

三星钻孔屏适配指导

1. 前言

钻孔屏 (Punch Hole), 即屏幕内相机开孔技术 (Hole In Display), 相比刘海屏, 可以带来保留窄边框的全面屏体验。三星此次突破手机屏幕工艺, 将钻孔屏作为 Galaxy A8s 上 “首次采用” 的技术, 而 Galaxy A8s 也将成为首个采用钻孔屏的机型。



在钻孔的区域, 要避免布局重要的控件、文本或其他内容, 对于安卓 P OS, 谷歌提供了统一的适配方案 (Cutout API); 然而由于 Galaxy A8s 上线版本为安卓 O OS, O OS 在市场中仍会存在很长一段时间, 因此强烈建议并感谢第三方合作伙伴按照下文方式适配三星 O OS 方案。

2. 分辨率信息

型号	分辨率	屏幕纵横比	Dpi
Galaxy A8s (SM-G8870)	1080*2340	19.5:9	403

3. 三星钻孔屏手机安卓 O OS 适配方法

3.1 如何识别钻孔屏

资源 "config_mainBuiltInDisplayCutout" 中若包含一个有内容的 cutout

spec , 可识别该型号为钻孔屏。

```
try {
    final Resources res = context.getResources();
    final int resId = res.getIdentifier("config_mainBuiltInDisplayCutout", "string", "android");
    final String spec = resId > 0 ? res.getString(resId): null;
    mHasDisplayCutout = spec != null && !TextUtils.isEmpty(spec);
} catch (Exception e) {
    Log.w(mLogTag, "Can not update hasDisplayCutout. " +
        e.toString());
}
```

3.2 如何获取钻孔屏区域

DisplayCutout API	
List<Rect> getBoundingRects()	返回 Rects 的列表 ,每个 Rects 都是屏幕上非功能区域的边界矩形。
int getSafeInsetBottom()	返回安全区域距离屏幕底部的距离，单位是 px
int getSafeInsetLeft ()	返回安全区域距离屏幕左边的距离，单位是 px
int getSafeInsetRight ()	返回安全区域距离屏幕右边的距离，单位是 px
int getSafeInsetTop ()	返回安全区域距离屏幕顶部的距离，单位是 px

Anroid8.1 可以通过代码反射调用 DisplayCutout 的主要 API

- 1) 通过 WindowInsets 的 getDisplayCutout 反射调用获得 DisplayCutout 的实例。

```
(Sample) Reflection 1
Method method = WindowInsets.class.getDeclaredMethod("getDisplayCutout");
Object displayCutoutInstance = method.invoke(windowInsets);
```

- 2) 通过实例做 getSafeInsetTop, getSafeInsetBottom,, getSafeInsetLeft,,

getSafeInsetRight, getBoundingRects 方法的反射

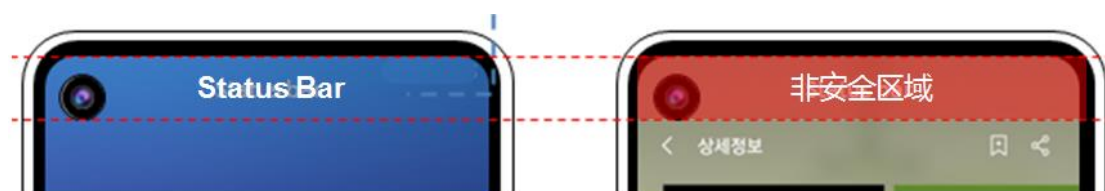
(Sample) Reflection 1

```
Rect safeInsets = new Rect();
List<Rect> boundingRects = new ArrayList<>();
Class cls = displayCutoutInstance.getClass();
int top = (int)cls.getDeclaredMethod("getSafeInsetTop").invoke
(displayCutoutInstance);
int bottom = (int)cls.getDeclaredMethod("getSafeInsetBottom").invoke
(displayCutoutInstance);
int left = (int)cls.getDeclaredMethod("getSafeInsetLeft").invoke
(displayCutoutInstance);
int right = (int)cls.getDeclaredMethod("getSafeInsetRight").invoke
(displayCutoutInstance);
safeInsets.set(left, top, right, bottom);
boundingRects.addAll((List<Rect>)cls.getDeclaredMethod("getBoundingRects").
invoke(displayCutoutInstance));
```

