1. Popuwindow不能在Activity的生命周期内操作。

尝试在打开界面的时候让Popuwindow默认显示出来，一开始的思路是在onCreate()中直接创建Popuwindow，并showAtLocation.后来。。。后来就奔溃了！原来，如果在Activity生命周期内运行Popuwindow,由于popuwindow的运行机制，会导致程序被Crash掉。。。

2.popWin.setOutsideTouchable(true); 这个方法，一看就知道是设置点击Popuwindow区域外的地方时隐藏 但是必须要为Popuwindow设置背景！！！才能实现

加入以下代码便能成功：

popWin.setBackgroundDrawable(new BitmapDrawable);//即只要给他设置了背景就行

以上方法不能用在fragment里弹出的popupWindow,解决方法是在popupWindow的button添加监听，dismiss该window

　　比如弹出框的布局：

http://images.cnblogs.com/OutliningIndicators/ExpandedBlockStart.gif

[复制代码](javascript:void(0);)

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:background="#FFBBFFBB"

android:orientation="vertical" >

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="10dp"

android:text="Hello My Window"

android:textSize="20sp" />

<Button

android:id="@+id/button1"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="10dp"

android:text="Button"

android:textSize="20sp" />

</LinearLayout>

[复制代码](javascript:void(0);)

　　Activity的布局中只有一个按钮，按下后会弹出框，Activity代码如下：

[复制代码](javascript:void(0);)

package com.example.hellopopupwindow;

import android.os.Bundle;

import android.app.Activity;

import android.content.Context;

import android.util.Log;

import android.view.LayoutInflater;

import android.view.MotionEvent;

import android.view.View;

import android.view.View.OnClickListener;

import android.view.View.OnTouchListener;

import android.view.ViewGroup.LayoutParams;

import android.widget.Button;

import android.widget.PopupWindow;

import android.widget.Toast;

public class MainActivity extends Activity {

private Context mContext = null;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

mContext = this;

Button button = (Button) findViewById(R.id.button);

button.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

showPopupWindow(view);

}

});

}

private void showPopupWindow(View view) {

// 一个自定义的布局，作为显示的内容

View contentView = LayoutInflater.from(mContext).inflate(

R.layout.pop\_window, null);

// 设置按钮的点击事件

Button button = (Button) contentView.findViewById(R.id.button1);

button.setOnClickListener(new OnClickListener() {

@Override

public void onClick(View v) {

Toast.makeText(mContext, "button is pressed",

Toast.LENGTH\_SHORT).show();

}

});

final PopupWindow popupWindow = new PopupWindow(contentView,

LayoutParams.WRAP\_CONTENT, LayoutParams.WRAP\_CONTENT, true);

popupWindow.setTouchable(true);

popupWindow.setTouchInterceptor(new OnTouchListener() {

@Override

public boolean onTouch(View v, MotionEvent event) {

Log.i("mengdd", "onTouch : ");

return false;

// 这里如果返回true的话，touch事件将被拦截

// 拦截后 PopupWindow的onTouchEvent不被调用，这样点击外部区域无法dismiss

}

});

// 如果不设置PopupWindow的背景，无论是点击外部区域还是Back键都无法dismiss弹框

// 我觉得这里是API的一个bug

popupWindow.setBackgroundDrawable(getResources().getDrawable(

R.drawable.selectmenu\_bg\_downward));

