Glimmertask7-异常处理

Exception

Name	Descripton
NullPointerException	尝试访问或操作一个 null 对象的成员或方法
IndexOutOfBoundsException	访问数组时索引超出了数组的有效范围
ArithmeticException	算术运算中的错误，如分母为0
NumberFormatException	尝试将一个不能解析为数字的字符串转换为数字
IOException	在进行输入输出操作时发生了错误
ClassNotFoundException	加载的类不存在或未找到
SQLException	在执行数据库操作时发生错误
FileNotFoundException	未找到文件

Error

Name	Description
OutOfMemoryError	JVM没有足够的内存去为对象分配空间
StackOverflowError	程序递归调用太深，导致栈溢出
NoSuchMethodError	调用不存在的方法
NoClassDefFoundError	定义类找不到
AbstractMethodError	调用一个抽象方法(抽象方法无法被调用)

当发生Exception和Error时，程序的处理态度分别应该是什么？

Exception

- Exception是相对可控的异常，程序自身可以通过try-catch语句对异常进行捕获和处理。

Error

- Error表示严重的程序或系统问题，这些问题不是程序自身可以解决的，这些问题应该由开发者尽力避免

cheked和uncheked异常

Checked Exception（受检异常）

受检异常必须由try-catch语句来捕获的异常，如果不捕获编译器会报错

Unchecked Exception（非受检异常）

非受检异常是由程序内在逻辑错误导致的，编译器不会强制将这些异常抛出

Checked Exception	Unchecked Exception
FileNotFoundException	IndexOutOfBoundsException
SQLException	NullPointerException
IOException	ArithmeticException

模拟银行取款

```
class InsufficientFundsException extends Exception {
    public InsufficientFundsException(String message) {
        super(message);
    }
}

class BankAccount {
    private double balance;

    public BankAccount(double initialBalance) {
        this.balance = initialBalance;
    }

    public double getBalance() {
        return balance;
    }

    public void withdraw(double amount) throws InsufficientFundsException {
        if (amount > balance) {
            throw new InsufficientFundsException("余额不足，无法取款。当前余额: " +
balance);
        }
        balance -= amount;
    }
}

public class BankAccountExample {
    public static void main(String[] args) {
        BankAccount account = new BankAccount(Math.random()*200);

        try {
            System.out.println("当前余额: " + account.getBalance());
            account.withdraw(150.0);
            System.out.println("取款成功。");
        } catch (InsufficientFundsException e) {
            System.err.println("错误: " + e.getMessage());
        }

        System.out.println("程序结束");
```

```
    }
}
```

所有可能流程

1. 正常取款：

- 先创建一个BankAccount对象,随机生成一个0~200的数
- 打印余额
- 尝试取出150元
- 成功取出,打印“取款成功”
- 打印“程序结束”

2. 余额不足：

- 先创建一个BankAccount对象,随机生成一个0~200的数
- 打印余额
- 尝试取出150元
- 失败，抛出InsufficientFundsException异常，进入catch语句
- 捕获异常，打印“错误: 余额不足，无法取款。当前余额: ……”
- 打印“程序结束”