**账本APP测试**

1.概述

测试目的：本次测试主要检查我们这个账簿APP的各项基本类\函数的正确性，以及软件功能的正确性。

2.测试环境

硬件环境：Android手机一台，PC机一台

软件环境：Android Studio，jdk1.8,Record Espresso

3.测试项目

3.1.单元测试：主要测试两个主要基础类的函数。

Class Record：该类是整个APP中的基本数据类，它包含：姓名、收入或支出金额、收入或支出日期、收入或支出原因。原因我们给出的是一个列表：如下

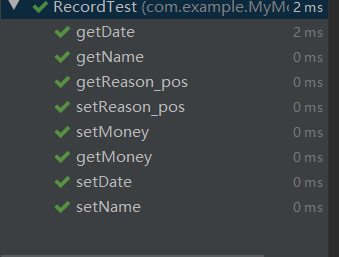
<?xml version="1.0" encoding="utf-8"?>  
<resources>  
<string-array name="Data">  
 <item>吃饭</item>  
 <item>购物</item>  
 <item>出行</item>  
 <item>工资</item>  
 <item>贩卖</item>  
 <item>随礼</item>  
 <item>其他</item>  
</string-array>

于是，我们在定义Record类中的原因时采用的是int类型。

测试代码：

package com.example.MyMoney.Data;  
  
import org.junit.Test;  
  
import static org.junit.Assert.\*;  
  
public class RecordTest {  
 Record record = new Record("张三",500,"2020-12-20",1);  
  
 @Test  
 public void getName() {  
 *assertEquals*("张三",record.getName());  
 }  
  
 @Test  
 public void getMoney() {  
 *assertTrue*(500==record.getMoney());  
 }  
  
 @Test  
 public void getDate() {  
 *assertEquals*("2020-12-20",record.getDate());  
 }  
  
 @Test  
 public void getReason\_pos() {  
 *assertTrue*(1==record.getReason\_pos());  
 }  
  
 @Test  
 public void setMoney() {  
 record.setMoney(300);  
 *assertFalse*(500==record.getMoney());  
 *assertTrue*(300==record.getMoney());  
 }  
 @Test  
 public void setName() {  
 record.setName("李四");  
 *assertEquals*("李四",record.getName());  
 *assertNotEquals*("张三",record.getName());  
 }  
 @Test  
 public void setDate() {  
 record.setDate("2020-12-21");  
 *assertEquals*("2020-12-21",record.getDate());  
 *assertNotEquals*("2020-12-20",record.getDate());  
 }  
 @Test  
 public void setReason\_pos() {  
 record.setReason\_pos(2);  
 *assertTrue*(2==record.getReason\_pos());  
 }  
}

测试结果：



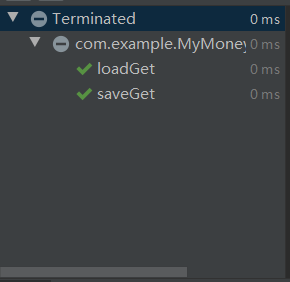
全部通过，表明在该类中的所有函数执行正常。

Class DataBank:该类主要是我们APP保存和加载数据的一个类，整个APP中涉及的收入，支出账目数据都由该类进行保存或加载。

测试代码：

package com.example.MyMoney.Data;  
  
import android.content.Context;  
  
import androidx.test.core.app.ApplicationProvider;  
  
import org.junit.After;  
import org.junit.Before;  
import org.junit.Test;  
  
import java.util.ArrayList;  
  
import static org.junit.Assert.\*;  
  
public class DataBankTest {  
 DataBank dataBank;  
 Context context;  
 Record record = new Record("张三",500,"2020-12-20",1);  
 private ArrayList<Record> infoArrayList\_get = new ArrayList<>();  
  
 @Before  
 public void setUp() throws Exception {  
 context = ApplicationProvider.*getApplicationContext*();  
 dataBank = new DataBank(context);  
 }  
  
 @After  
 public void tearDown() throws Exception {  
 }  
  
 @Test  
 public void loadGet() {  
 //APP第一次打开，dataBank中的数据应为0  
 dataBank.loadGet();  
 infoArrayList\_get=dataBank.getInfoArrayList\_get();  
 //判断是否为0  
 *assertTrue*(infoArrayList\_get.size()==0);  
 }  
  
