



Unisoc Confidential For hiar

Android 10.0 壁纸客制化指导手册

文档版本
发布日期

V1.2
2020-04-27

版权所有 © 紫光展锐科技有限公司。保留一切权利。

本文件所含数据和信息都属于紫光展锐所有的机密信息，紫光展锐保留所有相关权利。本文件仅为信息参考之目的提供，不包含任何明示或默示的知识产权许可，也不表示有任何明示或默示的保证，包括但不限于满足任何特殊目的、不侵权或性能。当您接受这份文件时，即表示您同意本文件中内容和信息属于紫光展锐机密信息，且同意在未获得紫光展锐书面同意前，不使用或复制本文件的整体或部分，也不向任何其他方披露本文件内容。紫光展锐有权在未经事先通知的情况下，在任何时候对本文件做任何修改。紫光展锐对本文件所含数据和信息不做任何保证，在任何情况下，紫光展锐均不负责任任何与本文件相关的直接或间接的、任何伤害或损失。

请参照交付物中说明文档对紫光展锐交付物进行使用，任何人对紫光展锐交付物的修改、定制化或违反说明文档的指引对紫光展锐交付物进行使用造成的任何损失由其自行承担。紫光展锐交付物中的性能指标、测试结果和参数等，均为在紫光展锐内部研发和测试系统中获得的，仅供参考，若任何人需要对交付物进行商用或量产，需要结合自身的软硬件测试环境进行全面的测试和调试。非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

Unisoc Confidential For hiar

紫光展锐科技有限公司



前言

概述

本文档主要介绍了 UNISOC Android 10.0 平台 Wallpaper 应用的 UI 显示、关键类以及如何客制化壁纸资源和壁纸相关功能，概述了动态壁纸在 Go 和非 Go 版本的表现。

读者对象


本文档主要适用于所有使用 UNISOC Android 10.0 平台的开发和维护人员。

缩略语

缩略语	英文全名	中文解释
UI	User Interface	用户界面
AOSP	Android Open Source Project	安卓开放源代码项目
OOM	Out Of Memory	内存溢出
FWVGA	Full Wide Video Graphics Array	扩大的 WVGA 分辨率
HD	High Definition	高清
FHD	Full High Definition	全高清

符号约定

在本文中可能出现下列标志，它所代表的含义如下。

符号	说明
 说明	用于突出重要/关键信息、补充信息和小窍门等。 “说明”不是安全警示信息，不涉及人身、设备及环境伤害。

变更信息

文档版本	发布日期	修改说明
V1.0	2019-09-29	第一次正式发布。
V1.1	2019-12-26	适用产品信息增加 UIS8581E\SL8541E 平台。
V1.2	2020-01-16	更新模板；删除适用产品和适用版本信息；章节架构调整。

关键字

Wallpaper、壁纸。

Unisoc Confidential For hiar

目 录

1 概述	1
1.1 UI 新特性	1
1.2 核心类	6
1.3 其他常用类	7
1.4 小结	8
2 壁纸资源客制化	9
2.1 默认壁纸资源	9
2.1.1 默认宽屏壁纸	9
2.1.2 默认单屏壁纸	9
2.2 预置壁纸资源	10
2.2.1 预置宽屏壁纸	10
2.2.2 预置单屏壁纸	11
2.2.3 删除或增加预置壁纸	11
3 壁纸功能客制化	14
3.1 禁用锁屏壁纸功能	15
3.1.1 Android 10.0-Go(512M)版本开启锁屏壁纸功能	15
3.1.2 Android 10.0-Go(1G)版本关闭锁屏壁纸功能	16
3.1.3 非 Go 版本关闭锁屏壁纸功能	16
3.2 壁纸解码内存控制	17
3.3 隐藏默认壁纸功能	18
4 动态壁纸	19
5 参考文档	20

图目录

图 1-1 设备当前壁纸	2
图 1-2 壁纸类别界面	3
图 1-3 壁纸缩略图界面	4
图 1-4 壁纸预览界面	5
图 1-5 设置壁纸提示语	6

Unisoc Confidential For hiar

表目录

表 3-1 不同版本壁纸相关功能默认配置.....	14
---------------------------	----

Unisoc Confidential For hiar

1 概述

在 Android 中，壁纸分为静态壁纸与动态壁纸两种。静态壁纸是一张图片，而动态壁纸则以动画或视频为表现形式，或者可以对用户的操作作出反应。

从 Android 10.0 开始，Google 在 AOSP 代码中新增了一个全新的壁纸应用 WallpaperPicker2，路径如下：

/packages/apps/WallpaperPicker2

UNISOC 在 Android 10.0 上使用 WallpaperPicker2 替换掉了 Android 9.0 上的 WallpaperPicker。相比 WallpaperPicker，WallpaperPicker2 能够分类显示壁纸资源，提示壁纸设置成功与否，而且在 UI 显示上更加美观。

