Mocor5 GridHome 客制化文档

版本： V1.0

日期：2018-05-31

声明

本文件所含数据和信息都属于紫光展锐机密及紫光展锐财产，紫光展锐保留所有相关权利。当您接受这份文件时，即表示您同意此份文件内含机密信息，且同意在未获得紫光展锐同意前，不使用或复制、整个或部分文件。紫光展锐有权在未经事先通知的情况下，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何与文件相关的直接或间接的、任何伤害或损失。

前　言

文档说明

本文档描述了如何对 Mocor5 中桌面应用 SimpleHome 的 GridHome 模式进行客制化，希望读者通过本文档，在对 GridHome 进行客制化的时候有一定的帮助。

阅读对象

本文档适合基于 Mocor5 项目的桌面应用进行开发、维护以及客制化的人员参考。

内容介绍

本文档包括三个章节，分别为：

- 第一章：概述。简单介绍 GridHome；
- 第二章：布局客制化。主要介绍 GridHome 桌面布局和如何客制化；
- 第三章：功能客制化。主要介绍 GridHome 一些功能的客制化；

文档约定

本文档采用下面醒目标志来表示在操作过程中应该特别注意的地方。

注意：

提醒操作中应注意的事项。

说明：

说明比较重要的事项。

相关文档

Mocor5 SimpleHome 框架介绍

Mocor5 ClassicHome 客制化文档

目 录

第 1 章 概述 1

第 2 章 布局客制化 2

2.1 桌面布局 2

2.1.1 布局介绍 2

2.1.2 调整布局 2

2.1.3 图标预置 2

2.1.4 时钟控件 3

2.1.5 农历 4

2.2 如何增删页面 5

2.2.1 增加页面 5

2.2.2 删除页面 6

2.3 亲情号 7

2.3.1 配置亲情号背景颜色 7

2.4 工具和应用文件夹 8

2.4.1 工具 8

2.4.2 应用文件夹 9

第 3 章 功能客制化 11

3.1 未读功能 11

3.1.1 如何添加未读功能 11

3.2 一键加速 13

3.2.1 调整一键加速位置 14

3.3 按键功能 14

3.4 手电筒 15

3.5 语音播报 15

图目录

图 1-1 ClassicHome 1
图 1-2 GridHome 1
图 2-1 桌面布局 2
图 2-3 亲情号 7
图 2-4 工具 8
图 3-1 未读显示 11
图 3-2 一键加速 13

第1章 概述

我司的 SimpleHome 桌面是一款针对 Mocor5 项目在 feature phone 上开发的 launcher。

该应用内部包含了两种不同的 UI 形态，分别称为 ClassicHome（图 1-1）和 GridHome（图 1-2）。可以根据项目需要自行选择终端显示的 UI 模式。另外，在桌面中可以通过长按某些按键达到快速打开某些界面的功能。

图 1-1 ClassicHome

图 1-2 GridHome

下面详细介绍 GridHome 应用的布局和功能客制化方法。

第2章 布局客制化

2.1 桌面布局

2.1.1 布局介绍

如图 2-1 所示，GridHome 的基本桌面布局。目前是采用 ViewPager + Fragment 架构，每一个页面对应一个 Fragment，每个 Fragment 内预置一个 GridView 控件来显示图标。

图 2-1 桌面布局

2.1.2 调整布局

目前每页是 2×3 的布局，如果需要调整为其他布局例如 3×3，请定位到如下文件：

SimpleHome\gridhome\res\values\config.xml

<integer name="grid_view_column_num">3</integer>//每行个数

<integer name="grid_view_max_items">9</integer> //最大个数

2.1.3 图标预置

桌面图标依照一定顺序和页面进行排列显示，需要配置文件中进行预置。

图标大小请修改如下配置，大小请不要超过背景 item 的宽高。

SimpleHome\gridhome\res\values\dimens.xml

<dimen name="app_image_width">48dp</dimen>

<dimen name="app_image_height">48dp</dimen>

1) 图标位置、背景是在如下配置文件中进行配置。

SimpleHome\gridhome\res\value\arrays.xml

以**联系人**为例：

```
<string-array name="contacts">
    <item>@string/contacts_package</item>//包名
    <item>@drawable/app_contacts</item>//定制的图标
    <item>@string/contacts_activity</item>//类名
    <item>@color/contacts_bg_color</item>//背景颜色
</string-array>
```