 @Test  
 public void saveGet() {  
 //APP第一次打开，dataBank中的数据应为0  
 dataBank.loadGet();  
 infoArrayList\_get=dataBank.getInfoArrayList\_get();  
 //给infoArrayList\_get添加一条记录  
 infoArrayList\_get.add(record);  
 //保存该条记录  
 dataBank.saveGet();  
 //第二次加载，dataBank中的infoArrayList\_get长度数据应为1  
 dataBank.loadGet();  
 infoArrayList\_get=dataBank.getInfoArrayList\_get();  
 //判断是否为1  
 *assertTrue*(infoArrayList\_get.size()==1);  
 }  
  
}

测试结果：



全部通过，表示DataBank中的数据保存和加载功能完善，能够正确保存和加载数据。

3.3.压力测试：重复执行某操作添加操作100次（以后的测试可以大大增加重复次数），测试APP抗压能力：

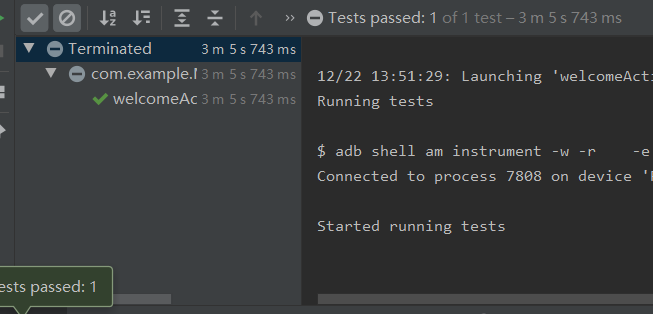
测试代码：

package com.example.MyMoney;  
  
  
import android.view.View;  
import android.view.ViewGroup;  
import android.view.ViewParent;  
  
import androidx.test.espresso.DataInteraction;  
import androidx.test.espresso.ViewInteraction;  
import androidx.test.filters.LargeTest;  
import androidx.test.rule.ActivityTestRule;  
import androidx.test.runner.AndroidJUnit4;  
  
import org.hamcrest.Description;  
import org.hamcrest.Matcher;  
import org.hamcrest.TypeSafeMatcher;  
import org.junit.Rule;  
import org.junit.Test;  
import org.junit.runner.RunWith;  
  
import static androidx.test.espresso.Espresso.*onData*;  
import static androidx.test.espresso.Espresso.*onView*;  
import static androidx.test.espresso.Espresso.*pressBack*;  
import static androidx.test.espresso.action.ViewActions.*click*;  
import static androidx.test.espresso.matcher.ViewMatchers.*withId*;  
import static java.lang.Thread.*sleep*;  
import static org.hamcrest.Matchers.*allOf*;

@LargeTest  
@RunWith(AndroidJUnit4.class)  
public class welcomeActivityTest {  
  
 @Rule  
 public ActivityTestRule<welcomeActivity> mActivityTestRule = new ActivityTestRule<>(welcomeActivity.class);  
  
 @Test  
 public void welcomeActivityTest() {  
 for(int i = 0;i<100;i++) {

//找到Button控件，让其重复点击100次  
 ViewInteraction appCompatButton = *onView*(*allOf*(*withId*(R.id.*update\_button*), *withText*("+")));  
 appCompatButton.perform(*click*());  
 *pressBack*();  
 try {  
 *sleep*(1000);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
  
 private static Matcher<View> childAtPosition(  
 final Matcher<View> parentMatcher, final int position) {  
  
 return new TypeSafeMatcher<View>() {  
 @Override  
 public void describeTo(Description description) {  
 description.appendText("Child at position " + position + " in parent ");  
 parentMatcher.describeTo(description);  
 }  
  
 @Override  
 public boolean matchesSafely(View view) {  
 ViewParent parent = view.getParent();  
 return parent instanceof ViewGroup && parentMatcher.matches(parent)  
 && view.equals(((ViewGroup) parent).getChildAt(position));  
 }  
 };  
 }  
}

