popupwindow、dialog、dialogfragment   
dialog pk dialogfragment：   
Dialog已经不推荐使用了，因为使用它在横竖屏切换等场合容易发生窗体泄漏;Dialog的替代是DialogFragment，它不会出现窗体泄漏的问题，而且兼具Dialog和Fragment的特点，即你既可以通过重写onCreateDialog来创建，也可以通过重写onCreateView来创建；但是注意不能同时重写这两个方法

popupwindow pk dialogfragment：   
popupwindow一般以关联某一个控件的弹窗，你看它的显示方法， public void showAsDropDown(View anchor) {，anchor相对这个控件显示，恩，明白了，需要关联一个控件，显示在这个控件的某个位置时用popupwindow。dialogfragment是整个页面的弹窗

android中实现弹窗的方式如下：

一.Dialog

不设置style，默认的样式很丑。

一个小例子：

public class LoadingDialog extends android.app.Dialog {

private Activity activity;

protected LoadingDialog(Activity activity) {

super(activity, R.style.FloadNormalDialogStyle);

this.activity = activity;

View view = LayoutInflater.from(activity).inflate(R.layout.layout\_loading\_dialog, null);

setContentView(view);

}

public void showLoading() {

if (null == activity || activity.isFinishing()) {

return;

}

setCancelable(false);

show();

}

public void hideLoading() {

if (isShowing()) {

dismiss();

}

}

二.PopUpWindow

虽然AlertDialog对话框基本够满足吊丝程序员日常开发了，但是 AlertDialog对话框还是不够灵活，因此出现了一个完全自定义，灵活度高的PopupWindow弹出式对话框。

一个小例子：

private void dialog6() {

View view = LayoutInflater.from(this).inflate(R.layout.items, null);

final PopupWindow popupWindow = new PopupWindow(view, WindowManager.LayoutParams.MATCH\_PARENT,

WindowManager.LayoutParams.WRAP\_CONTENT, true);

int x = 0;

int y = getStatusBarHeight() + getActionBarHeight();

popupWindow.showAsDropDown(mDialog1, Gravity.TOP, x, y);

//pw对话框设置半透明背景。原理：pw显示时，改变整个窗口的透明度为0.7，当pw消失时，透明度为1

final WindowManager.LayoutParams params = DialogActivity.this.getWindow().getAttributes();

params.alpha = 0.7f;

DialogActivity.this.getWindow().setAttributes(params);

view.findViewById(R.id.btn).setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

isExit = true;

popupWindow.dismiss();

params.alpha = 1f;

DialogActivity.this.getWindow().setAttributes(params);

}

});

//pw对话框消失监听事件

popupWindow.setOnDismissListener(new PopupWindow.OnDismissListener() {

@Override

public void onDismiss() {

params.alpha = 1f;

DialogActivity.this.getWindow().setAttributes(params);

}

});

}

三.Dialog和PopUpWindow的区别

就效果上来说：dialog和popupwindow可以做出完全一样的效果和位置。

1. AlertDialog非阻塞式对话框：对话框的显示不影响后台任务的执行。 PopupWindow阻塞式对话框：对话框弹出后阻塞后台任务执行，直到对话框消失。
2. AlertDialog默认半透明背景，PopupWindow默认没有半透明背景。 即Popupwindow不会给页面其他的部分添加蒙层，而Dialog会。
3. PopupWindow默认不响应Back键，除非设置pw.setBackgroundDrawable(new ColorDrawable(0x00000000));
4. PopupWindow默认没有标题，AlertDialog默认是有标题的，当然可以设置dialog.requestWindowFeature(Window.FEATURE\_NO\_TITLE);取消标题。
5. 二者显示的时候都要设置Gravity。如果不设置，Dialog默认是Gravity.CENTER。
6. Dialog没法设置宽为整个屏幕宽，总有点边界。Popupwindow可以(PopupWindow也可以设置有边界)。   
   我们在写程序的过程中可以根据自己的需要选择使用Popupwindow或者是Dialog。

四.DialogFragment

为何推荐dialogfragment

DialogFragment在android 3.0时被引入。是一种特殊的Fragment，用于在Activity的内容之上展示一个模态的对话框。典型的用于：展示警告框，输入框，确认框等等。在DialogFragment产生之前，我们创建对话框：一般采用AlertDialog和Dialog。注：官方不推荐直接使用Dialog创建对话框。     
使用DialogFragment来管理对话框，当旋转屏幕和按下后退键时可以更好的管理其声明周期，它和Fragment有着基本一致的声明周期。且DialogFragment也允许开发者把Dialog作为内嵌的组件进行重用，类似Fragment（可以在大屏幕和小屏幕显示出不同的效果）。上面会通过例子展示这些好处~     
使用DialogFragment至少需要实现onCreateView或者onCreateDIalog方法。onCreateView即使用定义的xml布局文件展示Dialog。onCreateDialog即利用AlertDialog或者Dialog创建出Dialog。

存在的一些问题

1.去掉默认标题

getDialog().requestWindowFeature(Window.FEATURE\_NO\_TITLE);

