窗口背景透明，设置背景透明度 #aarrggbb

aa的大小影响窗口透明度的大小

PopupWindow window;

View v = this.getLayoutInflater().inflate(R.layout.popupwindow, null);

window = new PopupWindow(v, 200, 300);

**window.setOutsideTouchable(true);**;//这里设置显示PopuWindow之后在外面点击是否有效。如果为false的话，那么点击PopuWindow外面并不会关闭PopuWindow。当然这里很明显只能在Touchable下才能使用。不设置此项则下面的捕获window外touch事件就无法触发。

　　变态

　　Drawable win\_bg = this.getResources().getDrawable(R.drawable.bg);

**window.setBackgroundDrawable(win\_bg);**

　　这个很，即使在XML里设置了background也认为是没有背景，必须在这里指定背景，如果不指定同样无法触发下面的Touch监听事件。

　　window.setTouchInterceptor(new OnTouchListener() {

　　@Override

　　public boolean onTouch(View v, MotionEvent event) {

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

　　window.dismiss();

　　return false;

　　}

　　});

**PopupWindow用法**  
  
使用PopupWindow可实现弹出窗口效果,，其实和AlertDialog一样，也是一种对话框，两者也经常混用，但是也各有特点。下面就看看使用方法。  
首先初始化一个PopupWindow，指定窗口大小参数。  
  
PopupWindow mPop = new PopupWindow(getLayoutInflater().inflate(R.layout.window, null),  
LayoutParams.WRAP\_CONTENT, LayoutParams.WRAP\_CONTENT);  
也可以分开写：  
LayoutInflater mLayoutInflater = (LayoutInflater) getSystemService(LAYOUT\_INFLATER\_SERVICE);  
//自定义布局  
ViewGroup menuView = (ViewGroup) mLayoutInflater.inflate(  
                    R.layout.window, null, true);  
PopupWindow mPop = new PopupWindow(menuView, LayoutParams.WRAP\_CONTENT,  
                    LayoutParams.WRAP\_CONTENT, true);  
当然也可以手动设置PopupWindow大小。  
mPop.setContentView(menuView );//设置包含视图  
mPop.setWidth(int )  
mPop.setHeight(int )//设置弹出框大小  
  
设置进场动画：  
mPop.setAnimationStyle(R.style.AnimationPreview);//设置动画样式  
  
  
mPop.setOutsideTouchable(true);//这里设置显示PopuWindow之后在外面点击是否有效。如果为false的话，那么点击PopuWindow外面并不会关闭PopuWindow。当然这里很明显只能在Touchable下才能使用。  
  
当有mPop.setFocusable(false);的时候，说明PopuWindow不能获得焦点，即使设置设置了背景不为空也不能点击外面消失，只能由dismiss()消失，但是外面的View的事件还是可以触发,back键也可以顺利dismiss掉。当设置为popuWindow.setFocusable(true);的时候，加上下面两行设置背景代码，点击外面和Back键才会消失。  
mPop.setFocusable(true);  
需要顺利让PopUpWindow dimiss（即点击PopuWindow之外的地方此或者back键PopuWindow会消失）；PopUpWindow的背景不能为空。必须在popuWindow.showAsDropDown(v);或者其它的显示PopuWindow方法之前设置它的背景不为空：  
  
mPop.setBackgroundDrawable(new ColorDrawable(0));  
  
  
  
mPop.showAsDropDown(anchor, 0, 0);//设置显示PopupWindow的位置位于View的左下方，x,y表示坐标偏移量  
  
mPop.showAtLocation(findViewById(R.id.parent), Gravity.LEFT, 0, -90);（以某个View为参考）,表示弹出窗口以parent组件为参考，位于左侧，偏移-90。  
mPop.setOnDismissListenerd(new PopupWindow.OnDismissListener(){})//设置窗口消失事件  
  
注：window.xml为布局文件  
  
总结：  
  
1、为PopupWindow的view布局，通过LayoutInflator获取布局的view.如:  
  