// 设置好参数之后再show

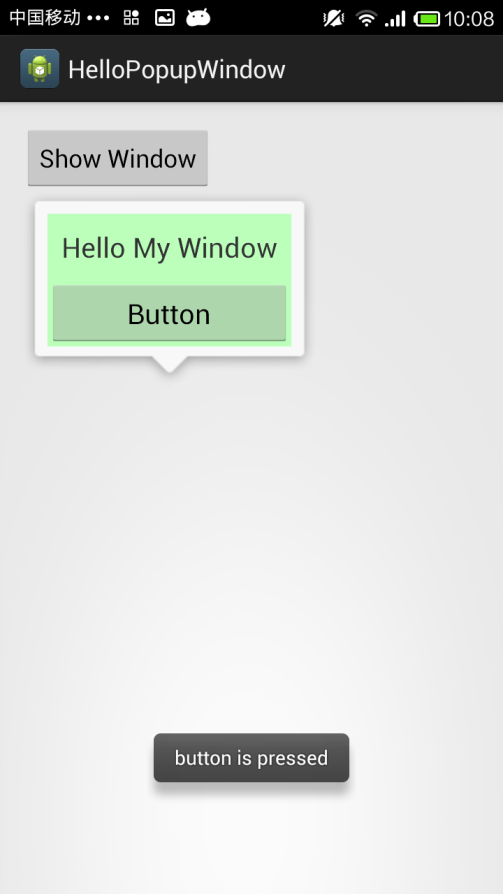
popupWindow.showAsDropDown(view);

}

}

[复制代码](javascript:void(0);)

　　弹出框的布局中有一个TextView和一个Button，Button点击后显示Toast，如图：



　　第一次实现的时候遇到了问题，就是弹出框不会在按下Back键的时候消失，点击弹框外区域也没有正常消失，搜索了一下，都说只要设置背景就好了。

　　然后我就找了个图片，果然弹框能正常dismiss了(见注释)。

**PopupWindow源码分析**

　　为了解答一下上面的问题，看看源码(**最新API Level 19，Android 4.4.2**)。

**1.显示方法**

　　显示提供了两种形式：

　　showAtLocation()显示在指定位置，有两个方法重载：

public void showAtLocation(View parent, int gravity, int x, int y)

public void showAtLocation(IBinder token, int gravity, int x, int y)

　　showAsDropDown()显示在一个参照物View的周围，有三个方法重载：

public void showAsDropDown(View anchor)

public void showAsDropDown(View anchor, int xoff, int yoff)

public void showAsDropDown(View anchor, int xoff, int yoff, int gravity)

　　最后一种带Gravity参数的方法是API 19新引入的。

　　弹出的方法中首先需要**preparePopup()**，最后再**invokePopup()**。

　　prepare的方法中可以看到有无背景的分别：

[复制代码](javascript:void(0);)

/\*\*

\* <p>Prepare the popup by embedding in into a new ViewGroup if the

\* background drawable is not null. If embedding is required, the layout

\* parameters' height is mnodified to take into account the background's

\* padding.</p>

\*

\* @param p the layout parameters of the popup's content view

\*/

private void preparePopup(WindowManager.LayoutParams p) {

if (mContentView == null || mContext == null || mWindowManager == null) {

throw new IllegalStateException("You must specify a valid content view by "

+ "calling setContentView() before attempting to show the popup.");

}

if (mBackground != null) {

final ViewGroup.LayoutParams layoutParams = mContentView.getLayoutParams();

int height = ViewGroup.LayoutParams.MATCH\_PARENT;

if (layoutParams != null &&

layoutParams.height == ViewGroup.LayoutParams.WRAP\_CONTENT) {

height = ViewGroup.LayoutParams.WRAP\_CONTENT;

}

// when a background is available, we embed the content view

// within another view that owns the background drawable

PopupViewContainer popupViewContainer = new PopupViewContainer(mContext);

PopupViewContainer.LayoutParams listParams = new PopupViewContainer.LayoutParams(

ViewGroup.LayoutParams.MATCH\_PARENT, height

);

popupViewContainer.setBackgroundDrawable(mBackground);

popupViewContainer.addView(mContentView, listParams);

mPopupView = popupViewContainer;

} else {

mPopupView = mContentView;

}

mPopupViewInitialLayoutDirectionInherited =

(mPopupView.getRawLayoutDirection() == View.LAYOUT\_DIRECTION\_INHERIT);

mPopupWidth = p.width;

mPopupHeight = p.height;