对应图标位置是放在第二页第一个位置：

```
<integer-array name="default_page_app_names">//页面数组
    <item>@array/contacts</item>//按数组顺序排列
    <item>@array/camera</item>
    <item>@array/phone</item>
    <item>@array/message</item>
</integer-array>
```

注意：如果某个页面应用个数少于最大个数，就会在该页面末位产生空位。

请尽量保证每页图标达到最大个数。

2.1.4 时钟控件

如图 2-2 所示，时钟控件在默认页显示，主要是显示系统当前时间和农历信息。

图 2-2 时钟控件

具体布局是在

SimpleHome\gridhome\res\layout\fragment_pager.xml

```
<LinearLayout
    android:id="@+id/default_clock_view"
    ......
    android:visibility="gone">
    <LinearLayout
        ........
        android:orientation="vertical">
        <RelativeLayout
            ......
            android:orientation="horizontal">
            <TextClock
                android:id="@+id/clock_view"//时间显示
                ......
                android:textColor="@color/clock_text_color"  />
            <TextClock
                ......
                android:textColor="@color/clock_text_color"  />
        </RelativeLayout>
        <TextClock
            android:id="@+id/format_date_view"//日期显示
            ......
            android:textColor="@color/clock_text_color"  />
        <TextView
            android:id="@+id/format_lunar_date_view"//农历显示，默认是隐藏的
            .......
            android:visibility="gone"/>
    </LinearLayout>
    ......
</LinearLayout>
```

2.1.5 农历

农历默认显示在时钟的下面，具体如图 2-2 所示。

- 若项目需要支持桌面显示农历日期，只需在/device/sprd/路径下找到对应的.mk 工程文件，并添加如下属性即可：

  PRODUCT_PROPERTY_OVERRIDES += \

  ro.simplehome.lunar=true

  不配置 ro.simplehome.lunar 属性，或者配置为 false 则均不支持显示农历日期。

- 更新农历的显示格式：

  请在如下函数中通过工具类 LunarCalendarConvertUtil 获取年月日后进行组合显示。

  gridhome\src\com\sprd\simple\fragment\DefaultWorkspaceFragment.java

```
private void updateLunarDateView(){
    Time time = new Time();
    time.set(System.currentTimeMillis());
    String lunarStr = LunarCalendarConvertUtil.buildLunarMonthDay(time, mContext)
                + "   " + LunarCalendarConvertUtil.bulidLunarYear(time, mContext);
    if(mLunarDateView != null) {
        mLunarDateView.setText(lunarStr);
    }
}
```

注意：

由于农历属于中国特有的一种历法， 因此，目前只有在中文简体和中文繁体下才支持该功能。

2.2 如何增删页面

当桌面空间放不下所有预置图标的时候，需要增加页面。

如 2.1.1 布局介绍，GridHome 设计框架是 ViewPager+Fragment.java 模式，每一个页面是一个 Fragment.java，如果需要增加、删除页面，就需要添加或者删除一个 Fragment。

2.2.1 增加页面

目前桌面待机是 4 页，以增加第 5 页为例：

1、 配置页面应用图标信息

SimpleHome\gridhome\res\value\arrays.xml

<integer-array name="five_page_app_names">

```
            <item>@array/call_log</item>
            <item>@array/calendar</item>
            <item>@array/gallery</item>
            <item>@array/fm</item>
            <item>@array/clock</item>
            <item>@array/settings</item>
        </integer-array>
```