下面简单介绍 WallpaperPicker2 的 UI 新特性和一些关键类。

1.1 UI 新特性

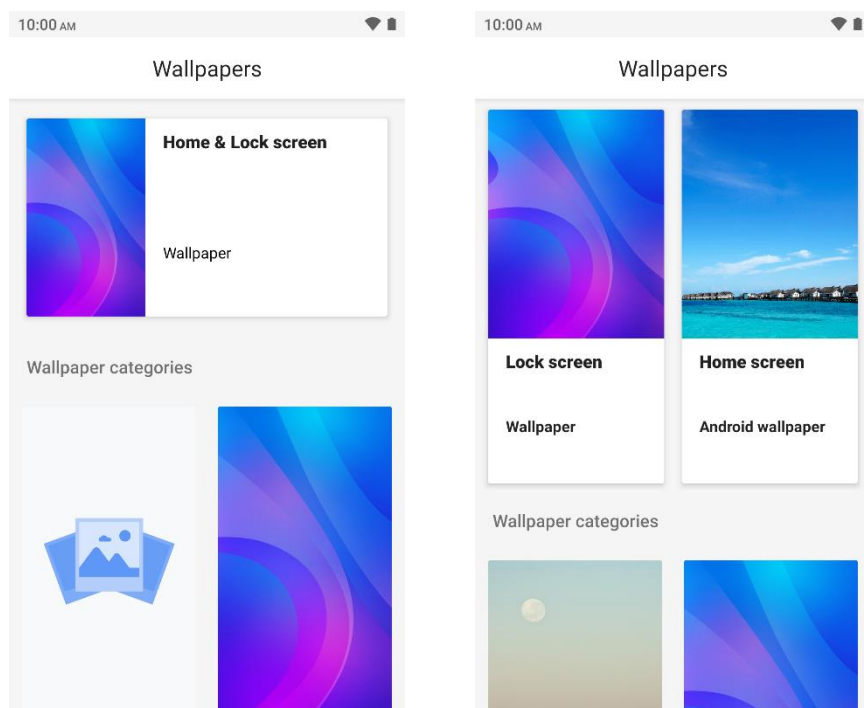
WallpaperPicker2 中有三种类型的界面：壁纸类别界面、壁纸缩略图界面以及壁纸预览界面。其中，壁纸缩略图界面是指某一类别壁纸的缩略图界面，比如设备上的壁纸缩略图界面。

不同于 WallpaperPicker，WallpaperPicker2 具有如下新特性：

- 显示当前壁纸信息

在壁纸类别界面，允许壁纸访问设备上的照片和媒体后，会在界面顶部显示系统当前的壁纸，如图 1-1 所示。左图表示设备当前主屏幕和锁定屏幕共用一张壁纸；右图中左边为锁定屏幕壁纸，右边为主屏幕壁纸。