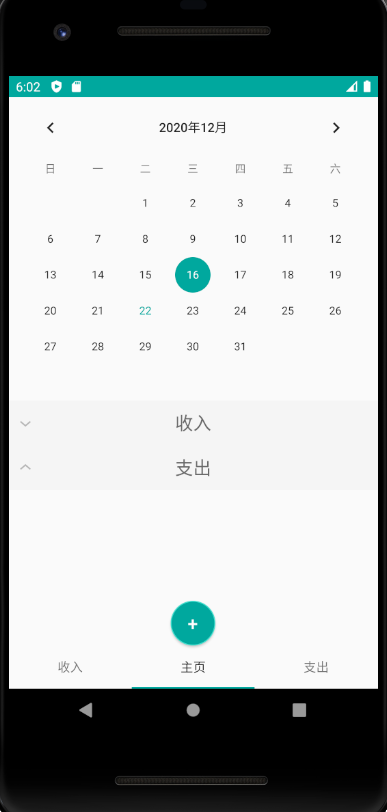
测试结果：



测试通过，APP经受住了这一百次的考验。

3.3全功能手动测试

1.进入APP，点击圆形按钮+,添加收入或支出或取消：

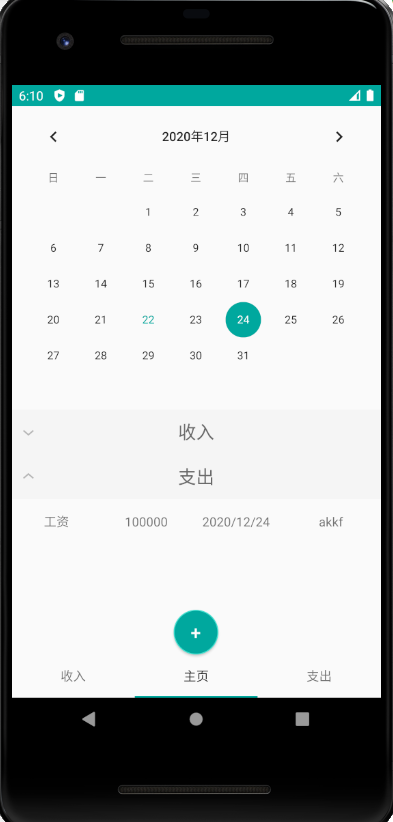
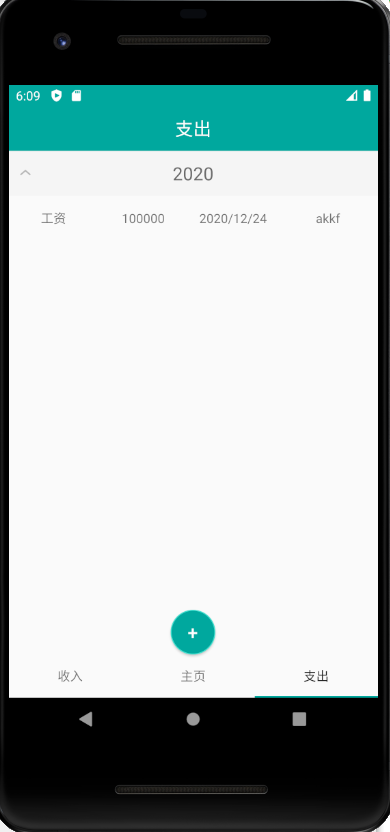
 

2.跳转界面，添加收支信息：

选择日期时弹出一个日历，供用户选择。

3.点击保存，在主页点击添加的日期，即可在下方查看是否有记录，或者滑动到收入或支出界面查看。

如上图，在支出中可看到（添加的为支出）。

重复上述操作，结果在目前所设置的所有功能均能正常运行，不过在添加收入时，程序出现了崩溃，经与开发工程师的交流改写，APP正常运行。

4.测试结论

本次测试的结果大体正确，APP的现有功能可以很完好的运行，测试基本通过，不过APP整体不太美观，功能比较单一，就是基本的增删改查。希望可以增加一个统计结果，以显示当月或当年所有收支费用统计。