以上 DisplayCutout 是谷歌提供的 P OS (Android9.0) 对缺口屏的集成解决方案 , 对 P OS 以下版本并未提供。为了便于有更多需求的应用适配 , 三星将 DisplayCutout 方案应用到 Android 8.1。更多信息可参看 : [DisplayCutout_Android81_eng.pdf](#)

3.3 如何获取系统状态栏高度

钻孔屏手机中 , 系统 Status Bar 完全包含了钻孔区域 , 因此 , 可以直接获取状态栏的高度作为非安全区域的高度 , 在进行页面设计和布局时 , 确保重要内容 (按键、菜单、文本等) 不要显示在非安全区域。



3.3.1 获取系统状态栏高度接口

```
public int getStatusBarHeight() {  
    int result = 0;  
    int resourceld = mApplicationContext.getResources().getIdentifier("status_bar_height",  
        "dimen", "android");  
    if (resourceld > 0) {  
        result = mApplicationContext.getResources().getDimensionPixelSize(resourceld);  
    }  
    TLog.d(TAG, "statusBarHeight : " + result);  
    return result;  
}
```

3.3.2 主要适配问题举例

1. 状态栏高度写死问题

没有从系统读取状态栏的高度值，而是直接使用固定值来显示高度，造成页面显示异常。



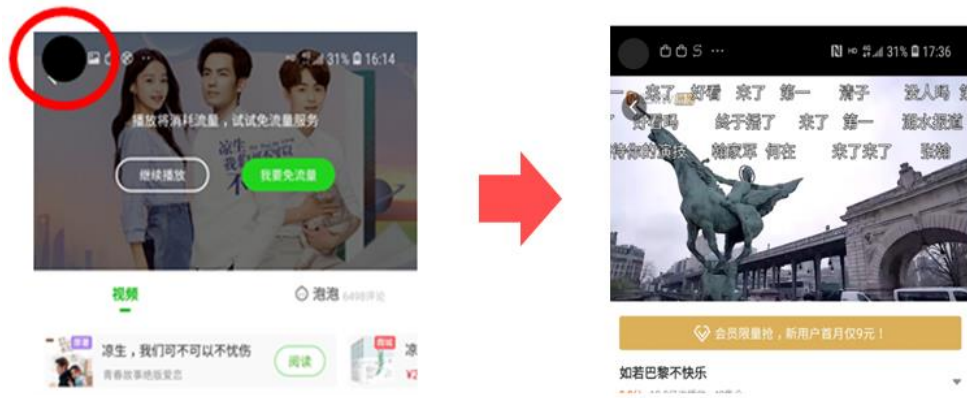
2. 沉浸式布局遮挡问题

页面不显示状态栏，导致重要内容被钻孔遮挡

两种方法：

1) 判断是否是钻孔屏，若是，将 window 更改为非全屏显示界面，即显示状态

栏。



2) 使用 `getStatusBarHeight()` 直接获取状态栏的高度，重要的元素布局在非安全区域外。



4. 三星钻孔屏手机安卓 P OS 适配方法

谷歌开放者网站称 P OS 版本为摄像头和扬声器预留的空间为屏幕缺口或凹口，上文提到的钻孔屏也属于缺口屏的一种，本节统一用缺口描述。

以下内容整理自谷歌开发者网站，详情请参看：

<https://developer.android.com/guide/topics/display-cutout/>:

1. 一个类

- Android9 支持最新的全面屏，其中包含为摄像头和扬声器预留空间的屏幕缺口（或凹口）。通过 [DisplayCutout](#) 类可确定非功能区域的位置和形状，这些区域不应显示内容。要确定这些屏幕缺口区域是否存在及其位置，请使用 [getDisplayCutout\(\)](#)函数。

2. 三种布局模式

- 全新的窗口布局属性 [layoutInDisplayCutoutMode](#)，让您的应用可以为设备屏幕缺口周围的内容进行布局。有三种模式可选，您可以将此属性设为下列值之一：

1) [LAYOUT_IN_DISPLAY_CUTOUT_MODE_DEFAULT](#)

