# **Snackbar花式使用指南**

Snackbar是Android Support Design Library库中的一个控件，可以在屏幕底部快速弹出消息，比Toast更加好用。本文对原生Snackbar进行了修改，使其更加灵活。

## 1.Snackbar基本介绍

使用Snackbar要导入com.android.support:design库。

Snackbar显示在所有屏幕其它元素之上(屏幕最顶层)，同一时间只能显示一个snackbar。

Snackbar的基本使用很简单，与Toast类似。

Snackbar.make(view, message\_text, duration)

.setAction(action\_text, click\_listener)

.show();

make()方法是生成Snackbar的。Snackbar需要一个控件容器view用来容纳，官方推荐使用CoordinatorLayout来确保Snackbar和其他组件的交互，比如滑动取消Snackbar、Snackbar出现时FloatingActionButton上移。显示时间duration有三种类型LENGTH\_SHORT、LENGTH\_LONG和LENGTH\_INDEFINITE。

setAction()方法可设置Snackbar右侧按钮，增加进行交互事件。如果不使用setAction()则只显示左侧message。

Snackbar.make(coordinatorLayout,"这是massage", Snackbar.LENGTH\_LONG).setAction("这是action", new View.OnClickListener() {

@Override

public void onClick(View v) {

Toast.makeText(MainActivity.this,"你点击了action",Toast.LENGTH\_SHORT).show();

}

}).show();

下面这张图演示了上面代码所实现的效果：Snackbar长显示、点击Action弹出toast提示以及Snackbar在CoordinatorLayout中滑动取消。

如果你想在Snackbar的显示时或消失时做些什么，可以调用Snackbar的setCallback()方法。

## 2.多彩Snackbar

Snackbar和Toast的默认样式都很单一，但是有时我们希望把不同类型信息区别显示，从而使用户更容易注意到提示信息。所以使Snackbar变色是一个好主意。

Snackbar的官方API只提供了setActionTextColor()这个方法修改Action的文字颜色，这怎么办？查源码吧，哪里不会点哪里。(><)

在源码中我们看到Snackbar中定义了一个继承自LinearLayout的内部类SnackbarLayout，Snackbar的样子就是由这个SnackbarLayout实现的。

SnackbarLayout中加载了R.layout.design\_layout\_snackbar\_include布局文件，打开后看到下面这段代码（我把padding、margin的具体数值也打了出来）：

<merge xmlns:android="http://schemas.android.com/apk/res/android"><TextView

android:id="@+id/snackbar\_text"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:paddingTop="14dp"

android:paddingBottom="14dp"

android:paddingLeft="12dp"

android:paddingRight="12dp"

android:textAppearance="@style/TextAppearance.Design.Snackbar.Message"

android:maxLines="2"

android:layout\_gravity="center\_vertical|left|start"

android:ellipsize="end"

android:textAlignment="viewStart"/>

<Button

android:id="@+id/snackbar\_action"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginLeft="0dp"

android:layout\_marginStart="0dp"

android:layout\_gravity="center\_vertical|right|end"

android:paddingTop="14dp"

android:paddingBottom="14dp"

android:paddingLeft="12dp"

android:paddingRight="12dp"

android:visibility="gone"

android:textColor="?attr/colorAccent"

style="?attr/borderlessButtonStyle"/></merge>

由命名可知，以snackbar\_text为名的TextView就是Snackbar左侧的message。

好了，我们开始修改Snackbar的背景颜色和message字体颜色吧。

public static void setSnackbarColor(Snackbar snackbar, int messageColor, int backgroundColor) {

View view = snackbar.getView();//获取Snackbar的view

if(view!=null){

view.setBackgroundColor(backgroundColor);//修改view的背景色

((TextView) view.findViewById(R.id.snackbar\_text)).setTextColor(messageColor);//获取Snackbar的message控件，修改字体颜色

}

}

很简单，没有几行代码。

本文最后提供的Snackbar封装类代码中定义了4种不同类型的信息：Info(妹子向你发来一条消息)、Confirm(妹子已收到你发出的消息)、Warning(妹子删除了你发出的消息)、Alert(妹子已将你拉黑)，分别用蓝色、绿色、橙色、红色来表示。

## 3.在Snackbar中增加图标

**短文本**

通常 Snackbar 的高度应该仅仅用于容纳所有的文本，而文本应该与执行的操作相关。Snackbar 中不能包含图标，操作只能以文本的形式存在。

**最多0-1个操作，不包含取消按钮**

当一个动作发生的时候，应当符合提示框和可用性规则。当有2个或者2个以上的操作出现时，应该使用提示框而不是 Snackbar，即使其中的一个是取消操作。如果 Snackbar 中提示的操作重要到需要打断屏幕上正在进行的操作，那么理当使用提示框而非 Snackbar。