2、 然后参考 ThirdPageAdapter.java、ThirdWorkspaceFragment.java 实现如下文件：

gridhome\src\com\sprd\simple\fragment\FiveWorkspaceFragment.java

gridhome\src\com\sprd\simple\adapter\ThirdPageAdapter.java

3、 最后在 Launcher.java 中把新增 Fragment 添加到 FragmentManger 中。

gridhome\src\com\sprd\simple\launcher：

```
public static final int sFIVE_WORKSPACE = 4;
```

然后在如下方法中中添加到 FragmentManager

```
public ArrayList<LauncherFragment> getFragments(int viewId, FragmentManager fm) {
        ……
        fragments.add(fiveWorkspaceFragment);
        return fragments;
}
```

4、 在获取应用文件夹的数据时候，需要过滤新增页面的应用数据。

```
public static Set<String> getDesktopSupportApps(Context context){
        Set<String> packageNameSet = new HashSet<String>();

        packageNameSet.addAll(getWorkspaceScreenApps(context,Launcher.sDEFAULT_WORK
        SPACE));
        ……

        packageNameSet.addAll(getWorkspaceScreenApps(context,Launcher.sFIVE_WORKSPA
        CE));
        return packageNameSet;
}
```

2.2.2 删除页面

删除页面与增加页面步骤相反。

1、 删除 Fragment、adapter 和布局文件。

2、 在 Launcher.java 中去除对应对象的使用。

3、 在获取应用文件夹内数据的方法中去除对删除页面数据的过滤。

2.3 亲情号

如图 2-3 所示，亲情号码页面是待机第一个页面，该页面是方便用户添加常用的亲情号码，便于用户快速拨打电话。

图 2-3 亲情号

该界面基本功能如下：

1、 按 OK 键，会进入联系人界面，然后添加联系人。

2、 选中已经设置过的亲情号，按 OK 键，会直接拨打该亲情电话。

3、 长按 OK 键，选中已经设置过的亲情号，可以删除该联系人，也可以进行重新编辑。

2.3.1 配置亲情号背景颜色

如上图 2-3，每一个亲情号码背景颜色不一致。

修改背景颜色，请定位到

packages\apps\SimpleHome\gridhome\res\value\arrays.xml

```
<string-array name="family_pic_bg">
    <item>@color/family_1_bg_color</item>
    <item>@color/family_2_bg_color</item>
    <item>@color/family_3_bg_color</item>
    <item>@color/family_4_bg_color</item>
```

```
        <item>@color/family_5_bg_color</item>
        <item>@color/family_6_bg_color</item>
    </string-array>
```

2.4 工具和应用文件夹

工具和应用文件夹都是 listview 布局，都是继承于

SimpleHome\gridhome\src\com\sprd\simple\launcher\BaseActivity.java

工具界面：

SimpleHome\gridhome\src\com\sprd\simple\launcher\ToolsActivity.java

应用文件夹界面

SimpleHome\gridhome\src\com\sprd\simple\launcher\AppFolderActivity.java

2.4.1 工具

如图 2-4 所示，工具界面是显示常用的一些工具软件，可以配置一些常用工具软件在工具界面显示。

图 2-4 工具

如何配置，请修改如下配置信息即可：

packages\apps\SimpleHome\gridhome\res\value\arrays.xml

```
    <array name="tool_apps">
        <item>com.android.sos</item>
```

```
        <item>com.sprd.firewall</item>
    <item>com.android.soundrecorder</item>

        <item>com.sprd.note</item>
        <item>com.sprd.appbackup</item>
        <item>com.sprd.fileexplorer</item>
        <item>com.sprd.sprdcalculator</item>
    </array>
```

