Atitit 异常处理 try catch 模式 vs exit handler模式

Try 相当于begin end

Catch相当于exit handler，，，不在执行出错航下边的语句。。

go语言为了更加简洁优雅，没有类似java的try...catch...这种写法，因为try...catch在某些情况下会嵌套使用，造成代码运行后不知道哪里就跑到了异常处理的代码里。

但是也有相应的异常处理机制。

需要记住的关键词有个，error，defer，panic，recover。

## 2 defer defer相当于java中的finally

defer相当于java中的finally，c中的析构函数。不管有没有发生异常的情况下，defer定义的函数都会被执行。一般用来进行资源回收等操作，防止程序员忘记这些操作。

defer语句的用法有两个优点：

1.让设计者永远也不会忘记关闭文件，有时当函数返回时常常忘记释放打开的资源变量。

2.将关闭和打开靠在一起，程序的意图变得清晰很多。

## 3 panic-recover运行时异常处理机制

Go语言中没有Java中那种try-catch-finally结构化异常处理机制，而使用panic()函数引发错误（等同于throw/raise），然后在defer语句中调用recover()函数捕获错误，这就是Go语言的异常恢复机制——panic-recover机制。

## **总结：**

1. go中error相当于java中的Exception，defer相当于java中的finally，panic相当于java中的throw，panic和recover一起使用相当于java中的try...catch...。

发布于 2019-05-20