上面这段是谷歌 [Material Design设计规范](http://wiki.jikexueyuan.com/project/material-design/components/snackbars-and-toasts.html" \t "_blank)中的话。

但是我就是想在Snackbar中加图标增加趣味性，引起用户注意怎么办？我就是想在Snackbar中放两个按钮进行可选非必要操作怎么办？我就是想整幺蛾子。︿(￣︶￣)︿

设计规范中的说法是有道理的，因为官方认为“Snackbar是一种针对操作的轻量级反馈机制”，做的麻烦了影响视觉感受。但是对于上述任性的开发者（或者是接了奇葩需求的苦逼开发者）我们也有解决方法。

前面我们提到过Snackbar的view是由SnackbarLayout实现的，而SnackbarLayout是继承自LinearLayout，那么我们新建一个布局添加进去不就行了么。(～o￣￣)～o...

public static void SnackbarAddView(Snackbar snackbar,int layoutId,int index) {

View snackbarview = snackbar.getView();//获取snackbar的View(其实就是SnackbarLayout)

Snackbar.SnackbarLayout snackbarLayout=(Snackbar.SnackbarLayout)snackbarview;//将获取的View转换成SnackbarLayout

View add\_view = LayoutInflater.from(snackbarview.getContext()).inflate(layoutId,null);//加载布局文件新建View

LinearLayout.LayoutParams p = new LinearLayout.LayoutParams(LinearLayout.LayoutParams.WRAP\_CONTENT,LinearLayout.LayoutParams.WRAP\_CONTENT);//设置新建布局参数

p.gravity= Gravity.CENTER\_VERTICAL;//设置新建布局在Snackbar内垂直居中显示

snackbarLayout.addView(add\_view,index,p);//将新建布局添加进snackbarLayout相应位置

}

上面的代码中，如果我们不设置向Snackbar中添加的布局文件的布局参数，新布局会显示在Snackbar内的顶部。使用上述任性方法的时候要注意新加布局的大小和Snackbar内文字长度，Snackbar过大或过于花哨了可不好看。

下面是使用示例。我们先新建一个布局，暂时命名为snackbar\_addview.xml,简单的放进了一个ImageView，图片就是android默认图标。

<?xml version="1.0" encoding="utf-8"?><LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"android:layout\_width="wrap\_content"android:layout\_height="wrap\_content"android:layout\_gravity="center\_vertical"

><ImageView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_gravity="center\_vertical"

android:src="@mipmap/ic\_launcher"/></LinearLayout>

然后在activity中写下snackbar的设置：

Snackbar snackbar= Snackbar.make(coordinatorLayout,"这是massage", Snackbar.LENGTH\_LONG);

SnackbarUtil.setSnackbarColor(snackbar,SnackbarUtil.blue);

SnackbarUtil.SnackbarAddView(snackbar,R.layout.snackbar\_addview,0);

snackbar.show();

## 4.SnackbarUtil

我将我常用的Snackbar相关设置封装成了一个类，大家可以根据自己的需求使用。

/\*\*

\* Created by 赵晨璞 on 2016/5/1.

\*/public class SnackbarUtil {

public static final int Info = 1;public static final int Confirm = 2;public static final int Warning = 3;public static final int Alert = 4;

public static int red = 0xfff44336;public static int green = 0xff4caf50;public static int blue = 0xff2195f3;public static int orange = 0xffffc107;

/\*\*

\* 短显示Snackbar，自定义颜色

\* @param view

\* @param message

\* @param messageColor

\* @param backgroundColor

\* @return

\*/public static Snackbar ShortSnackbar(View view, String message, int messageColor, int backgroundColor){

Snackbar snackbar = Snackbar.make(view,message, Snackbar.LENGTH\_SHORT);

setSnackbarColor(snackbar,messageColor,backgroundColor);

return snackbar;

}

/\*\*

\* 长显示Snackbar，自定义颜色

\* @param view

\* @param message

\* @param messageColor

\* @param backgroundColor

\* @return

\*/public static Snackbar LongSnackbar(View view, String message, int messageColor, int backgroundColor){

Snackbar snackbar = Snackbar.make(view,message, Snackbar.LENGTH\_LONG);

setSnackbarColor(snackbar,messageColor,backgroundColor);

return snackbar;