说明：

tool_apps 是方便在获取应用文件夹界面数据的时候快速过滤工具中已显示的应用。

```
    <integer-array name="tools_page_app_names">
        <item>@array/sos</item>
        <item>@array/fire_waller</item>
        <item>@array/recorder</item>
        <item>@array/notebook</item>
        <item>@array/backup</item>
        <item>@array/file_manager</item>
        <item>@array/calculator</item>
    </integer-array>
```

tools_page_app_names 是工具界面显示的应用图标信息，与 tool_apps 是一一对应的。

2.4.2 应用文件夹

如图 2-5 所示，应用文件夹显示除待机界面和工具界面外剩余的应用，例如新预置、新安装的一些应用。

图 2-5 应用文件夹

应用文件夹界面获取对应的应用数据，需要过滤待机界面和工具界面的应用。

具体见如下方法：

packages\apps\SimpleHome\gridhome\src\com\sprd\simple\util\PackageInfoUtil.java

```
public static List<AppInfo> getLauncherAttrAppForAppFolder(Context context) {
    List<ResolveInfo> resolveInfos = getAllLauncherAttrApplication(context);

    // find all apps on desktop and tools, then add to set.
    Set<String> notNeedExistAppFolder = new HashSet<String>();
    notNeedExistAppFolder.addAll(getDesktopSupportApps(context));
    notNeedExistAppFolder.addAll(getToolApps(context));
    notNeedExistAppFolder.remove("com.xx.xxx");
    //如果某些应用不需要在应用文件夹显示，可以通过 remove 过滤。
    //或者也可以去除过对应的应用的 Launcher 属性
    // remove those apks which they do not need to show in appFolder.
    return getAppsByFilter(context, resolveInfos, notNeedExistAppFolder);
}
```

第3章 功能客制化

3.1 未读功能

如图标 3-1，目前支持在待机页面显示未接电话和未读短信，桌面拨号和通话记录图标会显示未读个数。

图 3-1 未读显示

该方案主要是通过监听短信、电话数据库来监听未读个数变化。

具体代码请参考：

SimpleHome\gridhome\src\com\sprd\simple\util\ MissCallContentObserver.java

SimpleHome\gridhome\src\com\sprd\simple\util\ UnreadMessageContentObserver.java

SimpleHome\gridhome\src\com\sprd\simple\util\ UnreadInfoUtil.java

3.1.1 如何添加未读功能

添加未读功能主要有如下两个步骤：

1、 需要在对应界面注册和反注册数据库监听器。

```
@Override
protected void registerContentObservers() {
    mMmsSmsObserver = new UnreadMessageContentObserver(mContext, mHandler);
    mContext.getContentResolver().registerContentObserver(
```

```
                                    UnreadInfoUtil.MMSSMS_CONTENT_URI,
                                    true, mMmsSmsObserver);
    }
    @Override
     protected void unregisterContentObservers() {
         mContext.getContentResolver().unregisterContentObserver(mMmsSmsObserver);
     }
```

2、 在监听到数据库中，通过查询数据库的未读个数来通知 UI 更新。

```
    private void asyncGetUnreadInfo() {
        new AsyncTask<Void, Void, Integer>() {
            @Override
            protected Integer doInBackground(Void... params) {
                return UnreadInfoUtil.getUnreadMessageCount(mContext);
            }

            @Override
            protected void onPostExecute(Integer unReadMessageCount) {
                ……
                mHandler.obtainMessage(UnreadInfoUtil.MMSSMS_UNREAD_MESSAGE,
                        unReadMessageCount).sendToTarget();
            }
        }.execute();
    }
```

3、 UI 界面进行更新。

```
    case UnreadInfoUtil.MMSSMS_UNREAD_MESSAGE:
        int messageCount = Integer.parseInt(String.valueOf(msg.obj));
        mAdapter.setUnCount(messageCount);
        Log.i(TAG, "Handler messageCount = " + messageCount);
        break;
```