LayoutInflater inflater =(LayoutInflater)              
  
this.anchor.getContext().getSystemService(Context.LAYOUT\_INFLATER\_SERVICE);  
  
View textEntryView =  inflater.inflate(R.layout.paopao\_alert\_dialog, null);  
  
         
  
2、显示位置，可以有很多方式设置显示方式  
  
pop.showAtLocation(findViewById(R.id.ll2), Gravity.LEFT, 0, -90);  
  
或者  
  
pop.showAsDropDown(View anchor, int xoff, int yoff)  
  
   
  
3、进出场动画  
  
pop.setAnimationStyle(R.style.PopupAnimation);  
  
   
  
4、点击PopupWindow区域外部,PopupWindow消失  
  
   this.window = new PopupWindow(anchor.getContext());  
  
   
  
this.window.setTouchInterceptor(new OnTouchListener() {  
  
@Override  
  
public boolean onTouch(View v, MotionEvent event) {  
  
if(event.getAction() ==MotionEvent.ACTION\_OUTSIDE) {                
  
BetterPopupWindow.this.window.dismiss();  
  
return true;  
  
}  
  
return false;  
  
}  
  
});  
  
  
  
**PopupWindow 自适应宽度实例**  
  
  
@Override   
protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {          
     int maxWidth = meathureWidthByChilds() + getPaddingLeft() + getPaddingRight(); super.onMeasure(MeasureSpec.makeMeasureSpec(maxWidth,MeasureSpec.EXACTLY),heightMeasureSpec);     
}     
   public int meathureWidthByChilds() {       
      int maxWidth = 0;      
      View view = null;  
      for (int i = 0; i < getAdapter().getCount(); i++) {   
           view = getAdapter().getView(i, view, this);       
           view.measure(MeasureSpec.UNSPECIFIED, MeasureSpec.UNSPECIFIED);       
      if (view.getMeasuredWidth() > maxWidth){          
         maxWidth = view.getMeasuredWidth();      
      }         
  }         
   return maxWidth;    
}  
  
  
  
**PopupWindow自适应布局**  
  
Android自带的Menu菜单，常常无法满足我们的需求，所以就只有自己写menu菜单，通常的选择是用PopupWindow来实现自定义的menu菜单，先看代码，再来说明要注意的几点：  
  
    View menuView = inflater.inflate(R.layout.menu\_popwindow, null);    
    final PopupWindow p = new PopupWindow(mContext);    
    p.setContentView(menuView);    
    p.setWidth(ViewGroup.LayoutParams.FILL\_PARENT);    
    p.setHeight(ViewGroup.LayoutParams.WRAP\_CONTENT);    
    p.setAnimationStyle(R.style.MenuWindow);    
    p.setOnDismissListener(this);    
    p.setOutsideTouchable(false);    
    p.setBackgroundDrawable(new ColorDrawable(android.R.color.transparent));    
    p.setFocusable(true); // 如果把焦点设置为false，则其他部份是可以点击的，也就是说传递事件时，不会先走PopupWindow    
        
    mPopwindow = p;    
  
来说明其中的几点：  
  
  1. 为了让PopupWindow自适应屏幕的宽度，设置宽度时用ViewGroup.LayoutParams.FILL\_PARENT，为了自适应子布局的高度，设置高度时用ViewGroup.LayoutParams.WRAP\_CONTENT  
  
  2. 由于PopupWindow类没有继承ViewGroup类，所以inflater.inflate(int resource, ViewGroup root)方法的第二个参数只能传为null，传null会使最外层布局的android:layout\_xxx都不起作用。所以高度是以第二层布局为主  
  
  3. 为了设置背景和边距，其背景只能设置在第二层布局里，因第一层布局的android:layout\_marginXxx不起作用，而设置android:padding\_Xxx不会影响背景。  
  
  4. menu有一个特点，就是点外部，menu菜单要消失，要实现这个，有几个属性要一起设置：p.setOutsideTouchable(false);p.setBackgroundDrawable();p.setFocusable(true)