深圳市亿晟科技有限公司

API 开发文档

文档编号	YS-API-001	保密等级	I 级
作者	陈欢	最后修改日期	2020.05.29
审核人	卢健	最后审批日期	2020.06.02
批准人	李泉	最后批准日期	2020.06.04

修订记录

日期	版本	修订说明	修订人
2018.06.02	V1.0-20180602	初始版本	陈欢
2018.08.06	V2.0-20180806	1、增加网络部分，对以太网信息的获取 2、增加自动同步网络时间的状态获取	陈欢
2018.11.22	V2.1-20181122	增加背光是否打开和获取背光亮度	陈欢
2018.11.29	V2.2-20181129	增加休眠时间的控制	陈欢
2019.01.18	V3.0-20190118	增加获取 CPU 型号	陈欢
2019.02.28	V3.1-20190228	1、增加设置系统默认输入法 2、增加获取系统默认输入法	陈欢
2019.05.13	V3.2-20190513	增加设置系统当前默认语言	陈欢
2019.06.19	V3.3-20190619	增加获取系统 CPU 温度	陈欢
2019.06.28	V3.4-20190628	1、增加打开或关闭 adb 开关 2、增加替换开机动画	陈欢
2019.07.19	V3.5-20190719	增加设置默认 Launcher	陈欢
2020.01.13	V4.0-20200113	增加定时开关机	陈欢
2020.02.13	V4.1-20200213	增加 gpio 索引值控制	陈欢
2020.03.30	V4.2-20200330	1、增加 pppoe 拨号功能 2、增加升级固件设置是否弹窗的功能	陈欢
2020.05.21	V4.3-20200521	增加休眠唤醒方法	陈欢

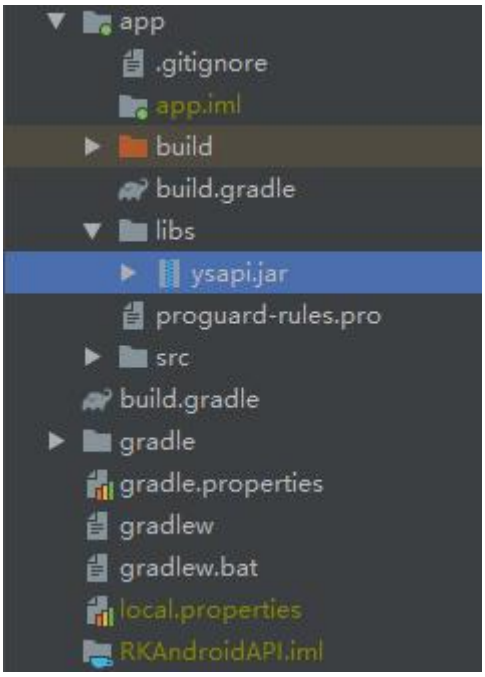
2020.05.29	V5.0-20200529	增加 3288 和 3399 9.0API	陈欢
2021.07.16	V5.1-20210716	1. 增加 3288 11.0 和 amlogicAPI 2. 增加开机自启功能 3. 增加守护进程功能 4. 增加安装白名单功能 5. 增加安装黑名单功能 6. Add the function of setting DPI	邹远航
2021.08.31	V5.2-20210831	1、增加厂商识别码的获取 2、增加 SN 号的获取 3、增加状态栏是否隐藏 4、增加隐藏显示状态栏 5、增加副屏截图 6、增加获取主屏幕类型 7、增加获取副屏幕类型 8、增加升级固件是否删除固件文件 9、增加静默卸载 10、增加是否设置了定时开关机 11、增加关机 12、增加打开或关闭网络 ADB 13、增加删除应用安装白名单	邹远航

		14、增加删除应用安装黑名单	

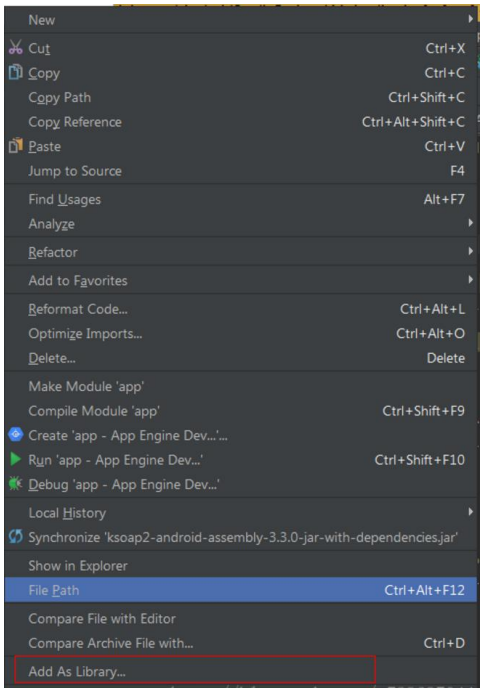
目录

第一章 如何在 Android Studio 中使用 API

(1) 将 ysapi.jar 复制到【工程目录\app\libs\】下；



(2) 右键点击 libs 文件夹中的 jar 文件选择 add as Library...然后选择 Model，这样也可以导入成功。



注意：所有的 api 调用

MyManager manager = MyManager.getInstance(this);

开始使用 API

首先要声明 MyManager 对象，然后就可以开始使用 API.如下面例子：

```
//声明 MyManager 对象
private MyManager manager;
manager = MyManager.getInstance(this);
//使用 API
```

调用 aidl 如不是点击事件应在回调钟调用

```
manager.setConnectClickInterface(new MyManager.ServiceConnectedInterface() {
            @Override
            public void onConnect() {
                manager.getApiVersion();
            }
        });
```

第二章　获取系统信息

2.1 获取 API 版本-日期信息

函数：public String getApiVersion ()

描述：获取目前 API 的版本-日期信息

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method getApiVersion()
     * @description  获取目前 API 平台-版本-日期信息，如果 API 发生修改就需要修改此处
     * @date    20180602
     * @author sky
     * @return  当前 API 的版本信息
    */
```

参数名/返回值	类型	说明	举例
返回值	String	版本-日期信息。	V1.0-20180602

范例：

```
MyManager manager = MyManager.getInstance(this);
Log.d(TAG,"API Version = " + manager.getApiVersion());
//输出 APIINFO: API Version = V1.0-20180602
```

2.2 获取设备的型号

函数：public String getAndroidModle()

描述：获取目前设备的型号

实现此接口的 API 版本：V1.0-20180602

```
/**
    * @method getAndroidModle()
    * @description 获取目前设备的型号，例如 rk3288、rk3399
    * @date   20180602
    * @author sky
    * @return 设备型号
   */
```

参数名/返回值	类型	说明	举例
返回值	String	设备的型号	Mstar Android TV

范例：