}

/\*\*

\* 自定义时常显示Snackbar，自定义颜色

\* @param view

\* @param message

\* @param messageColor

\* @param backgroundColor

\* @return

\*/public static Snackbar IndefiniteSnackbar(View view, String message,int duration,int messageColor, int backgroundColor){

Snackbar snackbar = Snackbar.make(view,message, Snackbar.LENGTH\_INDEFINITE).setDuration(duration);

setSnackbarColor(snackbar,messageColor,backgroundColor);

return snackbar;

}

/\*\*

\* 短显示Snackbar，可选预设类型

\* @param view

\* @param message

\* @param type

\* @return

\*/public static Snackbar ShortSnackbar(View view, String message, int type){

Snackbar snackbar = Snackbar.make(view,message, Snackbar.LENGTH\_SHORT);

switchType(snackbar,type);

return snackbar;

}

/\*\*

\* 长显示Snackbar，可选预设类型

\* @param view

\* @param message

\* @param type

\* @return

\*/public static Snackbar LongSnackbar(View view, String message,int type){

Snackbar snackbar = Snackbar.make(view,message, Snackbar.LENGTH\_LONG);

switchType(snackbar,type);

return snackbar;

}

/\*\*

\* 自定义时常显示Snackbar，可选预设类型

\* @param view

\* @param message

\* @param type

\* @return

\*/public static Snackbar IndefiniteSnackbar(View view, String message,int duration,int type){

Snackbar snackbar = Snackbar.make(view,message, Snackbar.LENGTH\_INDEFINITE).setDuration(duration);

switchType(snackbar,type);

return snackbar;

}

//选择预设类型private static void switchType(Snackbar snackbar,int type){

switch (type){

case Info:

setSnackbarColor(snackbar,blue);

break;

case Confirm:

setSnackbarColor(snackbar,green);

break;

case Warning:

setSnackbarColor(snackbar,orange);

break;

case Alert:

setSnackbarColor(snackbar,Color.YELLOW,red);

break;

}

}

/\*\*

\* 设置Snackbar背景颜色

\* @param snackbar

\* @param backgroundColor

\*/public static void setSnackbarColor(Snackbar snackbar, int backgroundColor) {

View view = snackbar.getView();

if(view!=null){

view.setBackgroundColor(backgroundColor);

}

}

/\*\*

\* 设置Snackbar文字和背景颜色

\* @param snackbar

\* @param messageColor

\* @param backgroundColor

\*/public static void setSnackbarColor(Snackbar snackbar, int messageColor, int backgroundColor) {

View view = snackbar.getView();

if(view!=null){

view.setBackgroundColor(backgroundColor);

((TextView) view.findViewById(R.id.snackbar\_text)).setTextColor(messageColor);

}

}

/\*\*

\* 向Snackbar中添加view

\* @param snackbar

\* @param layoutId

\* @param index 新加布局在Snackbar中的位置

\*/public static void SnackbarAddView( Snackbar snackbar,int layoutId,int index) {

View snackbarview = snackbar.getView();

Snackbar.SnackbarLayout snackbarLayout=(Snackbar.SnackbarLayout)snackbarview;

View add\_view = LayoutInflater.from(snackbarview.getContext()).inflate(layoutId,null);

LinearLayout.LayoutParams p = new LinearLayout.LayoutParams( LinearLayout.LayoutParams.WRAP\_CONTENT,LinearLayout.LayoutParams.WRAP\_CONTENT);

p.gravity= Gravity.CENTER\_VERTICAL;

snackbarLayout.addView(add\_view,index,p);

}

}

简单的使用示例：

SnackbarUtil.ShortSnackbar(coordinator,"妹子向你发来一条消息",SnackbarUtil.Info).show();

整出幺蛾子的使用示例：

Snackbar snackbar= SnackbarUtil.ShortSnackbar(coordinator,"妹子删了你发出的消息",SnackbarUtil.Warning).setActionTextColor(Color.RED).setAction("再次发送", new View.OnClickListener() {

@Override

public void onClick(View v) {

SnackbarUtil.LongSnackbar(coordinator,"妹子已将你拉黑",SnackbarUtil.Alert).setActionTextColor(Color.WHITE).show();

}

});

SnackbarUtil.SnackbarAddView(snackbar,R.layout.snackbar\_addview,0);

SnackbarUtil.SnackbarAddView(snackbar,R.layout.snackbar\_addview2,2);

snackbar.show();

这个示例中调用了两次SnackbarAddView()方法向Snackbar中添加了两个不同的自定义布局，效果如下（不建议大家这么玩 \_(:з」∠)＿ ）：