图1-1 设备当前壁纸



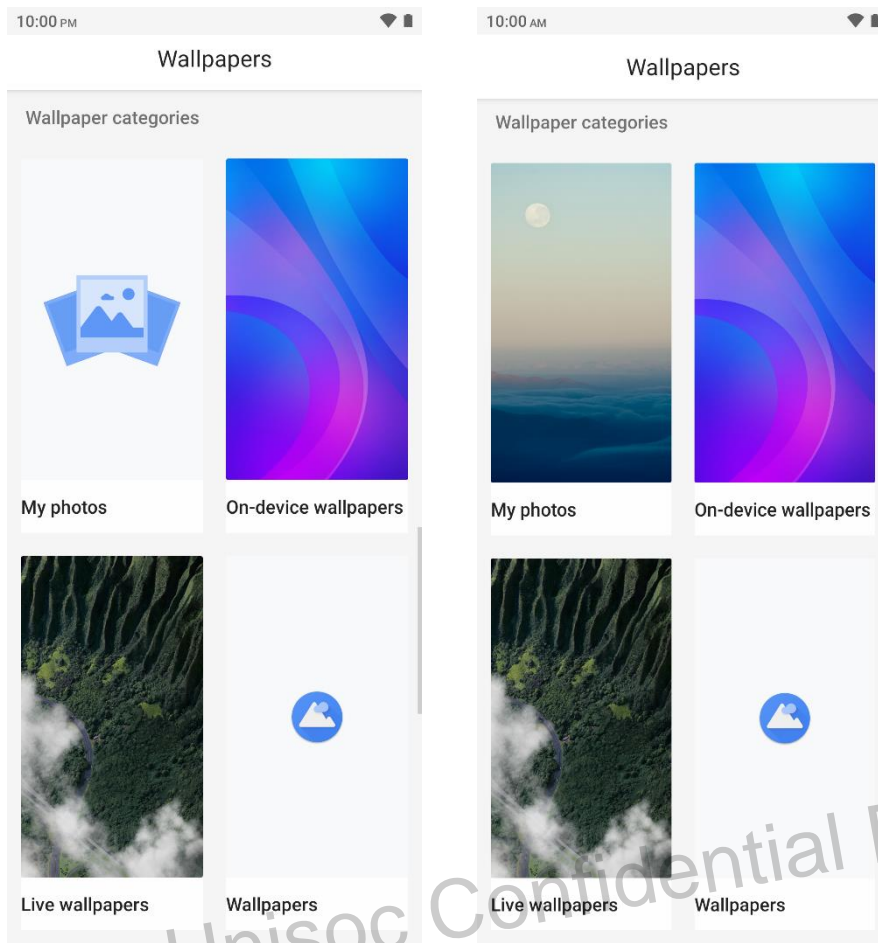
- 分类显示壁纸来源

壁纸类别界面将壁纸来源分为如下几类：

- 我的照片
进入文件管理器选取图片
- 设备上的壁纸
系统预置壁纸资源
- 动态壁纸
各种动态壁纸资源
- 壁纸
其他壁纸应用入口

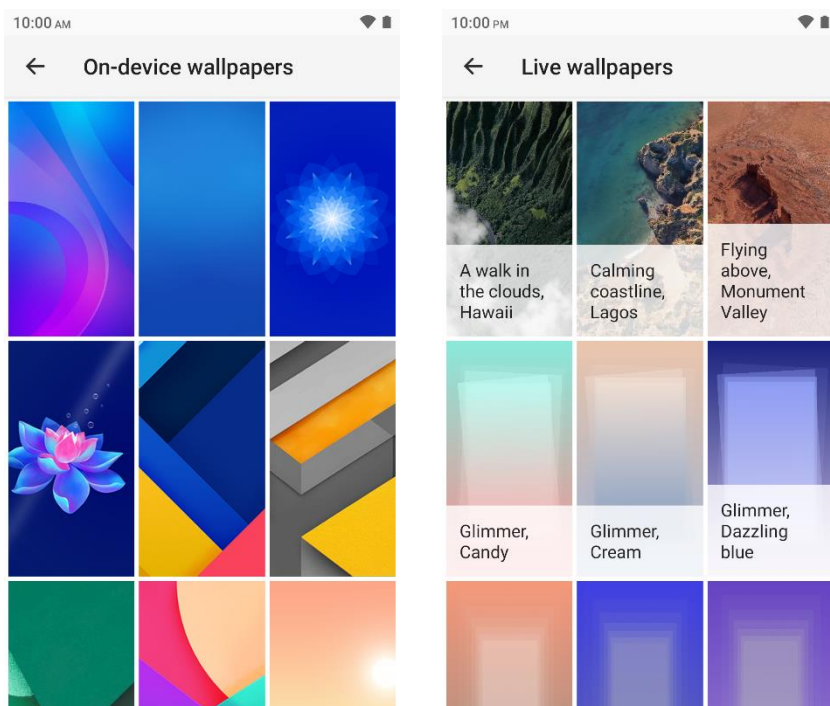
在赋予壁纸应用访问存储权限后，我的照片类别缩略图中会显示当前手机里日期最近的图片作为封面展示，具体显示如图 1-2 所示。

图1-2 壁纸类别界面



- 壁纸缩略图界面清晰、美观，且动态壁纸的缩略图底部可显示图片标题
此界面清晰、完整地罗列出对应类别的所有壁纸资源的缩略图，方便用户通过上下滑动查看并选择壁纸资源。具体显示如图 1-3 所示。

