《通讯录》设计说明书

学号： 姓名：

一、功能简介

本app为手机通讯录，包括联系人的添加，修改，单个删除以及批量删除的功能。

1.显示联系人列表功能

2.联系人添加功能

3.联系人修改功能

4.联系人删除功能

5.联系人编辑，批量操作，全选、全不选，删除功能

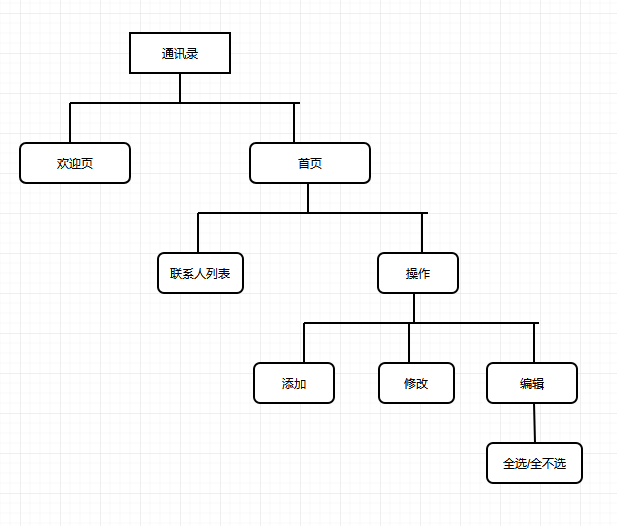


图1 通讯录 功能结构图

二、关键技术介绍

1. 在联系人存储时用到Sqlite数据库，完成联系人的存储功能

2. 在联系人列表某处用到ListView完成联系人显示功能

3. 在编辑全选处用到全选算法，完成批量删除功能

三、主要代码

1.MainActivity.java

public class MainActivity extends Activity {

private ListView lv;

private List<AddressBean> notes;

private MyAdapter adapter;

private TextView tvAdd;

private Button btnEdit, btnCheckAll;//编辑,全选

private boolean isShowSelect;

private TextView tvDelete;//删除按钮

private boolean isSelectAll = true;//true ：全选 false 全不选

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

//编辑

btnEdit = (Button) findViewById(R.id.btn\_edit);

//删除按钮

tvDelete = (TextView) findViewById(R.id.tv\_delete);

tvDelete.setVisibility(View.GONE);

//全选

btnCheckAll = (Button) findViewById(R.id.btn\_check\_all);

btnCheckAll.setVisibility(View.GONE);

//列表

lv = (ListView) findViewById(R.id.lv);

//添加按钮

tvAdd = (TextView) findViewById(R.id.tv\_add);

//查询数据库数据

notes = DbHelper.getInstance(this).getAllAddress();

//列表数据填充

adapter = new MyAdapter(this, notes);

lv.setAdapter(adapter);

//列表点击，进入详情

lv.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

Intent intent = new Intent(MainActivity.this, AddressDetailActivity.class);

intent.putExtra("id", notes.get(position).getId());

intent.putExtra("name", notes.get(position).getName());

intent.putExtra("phone", notes.get(position).getPhone());

startActivity(intent);

}

});

//长按删除

lv.setOnItemLongClickListener(new AdapterView.OnItemLongClickListener() {

@Override

public boolean onItemLongClick(AdapterView<?> parent, View view, final int position, long id) {

//列表对话框

final String[] items = {"删除"};

new AlertDialog.Builder(MainActivity.this)

.setItems(items, new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

switch (which) {

case 0:

//确认对话框

new AlertDialog.Builder(MainActivity.this)

.setMessage("确定删除该条通讯录吗？")

.setPositiveButton("确定", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

//删除通讯录

int noteId = notes.get(position).getId();

DbHelper.getInstance(MainActivity.this).deleteAddress(noteId);

//刷新列表

notes = DbHelper.getInstance(MainActivity.this).getAllAddress();

adapter.updateAll(notes);