查询未接电话和未读短信，有以下两个接口：

packages\apps\SimpleHome\gridhome\src\com\sprd\simple\util\ UnreadInfoUtil.java

public static int getUnreadMessageCount(Context context).

public static int getMissedCallCount(Context context){}

注意：详细步骤请参考待机第三页代码：

SimpleHome\gridhome\src\com\sprd\simple\fragment\ThirdWorkspaceFragment.java

3.2 一键加速

如图 3-2 所示，一键加速该功能是在待机桌面显示一个图标，点击该图标，会杀掉后台进程，进而释放内存。

图 3-2 一键加速

具体是在如下代码：

SimpleHome\gridhome\src\com\sprd\simple\fragment\ FourthWorkspaceFragment.java

1、 点击是否是一键加速的 item，如果是通过 handler 发消息启动杀进程的线程。

```
public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
    try {
        if (ICON_SPEED == position) {
            mSpeedLastClickTime = SystemClock.uptimeMillis();
            handler.removeMessages(CLEAN_BACKGROUND_PROCESS);
            handler.sendEmptyMessageDelayed(CLEAN_BACKGROUND_PROCESS,
                    DELAY_CLEAN_BACKGROUND_PROCESS_TIME);
        }
        ......
    }
```

```
        ......


    }
```

2、 启动杀进程的线程 SecurityThread。

```
    case CLEAN_BACKGROUND_PROCESS: {
        SecurityThread thread = new SecurityThread();
        thread.start();
        break;
    }
```

3、 SecurityThread 线程执行的是杀后台进程的操作。

4、 线程执行完，通知 UI 更新，并弹出 Toast，提示释放了几个进程和显示剩余内存。

```
    case STOP_ANIMATION: {
        int processCount = msg.arg1;
        int availableMemory = msg.arg2;
        img.clearAnimation();
        if(mAdapter instanceof FourthPageAdapter) {
            ((FourthPageAdapter) mAdapter).updateSpeedView();
        }
        Toast.makeText(getActivity(),
                getActivity().getResources().getString(
                R.string.clean_toast_prompt,
                processCount, availableMemory),
                Toast.LENGTH_LONG).show();
                break;
    }
```

3.2.1 调整一键加速位置

由于该功能不是一个独立应用，故该图标在预置的时候做了特殊处理。若需要调整该图标位置，需要把一键加速的整体逻辑跟随移动。

请尽量不要移动该图标位置，若确实需要调整，请提 CQ 进行处理。

3.3 按键功能

待机界面会对键盘进行按键事件处理，需要在 SimpleHome 中进行配置。

按键功能属于 SimpleHome 两种 UI 模式的公共部分，详情请参考《SimpleHome 框架介绍》文档。

可以统一在如下文件中进行配置：

SimpleHome\common\src\com\sprd\common\util\KeyCodeEventUtil.java

短按事件：

```
public static boolean pressKeyEventForMainActivity(Context context,
                                                   int keyCode, KeyEvent event) {
.....
}
```

长按事件：

```
public static boolean longPressKeyEventForMainActivity(Context context, int keyCode) {
.....
}
```

3.4 手电筒

由于手电筒功能受限于硬件，可以用闪光灯、专门的 LED 等、手机屏幕等来当做手电筒使用。

目前版本是采用兼容的方案，无论什么类型的手电筒，只需要在系统属性中配置对应节点即可。

详情请参考《simpleHome 框架介绍》文档。

3.5 语音播报

工具类 SimpleHome\common\src\com\sprd\common\util\TextToSpeechUtils.java 采用单例模式，完全集成了语音播报功能。对外只有一个成员方法：

```
public static void speak(String str, Context context)
```

其中，第一个参数为要播报的字符串内容，第二个参数为上下文。

因此，在需要语音播报的地方直接调用 TextToSpeechUtils.speak 方法，并传入播报内容和上下文即可。

说明：

TextToSpeechUtils 使用单例模式可以有效防止多语音同时播报的现象。