```
MyManager manager = MyManager.getInstance(this);
Toast.makeText(getApplicationContext(),"Model"+manager.getAndroidModle(),Toast.LENGTH_SHORT).show();
//输出  Model= rk3288
```

2.3 获取设备的系统版本

函数：public String getAndroidVersion()

描述：获取目前设备的 android 系统的版本

实现此接口的 API 版本：V1.0-20180602

```
/**
    * @method getAndroidVersion()
    * @description 获取目前设备的 android 系统的版本，例如 7.1 就返回 25
    * @date   20180602
```

```
     * @author sky
     * @return android 系统的版本
    */
```

参数名/返回值	类型	说明	举例
返回值	String	android 系统的版本	19

范例：

```
MyManager manager = MyManager.getInstance(this);
Toast.makeText(getApplicationContext(),"Version="+manager.getAndroidVersion(),Toast.LENGTH_SHORT).show();
//输出 Version=19
```

2.4 获取设备内存容量

函数：public String getRunningMemory()

描述：获取设备的硬件内存大小容量。

实现此接口的 API 版本：V1.0-20180602

```
 /**
     * @method getRunningMemory()
     * @description  获取设备的硬件内存大小容量
     * @date    20180602
     * @author sky
     * @return  内存大小
    */
```

参数名/返回值	类型	说明	举例
返回值	String	硬件内存大小，以 GB 为单位	2GB

范例：

MyManager manager = MyManager.getInstance(this);

Toast.makeText(getApplicationContext(),"RunningMemory"+manager.getRunningMemory(),Toast.LENGTH_SHORT).show();

//输出 RunningMemory2GB

2.5 获取设备存储容量

函数：public String getInternalStorageMemory ()

描述：获取设备的硬件内部存储大小容量

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method getInternalStorageMemory()
     * @description  获取设备的存储大小容量
     * @date 20180602
     * @author sky
     * @return  设备内部存储大小
    */
    public String getInternalStorageMemory();
```

参数名/返回值	类型	说明	举例
返回值	String	硬件内部存储大小，以 GB 为单位	16GB

范例：

MyManager manager = MyManager.getInstance(this);

Toast.makeText(getApplicationContext(),"InternalStorageMemory:"+manager.getRunningMemory(), Toast.LENGTH_SHORT).show();

//输出 InternalStorageMemory:16GB

2.6 获取设备固件版本

函数：public String getFirmwareVersion()

描述：获取设备的固件 SDK 版本。

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method getFirmwareVersion()
     * @description  获取设备的固件 SDK 版本。
     * @date    20180602
     * @author sky
     * @return  设备 SDK 版本
    */
```

参数名/返回值	类型	说明	举例
返回值	String	设备的固件版本	1.0

范例：

MyManager manager = MyManager.getInstance(this);

Toast.makeText(getApplicationContext(),"FW:"+manager.getFirmwareVersion(),Toast.LENGTH_SHORT).show();

//输出 FW:1.0

2.7 获取设备固件内核版本

函数：public String getKernelVersion()

描述：获取设备的固件内核版本

实现此接口的 API 版本：V1.0-20180802

```
/**
        * @method getKernelVersion()
        * @description  获取设备的固件内核版本
        * @date    20180802
        * @author sky
        * @return  内核版本
       */
```

参数名/返回值	类型	说明	举例
返回值	String	设备的固件内核版本	3.10.86

范例：

```
MyManager manager = MyManager.getInstance(this);
Log.e("SW kernel Version=” manager.getKernelVersion());
//输出 SW kernel Version =3.10.86
```

2.8 获取设备固件版本和编译日期

函数：public String getAndroidDisplay()

描述：获取设备的固件系统版本和编译日期

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method    getAndroidDisplay()
     * @description    获取固件版本号
     * @date    20180602
     * @author sky
     * @return    设备的固件版本号
    */
```

参数名/返回值	类型	说明	举例
返回值	String	设备的固件系统版本和编译日期	

范例：

MyManager manager = MyManager.getInstance(this);

Log.e("SWAndroidDisplay=" manager. getAndroidDisplay ());

//输出 SW AndroidDisplay= aosp_almond_dtmb-userdebug 6.0 MRA58K eng.ace.20180523.103817 test-keys

2.9 获取固件编译时间

函数：public String getFirmwareDate()

描述：获取固件编译的时间

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method getFirmwareDate()
     * @description 固件的编译时间
     * @date    20180602
     * @author sky
```

```
     * @return  固件编译的时间，比如 20180602
    */
```

参数名/返回值	类型	说明	举例
返回值	String	设备的固件编译日期	20180602

范例：

```
MyManager manager = MyManager.getInstance(this);
 Log.d("API 显示","getFirmwareDate(): "+ manager.getFirmwareDate());
//输出 API 显示：: getFirmwareDate(): 20180602
```

2.10 获取设备 CPU 型号

函数：public String getCPUType()

描述：获取设备 CPU 型号

实现此接口的 API 版本：V3.0-20190118

```
/**
     * @method getCPUType()
     * @description  获取设备 CPU 型号
     * @date    20190118
     * @author sky
     * @return  设备 CPU 型号
    */
```

参数名/返回值	类型	说明	举例
返回值	String	设备的 CPU 型号	

范例：

```
MyManager manager = MyManager.getInstance(this);
 Log.d("API 显示","设备 CPU 型号: "+ manager.getCPUType());
```

2.11 获取厂商识别码

函数：public String getVendorID()

描述：获取厂商识别码

实现此接口的 API 版本：V1.0-20180602

```
 /**
      * @method getVendorID()
      * @description  获取厂商识别码
      * @date    20180602
      * @author sky
      * @return  获取厂商识别码
     */
```

参数名/返回值	类型	说明	举例
返回值	String	返回厂商识别码	YSTA6212

范例：

```
MyManager manager = MyManager.getInstance(this);
ToastUtils.showToast(this, "厂商识别码=" + manager.getVendorID());
```

2.12 获取 SN 号

函数：public String getSn()

描述：获取 SN 号

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method getSn()
     * @description  获取 SN 号
     * @date 20180602
     * @author sky
     * @param
    */
```

参数名/返回值	类型	说明	举例
返回值	String	SN 号	2CE4WW1BKN

注意点：

范例：

```
MyManager manager = MyManager.getInstance(this);
ToastUtils.showToast(this, "sn=" + manager.getSn());
```

第三章 关机和重启

3.1 关机

函数： public void shutdown()