图1-3 壁纸缩略图界面



- 预览图片界面可查看图片的一些信息

在底部弹框中可查看当前预览图片的某些属性，不同来源的壁纸，显示内容不同。

- 我的照片

显示图片标题、作者、时间、设备型号等，如果以上均不存在，则显示我的照片。

- 设备上的壁纸

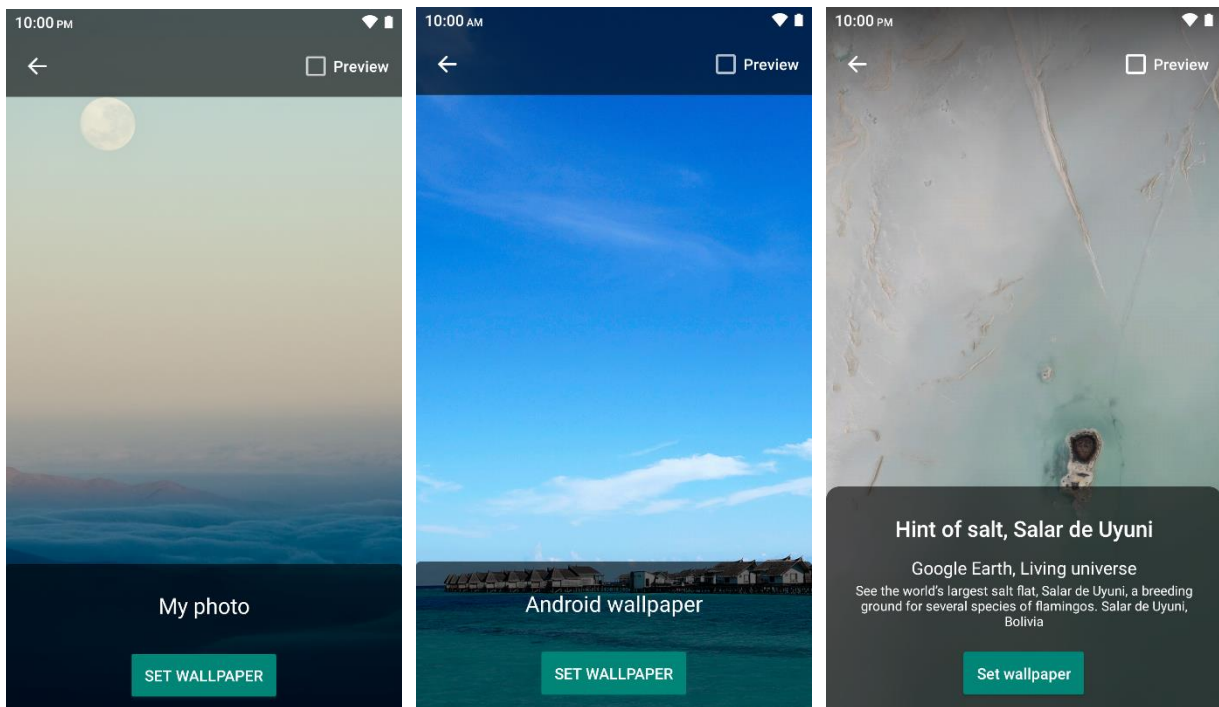
只有默认壁纸显示壁纸，其他统一显示安卓壁纸。

- 动态壁纸

显示动态壁纸的标题和作者。

具体显示如图 1-4 所示。

图1-4 壁纸预览界面

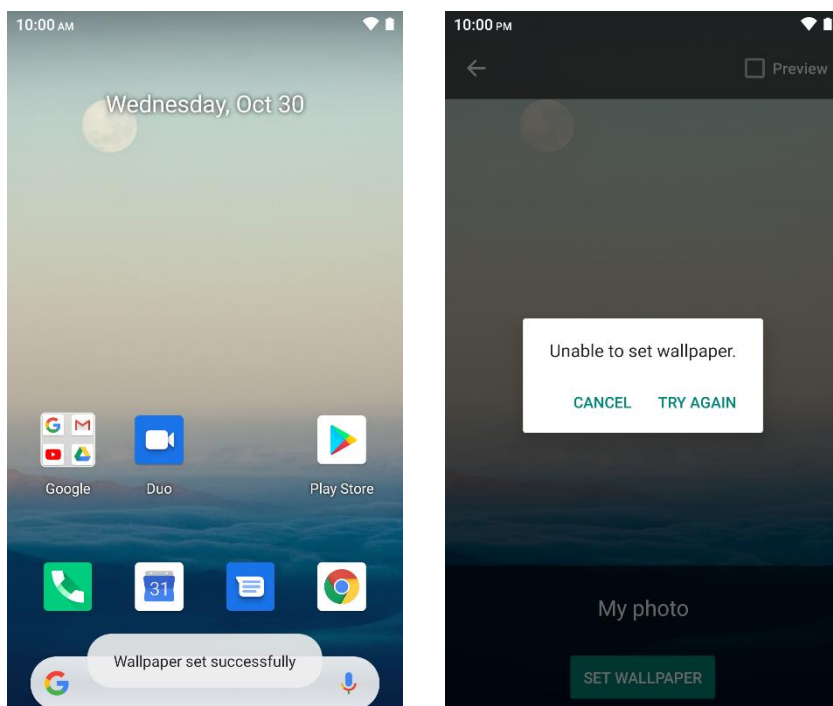


说明

可通过下拉底部弹框或者点击预览界面右上角的 checkbox 隐藏底部弹框，达到近乎全屏预览壁纸的效果。

- 壁纸设置结果提示语
壁纸设置完成后，以 **Toast** 形式提示用户壁纸设置成功。如设置失败，则提醒用户是否需要重新设置，优化了用户体验。具体表现如图 1-5 所示。

图1-5 设置壁纸提示语



1.2 核心类

- **TopLevelPickerActivity**
壁纸类别界面，默认与 CategoryFragment 绑定，用于显示当前系统壁纸和各类别壁纸缩略图。
- **CategoryFragment**
继承自 ToolbarFragment，在壁纸类别更新后通知 Adapter 更新 UI。此类定义了多个私有内部类，用于控制壁纸类别界面各部分的显示。
 - CategoryAdapter
负责将不同的壁纸类别数据与对应的 UI 绑定。
 - CategoryHolder
负责每个壁纸类别 Item 的 UI 显示。
 - SingleWallpaperMetadataHolder
无锁屏壁纸时的当前系统壁纸 Item。
 - TwoWallpapersMetadataHolder
当前系统的锁定屏幕壁纸和主屏幕壁纸的 Item。
 - PermissionNeededHolder
无 android.permission.READ_EXTERNAL_STORAGE 权限时，界面顶部的权限说明 Item。
- **WallpaperPickerDelegate**

TopLevelPickerActivity 的代理类，负责获取壁纸类别、申请权限、打开壁纸缩略图或壁纸预览界面等操作，是 TopLevelPickerActivity 与 DefaultCategoryProvider 以及其他壁纸界面之间通信的桥梁。

- **DefaultCategoryProvider**

壁纸类别提供者，启用一个 AsyncTask 后台获取所有的壁纸类别。

- **IndividualPickerFragment**

此为 IndividualPickerActivity 所加载的 fragment，显示某一壁纸类别的所有壁纸资源。资源 Item 显示的列数为 3 或者 4，与当前界面窗口宽度有关。

