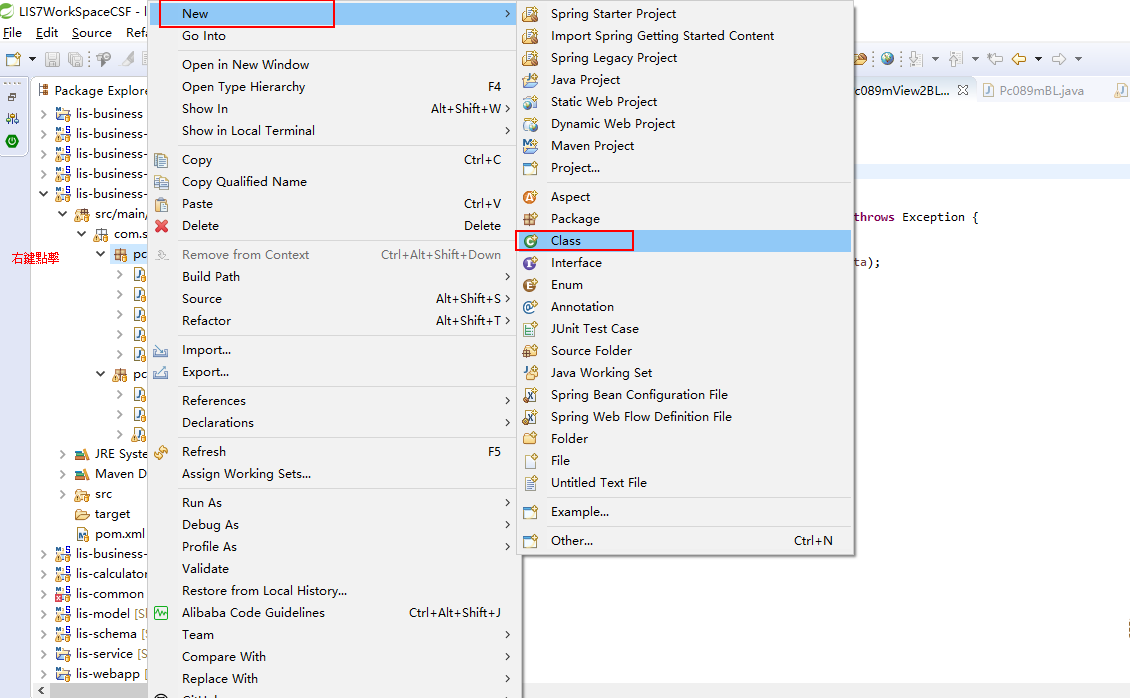
* 1. 建档方法

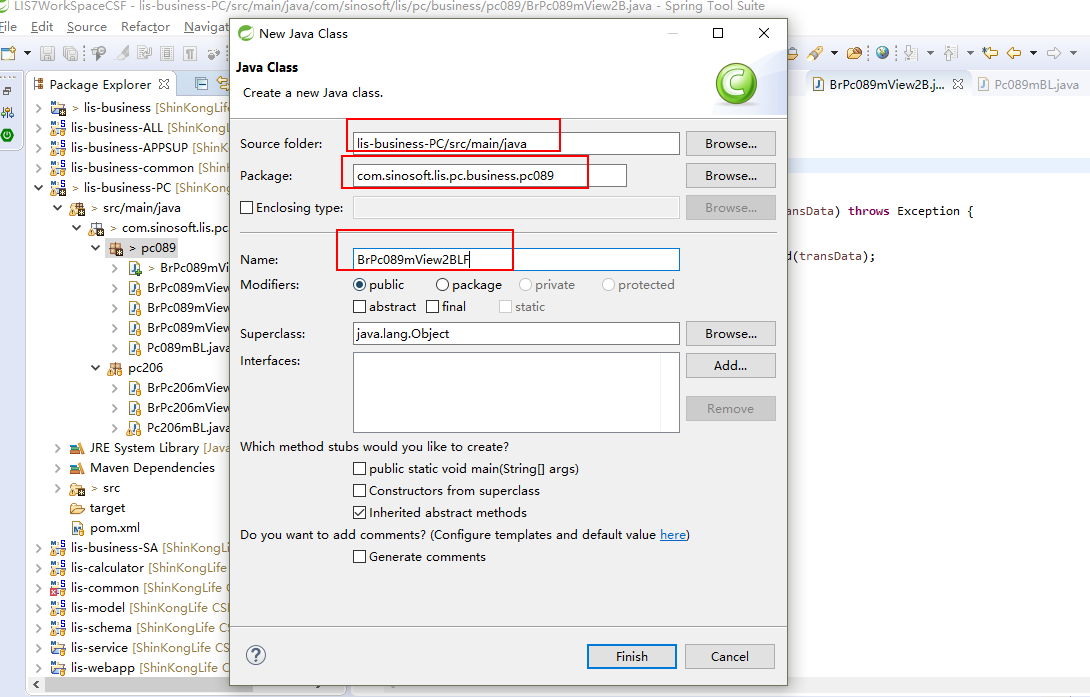
找到需要建立java档的具体路径，如，要在lis-business-PC包下建立新的java档，需要进行如下操作：