描述： 关机

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method shutdown()
     * @description 执行关机操作，走安卓标准关机流程
     * @date   20180602
     * @author sky
    */
```

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.shutdown();
```

3.2 重启

函数:：public void reboot()

描述： 重启

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method reboot()
     * @description 执行重启操作，走安卓标准重启流程
```

```
     * @date    20180602
     * @author sky
    */
    public void reboot();
```

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.reboot();
```

第四章 显示

4.1 截屏保存

函数：public boolean takeScreenshot(String path)

描述：截取当前全屏为图片并重命名到相应位置

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method takeScreenshot
     * @description 截屏，执行此方法可将系统当前显示画面截图保存在指定路径
     * @date   20180602
     * @author sky
     * @param path，保存图片的路径。例如/mnt/internal_sd/001.jpg
     * @return 是否截图成功，true 成功，false 失败
     *
    */
```

参数名/返回值	类型	说明	举例
参数名	String	保存绝对路径	“/mnt/internal_sd/001.jpg”
返回值	boolean	截图是否成功	True 成功保存截图 false 失败

注意点：

1、path 是否是可读写目录包括文件名。

2、使用此方法需要绑定 aidl 服务，在创建 MyManager 对象的时候绑定，方法如下：manager.bindAIDLService(this);

3、在 activity 销毁的时候需要调用 manager.unBindAIDLService(this);

范例：

//截屏并保存到"/mnt/internal_sd/001.jpg"

MyManager manager = MyManager.getInstance(this);

manager.takeScreenshot(Environment.getExternalStorageDirectory().getPath() +"/001.jpg");

4.2 设置屏幕旋转角度

函数：public void rotateScreen(Context context, String degree)

描述：设置屏幕顺时针旋转 N 角度

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method rotateScreen(Context context, String degree)
     * @description 旋转屏幕方向，0 度、90 度、180 度和 270 度
     * @date 20180602
     * @author sky
     * @param context，上下文对象。degree，旋转的角度（0/90/180/270）
    */
```

参数名/返回值	类型	说明	举例
degree	String	只支持 0 ， 90， 180， 270 四个角度。	"0" "90" "180" "270"

范例：

MyManager manager = MyManager.getInstance(this);

manager.rotateScreen(context,"90");

4.3 获取屏幕宽像素

函数：public int getDisplayWidth(Context context)

描述：获取显示屏分辨率宽 X 像素。

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method getDisplayWidth(Context context)
     * @description  获取屏幕宽
     * @date 20180602
     * @author sky
     * @param context，上下文对象
     * @return  屏幕宽度
    */
    public int getDisplayWidth(Context context)
```

参数名/返回值	类型	说明	举例
参数名	Context	上下文对象	
返回值	int	屏幕实际宽像素	

范例：MyManager manager = MyManager.getInstance(this);

manager.getDisplayWidth(context);

4.4 获取屏幕高像素

函数：public int getScreenHeight()

描述：获取显示屏分辨率高 Y 像素

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method getDisplayHeight(Context context)
     * @description  获取屏幕高
     * @date    20180602
     * @author sky
     * @param context，上下文对象
     * @return  屏幕高度
    */
```

参数名/返回值	类型	说明	举例
参数名	Context	上下文对象	
返回值	int	屏幕实际高像素	

范例：

```
//在 1080p 的屏获取分辨率
    MyManager    manager = MyManager.getInstance(this);
     int x = manager.getDisplayWidth(context);
     int y = manager.getDisplayHeight(context);
     Log.d("API 显示：","Width = "+x+" Height = "+y);
  //将输出 Width: Height= 1920:1080
```

4.5 导航栏状态设置

函数：public void hideNavBar(boolean hide)

描述：显示或隐藏导航栏

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method hideNavBar(boolean hide)
     * @description  设置显示或隐藏导航
     * @date    20180602
     * @author sky
     * @param hide，隐藏导航栏传入 true，显示传入 false
    */
```

参数名/返回值	类型	说明	举例
参数名	boolean	True：隐藏导航栏 False：显示导航栏	

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.hideNavBar(true);//隐藏导航栏的写法
```

4.6 导航栏状态查询

函数：public boolean getNavBarHideState()

描述：导航栏是否隐藏

实现此接口的 API 版本：V1.0-20180602

```
    /**
     * @method getNavBarHideState()
     * @description  获取导航栏的状态
     * @date    20180602
     * @author sky
     * @return  导航栏隐藏返回 true，显示则返回 false
```

```
*/
```

参数名/返回值	类型	说明	举例
返回值	boolean	true:隐藏， false:显示	

范例：

```
MyManager manager = MyManager.getInstance(this);
if(manager.getNavBarHideState()){
        Log.e("StatusBar hide.”);
}else{
        Log.e("StatusBar show.”);
}
```

4.7 设置滑出导航栏打开或关闭

函数：public void setSlideShowNavBar(boolean flag)

描述：打开或关闭滑出导航栏

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method setSlideShowNavBar(boolean flag)
     * @description 打开或关闭滑出导航栏
     * @date 20180602
     * @author sky
     * @param flag，打开滑出导航栏传入 true，关闭传入 false
    */
```

参数名/返回值	类型	说明	举例

参数名	boolean	True：打开滑出状态栏 False：关闭滑出状态栏	

范例：

MyManager manager = MyManager.getInstance(this);

manager.setSlideShowNavBar(true);//打开滑出导航栏

4.8 查询滑出导航栏是否打开

函数：public boolean isSlideShowNavBarOpen

描述：滑出导航栏是否打开

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method isSlideShowNavBarOpen()
     * @description  查询滑出导航栏选项是否打开
     * @date    20180602
     * @author sky
     * @return  滑出导航栏打开返回 true，关闭返回 false
    */
```

参数名/返回值	类型	说明	举例
返回值	boolean	true:打开， false:关闭	

范例：

```
MyManager manager = MyManager.getInstance(this);
if(manager.isSlideShowNavBarOpen){
        Log.e("SlideShowNavBar open.”);
```

```
}else{
        Log.e("SlideShowNavBar close”);
}
```

4.9 设置滑出通知栏打开或关闭

函数：public void setSlideShowNotificationBar(boolean enable)

描述：打开或关闭滑出通知栏

实现此接口的 API 版本：V1.0-20180602

```
 /**
      * @method setSlideShowNotificationBar(boolean flag)
      * @description  设置  打开或关闭下拉通知栏
      * @date 20180602
      * @author sky
      * @param flag，打开下拉通知栏传入 true，禁止下拉通知栏传入 false
     */
```