}

})

.setNegativeButton("取消", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

}

})

.show();

break;

}

}

})

.show();

return true;

}

});

//添加按钮点击

tvAdd.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(MainActivity.this, AddAddressActivity.class);

startActivity(intent);

}

});

//编辑

btnEdit.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

init();

}

});

//全选

btnCheckAll.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

if (isSelectAll) { //全选

for (AddressBean b : notes) {

b.setChecked(true);

}

} else { //全不选

for (AddressBean b : notes) {

b.setChecked(false);

}

}

isSelectAll = !isSelectAll;

if (!isSelectAll) {

btnCheckAll.setText("全不选");

} else {

btnCheckAll.setText("全选");

}

adapter.notifyDataSetChanged();

}

});

tvDelete.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

//拼接ids

String condition = "";

for (AddressBean b : notes) {

if (b.isChecked()) {

condition += b.getId() + ",";

}

}

if (TextUtils.isEmpty(condition)) {

Toast.makeText(MainActivity.this, "未选中", Toast.LENGTH\_SHORT).show();

return;

}

//去掉结尾的逗号

if (condition.endsWith(",")) {

condition = condition.substring(0, condition.length() - 1);

}

DbHelper.getInstance(MainActivity.this).deleteAddressByIds(condition);

//不显示勾选框

adapter.setShowSelect(false);

btnEdit.setText("编辑");

//删除后隐藏

tvDelete.setVisibility(View.GONE);

btnCheckAll.setVisibility(View.GONE);

isSelectAll = true;

//刷新列表

notes.clear();

adapter.setShowSelect(false);

notes.addAll(DbHelper.getInstance(MainActivity.this).getAllAddress());

adapter.updateAll(notes);

}

});

}

public void init() {

//默认全不勾选

for (AddressBean b : notes) {

b.setChecked(false);

}

isSelectAll = true;

isShowSelect = !isShowSelect;

if (isShowSelect) {

btnEdit.setText("取消");

tvAdd.setVisibility(View.GONE);

tvDelete.setVisibility(View.VISIBLE);

btnCheckAll.setVisibility(View.VISIBLE);

} else {

btnEdit.setText("编辑");

tvAdd.setVisibility(View.VISIBLE);

tvDelete.setVisibility(View.GONE);

btnCheckAll.setVisibility(View.GONE);

}

//显示勾选框

adapter.setShowSelect(isShowSelect);

}

@Override

protected void onResume() {

super.onResume();

notes = DbHelper.getInstance(this).getAllAddress();

adapter.updateAll(notes);

}

}

2.数据库工具类封装

/\*\*

\* 数据库工具类

\*/

public class DbHelper extends SQLiteOpenHelper {

public DbHelper(Context ctx) {

super(ctx, "AddressList", null, 1);

}

@Override

public void onCreate(SQLiteDatabase db) {

//创建表

db.execSQL("CREATE TABLE if not exists address(id integer PRIMARY KEY autoincrement,"

+ "name text, phone text)");

}

@Override

public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {

}

private static DbHelper dbManager;

public static DbHelper getInstance(Context ctx) {

if (dbManager == null) {

synchronized (DbHelper.class) {

if (dbManager == null) {

dbManager = new DbHelper(ctx);

}

}

}

return dbManager;

}

/\*\*

\* 保存通讯录

\*

\* @param bean

\*/

public void saveAddress(AddressBean bean) {

SQLiteDatabase db = getWritableDatabase();

if (db != null) {

db.execSQL("INSERT INTO address(name, phone) VALUES ('" + bean.getName() + "', '" + bean.getPhone() + "')");

}

}

/\*\*

\* 删除通讯录

\*

\* @param id

\*/

public void deleteAddress(int id) {

SQLiteDatabase db = getWritableDatabase();

if (db != null) {

db.execSQL("DELETE FROM address WHERE id = " + id);

