

异常处理

Java语言基础

CTRL+ALT+T选中需要被包裹的位置

在程序执行过程中发生的不正常情况称为异常，开发过程中的语法错误和逻辑错误不是异常
执行过程中发生的所有异常事件可以分为两类：

- Error错误，Java虚拟机无法解决的严重问题，例如JVM系统内部的错误，资源耗尽等严重情况，Error是严重错误，程序会崩溃
- Exception：其他因编程错误或者偶然的外在因素导致的一般性问题，可以使用针对性的代码进行处理，例如空指针访问，读取不存在的文件等，此外Exception分为两类，运行时异常和编译时异常
常见运行时异常：
 - 空指针异常NullPointerException
 - 算数异常ArithmeticException
 - 数组下标越界异常ArrayIndexOutOfBoundsException\
 - 类型转换异常ClassCastException
 - 字符串转换异常NumberFormatException

throws将异常抛出，交给调用者来处理

异常处理try-catch

try-catch-finally，程序员在代码中捕获发生的异常

```
try{可能存在异常的代码}  
catch(Exception e){捕获异常进行处理，系统将异常封装成Exception对象e，传递给catch}  
finally{不管是否捕获异常始终执行}
```

- 如果发生了异常，异常后面的代码不会执行，直接进入catch块
- 如果异常没有发生则程序执行try的代码块，不会进入到catch
- 如果不管如何都执行某段代码 则放入finally中
- 如果希望对多个异常进行处理则可以并列使用catch分别捕获对应异常，要求子类异常写在前面父类异常写在后面
- 如果try-catch中提前返回值，finally语句仍然会继续执行然后回头执行返回语句

异常处理throws

如果一个方法可能生成某种异常，但是并不能确定如何处理这些异常，则此方法应显式声明抛出异常，表明该方法将不对这些异常进行处理，而由方法的调用者处理

在方法声明中用throws语句可以声明抛出异常的列表，throws后面的异常类型可以是方法中产

生的异常类型，也可以是它的父类。也可以是异常列表

对于编译异常程序必须处理

对于运行异常程序不处理则默认throws处理

子类重写父类方法的时候，要么子类抛出的异常与父类的一致，要么是父类异常的子类异常

如果其他方法调用了一个抛出编译异常的方法则该方法也会抛出编译异常，调用抛出运行异常的方法不会抛出异常

自定义异常

定义一个类，并且继承Exception或者RuntimeException

如果继承Exception则属于编译异常，继承RuntimeException则属于运行异常

一般情况下自定义异常继承RuntimeException即运行异常，好处是可以使用自定义的处理机制

throws VS throw

	意义	位置	后面跟的东西
throws	异常处理的一种方式	方法声明处	异常类型
throw	手动生成异常对象的关键字	方法体	异常对象