参数名/返回值	类型	说明	举例
参数名	boolean	True：打开滑出状态栏 False：关闭滑出状态栏	

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.setSlideShowNavBar(true);//打开下拉通知栏
```

4.10 查询滑出通知栏状态

函数：public boolean isSlideShowNotificationBarOpen()

描述：滑出通知栏是否打开

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method isSlideShowNotificationBarOpen()
     * @description 查询下拉通知栏是否打开
     * @date 20180602
     * @author sky
     * @return 下拉通知栏打开返回 true，否则返回 false
    */
```

参数名/返回值	类型	说明	举例
返回值	boolean	true:打开， false:关闭	

范例：

```
MyManager manager = MyManager.getInstance(this);
if(manager.isSlideShowNavBarOpen){
        Log.e("SlideShowNavBar open.”);
}else{
        Log.e("SlideShowNavBar close”);
}
```

4.11 设置屏幕亮度

函数：public void changeScreenLight(int value)

描述：调节屏幕亮度

实现此接口的 API 版本：V1.0-20180828

```
/**
     * @method changeScreenLight(int value)
     * @description  设置屏幕亮度
     * @date 20180828
     * @author sky
     * @param value，是按照 1 到 100 设置的，设置 100 即代表最大亮度，1 代表最小亮度
    */
```

参数名/返回值	类型	说明	举例
参数名	int	亮度的具体值（1--100）	50

注意点：参数值是按照 1 到 100 设置的，设置 100 即代表最大亮度，1 代表最小亮度

范例：manager.changeScreenLight(50);

4.12 关闭背光

函数：public void turnOffBacklight()

描述：熄灭屏幕，只关背光，却不进入休眠，软件继续运行。

实现此接口的 API 版本：V1.0-20180602

```
 /**
     * @method turnOffBackLight()
     * @description  关闭屏幕背光，仅仅是关闭背光，其他系统功能照旧
     * @date    20180602
     * @author sky
    */
```

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.turnOffBacklight();
```

4.13 开背光

函数：public void turnOnBacklight()

描述：开背光。

实现此接口的 API 版本：V1.0-20180602

```
 /**
      * @method turnOnBackLight()
      * @description 打开屏幕背光，跟 turnOffBackLight()方法相对应
      * @date   20180602
      * @author sky
      */
```

范例：manager.turnOnBacklight();

4.14 获取背光是否打开

函数：public boolean isBacklightOn()

描述：获取背光是否打开。

实现此接口的 API 版本：V2.1-20181122

```
  /**
      * @method isBacklightOn()
      * @description 获取背光是否打开
      * @date   20181122
      * @author sky
      * @return 当前背光是开返回 true，否则返回 false
     */
```

参数名/返回值	类型	说明	举例
返回值	boolean	True 表示背光打开 false 表示背光关闭	

范例：

MyManager manager = MyManager.getInstance(this);

manager.isBacklightOn();

4.15 获取背光亮度值

函数：public int getSystemBrightness()

描述：获取背光亮度值

实现此接口的 API 版本：V2.1-20181122

```
/**
     * @method getSystemBrightness()
     * @description 获取当前设备背光亮度
     * @date    20181122
     * @author sky
     * @return int，返回的亮度范围是 0-100
    */
```

参数名/返回值	类型	说明	举例
返回值	int	取到的亮度范围是从 0 到 100	

范例：

MyManager manager = MyManager.getInstance(this);

manager.getSystemBrightness();

4.16 关闭 HDMI 输出

函数：public void turnOffHDMI()

描述：关闭 HDMI 输出信号

实现此接口的 API 版本：V1.0-20180602

```
/**
    * @method turnOffHDMI()
    * @description 关闭 HDMI 输出，仅仅关闭信号输出，其他系统功能不影响
    * @date 20180602
    * @author sky
    */
```

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.turnOffBacklight();
```

4.17 打开 HDMI 输出

函数：public void turnOnHDMI()

描述：打开 HDMI 输出信号

实现此接口的 API 版本：V1.0-20180602

```
/**
    * @method turnOnHDMI()
    * @description 打开 HDMI 输出，与 turnOffHDMI()相对应
    * @date 20180602
    * @author sky
   */
```

范例：

MyManager manager = MyManager.getInstance(this);

manager.turnOnBacklight();

4.18 状态栏状态设置

函数：public void hideStatusBar(boolean hide)

描述：显示或隐藏状态栏

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method hideStatusBar(boolean hide)
     * @description  设置显示或隐藏状态栏
     * @date    2021/1/12
     * @author sky
     * @param hide，隐藏状态栏传入 true，显示传入 false
    */
```

参数名/返回值	类型	说明	举例
参数名	boolean	True：隐藏状态栏 False：显示状态栏	

范例：

MyManager manager = MyManager.getInstance(this);

manager.hideStatusBar(true);//隐藏状态栏的写法

4.19 状态栏状态查询

函数：public boolean getStatusBar()

描述：状态栏是否隐藏

实现此接口的 API 版本：2021/1/12

```
/**
    * @method getStatusBar()
    * @description  获取状态栏的状态
    * @date    20180602
    * @author sky
    * @return  状态栏隐藏返回 true，显示则返回 false
   */
```

参数名/返回值	类型	说明	举例
返回值	boolean	true:隐藏， false:显示	

范例：

```
MyManager manager = MyManager.getInstance(this);
if(manager.getStatusBar()){
        Log.e("StatusBar hide.”);
}else{
        Log.e("StatusBar show.”);
}
```

4.20 副屏截图

函数：public boolean viceScreenshot(String path)

描述：截取当前全屏为图片并重命名到相应位置

实现此接口的 API 版本：V1.0-20180602

```
/**
    * @method viceScreenshot
```

```
     * @description  截屏，执行此方法可将系统当前显示画面截图保存在指定路径
     * @date    20180602
     * @author sky
     * @param path，保存图片的路径。例如/mnt/internal_sd/001.jpg
     * @return  是否截图成功，true 成功，false 失败
     *
    */
```

参数名/返回值	类型	说明	举例
参数名	String	保存绝对路径	“/mnt/internal_sd/001.jpg”
返回值	boolean	截图是否成功	True 成功保存截图 false 失败

注意点：

1、path 是否是可读写目录包括文件名。

2、使用此方法需要绑定 aidl 服务，在创建 MyManager 对象的时候绑定，方法如下：manager.bindAIDLService(this);

3、在 activity 销毁的时候需要调用 manager.unBindAIDLService(this);

范例：

```
//截屏并保存到"/mnt/internal_sd/001.jpg"
MyManager manager = MyManager.getInstance(this);
manager.viceScreenshot(Environment.getExternalStorageDirectory().getPath() +"/001.jpg");
```

4.21 主屏幕类型

函数：public String getHomeScreenType()

描述：主屏幕类型

实现此接口的 API 版本：2021/1/12