1. 要找到需要建立新档的合适的路径,如：

lis-business-PC\src\main\java\com\sinosoft\lis\pc\business\pc089\

在上边的路径上建立BrPc089mView2BLF.java的操作爲：

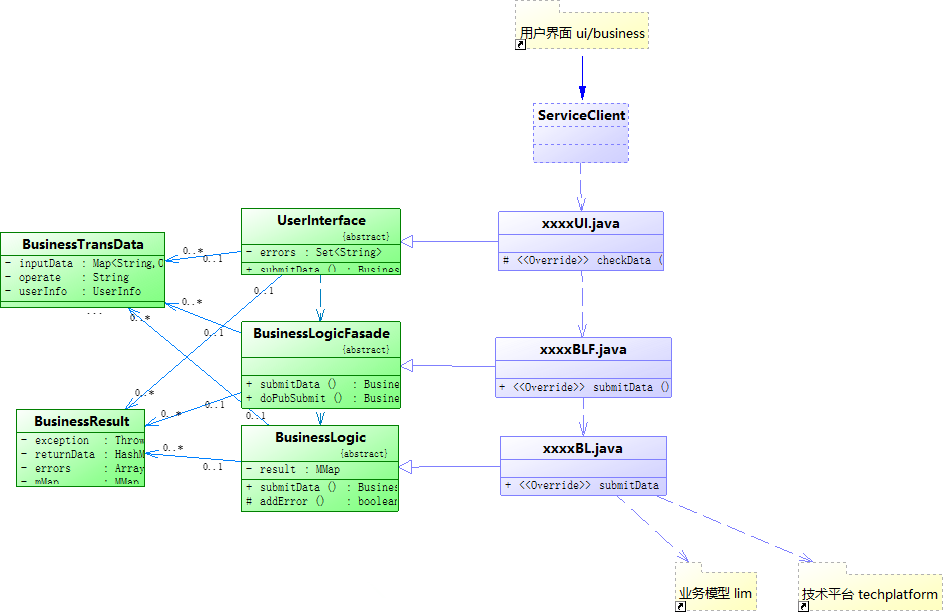




1. 建立的档是空白档，需要自行添加需要继承的父类别，并添加实现业务逻辑相关的代码。

# UI、BLF、BL三层结构说明

UI、BLF、BL的整体结构如下图所示：



下面将分别对这三层结构进行相应的说明，具体调用存储过程的使用方法和规范，参照第5章 存储过程的调用章节。

* 1. UI
     1. 功能概述

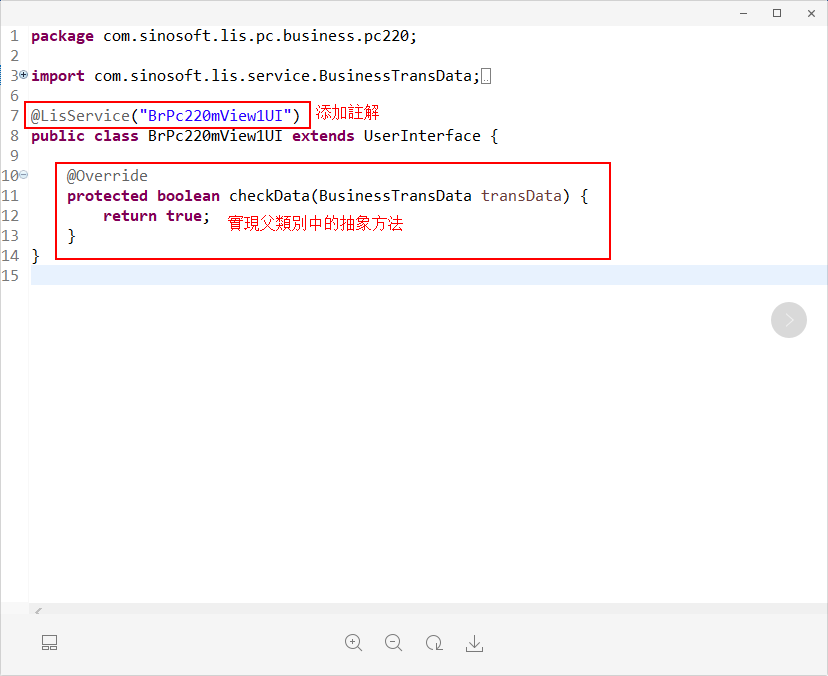
UI层为前台模块提供服务（LisService），前台模块使用CallService访问后台将数据传给UI，UI不进行任何操作，将数据传递给BLF，然后接收BLF的资料并返回给前台模块。

开发人员针对UI不需要做过多的事情，在UI的父类别中已经自动将数据传递到BLF，并接收返回的数据，返回到了前台。

* + 1. 实作

所有的UI都应该继承UserInterface（com.sinosoft.lis.service），并需要在UI上添加@ LisService标签，便签的名字，要跟当前类别的名字相同。作爲前台controller进行callService时的唯一识别标记。然后只需要实现父类别界面的checkData方法界面，直接return true即可。一个完整的UI如下。

* + 1. 完整UI档实例



* 1. BLF
     1. 功能概述

BLF层调用BL完成后续操作，在BL中执行完存储过程，会得到执行存储过程之后的结果。然后把执行结果返回到给BLF并在BLF的父类别中返回给UI，然后UI会自动将处理结果返回给前台controller。