- **PreviewFragment**

壁纸预览和设置界面，是 StandalonePreviewActivity 和 PreviewActivity 的 content view，利用 SubsamplingScaleImageView 控件可以通过缩放、拖拽来预览壁纸。底部弹框中显示当前预览图片的来源和设置壁纸按钮。该类监听设置壁纸弹框 SetWallpaperDialogFragment 中各选项的 onClick 事件，从而触发设置壁纸的逻辑。

- **StandalonePreviewActivity**

从第三方应用（如图库）设置壁纸时，跳转到壁纸应用所显示的壁纸预览和设置界面。

- **PreviewActivity**

从壁纸应用内部选择一张图片设置壁纸时，显示的壁纸预览和设置界面。

1.3 其他常用类

- **WallpaperSetter**

负责壁纸设置的相关操作。

- **ImageWallpaperInfo**

继承自 WallpaperInfo，对外提供 URI 图片资源的解析方法、图片来源等属性以及图片所属壁纸类别等。

- **AppResourceWallpaperInfo**

继承自 WallpaperInfo，对外提供 Resource 图片资源的解析方法、图片来源等属性以及图片所属壁纸类别等。

- **DefaultWallpaperInfo**

继承自 WallpaperInfo，对外提供默认壁纸资源的解析方法、默认壁纸图片各种属性以及图片所属壁纸类别等。

- **ContentUriAsset**

继承自 StreamableAsset，与 ImageWallpaperInfo 对应，封装了 URI 图片的解析和裁剪以及 load 图片到指定的 view 等操作。

- **ResourceAsset**

继承自 StreamableAsset，与 AppResourceWallpaperInfo 对应，封装了 Resource 图片的解析、以及 load 图片到指定的 view 等操作。

- **BuiltInWallpaperAsset**

继承自 Asset，与 DefaultWallpaperInfo 对应，封装了默认壁纸的解析、以及 load 图片到指定的 view 等操作。

1.4 小结

本章简要介绍了 Android 10.0 上壁纸应用 WallpaperPicker2 的新特性和一些核心类，接下来的各章节将依次介绍静态壁纸的资源客制化、功能客制化以及动态壁纸功能。

Unisoc Confidential For hiar

2

壁纸资源客制化

静态壁纸是一张图片，利用 SystemUI 中的 ImageWallpaper 这个 service 显示在壁纸窗口中。而每个动态壁纸都对应一个自身的 WallpaperService。AOSP 代码中只提供了动态壁纸框架 LiveWallpapersPicker，并没有具体的动态壁纸服务，故这里的壁纸资源客制化主要是针对静态壁纸而言的。

按照壁纸的大小和表现，可将其分为单屏壁纸和宽屏壁纸。

- 单屏壁纸
壁纸的宽高与屏幕分辨率大小一致的壁纸资源，在设置为主屏幕壁纸后，不能跟随桌面上页面的滑动而左右滚动。
- 宽屏壁纸
宽高比大于屏幕宽高比的壁纸资源。在设置为主屏幕壁纸后，能够跟随桌面上页面的滑动而左右滚动。

通常会将壁纸宽度为 2 倍屏幕宽度，且壁纸高度等于屏幕高度的宽屏壁纸称为标准宽屏壁纸。本文提到的宽屏壁纸均是指标准宽屏壁纸。

说明

UNISOC Android 10.0 平台 Go 版本中的默认壁纸和预置壁纸均为单屏壁纸，在非 Go 版本中两者则为宽屏壁纸。

2.1 默认壁纸资源

默认壁纸是指设备首次开机时待机界面显示的背景壁纸，属于必选配置。默认壁纸资源路径为：
`/frameworks/base/core/res/res/drawable-nodpi/default_wallpaper.png`

单屏默认壁纸是通过 overlay 实现的，故在修改单屏默认壁纸时，需要先确定单屏默认壁纸的 overlay 路径。

2.1.1 默认宽屏壁纸

默认宽屏壁纸资源路径，即 Google 原生默认壁纸资源路径，如下：

`/frameworks/base/core/res/res/drawable-nodpi/default_wallpaper.png`

在客制化时直接将这里的 `default_wallpaper.png` 改为指定的宽屏壁纸图片即可（默认壁纸名称不能改变）。当然，考虑不同项目的兼容性，也可以通过 overlay 的方式指定项目的默认宽屏壁纸。

2.1.2 默认单屏壁纸

默认单屏壁纸资源路径为：

`/vendor/sprd/resource/wallpapers/XXX/overlay/frameworks/base/core/res/res/drawable-nodpi/default_wallpaper.png`

这里的 XXX 表示屏幕分辨率，目前有两种选择：FWVGA 和 HD，其下存放着对应分辨率大小的单屏壁纸资源，包括默认壁纸和系统预置壁纸。

FWVGA/HD 分辨率的默认单屏壁纸

直接修改/vendor/sprd/resource/wallpapers/XXX/overlay/frameworks/base/core/res/res/drawable-nodpi/default_wallpaper.png 路径中项目对应分辨率下的默认单屏壁纸资源图片即可。