```
    /**
     * @method getHomeScreenType()
```

```
     * @description  主屏幕类型
     * @date    20180602
     * @author sky
     * @return  主屏幕类型
    */
```

参数名/返回值	类型	说明	举例
返回值	String	主屏幕类型	HDMI

范例：

```
MyManager manager = MyManager.getInstance(this);
ToastUtils.showToast(this, "主屏幕类型为:" + manager.getHomeScreenType());
```

4.22 副屏幕类型

函数：public String getSecondaryScreenType()

描述：副屏幕类型

实现此接口的 API 版本：2021/1/12

```
    /**
     * @method getSecondaryScreenType()
     * @description  副屏幕类型
     * @date    20180602
     * @author sky
     * @return  副屏幕类型
    */
```

参数名/返回值	类型	说明	举例

返回值	String	副屏幕类型	HDMI

范例：

MyManager manager = MyManager.getInstance(this);

ToastUtils.showToast(this, "副屏幕类型为:" + manager.getSecondaryScreenType());

4.23 获取 DPI

函数：public String getDpi()

描述：获取 DPI

实现此接口的 API 版本：V5.1-20210716

```
/**
    * @param * @param
    * @return  返回 dpi
    * @method getDpi
    * @description  获取 DPI
    * @date: 2021/1/12
    * @author: zouyuanhang
    */
```

参数名/返回值	类型	说明	举例
参数	String	返回当前 DPI	160

范例：

MyManager manager = MyManager.getInstance(this);

manager.getDpi();

4.24 设置 DPI

函数：public boolean setDpi(int value)

描述：设置 DPI

实现此接口的 API 版本：V5.1-20210716

```
/**
    * @param * @param value        0=160 ,1=240,2=360,4=480
    * @return
    * @method setDpi
    * @description  设置 DPI
    * @date: 2021/1/12
    * @author: zouyuanhang
    */
```

参数名/返回值	类型	说明	举例
参数	int	传入的 dpi 值 0=160 ,1=240,2=360,3 =480	0

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.setDpi(0);
```

第五章 安装升级

5.1 固件升级

函数：public void upgradeSystem(String absolutePath)

描述：升级固件

实现此接口的 API 版本：V1.0-20180602

```
/**
    * @method upgradeSystem(String absolutePath)
    * @description 升级固件
    * @date 20180602
    * @author sky
    * @param absolutePath，待升级固件放置的绝对路径
   */
```

参数名/返回值	类型	说明	举例
参数名	String	固件存放的绝对路径	/mnt/internal_sd/update.img

注意点：

1. 函数将会自动重启系统进行 update.img 升级。
2. 保证 img 文件存在，完整可用。
3. 文件名一定为 update.img 或 updat.zip。

范例：

```
MyManager manager = MyManager.getInstance(this);
manager. upgradeSystem("/mnt/internal_sd/update.img");
```

5.2 recovery 模式

函数: public void rebootRecovery()

描述：将重启进入 recovery 模式

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method rebootRecovery()
     * @description  恢复出厂设置
     * @date 20180602
     * @author sky
    */
```

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.rebootRecovery();
```

5.3 静默安装 APK

函数：public boolean silentInstallApk(String apkPath)

描述：静默安装 APK 应用。

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method silentInstallApk(String apkPath)
     * @description  静默安装 apk，采用的方法是 pm install -r
     * @date 20180602
     * @author sky
     * @param apkPath，需要安装的 apk 的绝对路径
     * @return  静默安装成功返回 true，否则返回 false
    */
```

参数名/返回值	类型	说明	举例
参数名	String	待安装 APK 的绝对路径	"/mnt/internal_sd/test.apk"
返回值	Boolean	静默安装是否成功	

注意点：

1. APK 文件存在，完整可用。

范例：

```
//安装 test.apk

MyManager manager = MyManager.getInstance(this);

 boolean success = manager.silentInstallApk("/mnt/usb/0000-0000/YiShengTest.apk");

 ToastUtils.showToast(this,"静默升级是否成功" + success);
```

5.4 设置升级固件是否弹窗

函数：public void setUpdateSystemWithDialog(boolean flag)

描述：设置升级固件是否弹窗

实现此接口的 API 版本：V4.2-20200330

```
 /**
      * @method setUpdateSystemWithDialog(boolean flag)
      * @description  设置升级固件是否弹窗，默认是不弹窗
      * @date 20200330
      * @author sky
      * @param flag，true 升级时弹窗，false 升级不弹窗
     */
```

参数名/返回值	类型	说明	举例
参数名	boolean	控制升级固件是否弹窗	True 弹窗　false 不弹窗

范例：

MyManager manager = MyManager.getInstance(this);

manager. setUpdateSystemWithDialog(true);//设置升级固件弹窗确认

5.5 设置升级固件时后，固件包是否从文件中删除

函数：public void setUpdateSystemDelete(boolean flag)

描述：设置升级固件时后，固件包是否从文件中删除

实现此接口的 API 版本：V4.2-20200330

```
/**
    * @method setUpdateSystemDelete(boolean flag)
    * @description  设置升级固件时后，固件包是否从文件中删除，默认是不删除
    * @date 20200330
    * @author sky
    * @param flag，true 升级成功后删除，false 升级成功后不删除
   */
```

参数名/返回值	类型	说明	举例
参数名	boolean	控制升级固件是否删除	True 删除　false 不删除

范例：

MyManager manager = MyManager.getInstance(this);

manager. setUpdateSystemDelete(true);//设置升级成功删除固件

5.6 静默卸载 apk

函数：public boolean unInstallApk(String packagename)

描述：静默卸载 apk

实现此接口的 API 版本：V5.1-20210419

```
    /**
```

```
     * @methodunInstallApk(String packagename)
     * @description  静默卸载 apk
     * @date 20210419
     * @author sky
     * @param packagename  需要卸载的包名
    */
```

参数名/返回值	类型	说明	举例
参数名	String	需要卸载的包名	com.bjw.ComAssistant

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.selfStart("com.bjw.ComAssistant");
```

第六章 网络

6.1 查询以太网 MAC 地址

函数：public String getEthMacAddress()

描述：获取以太网的 MAC 地址

实现此接口的 API 版本：V2.0-20180806

```
/**
     * @method getEthMacAddress()
     * @description 获取以太网 mac 地址
     * @date 20180806
     * @author sky
     * @return 返回以太网 mac 地址，例如 30:1F:9A:61:BA:8F
    */
```

