【案例8-3】 模拟银行存取钱功能

### 【案例介绍】

**1.任务描述**

在银行办理业务时，通常银行会开多个窗口，客户排队等候，窗口办理完业务，会呼叫下一个用户办理业务。本案例要求编写一个程序模拟银行存取钱业务办理。假如有两个用户在存取钱，两个用户分别操作各自的账户，并在控制台打印存取钱的数量以及账户的余额。

**2.运行结果**

运行结果如图8-1所示。

电脑屏幕的照片

描述已自动生成

图8-1 运行结果

### 【案例目标】

* 学会分析”模拟银行存取钱功能”任务实现的逻辑思路。
* 能够独立完成”模拟银行存取钱功能”程序的源代码编写、编译以及运行。
* 通过存取款线程理解多线程安全问题的发生原因，并掌握如何解决多线程安全问题。

### 【案例思路】

1. 通过任务描述和运行结果可以看出，该任务需要使用多线程的相关知识来是实现。由于两个用户操作各自的账户，因此我们需要创建两个线程完成每个用户的操作。这里我们使用实现Runnable接口的方法来创建线程。
2. 既然是储户去银行存款，那么可以得出该任务会涉及到三个类，分别是银行类、储户类和测试类。
3. 定义一个实体类作为账务的集合，包括用户名、登录名、登录密码、钱包、取钱时间和存钱时间等字段。
4. 在银行类中需要定义一个账户的实体类、一个存钱的方法、一个取钱的方法、查询余额的方法和获取当前用户的方法。获取等前用户方法需要使用synchronized线程锁判断是是哪一位用户，在存钱和取钱的方法中先调用获取用户方法判断操作者，再进行存取钱操作，需要注意的是在进行取钱操作是，需要判断余额是否大于需要取的钱数。
5. 在测试类中使用for循环调用线程模拟用户存取钱操作。

### 【案例代码】

1. 创建用户类

定义一个用户的类，根据用户实现多人同时存取钱功能，如文件8-1所示。

文件8-1 User.java

1. package chapter0403;

import java.util.Date;

public class User {

private String u\_name;//用户名

private String u\_login\_name;//登录名 卡的id

private String u\_login\_pwd;//登录密码

private String u\_wallet;//钱包

private Date draw\_money\_time;//取钱时间

private Date save\_money\_time;//存钱时间

public User(){}

public User(String u\_name, String u\_login\_name, String u\_login\_pwd,

String u\_wallet) {

this.u\_name = u\_name;

this.u\_login\_name = u\_login\_name;

this.u\_login\_pwd = u\_login\_pwd;

this.u\_wallet = u\_wallet;

}

public User(String u\_name, String u\_login\_name, String u\_login\_pwd,

String u\_wallet, Date draw\_money\_time, Date save\_money\_time) {

this.u\_name = u\_name;

this.u\_login\_name = u\_login\_name;

this.u\_login\_pwd = u\_login\_pwd;

this.u\_wallet = u\_wallet;

this.draw\_money\_time = draw\_money\_time;

this.save\_money\_time = save\_money\_time;

}

public String getU\_name() {

return u\_name;

}

public void setU\_name(String u\_name) {

this.u\_name = u\_name;

}

public String getU\_login\_name() {

return u\_login\_name;

}

public void setU\_login\_name(String u\_login\_name) {

this.u\_login\_name = u\_login\_name;

}

public String getU\_login\_pwd() {

return u\_login\_pwd;

}

public void setU\_login\_pwd(String u\_login\_pwd) {

this.u\_login\_pwd = u\_login\_pwd;

}

public String getU\_wallet() {

return u\_wallet;

}

public void setU\_wallet(String u\_wallet) {

this.u\_wallet = u\_wallet;

}

public Date getDraw\_money\_time() {

return draw\_money\_time;

}

public void setDraw\_money\_time(Date draw\_money\_time) {

this.draw\_money\_time = draw\_money\_time;

}

public Date getSave\_money\_time() {

return save\_money\_time;

}

public void setSave\_money\_time(Date save\_money\_time) {

this.save\_money\_time = save\_money\_time;