其他分辨率的默认单屏壁纸

步骤 1 创建项目对应分辨率的单屏壁纸资源路径（以 FHD 为例）

/vendor/sprd/resource/wallpapers/FHD/overlay/frameworks/base/core/res/res/drawable-nodpi/

步骤 2 然后将默认单屏壁纸资源命名为 default_wallpaper.png，并拷贝到该路径下。

步骤 3 在项目的工程配置文件中配置壁纸的 overlay 路径

/device/sprd/project_name/board_name.mk

```
+ WPDIR := vendor/sprd/resource/wallpapers/FHD
- DEVICE_PACKAGE_OVERLAYS := $(BOARD_DIR)/overlay $(PLAT_DIR)/overlay
$(PLAT_COMM)/overlay
+ DEVICE_PACKAGE_OVERLAYS := $(BOARD_DIR)/overlay $(PLAT_DIR)/overlay
$(PLAT_COMM)/overlay $(WPDIR)/overlay
```

即在 DEVICE_PACKAGE_OVERLAYS 上添加了新的 overlay 路径(WPDIR)/overlay。

---结束

2.2 预置壁纸资源

本文中的预置壁纸资源是指预置到壁纸应用中的那些壁纸资源，默认位于/packages/apps/WallpaperPicker2/res/drawable-nodpi/

其中，单屏预置壁纸资源位于 vendor 下的 wallpaper overlay 路径下。

该项为可选配置，可根据项目需求自行定制系统预置壁纸的数量和资源。

2.2.1 预置宽屏壁纸

直接修改/packages/apps/WallpaperPicker2/res/drawable-nodpi/下的宽屏预置壁纸和缩略图资源即可。也可通过 overlay 方式为快速修改不同项目的预置壁纸资源。

说明

修改前后缩略图大小保持不变，预置壁纸资源和缩略图资源的名称保持不变。

2.2.2 预置单屏壁纸

单屏预置壁纸资源路径为：

/vendor/sprd/resource/wallpapers/XXX/overlay/packages/apps/WallpaperPicker2/res/drawable-nodpi/。
这里的 XXX 表示屏幕分辨率，目前 UNISOC 平台只对 FWVGA 和 HD 两种分辨率做了适配。

FWVGA/HD 分辨率的预置单屏壁纸

直接修改对应分辨率路径

/vendor/sprd/resource/wallpapers/XXX/overlay/packages/apps/WallpaperPicker2/res/drawable-nodpi/中的壁纸资源和缩略图即可。

其他分辨率的预置单屏壁纸

步骤 1 创建项目对应分辨率的单屏壁纸资源路径（以 FHD 为例）

```
/vendor/sprd/resource/wallpapers/FHD/overlay/packages/apps/WallpaperPicker2/res/drawable-nodpi/
```

步骤 2 将预置壁纸以及缩略图资源拷贝到该路径下。

步骤 3 在项目的工程配置文件中配置壁纸的 overlay 路径

```
/device/sprd/project_name/board_name.mk
```

```
+ WPDIR := vendor/sprd/resource/wallpapers/FHD
- DEVICE_PACKAGE_OVERLAYS := $(BOARDADDR)/overlay $(PLATDIR)/overlay
$(PLATCOMM)/overlay
+ DEVICE_PACKAGE_OVERLAYS := $(BOARDADDR)/overlay $(PLATDIR)/overlay
$(PLATCOMM)/overlay $(WPDIR)/overlay
```

即在 DEVICE_PACKAGE_OVERLAYS 上添加了新的 overlay 路径(WPDIR)/overlay。

----结束

2.2.3 删除或增加预置壁纸

预置壁纸属于壁纸类别中设备上的壁纸类，其加载是在 DefaultCategoryProvider 的静态内部类 FetchCategoriesTask 中实现的，加载的预置壁纸来源包括默认壁纸、Partner 中的预置壁纸以及 WallpaperPicker2 应用自身的预置壁纸。

在 WallpaperPicker2 中定义了一个 string-array 类型的资源代表所有的应用内预置壁纸：

/packages/apps/WallpaperPicker2/res/values-nodpi/wallpapers.xml

```
<resources>
  <string-array name="wallpapers" translatable="false">
    <item>wallpaper_00</item>
    <item>wallpaper_01</item>
    <item>wallpaper_02</item>
```

```
<item>wallpaper_03</item>
<item>wallpaper_04</item>
<item>wallpaper_05</item>
<item>wallpaper_06</item>
<item>wallpaper_07</item>
<item>wallpaper_08</item>
<item>wallpaper_09</item>
<item>wallpaper_10</item>
<item>wallpaper_11</item>
</string-array>
</resources>
```

上面代码中 item 项与 WallpaperPicker2 中预置壁纸资源名称一一对应。

删除预置壁纸

只需删除上面/packages/apps/WallpaperPicker2/res/values-nodpi/wallpapers.xml 中的对应 item 项即可。考虑到应用所占内存，可进一步删除对应的壁纸资源以节省内存。

