cats 后端代码修改规范

1. 如果使用该规范
   1. 写代码时参考该规范
   2. 写完一个方法使用该规范自己review下
   3. Review代码时参考该规范
2. 注释
   1. 每个类开始添加作者相关信息
   2. 注释也要拷贝过来
3. 日志
   1. 进入方法之后打印DEBUG级别的日志，日志内容包括方法名、参数名、参数值
   2. Error级别日志使用格式log.error(“error message”, e)
4. 类型对应
   1. 使用相同的类型对应(此处添加一个list)
5. 类名修改规则：
   1. vb中以名词命名的类修改之后的java类名为名词+BO
   2. vb中以名词命名的类若有较多的属性字段，则这些属性单独提取一个VO类
   3. vb中以DAL为后缀名的类修改之后的java类名为名词+DAO
6. 方法修改规则：
   1. 方法名首字母改为小写
   2. 方法参数中byVal和byRef的传值类型对应修改为：byVal为值传递，byRef为传对象的引用。注意：vb默认使用的是值传递，java默认的是引用传递，因此在改成java后，如果期待的行为是值传递的话，在方法开始要复制该对象，然后使用对象的副本。
   3. vb方法中的shared对应java方法中的static
7. 变量修改规则

修改之后java中变量名称和类型与vb中保持一致

1. 新添加的公共工具类

不要自己随意添加公共工具类，先看是否现有的工具类是否已经有了该方法，如果没有是否可以加入到现有公共类中，否则新建公共类与大家讨论过，一起确定类名

1. 数据库操作
   1. 参数不需要@
   2. 重写类似 .NET平台中的DataSet结构的类，将ResultSet封装到此DataSet中
   3. 所有方法需要catch异常并对其按照异常章节处理
   4. 方法结束后要关闭连接释放相关资源
2. 异常

在进行数据库操作的时候，对以检查异常进行try catch，并在catch中先记录日志，INFO级别的日志内容为 方法名、参数名、参数值和调用的SQL语句，ERROR级别的日志内容为所捕获的异常信息，以及调用的SQL语句，并抛出自定义的RuntimeException（CATException）,异常只在DAO层抛出，BO层不进行捕捉也不进行重新抛出。

1. 提交代码过程
   1. Update最新代码
   2. 所有单元测试跑过
   3. 提交代码
2. JAR包

源代码中（非测试代码）要使用某个Jar时，要提供其license和下载url，使用前请

**CVS上的代码要保持无编译错误，所有单元测试都通过，checkout后就可运行**

Sample:

**public** **static** **boolean** DeleteActionAccess(String pUserID, String pActionTypeID)

{

*logger*.debug(CATSConstants.*ENT\_FUN* + **new** Exception().getStackTrace()[0].getMethodName());

String paras = "pUserID: " + pUserID + ",pActionTypeID: " + pActionTypeID;

*logger*.debug(paras);

String vSQL="";

**try**

{

vSQL = StringUtil.*format*("DELETE FROM CATSActionAccess WHERE UserID = '{0}' AND ActionTypeID = '{1}'", pUserID,

pActionTypeID);

SqlDBAccess.*execCommand*(vSQL);

**return** **true**;

}

**catch** (Exception err)

{

*logger*.info(paras);

*logger*.info(vSQL);

*logger*.error(err.getMessage(), err);

**throw** **new** CATException(err.getMessage());

}

}