参数名/返回值	类型	说明	举例
返回值	String	以太网的 MAC 地址	30:1F:9A:61:BA:8F

范例：

```
MyManager manager = MyManager.getInstance(this);
Log.e("ETH MAC=” manager.getEthMacAddress ());
//输出 ETH MAC =30:1F:9A:61:BA:8F
```

6.2 查询以太网连接模式

函数：public String getEthMode()

描述：获取当前以太网的模式

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method    getEthMode()
     * @description  获取以太网的模式，静态或动态
     * @date 20180602
     * @author sky
     * @return  静态模式返回 StaticIp，动态模式返回 DHCP
    */
```

参数名/返回值	类型	说明	举例
返回值	String	以太网模式	StaticIp（静态） DHCP（动态）

注意点：

①、在获取以太网静态 IP 之前，需要绑定获取以太网静态 IP 的 aidl 服务，在创建 MyManager 对象的时候绑定，方法如下：manager.bindAIDLService(this);

②、在 activity 销毁的时候需要调用 manager.unBindAIDLService(this);

范例：

```
MyManager manager = MyManager.getInstance(this);
Log.d("API 显示","当前模式 ="+manager.getEthMode());
//输出 当前模式 = DHCP
```

6.3 查询以太网开关状态

函数：public boolean getEthStatus()

描述：获取当前以太网开关是否开启

实现此接口的 API 版本：V2.0-20180806

```
/**
```

```
 * @method getEthStatus()
 * @description  获取以太网的开关状态
 * @date    20180806
 * @author sky
 * @return  以太网开关打开返回 true，开关关闭返回 false
*/
```

参数名/返回值	类型	说明	举例
返回值	boolean	以太网开关状态	true（打开） false（关闭）

注意点：

①、在获取以太网是否开启之前，需要绑定 aidl 服务，在创建 MyManager 对象的时候绑定，方法如下：manager.bindAIDLService(this);

②、在 activity 销毁的时候需要调用 manager.unBindAIDLService(this);

范例：

```
MyManager manager = MyManager.getInstance(this);
Log.d("API 显示","以太网是否打开 ="+manager.getEthStatus());
//输出  以太网是否打开 = false
```

6.4 查询以太网的子网掩码

函数：public String getNetMask ()

描述：获取当前以太网的子网掩码

实现此接口的 API 版本：V2.0-20180806

```
/**
 * @method getNetMask()
 * @description  获取以太网的子网掩码
```

```
     * @date    20180806
     * @author sky
     * @return  返回当前设备以太网的子网掩码，例如 255.255.255.0
    */
```

参数名/返回值	类型	说明	举例
返回值	String	子网掩码	255.255.255.0

注意点：

①、在获取以太网是否开启之前，需要绑定 aidl 服务，在创建 MyManager 对象的时候绑定，方法如下：manager.bindAIDLService(this);

②、在 activity 销毁的时候需要调用 manager.unBindAIDLService(this);

范例：

```
MyManager manager = MyManager.getInstance(this);
Log.d("API 显示","以太网的子网掩码 ="+manager.getNetMask());
//输出  以太网的子网掩码 = 255.255.255.0
```

6.5 查询以太网的网关

函数：public String getGateway ()

描述：获取当前以太网的网关

实现此接口的 API 版本：V2.0-20180806

```
 /**
     * @method getGateway()
     * @description  获取以太网的网关
     * @date    20180806
```

```
     * @author sky
     * @return  返回当前设备以太网的网关，例如 192.168.1.1
    */
```

参数名/返回值	类型	说明	举例
返回值	String	网关	192.168.1.1

注意点：

①、在获取以太网是否开启之前，需要绑定 aidl 服务，在创建 MyManager 对象的时候绑定，方法如下：manager.bindAIDLService(this);

②、在 activity 销毁的时候需要调用 manager.unBindAIDLService(this);

范例：

```
MyManager manager = MyManager.getInstance(this);
Log.d("API 显示","以太网的网关 ="+manager.getGateway());
//输出  以太网的网关 = 192.168.1.1
```

6.6 查询以太网的 Dns1

函数：public String getEthDns1 ()

描述：获取当前以太网的 Dns1

实现此接口的 API 版本：V2.0-20180806

```
    /**
     * @method getEthDns1()
     * @description  获取以太网的 DNS1
     * @date    20180806
     * @author sky
     * @return  返回当前设备以太网的 DNS1，例如 192.168.1.1
```

```
*/
```

参数名/返回值	类型	说明	举例
返回值	String	Dns1	192.168.1.1

注意点：

①、在获取以太网是否开启之前，需要绑定 aidl 服务，在创建 MyManager 对象的时候绑定，方法如下：manager.bindAIDLService(this);

②、在 activity 销毁的时候需要调用 manager.unBindAIDLService(this);

范例：

```
MyManager manager = MyManager.getInstance(this);
Log.d("API 显示","以太网的 dns1 ="+manager.getGateway());
//输出  以太网的 dns1 = 192.168.1.1
```

6.7 查询以太网 Dns2

函数：public String getEthDns2()

描述：获取当前以太网的 Dns2

实现此接口的 API 版本：V2.0-20180806

```
/**
    * @method getEthDns2()
    * @description  获取以太网的 DNS2
    * @date    20180806
    * @author sky
    * @return  返回当前设备以太网的 DNS2，例如 192.168.1.1
    */
```

参数名/返回值	类型	说明	举例
返回值	String	Dns2	192.168.1.1

注意点：

①、在获取以太网是否开启之前，需要绑定 aidl 服务，在创建 MyManager 对象的时候绑定，方法如下：manager.bindAIDLService(this);

②、在 activity 销毁的时候需要调用 manager.unBindAIDLService(this);

范例：

```
MyManager manager = MyManager.getInstance(this);
Log.d("API 显示","以太网的 dns2 ="+manager.getGateway());
//输出  以太网的 dns2 = 192.168.1.1
```

6.8 设置以太网为动态获取模式

函数：public void setDhcpIpAddress(Context context)

描述：设置以太网模式为动态获取

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method setDhcpIpAddress(Context context)
     * @description  设置以太网的模式为动态
     * @date 20180602
     * @author sky
     * @param context，上下文对象
    */
```

参数名/返回值	类型	说明	举例
参数名	Context	上下文对象	

范例：

MyManager manager = MyManager.getInstance(this);

manager.setDhcpIpAddress(this);

6.9 设置以太网为静态地址模式

函数：public void setStaticEthIPAddress(String IPaddr, String gateWay,String mask, String dns1, String dns2)