📖 说明

壁纸资源可能位于壁纸的 overlay 路径下。

添加预置壁纸

步骤 1 在/packages/apps/WallpaperPicker2/res/values-nodpi/wallpapers.xml 中添加 item 项

```
<resources>
<string-array name="wallpapers" translatable="false">
    .....
    <item>wallpaper_10</item>
    <item>wallpaper_11</item>
    <item>wallpaper_new</item> <!--新增壁纸资源 -->
</string-array>
</resources>
```

步骤 2 定位到壁纸资源路径，可能为/packages/apps/WallpaperPicker2/res/drawable-nodpi/或者壁纸资源 overlay 路径，将名称为 wallpaper_new 的壁纸资源放到其中。

----结束

📖 说明

在添加预置壁纸时，强烈建议同时添加对应的壁纸缩略图资源，缩略图大小与平台已有的其他壁纸资源缩略图大小保持一致即可。

Unisoc Confidential For hiar

3 壁纸功能客制化

UNISOC 平台开发了禁用锁屏壁纸（锁屏界面无独立壁纸）和壁纸解码内存控制（限制壁纸应用加载的图片大小，防止 OOM）功能，两者均是利用 `prop` 属性控制的。其中，两个功能在不同版本的默认配置情况如表 3-1 所示。

表3-1 不同版本壁纸相关功能默认配置

功能	Android 10.0-Go (512M)	Android 10.0-Go (1G)	Android 10.0
禁用锁屏壁纸 <code>ro.lockwallpaper.enable</code>	Y	N	N
壁纸解码内存控制 <code>ro.wallpaper.mem.maxsize</code> (单位: MB)	64	96	256

从表 3-1 可以看出，禁用锁屏壁纸功能默认在 Android 10.0-Go(512M)版本中开启，即 Android 10.0-Go(512M)版本默认不支持单独设置锁屏壁纸，这样设计主要是出于节省系统常驻 RAM 考虑。

壁纸解码内存控制功能全版本开启，其中 Android 10.0-Go(512M)版本配置为 64M，Android 10.0-Go(1G)版本配置为 96M，其他版本默认为 256M。

当图片解码所消耗的 RAM size 超过上述阈值时，会对原图片下采样。该方案可能会稍微牺牲一点壁纸的清晰度，这种牺牲非常小，用户一般感受不到。

以上默认配置是在如下文件中完成的，可根据项目情况灵活调整。

/vendor/sprd/generic/misc/launchercfg/LauncherFeatures.mk

```
WALLPAPER_FEATURES ?=
```

```
#####config wallpaper begin#####
```

```
# config wallpaper features
```

```
ifeq ($(strip $(WALLPAPER_FEATURES)),)
```

```
    ifeq ($(strip $(PRODUCT_GO_DEVICE)),true)
```

```
        ifeq ($(strip $(CHIPRAM_DDR_CUSTOMIZE_SIZE)),0x20000000)
```

```
            WALLPAPER_FEATURES += \
```

```
                ro.lockwallpaper.enable=false \
```

```
                ro.wallpaper.mem.maxsize=64
```

```
        else
```

```

        WALLPAPER_FEATURES += \
            ro.wallpaper.mem.maxsize=96
    endif
endif
endif

ifneq ($(strip $(WALLPAPER_FEATURES)),)
    ADDITIONAL_BUILD_PROPERTIES += $(WALLPAPER_FEATURES)

    $(warning "config done, wallpaper features is: $(WALLPAPER_FEATURES)")
endif

```

以上是壁纸功能的默认配置。可在项目的工程配置文件中自定义需开启的壁纸功能，如下：

/device/sprd/project_name/board_name.mk

```

+ WALLPAPER_FEATURES += \
+     ro.lockwallpaper.enable=false \
+     ro.wallpaper.mem.maxsize=64

```

说明

一旦在项目工程配置文件中定义了 WALLPAPER_FEATURES，在默认配置文件 LauncherFeatures.mk 中的配置则不再生效，故需在工程配置文件中自定义所需开启/关闭的所有壁纸功能。

上面 ro.wallpaper.mem.maxsize 的取值建议根据项目 RAM 的大小自行调整，建议取值为 2 的 N 次幂，默认为 256，单位 M。

3.1 禁用锁屏壁纸功能

开启该功能后，不再显示锁屏壁纸，且设置壁纸时直接默认设置为主屏幕壁纸。目前，UNISOC 平台只在 Android 10.0-Go（512M）版本上默认开启该功能。

3.1.1 Android 10.0-Go(512M)版本开启锁屏壁纸功能

/vendor/sprd/generic/misc/launchercfg/LauncherFeatures.mk

```

--- a/launchercfg/LauncherFeatures.mk
+++ b/launchercfg/LauncherFeatures.mk

@@ -14,7 +14,7 @@ ifeq ($(strip $(WALLPAPER_FEATURES)),)

    ifeq ($(strip $(PRODUCT_GO_DEVICE)),true)

```

```
ifeq ($(strip $(CHIPRAM_DDR_CUSTOMIZE_SIZE)),0x20000000)

    WALLPAPER_FEATURES += \
-        ro.lockwallpaper.enable=false \
+        ro.lockwallpaper.enable=true \
        ro.wallpaper.mem.maxsize=64
else
    WALLPAPER_FEATURES += \
```