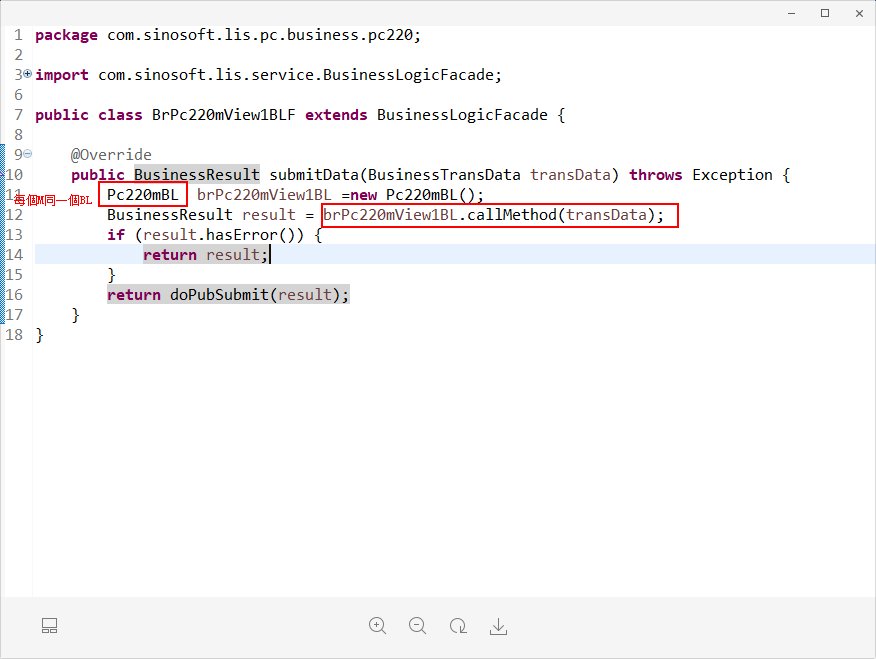
* + 1. 实作

所有BLF都应该继承BusinessLogicFacade。

在现行系统中，BLF负责进行各个view对相应BL的调用，并执行BL中对存储过程的呼叫。

对存储过程的呼叫逻辑，在第5章中进行说明。

* + 1. 完整BLF档实例



* 1. BL
     1. 功能概述

BL主要是对存储过程进行调用，LIS7采用sqlj的方式对存储过程进行调用，并会将结果返回到前台，供后续逻辑使用。

BL的父类别会根据controller中要执行方法的名字标识，自动去BL中找到相应的方法，执行方法中调用的存储过程，并将存储过程的返回值和出参传给BLF，进而返回给台前controller（具体实现方法，请在第5章查阅）。

* + 1. 实作

所有BL都应该继承BusinessLogic。