2) [LAYOUT_IN_DISPLAY_CUTOUT_MODE_SHORT_EDGES](#)

3) [LAYOUT_IN_DISPLAY_CUTOUT_MODE_NEVER](#)

对应的解释为：

- 1) 仅仅当系统提供的 bar 完全包含了缺口区时才允许 window 延伸到缺口区，否则 window 不会和屏幕缺口区重叠。

//在实际使用中，这意味着如果窗口没有设置 [FLAG_FULLSCREEN](#) 或

[View.SYSTEM_UI_FLAG_FULLSCREEN](#)（或

[View.SYSTEM_UI_FLAG_HIDE_NAVIGATION](#)），如果缺口在屏幕顶部（或底部），

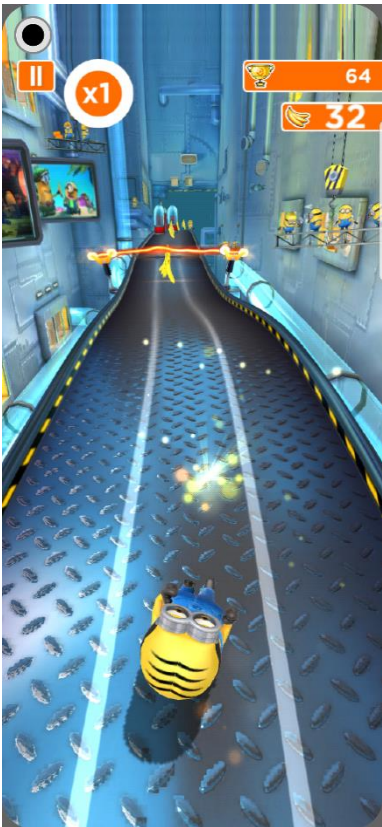
它可以延伸到竖屏的缺口区域。否则（即全屏或横屏）它不会和缺口区域重叠。

也就是说此模式在全屏下不允许使用缺口区域。

- 2) 允许 window 延伸到短边的缺口区。

//窗口必须确保没有重要元素和缺口区域重叠。另外，在这种模式下，无论窗口是

否隐藏了状态栏，在竖屏和横屏下都可以延伸到缺口区域。



3) 不允许 window 延伸到缺口区。

3. 新增主要接口介绍

3.1 如果有缺口区域返回位置

```
class WindowInsets {DisplayCutout getDisplayCutout();}
```

public DisplayCutout getDisplayCutout ()	
返回	
DisplayCutout	如果有缺口区域返回位置

3.2 缺口位置接口

```
class DisplayCutout {int getSafeInsetLeft();int getSafeInsetTop();int  
getSafeInsetRight();int getSafeInsetBottom();Region getBounds();}
```

public final class DisplayCutout	
方法	
List<Rect> getBoundingRects()	返回 Rects 的列表 ,每个 Rects 都是屏幕上非功能区域的边界矩形。
int getSafeInsetBottom()	返回安全区域距离屏幕底部的距离 , 单位是 px
int getSafeInsetLeft ()	返回安全区域距离屏幕左边的距离 , 单位是 px
int getSafeInsetRight ()	返回安全区域距离屏幕右边的距离 , 单位是 px
int getSafeInsetTop ()	返回安全区域距离屏幕顶部的距离 , 单位是 px

4. 提示

- 内容如果非放在危险区不可 , 尽量使用 SafeInsets
- 如需动态切换 fullscreen 模式 , 就用 SHORT_EDGES 或 NEVER 模式
- 如果 APP 没有特别使用 P 的 API :
 - Status bar 高度 >= 缺口区域高度
 - Fullscreen 或 横屏时 , 必须 letterbox
- 避免 Status Bar 硬代码 :



```
<dimen name="status_bar_size">
    25dp
</dimen>
```



```
WindowInsets
View.OnApplyWindowInsetsListener
```


- Windows v.s. Screen 坐标

- 当进入 letterbox 状态时：

- (0,0)的 screen 坐标 \neq (0,0)的 window 坐标

- MotionEvent.getRawX/Y()返回 screen 坐标

- 使用 getLocationOnScreen()转换到视图的坐标空间