3.1.2 Android 10.0-Go(1G)版本关闭锁屏壁纸功能

/vendor/sprd/generic/misc/launchercfg/LauncherFeatures.mk

```
--- a/launchercfg/LauncherFeatures.mk
+++ b/launchercfg/LauncherFeatures.mk
@@ -19,6 +19,10 @@ ifeq ($(strip $(WALLPAPER_FEATURES)),)
    else
        WALLPAPER_FEATURES += \
            ro.wallpaper.mem.maxsize=96
+
+    ifeq ($(strip $(CHIPRAM_DDR_CUSTOMIZE_SIZE)),0x40000000)
+        WALLPAPER_FEATURES += \
+            ro.lockwallpaper.enable=false
+    endif
    endif
endif
endif
endif
```

3.1.3 非 Go 版本关闭锁屏壁纸功能

/vendor/sprd/generic/misc/launchercfg/LauncherFeatures.mk

```
--- a/launchercfg/LauncherFeatures.mk
+++ b/launchercfg/LauncherFeatures.mk
```

```
@@ -20,6 +20,9 @@ ifeq ($(strip $(WALLPAPER_FEATURES)),)
    WALLPAPER_FEATURES += \
        ro.wallpaper.mem.maxsize=96
endif
+ else
+     WALLPAPER_FEATURES += \
+         ro.lockwallpaper.enable=false
endif
endif
```

3.2 壁纸解码内存控制

在壁纸预览时，若选择预览的图片较大，会导致壁纸应用占用内存急剧增加，造成壁纸预览界面黑屏或者系统重启等问题。为了解决该问题，UNISOC 平台新增了壁纸解码内存控制功能，通过 **prop** 属性 **ro.wallpaper.mem.maxsize** 配置壁纸预览图片的最大可用内存，超出该阈值的图片则降低采样率来减少内存消耗。

ro.wallpaper.mem.maxsize 在不同版本的配置见表 3-1。其中，256M 为 JAVA 代码中定义的默认值，即如果版本中没有配置 **ro.wallpaper.mem.maxsize** 这个 **prop** 属性，则会采用 256M 作为默认阈值。如需客制化，请在 `/vendor/sprd/generic/misc/launchercfg/LauncherFeatures.mk` 中自定义：

```
# config wallpaper features
ifeq ($(strip $(WALLPAPER_FEATURES)),)
    ifeq ($(strip $(PRODUCT_GO_DEVICE)),true)
        ifeq ($(strip $(CHIPRAM_DDR_CUSTOMIZE_SIZE)),0x20000000)
            WALLPAPER_FEATURES += \
                ro.lockwallpaper.enable=false \
                ro.wallpaper.mem.maxsize=64
        else
            WALLPAPER_FEATURES += \
                ro.wallpaper.mem.maxsize=96
        endif
    endif
endif
```


3.3 隐藏默认壁纸功能

隐藏“设备上的壁纸”类别中的默认壁纸的步骤如下：

步骤 1 添加 Log 确定壁纸的 Partner 应用

WallpaperPicker2/src/com/android/wallpaper/module/DefaultPartnerProvider.java

```
public DefaultPartnerProvider(Context ctx) {
    .....
} else {
    mPackageName = null;
    mResources = null;
}
+ Log.d("DefaultPartnerProvider", "mPackageName = " + mPackageName);
}
```

步骤 2 如果 Partner 存在，则只需在 Partner 所在应用的 config.xml 中添加

```
+ <bool name="default_wallpapper_hidden">true</bool>
```

反之，请按如下修改

WallpaperPicker2/src/com/android/wallpaper/module/DefaultPartnerProvider.java

```
public boolean shouldHideDefaultWallpaper() {
    .....
    if (res != null) {
        final int resId = res.getIdentifier(
            RES_DEFAULT_WALLPAPER_HIDDEN, /* defType */ "bool", mPackageName);
        return resId != 0 && res.getBoolean(resId);
    }
- return false;
+ return true;
}
```

----结束

4

动态壁纸

通常第三方动态壁纸应用只包含动态壁纸的实现，其预览和设置是利用系统的 LiveWallpapersPicker 完成的。LiveWallpapersPicker 是预览和设置动态壁纸的框架，位于/packages/wallpapers/LivePicker/。

Google 原生从 Android 9.0 开始，Go 版本默认不支持动态壁纸功能。

根据 Google 发布的《Android Q-Go Device Configuration Guide》文档要求，Android 10.0-Go 版本禁用动态壁纸功能。因此，在 Go 版本中，即使安装了动态壁纸应用也是无法设置动态壁纸的。可根据如下代码判断系统是否支持动态壁纸功能。

```
final PackageManager pm = mContext.getPackageManager();  
  
pm.hasSystemFeature(PackageManager.FEATURE_LIVE_WALLPAPER);
```

Unisoc Confidential For hiar

5

参考文档

1. 《Android Q-Go Device Configuration Guide》

Unisoc Confidential For hiar