在BL中首先要把一个DefaultContext进行实例化，即使用：

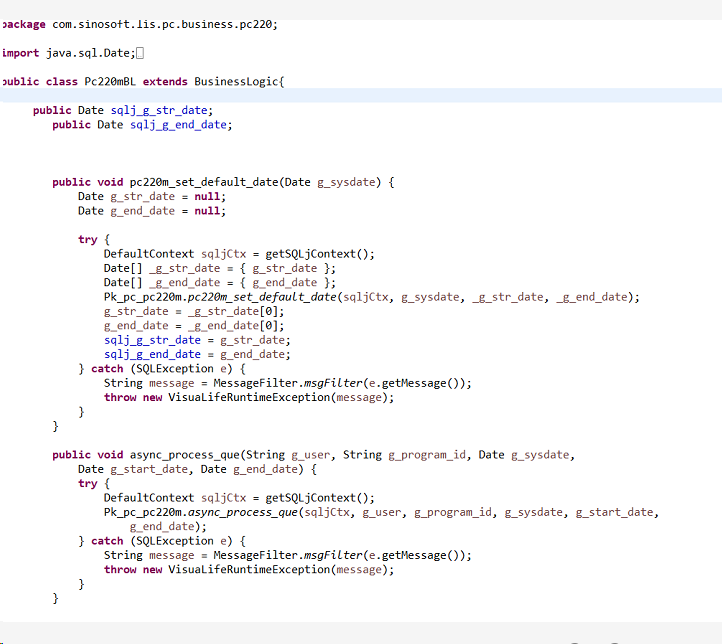
DefaultContext sqljCtx = getSQLjContext();

然后，调用执行sqlj中的存储过程，并把出参放到全域的public变量中。

即如下图中的：

sqlj\_g\_str\_date = g\_str\_date;

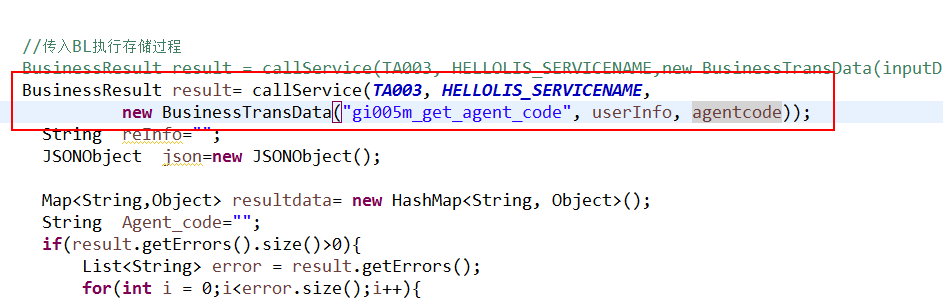
* + 1. 完整BL档实例



1. 存储过程的调用
   1. Controller中的操作

在准备完页面的数据之后，要将页面上的数据传递到controller中，（controller在页面设计规范及标准白皮书中有详细介绍），并在controller进行相应的封装，具体是说有的值需要放入到type实体类别中。