2.如何加动画

//设置dialog的 进出 动画

getDialog().getWindow().setWindowAnimations(R.style.animate\_dialog);

http://blog.csdn.net/lmj623565791/article/details/37815413

3.如何把dialog fragment选择的值传递给activity   
接口回调

4.设置DialogFragment 的宽高   
设置DialogFragment 的宽高，在 onCreatView中设置是没有效果的，需要在onStart方法中设置。

@Override

public void onStart() {

/\*设置对话框的宽高\*/

getDialog().getWindow().getAttributes().width=getResources().getDisplayMetrics().widthPixels-200;

/\*下面的方式设置也行\*/

// getDialog().getWindow().setLayout(getResources().getDisplayMetrics().widthPixels-200, getDialog().getWindow().getAttributes().height);

super.onStart();

}

5.推荐的用法

/\*\*

\* 这种封装方式非常的好，可以给给外界提供一个数据访问的入口

\*

\* @return

\*/

public static DialogFragmentTest getInstance(List<DialogFragmentTestDemo> demoList) {

DialogFragmentTest dialogFragmentTest = new DialogFragmentTest();

Bundle b = new Bundle();

b.putSerializable("demoList", (Serializable) demoList);

dialogFragmentTest.setArguments(b);

return dialogFragmentTest;

}

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13

例子：

dialogFragment：

public class DialogFragmentTest extends DialogFragment implements View.OnClickListener {

private GridView mGridview;

private ImageView mCloseDialog;

private List<DialogFragmentTestDemo> demoList;

public static final String DIALOGFRAGMENTTEST = "dialogfragmenttest";

private CallBack mCallBack;

public void setCallBack(CallBack callBack) {

mCallBack = callBack;

}

public interface CallBack {

void getChoicedTxt(String str);

}

/\*\*

\* 这种封装方式非常的好，可以给给外界提供一个数据访问的入口

\*

\* @return

\*/

public static DialogFragmentTest getInstance(List<DialogFragmentTestDemo> demoList) {

DialogFragmentTest dialogFragmentTest = new DialogFragmentTest();

Bundle b = new Bundle();

b.putSerializable("demoList", (Serializable) demoList);

dialogFragmentTest.setArguments(b);

return dialogFragmentTest;

}

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

this.setCancelable(true);

}

@Override

public void onStart() {

super.onStart();

// 设置dialog的layout

DisplayMetrics dm = new DisplayMetrics();

getActivity().getWindowManager().getDefaultDisplay().getMetrics(dm);

WindowManager.LayoutParams layoutParams = getDialog().getWindow().getAttributes();

layoutParams.width = dm.widthPixels;

layoutParams.height = layoutParams.WRAP\_CONTENT;

layoutParams.gravity = Gravity.BOTTOM;

getDialog().getWindow().setAttributes(layoutParams);

}

@Override

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {

getDialog().requestWindowFeature(Window.FEATURE\_NO\_TITLE);

getDialog().getWindow().setWindowAnimations(R.style.animate\_dialog);

View view = inflater.inflate(R.layout.dialogfragment\_test, container);

mGridview = (GridView) view.findViewById(R.id.gdv\_content);

mCloseDialog = (ImageView) view.findViewById(R.id.cancel\_dialog);

mCloseDialog.setOnClickListener(this);

handleArgs();

getDialog().getWindow().setBackgroundDrawable(new ColorDrawable(Color.WHITE));

return view;

}

private void handleArgs() {

Bundle bundle = getArguments();

demoList = (List<DialogFragmentTestDemo>) bundle.getSerializable("demoList");

GridViewAdapter gridViewAdapter = new GridViewAdapter(getActivity(), demoList);

mGridview.setAdapter(gridViewAdapter);

mGridview.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

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

Toast.makeText(getActivity(), demoList.get(position).getStr(), Toast.LENGTH\_LONG).show();

mCallBack.getChoicedTxt(demoList.get(position).getStr());

}

});

}

@Override

public void onClick(View v) {

switch (v.getId()) {

case R.id.cancel\_dialog:

dismissDialog();

break;

default:

}

}

public void dismissDialog() {

android.app.Fragment prev = getFragmentManager().findFragmentByTag(DIALOGFRAGMENTTEST);

if (prev != null) {

DialogFragment df = (DialogFragment) prev;

df.dismiss();

}

}

}

Activity：

final Button viewById = (Button) findViewById(R.id.btn\_dialogfragment\_test);

viewById.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

ArrayList<DialogFragmentTestDemo> demos = new ArrayList<DialogFragmentTestDemo>();

for (int i = 0; i < 3; i++) {

DialogFragmentTestDemo demo = new DialogFragmentTestDemo();

demo.setStr("我是" + i);

demos.add(demo);

}

;

DialogFragmentTest instance = DialogFragmentTest.getInstance(demos);

instance.setCallBack(new DialogFragmentTest.CallBack() {

@Override

public void getChoicedTxt(String str) {

viewById.setText(str);

}

});

instance.show(getFragmentManager(), DialogFragmentTest.DIALOGFRAGMENTTEST);

}

});

效果：   