}

}

1. 创建银行业务类

定义一个业务类，实现用户的存取钱功能，如文件8-2所示。

文件8-2 Bank.java

1. package chapter0403;

import java.math.BigDecimal;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

public class Bank {

private List<User> userList=new ArrayList<>();

public Bank(List<User> userList) {

this.userList = userList;

}

public List<User> getUserList() {

return userList;

}

public void setUserList(List<User> userList) {

this.userList = userList;

}

//存钱

public Boolean saveMoney(String card,String pwd,String moneyNum){

User u=getUserByCard(card);

synchronized (Bank.class) {

if (u.getU\_login\_name().equals(card) &&

u.getU\_login\_pwd().equals(pwd)) {

BigDecimal oldData=new BigDecimal(u.getU\_wallet());

BigDecimal money=new BigDecimal(moneyNum);

u.setU\_wallet(oldData.add(money).toString());

u.setSave\_money\_time(new Date());

System.out.println(Thread.currentThread().getName()+"存钱

---->"+u.getU\_name()+"在"+new SimpleDateFormat("yyyy-MM-dd

HH:mm:ss").format(u.getSave\_money\_time())+"存["+moneyNum+"]钱，余额：

"+u.getU\_wallet());

return true;

}

}

System.out.println(getUserByCard(card).getU\_name()+"存钱失败");

return false;

}

//取钱

public Boolean getMoney(String card,String pwd,String moneyNum){

User u=getUserByCard(card);

synchronized (Bank.class) {

if (u!=null && u.getU\_login\_name().equals(card) &&

u.getU\_login\_pwd().equals(pwd)) {

BigDecimal oldData=new BigDecimal(u.getU\_wallet());

BigDecimal money=new BigDecimal(moneyNum);

if(oldData.compareTo(money)>=0){

u.setU\_wallet(oldData.subtract(money).toString());

u.setDraw\_money\_time(new Date());

System.out.println(Thread.currentThread().getName()+"取钱

---->"+u.getU\_name()+"在"+new SimpleDateFormat("yyyy-MM-dd

HH:mm:ss").format(u.getDraw\_money\_time())+"取["+moneyNum+"]钱

,余额："+u.getU\_wallet());

return true;

}else {

System.out.println(getUserByCard(card).getU\_name()+"

要取["+moneyNum+"]钱,但余额不足");

return false;

}

}

}

System.out.println(card+"取钱失败");

return false;

}

//查询余额

public String balanceEnquiry(String card,String pwd){

for(User u :this.userList){

if(u.getU\_login\_name().equals(card)&&

u.getU\_login\_pwd().equals(pwd)){

System.out.println(Thread.currentThread().getName()+":"

+u.getU\_name()+"余额："+u.getU\_wallet());

return u.getU\_wallet();

}

}

System.out.println(Thread.currentThread().getName()+":"+card+"

操作失败");

return null;

}

//获取当前用户

public synchronized User getUserByCard(String card){

for(User u :this.userList){

if(u.getU\_login\_name().equals(card)){

return u;

}

}

return null;

}

public void delayTime(Integer nim){

try {

Thread.sleep(nim);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

}

1. 创建测试类

定义测试类，再类中创建客户对象，并创建和开启线程执行存取钱操作，如文件8-3所示。

1. package chapter0403;

import java.util.ArrayList;

import java.util.List;

public class BankText {

public static void main(String[] args) throws Exception {

User u = new User("张三", "132466", "123", "100");

User uu = new User("李四", "4600882", "123", "0");

List<User> list = new ArrayList<>();

list.add(u);

list.add(uu);

Bank atm = new Bank(list);//初始化数据 模拟

Thread t = new Thread() {

public void run() {

for (int i = 0; i < 10; i++) {

atm.saveMoney("132466", "123", "12");

atm.delayTime(250);

atm.getMoney("4600882", "123", "14");

atm.delayTime(250);

}

}

};

Thread tt = new Thread() {

public void run() {

for (int i = 0; i < 10; i++) {

atm.getMoney("132466", "123", "2");

atm.delayTime(250);

atm.saveMoney("4600882", "123", "12");

atm.delayTime(250);

}

}

};

t.start();

tt.start();

}

}