备准好数据之后，执行callService的方法，如下：



callService()方法有3个参数，第一个参数，是所属模块，即TA003；第二个参数，即后台UI即service的名称，如上图：HELLOLIS\_SERVICENAME；第三个参数是BusinessTransData类别，内部封装了两个固定参数，和多个不固定参数：两个固定参数分别是要执行的BL中的方法和UserInfo信息。多个不固定参数是要执行存储过程的参数。

BusinessResult类别是后台逻辑执行后，传回来的返回值。

返回值是一个map，

Map<String,Object> map =result.getReturnData();

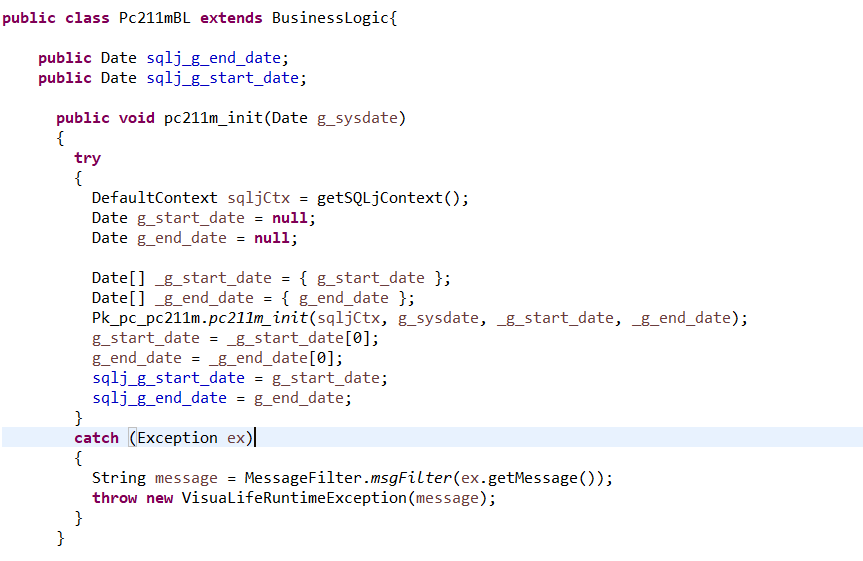
如果调用的存储过程，有返回值，则使用以下方法接取返回值。

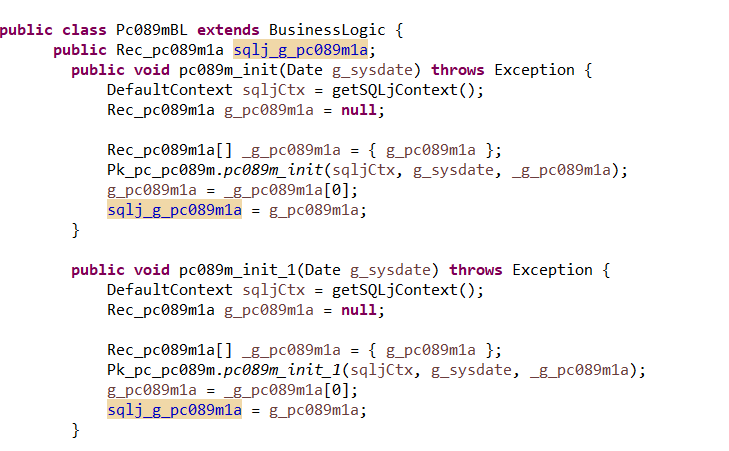
String returnData = (String)map.get(“returnData”);

如果要获取出参,则方法如下：

|  |
| --- |
| Date StrDate=(Date) result.getReturnData().get("sqlj\_g\_start\_date");  Rec\_pc089m1a pc089m1a  =(Rec\_pc089m1a) result.getReturnData().get("sqlj\_g\_pc089m1a"); |