}

}

/\*\*

\* 批量删除通讯录

\*

\* @param ids 如： 1,2,3

\*/

public void deleteAddressByIds(String ids) {

SQLiteDatabase db = getWritableDatabase();

if (db != null) {

db.execSQL("DELETE FROM address WHERE id in(" + ids + ")");

}

}

/\*\*

\* 更新通讯录

\*

\* @param record

\*/

public void updateAddress(AddressBean record) {

SQLiteDatabase db = getWritableDatabase();

ContentValues contentValues = new ContentValues();

contentValues.put("name", record.getName());

contentValues.put("phone", record.getPhone());

if (db != null) {

db.update("address", contentValues, "id = ?", new String[]{record.getId() + ""});

}

}

/\*\*

\* 获取所有通讯录

\*

\* @return

\*/

public List<AddressBean> getAllAddress() {

List<AddressBean> records = new ArrayList<AddressBean>();

SQLiteDatabase db = getWritableDatabase();

if (db != null) {

Cursor cursor = db.rawQuery("SELECT \* FROM address", null);

while (cursor.moveToNext()) {

String name = cursor.getString(cursor.getColumnIndex("name"));

String phone = cursor.getString(cursor.getColumnIndex("phone"));

int id = cursor.getInt(cursor.getColumnIndex("id"));

AddressBean note = new AddressBean();

note.setId(id);

note.setName(name);

note.setPhone(phone);

records.add(note);

}

cursor.close();

}

return records;

}

}

3.ListView的适配器

/\*\*

\* 列表数据适配器

\*/

public class MyAdapter extends BaseAdapter {

LayoutInflater inflater;

List<AddressBean> notes;

private boolean isShowSelect;

public MyAdapter(Context context, List<AddressBean> notes) {

inflater = LayoutInflater.from(context);

this.notes = notes;

}

public void updateAll(List<AddressBean> notes) {

this.notes = notes;

notifyDataSetChanged();

}

@Override

public int getCount() {

return notes == null ? 0 : notes.size();

}

@Override

public Object getItem(int position) {

return notes == null ? null : notes.get(position);

}

@Override

public long getItemId(int position) {

return position;

}

@Override

public View getView(final int position, View convertView, ViewGroup parent) {

ViewHolder viewHolder;

if (convertView == null) {

convertView = inflater.inflate(R.layout.list\_item, parent, false);

viewHolder = new ViewHolder();

viewHolder.tvName = (TextView) convertView.findViewById(R.id.tv\_name);

viewHolder.tvPhone = (TextView) convertView.findViewById(R.id.tv\_phone);

viewHolder.ivSelect = (ImageView) convertView.findViewById(R.id.iv\_select);

convertView.setTag(viewHolder);

} else {

viewHolder = (ViewHolder) convertView.getTag();

}

viewHolder.tvName.setText(notes.get(position).getName());

viewHolder.tvPhone.setText(notes.get(position).getPhone());

//是否显示右侧勾选框

if (isShowSelect) {

viewHolder.ivSelect.setVisibility(View.VISIBLE);

} else {

viewHolder.ivSelect.setVisibility(View.GONE);

}

if (notes.get(position).isChecked()) {

viewHolder.ivSelect.setImageResource(R.mipmap.ic\_checked);

} else {

viewHolder.ivSelect.setImageResource(R.mipmap.ic\_unchecked);

}

viewHolder.ivSelect.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

notes.get(position).setChecked(!notes.get(position).isChecked());

notifyDataSetChanged();

}

});

return convertView;

}

public void setShowSelect(boolean isShowSelect) {

this.isShowSelect = isShowSelect;

notifyDataSetChanged();

}

private static class ViewHolder {

TextView tvName;

TextView tvPhone;

ImageView ivSelect;

}

}

1. 通讯录实体类

/\*\*

\* 通讯录实体类

\*/

public class AddressBean {

private int id;

private String name; //姓名

private String phone; //电话

private boolean isChecked;//是否选中

//省略get、set

}