描述：设置以太网的模式为静态地址

实现此接口的 API 版本：V1.0-20180602

```
 /**
      * @method setStaticEthIPAddress(String IPaddr, String gateWay, String mask, String dns1, String dns2)
      * @description  设置以太网的模式为静态，并指定相关参数
      * @date 20180602
      * @author sky
      * @param   IPaddr，设置的 IP 地址。gateWay，设置的网关。mask，设置的子网掩码。dns1 和 dns2 是设置的 dns 值
      */
```

参数名/返回值	类型	说明	举例
IPaddr	String	IP 地址	"192.168.1.125"
gateWay	String	网关	"192.168.1.1"
mask	String	子网掩码	"255.255.255.0"
dns1	String	DNS	"192.168.1.1"

ipLength	String	DNS	"0.0.0.0"

注意点：网关、子网掩码和 DNS 都要设置为合法数据

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.setEthIPAddress("192.168.1.125", "192.168.1.1","255.255.255.0","192.168.1.1","0.0.0.0");
```

6.10 获取以太网动态 IP 地址

函数：public String getDhcpIpAddress()

描述：获取以太网的动态 IP 地址

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method getDhcpIpAddress()
     * @description  获取以太网的动态 IP 地址
     * @date    20180602
     * @author sky
     * @return  返回以太网动态 ip 地址
    */
```

参数名/返回值	类型	说明	举例
返回值	String	以太网的动态 IP 地址	192.168.1.100

范例：

```
MyManager manager = MyManager.getInstance(this);
Log.d("API 显示","ETH IP ="+myManager.getDhcpIpAddress());
//输出 ETH IP =192.168.1.103
```

6.11 查询以太网的静态 IP 地址

函数：public String getStaticEthIPAddress()

描述：获取以太网的静态 IP 地址

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method getStaticEthIPAddress()
     * @description  获取以太网的静态 IP
     * @date    20180602
     * @author sky
     * @return  返回以太网的静态 ip 地址
    */
```

参数名/返回值	类型	说明	举例
返回值	String	以太网的静态 IP 地址	192.168.1.100

注意点：

①、在获取以太网静态 IP 之前，需要绑定获取以太网静态 IP 的 aidl 服务，在创建 myManager 对象的时候绑定，方法如下：myManager.bindAIDLService(this);

②、在 activity 销毁的时候需要调用 myManager.unBindAIDLService(this);

范例：

```
MyManager manager = MyManager.getInstance(this);
Log.d("API 显示","ETH IP ="+manager.getStaticEthIPAddress();
//输出 ETH IP =192.168.1.103
```

6.12 控制以太网开关

函数：public void ethEnabled(boolean enable)

描述：开关以太网

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method ethEnabled(boolean enable)
     * @description 控制以太网开关
     * @date 20180602
     * @author sky
     * @param enable，打开以太网开关传入 true，关闭以太网开关传入 false
    */
```

参数名/返回值	类型	说明	举例
返回值	boolean	开关以太网	true 开　false 关

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.ethEnabled(false);//关以太网
```

6.13 pppoe 拨号上网

函数：public void setPppoeDial(Context context,String userName,String password)

描述：设置 pppoe 拨号上网

实现此接口的 API 版本：V4.2-20200330

参数名/返回值	类型	说明	举例

参数名	Context	上下文对象	
参数名	String	拨号上网的用户名	
参数名	String	拨号上网的密码	

范例：

MyManager manager = MyManager.getInstance(this);

manager.setPppoeDial(this,”username”,”password”);

第七章 存储

7.1 查询外部存储 SD 卡路径

函数：public String getSDcardPath()

描述：获取外部存储 SD 卡路径

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method getSDcardPath()
     * @description 获取外置 SD 卡路径
     * @date 20180602
     * @author sky
     * @return 返回外置 SD 卡路径
    */
```

参数名/返回值	类型	说明	举例
返回值	String	外部存储 SD 卡绝对路径	

范例：

```
MyManager manager = MyManager.getInstance(this);
Log.d("API 测试","SD="+manager. getSDcardPath());
//输出 API: 外置 SD 卡路径 = /mnt/external_sd
```

7.2 查询外部存储 U 盘路径

函数：public String getUSBStoragePath(int num)

描述：获取外部存储 U 盘路径

实现此接口的 API 版本：V1.0-20180602

```
/**
 * @method getUSBStoragePath(int num)
 * @description 获取外置 U 盘路径
 * @date 20180602
 * @author sky
 * @param num，从 0 开始，表示第几个 U 盘
 * @return u 盘的路径
*/
```

参数名/返回值	类型	说明	举例
num	int	U 盘盘符	0,1,2,3
返回值	String	外部存储 U 盘绝对路径	

注意点：

需在源码设置路径，才能正常运行 Demo

范例：

```
MyManager manager = MyManager.getInstance(this);
Log.d("API 测试","USB Path = "+manager.getUSBStoragePath(0));
//输出/mnt/usb_storage/USB_DISK2/udisk0;
```

7.3 查询串口绝对路径

函数：public String getUartPath(String uart)

描述：获取串口绝对路径。

实现此接口的 API 版本：V1.0-20180602

```
/**
 * @method getUartPath(String uart)
 * @description 获取串口的绝对路径
```

```
     * @date 20180602
     * @author sky
     * @param uart，传入的是串口的序列号，比如串口 0 就传入 TTYS0
     * @return  返回指定串口的绝对路径
    */
```

参数名/返回值	类型	说明	举例
Uart	String	串口对应的端口号- uart 0, uart 1, uart 2, uart 3	
返回值	String	UART 绝对路径	

范例：

MyManager manager = MyManager.getInstance(this);

Log.d(“API 测试”,"UART-1=”+manager. getUartPath (uart1));

//输出 UART-1=/dev/ttyS1

第八章　定时开关机

8.1 设置周模式的定时开关机

函数：public void setPowerOnOffWithWeekly(int[] powerOnTime,int[] powerOffTime,int[] weekdays)

描述：设置定时开关机，周模式，一天只有一组开关机时间

实现此接口的 API 版本：V4.0-20200115

```
/**
    * @method setPowerOnOffWithWeekly(int[] powerOnTime,int[] powerOffTime,int[] weekdays)
    * @description 周模式设置定时开关机，一天只有一组开关机时间
    * @date 20200113
    * @author sky
    * @param powerOnTime，开机时间，例如{8,30}。powerOffTime，关机时间，例如{18,30}。
    *          weekdays，周一到周日工作状态,1 为开机，0 为不开机。例如{1,1,1,1,1,0,0}，是指周一到周五执行开关机
    */
```

参数名/返回值	类型	说明	举例
powerOnTime	Int[ ]	开机时间，时分	{8,30}
powerOffTime	Int[ ]	关机时间，时分	{18,30}
weekdays	Int[ ]	周一到周日的工作状态	{1,1,1,1,0,0,1}