其中，sqlj\_g\_start\_date和sqlj\_g\_pc089m1a爲在BL中定义的全域变量的名字。如：



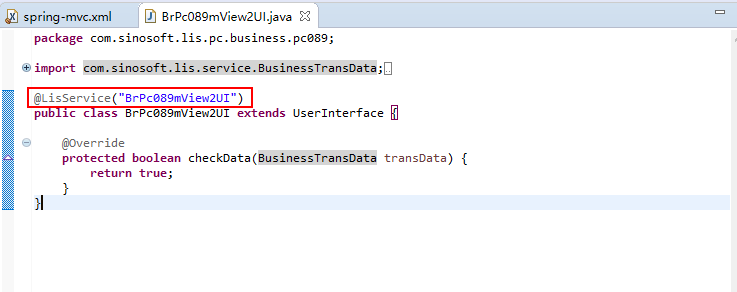


上图中，gi005m\_get\_agent\_code是在BL类别中的一个方法，方法中封装了对存储过程的调用，如下所示：（完整BL代码见上文）

|  |
| --- |
| *public String gi005m\_get\_agent\_code(String g\_agent\_code) {*  *try {*  *DefaultContext sqljCtx = getSQLjContext();*  *String result = Pk\_gi\_gi005ms1.gi005m\_get\_agent\_code(sqljCtx,*  *g\_agent\_code);*  *return result;*  *} catch (Exception ex) {*  *String message = MessageFilter.msgFilter(ex.getMessage());*  *throw new VisuaLifeRuntimeException(message);*  *}*  *}* |

* 1. 在UI中的逻辑

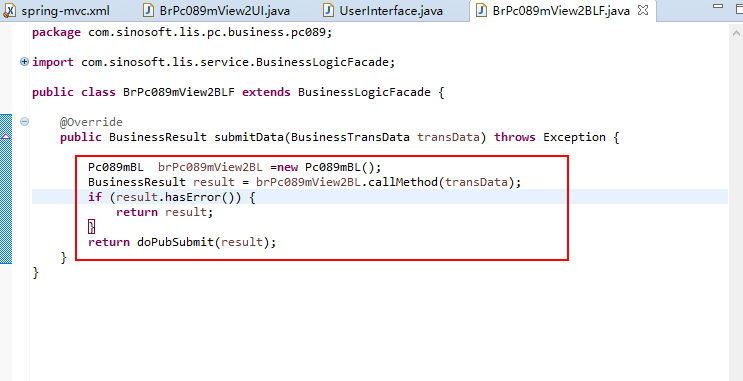
UI是前台调用后台的入口，根据controller中要访问的service的名称，去访问相对应的UI。在UI中有@LisService的注解对应controller中请求的UI的名称。



UI继承了UserInterface类别，在本类别中通过submit方法对BLF进行了调用，此处只需要实现如上图的checkData()方法即可。

* 1. 在BLF中的逻辑

BLF中执行调用BL的callMethod方法，会根据controller传过来的参数值自动执行相应的方法，BL中方法执行完毕后，会返回BusinessResult类别的执行结果。返回到UI，从UI返回到前台。



* 1. 在BL中的逻辑

BL的父类别实作了callMethod的方法，BLF调用的方法即爲此方法，代码如下所示：



BL中的代码在上文中已经贴出。

这样，就完成了一次存储过程的调用。

1. 实体工具类别

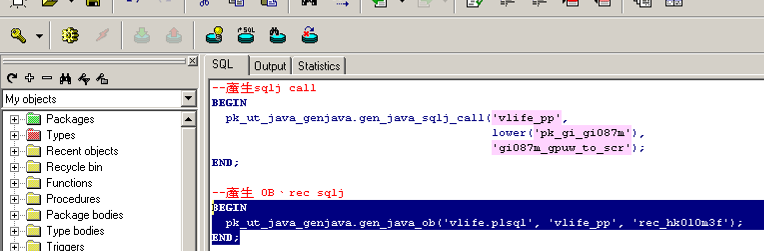
调用数据库存储过程时会用到自订的数据库类别，LIS7使用UIX原有系统的rec类型来调用存储过程，实体类别的生成方法见ADF教学文档第一章，LIS7只用到了其中set值和get值的方法，获得VO的方法没有使用。是以jar档的形式引用到系统中。具体产生方法，第7章有说明。

