日志记录经验总结

程序的报错机制可以使程序更加健壮，传统面向过程编程中通过返回值来判断是否成功，即返回true或false进行判断，或者通过返回状态码来进行标识程序状态；而java面向对象编程则对程序的异常进行对象化，封装信息、原因等，并通过try catch 进行捕捉处理。

另外在一些rpc调用（http/api开放接口）可通过返回状态码的方式来判错

在java项目中建议抛出异常，分层捕捉集中处理，并针对不同异常类型进行分别处理：如记录日志。

|  |
| --- |
| Try  {  ....QueryFile( "C:\abc.txt" );  }  catch(SQLException e ) {  ....//代码进入这里说明发生【sql错误】  }  catch(ArithmeticException e ) {  ....//代码进入这里说明发生【算术错误】  } |

1. 记录技术性异常而不记录用户异常

用户异常如传入参数有错误，用户名已存在等称为用户异常，

技术性异常如 文件读取错误等

如果记录用户异常可能会因为日志实体太长，而忽略有意义的日志信息

1. 捕捉后记录日志不要在抛出新的异常
2. 输出导致异常的原因