}

[复制代码](javascript:void(0);)

**2.背景是否为空对Touch事件的影响**

　　如果有背景，则会在contentView外面包一层**PopupViewContainer**之后作为mPopupView，如果没有背景，则直接用contentView作为mPopupView。

　　而这个**PopupViewContainer**是一个内部私有类，它继承了FrameLayout，在其中重写了Key和Touch事件的分发处理：

[复制代码](javascript:void(0);)

@Override

public boolean dispatchKeyEvent(KeyEvent event) {

if (event.getKeyCode() == KeyEvent.KEYCODE\_BACK) {

if (getKeyDispatcherState() == null) {

return super.dispatchKeyEvent(event);

}

if (event.getAction() == KeyEvent.ACTION\_DOWN

&& event.getRepeatCount() == 0) {

KeyEvent.DispatcherState state = getKeyDispatcherState();

if (state != null) {

state.startTracking(event, this);

}

return true;

} else if (event.getAction() == KeyEvent.ACTION\_UP) {

KeyEvent.DispatcherState state = getKeyDispatcherState();

if (state != null && state.isTracking(event) && !event.isCanceled()) {

dismiss();

return true;

}

}

return super.dispatchKeyEvent(event);

} else {

return super.dispatchKeyEvent(event);

}

}

@Override

public boolean dispatchTouchEvent(MotionEvent ev) {

if (mTouchInterceptor != null && mTouchInterceptor.onTouch(this, ev)) {

return true;

}

return super.dispatchTouchEvent(ev);

}

@Override

public boolean onTouchEvent(MotionEvent event) {

final int x = (int) event.getX();

final int y = (int) event.getY();

if ((event.getAction() == MotionEvent.ACTION\_DOWN)

&& ((x < 0) || (x >= getWidth()) || (y < 0) || (y >= getHeight()))) {

dismiss();

return true;

} else if (event.getAction() == MotionEvent.ACTION\_OUTSIDE) {

dismiss();

return true;

} else {

return super.onTouchEvent(event);

}

}

[复制代码](javascript:void(0);)

**由于PopupView本身并没有重写Key和Touch事件的处理，所以如果没有包这个外层容器类，点击Back键或者外部区域是不会导致弹框消失的。**

**补充Case: 弹窗不消失,但是事件向下传递**

　　如上所述:

**设置了PopupWindow的background,点击Back键或者点击弹窗的外部区域,弹窗就会dismiss.**

**相反,如果不设置PopupWindow的background,那么点击back键和点击弹窗的外部区域,弹窗是不会消失的.**

　　那么,如果我想要一个效果,**点击外部区域,弹窗不消失,但是点击事件会向下面的activity传递**,比如下面是一个WebView,我想点击里面的链接等.

　　研究了半天，说是要给Window设置一个Flag，

**WindowManager.LayoutParams.FLAG\_NOT\_TOUCH\_MODAL**

　　看了源码，这个Flag的设置与否是由一个叫mNotTouchModal的字段控制，但是设置该字段的set方法被标记为@hide。

　　所以要通过反射的方法调用：

[复制代码](javascript:void(0);)

　　/\*\*

\* Set whether this window is touch modal or if outside touches will be sent

\* to

\* other windows behind it.

\*

\*/

public static void setPopupWindowTouchModal(PopupWindow popupWindow,

boolean touchModal) {

if (null == popupWindow) {

return;

}

Method method;

try {

method = PopupWindow.class.getDeclaredMethod("setTouchModal",

boolean.class);

method.setAccessible(true);

method.invoke(popupWindow, touchModal);

}

catch (Exception e) {

e.printStackTrace();

}

}

[复制代码](javascript:void(0);)

　　然后在程序中：

UIUtils.setPopupWindowTouchModal(popupWindow, false);

　　该popupWindow外部的事件就可以传递给下面的Activity了。