1. 存储过程的工具类别

调用存储过程使用存储过程的工具类别，也是已jar档的形式引入到LIS7的系统中，具体生成sqlj的方法，见ADF教学文档。引用如下：

SQLj相关档案、指令产生工具

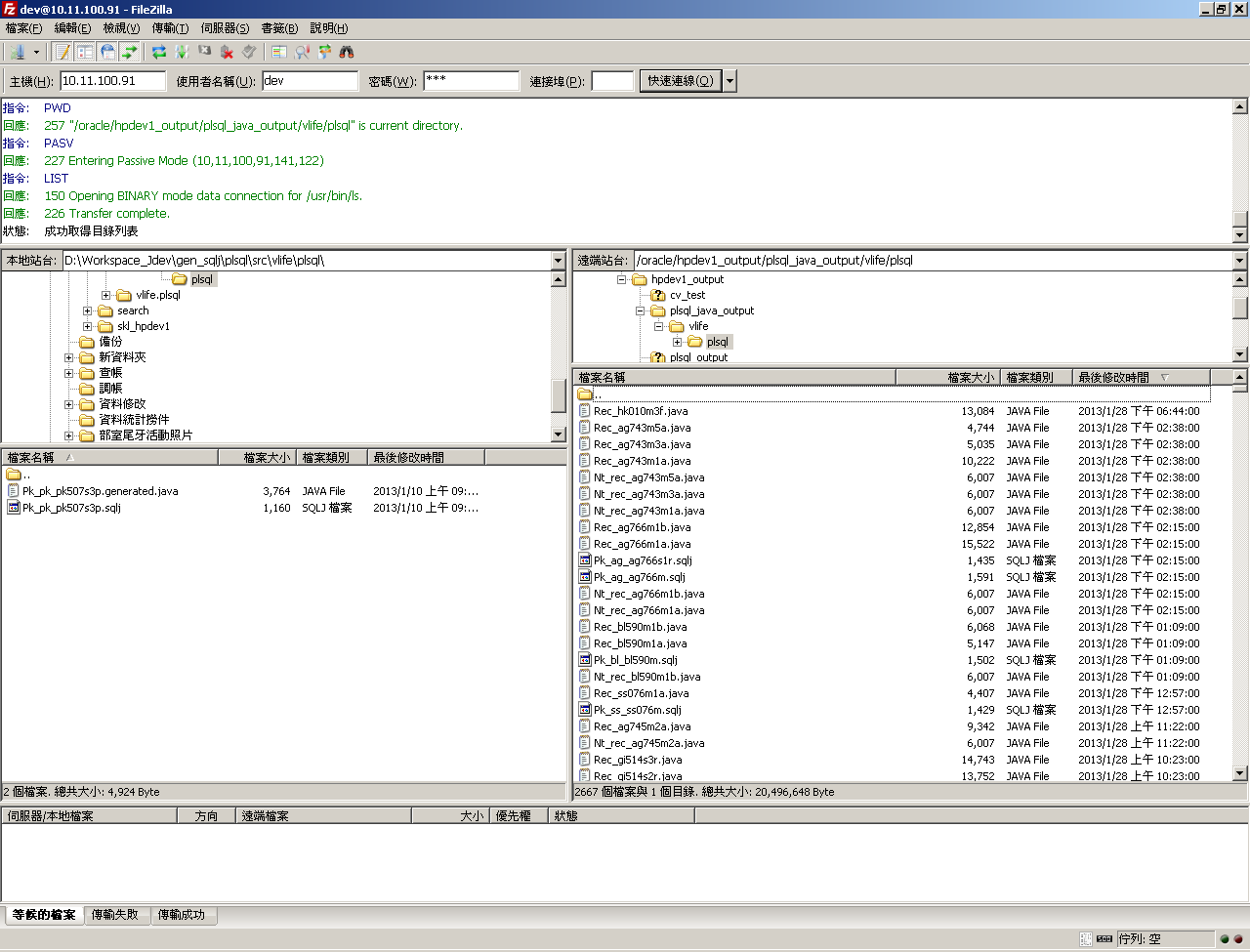
* 工具名称：gen\_sqlj.sql
* 另外需要安装的工具：FileZilla FTP
* 使用方法：于开发环境的PL/SQL Developer打开档案，找出符合需要的指令区段，将名称改好后选取执行，档案结果会放到DB后端的plsql\_java\_output资料夹底下。例如：欲产生vlife\_pp底下的rec\_hk010m3f.java档



