Android运算器

开发设计思路：

View：

整个应用由两个activity组成，首页如下：

点击屏幕中央按钮开始答题；对应代码在mainactivity中点按钮触发start事件

public void start(View v) {

Intent intent = new Intent(getApplicationContext(), ShitiActivity.class);

startActivity(intent);

}

通过这段代码跳转到ShiftActivity即答题页面对应的activity



在该页中设置了两个按钮，分别对应reset,tijiao

放题目的位置放了个listview方便加载大量数据

点击重置会重新更新adpater,listview中也会相应的修改数据

点击提交会弹出dialog对话框



点取消对话框会消失，点确定会开始计算答对的题目数弹出一个toast提示答对的题数及分数

算法分析：

当程序调到shiftActivity时

先初始化问题集：

ArrayList<Question>();

**public** **void** updatequestion() {//更新问题集

q.clear();

**if**(myadapter!=**null**){

myadapter.notifyDataSetChanged();}

Question q1 = **new** Question();

q1.setXuhao(0+"");

q.add(q1);

//先一次出20个题出来

**for** (**int** i = 1; i < 20; i++) {// 检查后19道题是否重复

Question q2 = **new** Question();

**for** (**int** j = 0; j < i; j++) {

Question q3 = q.get(j);

**while** (q2.getA() == q3.getA() && q2.getB() == q3.getB()

&& q2.getZ() == q3.getZ()) {

q2 = **new** Question();// 要是问题重复就重新出

}

}

q2.setXuhao(i+"");

q.add(q2);

}

}

上面这个函数实现提出20个不同的问题的功能，每new一个合格的问题都往回查前面有没有重复的问题。

而每个合格的问题的提出在Question类的构造方法中实现：

**public** **class** Question {

**private** **int** a;

**private** **int** b;

**private** **int** z;

**private** **int** anwser;//正确答案

**private** String xuhao;//题号

**private** String yoghuzhi;//用户给的结果

**public** String getXuhao() {

**return** xuhao;

}

**public** **void** setXuhao(String xuhao) {

**this**.xuhao = xuhao;

}

**public** String getYoghuzhi() {

**return** yoghuzhi;

}

**public** **void** setYoghuzhi(String yoghuzhi) {

**this**.yoghuzhi = yoghuzhi;

}

**public** Question() {//构造合格问题

**int** result=0;

**do**{

a=(**int**) (Math.*random*()\*100);

b=(**int**) (Math.*random*()\*100);

z=(**int**) (Math.*random*()\*2)+1;

**switch** (z) {//0代表+,1代表-

**case** 2:

result=a+b;

**break**;

**case** 1:

result=a-b;

**break**;

}

}**while**(result<0||result>100);

anwser=result;

}

**public** **int** getA() {

**return** a;

}

**public** **void** setA(**int** a) {

**this**.a = a;

}

**public** **int** getB() {

**return** b;

}

**public** **void** setB(**int** b) {

**this**.b = b;

}

**public** **int** getZ() {

**return** z;

}

**public** **void** setZ(**int** z) {

**this**.z = z;

}

**public** **int** getAnwser() {

**return** anwser;

}

对于用户输入数据的判断，只需从问题集中取出正确结果和用户输入值进行比较就可以实现

**int** right=0;//对的题数

**for** (**int** i = 0; i < 20; i++) {

**int** youresult=0;

Question q1 = q.get(i);

String s1=q1.getYoghuzhi().toString().trim();

**if**("".equals(s1)){

Toast.*makeText*(ShitiActivity.**this**, "您的第"+i+"题没做，请继续", 1).show();

dialog.dismiss();

**return**;

}//没填当0

**else**{

youresult= Integer.*parseInt*(s1);

**if**( q1.getAnwser()==youresult){

right++;

}}

}

dialog.dismiss();

Toast.*makeText*(ShitiActivity.**this**, "你本次共做对"+right+"道题,得分："+right\*5, 1).show();

}

问题分析：

整个应用的算法并不难实现，但是在把数据展示到listview的过程中，涉及到到item的edittext中取数据的操作，而用户随时都有可能修改数据，所以问题集中的用户输入的结果也应该实时更新，所以在adapter中通过settag的方式绑定当前position的edittext而且设置了textchange的监听来实时监测修改最后用户输入的值。

}