JNI 攻略之九－－操作异常

江苏 无锡 缪小东

本篇主要介绍如何在 JNI 处理 java 异常，这种异常可以 java 内部方法的，也可以是 JNI 中自己创建并且抛出的！同时还复习了 JNI 中操作 java 对象的方法！

一、Java 类及两个辅助类

```
//下面是一个包含本地代码的 java 类！该本地方法调用 java 方法，同时可能抛出异常。
public class ExceptionAccess{
    private static native void doThrowException(int age)throws AgeOutofBoundsException;
            //本地方法，调用 Student 类的 setAge 方法，抛出异常
    public static void main(String[] args){
        try   {
            System.out.println("We are going to set the age ! age =    20");
            ExceptionAccess.doThrowException(20);                    //正常处理
            System.out.println();
            System.out.println();
            System.out.println("We are going to set the age ! age = 101"); //抛出异常
            ExceptionAccess.doThrowException(101);
        }catch(AgeOutofBoundsException e ){
            System.out.println("In Java:\n\t" + e);
        }
    }

    static {                                                    //不用说了吧！
        System.loadLibrary("ExceptionAccess");
    }
}

//以下是一个简单的辅助类！包含一个抛出异常的方法！
class Student{
    private int age ;
    public void setAge(int age)throws AgeOutofBoundsException{
        if(age > 100|| age<0) throw new AgeOutofBoundsException("\nAge is out of bound!Please
check your age !\n");
        this.age = age ;

    }
    public int getAge(){
        return age ;
```

```
    }
}

//简单的异常类！继承父异常 Exception！
class AgeOutofBoundsException extends Exception{
    AgeOutofBoundsException(){
        super();
    }
    AgeOutofBoundsException(String reason){
        super(reason);
    }
}
```

二、本地代码实现

```
//以下是 JNI 中本地方法的实现

#include <jni.h>

JNIEXPORT void JNICALL Java_ExceptionAccess_doThrowException(JNIEnv *env, jclass cls , jint age ){
    jclass stucls ;                                 //Student 类的类
    jmethodID cmid ,setmid ,getmid ;                //Student 类的构造器方法、set/get 方法
    jobject stuobj ;                                //Student 类的一个对象
    jthrowable exc;                                 //JNI 中的异常——本篇的重点哦！

    jint result ;                                   //简单的返回结果

    stucls = (*env)->FindClass(env, "Student");     //得到 Student 类的类
    if (stucls == NULL) {       return ;    }
    cmid = (*env)->GetMethodID(env, stucls,"<init>", "()V");    //构造 Student 类
                    //在 JNI 中，构造器其实就是一个名称为"<init>"的方法，返回值为 void
    if (cmid == NULL) {     return ;    }
    stuobj = (*env)->NewObject(env, stucls, cmid, NULL);        //创建该 Student 类的实例

    setmid=(*env)->GetMethodID(env, stucls, "setAge", "(I)V");  //得到 Student 类的 set 方法
    if (setmid == NULL) {   return;     }
    (*env)->CallVoidMethod(env, stuobj, setmid,age);
        //调用 Student 类的 set 方法，输入为本地方法中的参数 age 哦！仔细看清楚了！
    exc = (*env)->ExceptionOccurred(env);               //从 env 中得到是否发生异常
    if (exc) {                                      //异常发生，则……
        jclass newExcCls;                           //在 JNI 中创建一个异常对象
        (*env)->ExceptionDescribe(env);             //得到异常的描述
        (*env)->ExceptionClear(env);                //清除异常
```

```
        newExcCls = (*env)->FindClass(env,"AgeOutofBoundsException"); //建立一个对于的异常
        if (newExcCls == NULL) {return;}
        (*env)->ThrowNew(env, newExcCls, "Thrown from C code!\n Age is out of Bound !\n");
                                                //在 JNI 中抛出异常！ 异常到此结束！
    }

    getmid=(*env)->GetMethodID(env, stucls, "getAge", "()I");       //得 getAge 方法
    if (getmid == NULL) {    return;    }
    result = (*env)->CallIntMethod(env, stuobj, getmid);            //调用 getAge 方法
    printf("We are going to set the age ! age =    %d\n",result);        //在本地方法中输出结果

}
```

结束了！就这么容易！下篇可能比较复杂哦！待续！

更多精彩请关注：

http://blog.163.com/miaoxiaodong78/