**1**

**2**

打开FileZilla连线到开发环境的DB后端(主机：10.11.100.91，账号、密码：dev)，并选择到plsql\_java\_output/vlife/plsql此路径底下



**3**

**2**

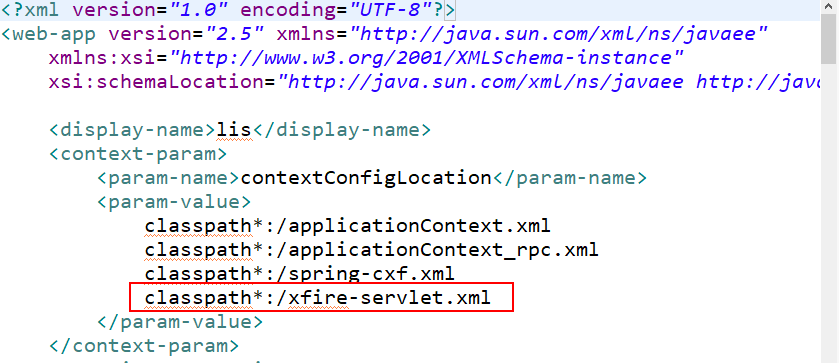
**1**

* 后续请看**SKL- Std-DEV Advance Skill.****doc里的How to 自行产生plsql.jar里面没有的class**

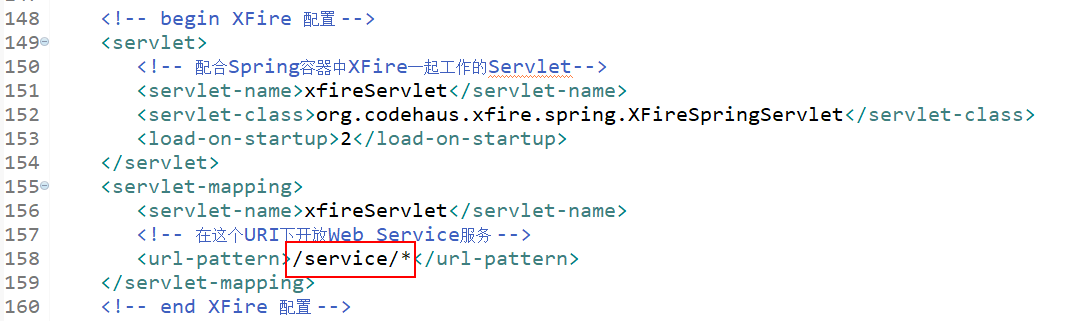
1. Webservice

在web.xml(lis-webapp-frame/src/main/webapp/WEB-INF/)中,添加:

classpath\*:/xfire-servlet.xml



增加webservice的servlet配置：



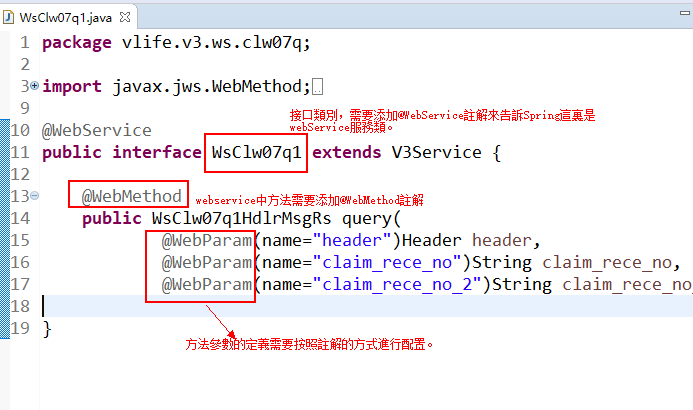
然后是webservice的设定档的相关设定：



当实作一个service时，只需添加相应的注解即可。如下图所示：

在lis-webapp-common\src\main\java\vlife\v3\ws\档路径下，添加需要被xfire注册的webService服务类别：

首先需要创建界面类别：



实作类别：



<建表sql > 文件和 <template.sql >文件