注意点：

此方法一天只能有一组时间设入，并且开机时间在前，关机时间在后

范例：

```
MyManager manager = MyManager.getInstance(this);
int [] timeonArray = new int{8,30};
```

int [] timeoffArray = new int{18,30};

Int[] weekdays = new int{1,1,1,1,1,0,0};//周一到周日工作状态,1 为开机，0 为不开机

manager.setPowerOnOffWithWeekly(timeonArray,timeoffArray,weekdays);

设置上述时间将会在每周一到周五的 8:30 开机，18:30 关机。

8.2 设置一组定时开关机

函数：public void setPowerOnOff(int[] powerOnTime,int[] powerOffTime)

描述：设置一组定时开关机，一天可设置多次，需传入年月日时分

实现此接口的 API 版本：V4.0-20200115

```
/**
    * @method setPowerOnOff(int[] powerOnTime,int[] powerOffTime)
    * @description 设置一组定时开关机时间数据，需要传入年月日时分
    * @date 20200113
    * @author sky
    * @param powerOnTime，开机时间，例如{2020,1,10,20,48}，powerOffTime，关机时间，例如{2020,1,10,20,38}。
    */
```

参数名/返回值	类型	说明	举例
powerOnTime	Int[ ]	开机时间，年月日时分	{2020,01,13,18,40}
powerOffTime	Int[ ]	关机时间，年月日时分	{2020,01,13,18,30}

注意点：

此方法设置的时候，关机时间在前，开机时间在后

范例：

MyManager manager = MyManager.getInstance(this);

int [] timeoffArray = new int{2018,1,10,20,38};

int [] timeonArray = new int{2018,1,10,20,48};

manager.setPowerOnOff(timeonArray,timeoffArray);

设置上述时间将会在 2018 年 1 月 10 号，20:38 关机，20:48 开机

注意：该方法同样适用于只设置开机时间，只需将关机时间传 0 即可，即

int [] timeoffArray = new int{0,0,0,0,0};

8.3 获取当前定时开关机模式

函数：public String getPowerOnMode()

描述：获取定时开关机模式

实现此接口的 API 版本：V4.0-20200115

```
/**
     * @method getPowerOnMode()
     * @description  获取定时开关机的模式
     * @date    20200113
     * @author sky
     * @return    "0"是指在定时开关机本地设置的开关机时间。"2"是指用广播的方式调用 setPowerOnOffWithWeekly 方法。"1"是指用广播的方式调用 setPowerOnOff 方法。
     */
```

参数名/返回值	类型	说明	举例
返回值	String	开关机模式	“0”

范例：

MyManager manager = MyManager.getInstance(this);

manager.getPowerOnMode();

8.4 获取当前定时开关机的开机时间

函数：public String getPowerOnTime()

描述：获取定时开关机开机时间

实现此接口的 API 版本：V4.0-20200115

```
/**
    * @method getPowerOnTime()
    * @description 获取当前设备的开机时间
    * @date 20200113
    * @author sky
    * @return 返回当前设置的开机时间，例如 202001132025，是指 2020 年 1 月 13 号 20:25 开机
    */
```

参数名/返回值	类型	说明	举例
返回值	String	定时开关机开机时间	202001132025

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.getPowerOnTime();
```

8.5 获取当前定时开关机的关机时间

函数：public String getPowerOffTime()

描述：获取定时开关机关机时间

实现此接口的 API 版本：V4.0-20200115

```
/**
    * @method getPowerOffTime()
    * @description 获取当前设备的关机时间
```

```
     * @date 20200113
     * @author sky
     * @return  返回当前设置的开机时间，例如 202001132020，是指 2020 年 1 月 13 号 20:20 关机
     */
```

参数名/返回值	类型	说明	举例
返回值	String	定时开关机关机时间	202001132020

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.getPowerOffTime();
```

8.6 获取设备上一次的定时开关机的开机时间

函数：public String getLastestPowerOnTime()

描述：获取设备上一次执行过的开机时间

实现此接口的 API 版本：V4.0-20200115

```
 /**
     * @method getLastestPowerOnTime()
     * @description  获取设备上一次执行过的开机时间
     * @date 20200113
     * @author sky
     * @return  返回设备上一次的开机时间，例如 202001132025，是指在 2020 年 1 月 13 号 20:25 执行了开机操作
     */
```

参数名/返回值	类型	说明	举例
返回值	String	上一次的定时开关机开机时间	202001132025

范例：

MyManager manager = MyManager.getInstance(this);

manager.getLastestPowerOnTime()；

8.7 获取设备上一次的定时开关机的关机时间

函数：public String getLastestPowerOffTime()

描述：获取设备上一次执行过的开机时间

实现此接口的 API 版本：V4.0-20200115

```
/**
    * @method getLastestPowerOffTime()
    * @description 获取设备上一次执行过的开机时间
    * @date 20200113
    * @author sky
    * @return 返回设备上一次的开机时间，例如 202001132020，是指在 2020 年 1 月 13 号 20:20 执行了开机操作
    */
```

参数名/返回值	类型	说明	举例
返回值	String	上一次的定时开关机关机时间	202001132020

范例：

MyManager manager = MyManager.getInstance(this);

manager.getLastestPowerOffTime()；

8.8 清除定时开关机数据

函数：public void clearPowerOnOffTime()

描述：清除定时开关机时间

实现此接口的 API 版本：V4.0-20200115

```
/**
     * @method clearPowerOnOffTime()
     * @description  清除定时开关机时间
     * @date    20200113
     * @author sky
    */
```

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.clearPowerOnOffTime();
```

8.9 获取定时开关机版本

函数：public String getVersion()

描述：获取定时开关机版本号

实现此接口的 API 版本：V4.0-20200115

参数名/返回值	类型	说明	举例
返回值	String	定时开关机版本	YS_1.0_20180312

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.getVersion();
```

8.10 获取设备是否设置了定时开关机

函数：public boolean isSetPowerOnTime()

描述：获取设备是否设置了定时开关机

实现此接口的 API 版本：V4.0-20200115

参数名/返回值	类型	说明	举例
返回值	boolean	是否设置了定时开关机	True

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.isSetPowerOnTime();
```

第九章　GPIO 索引值控制

9.1 设置 io 口的状态是输入或输出

函数：public boolean setGpioDirection(int gpio, int arg)

描述：根据 io 口值设置该 gpio 是输入还是输出

实现此接口的 API 版本：V4.1-20200213

```
/**
    * @method setGpioDirection(int gpio, int arg)
    * @description  设置 io 口的状态是输入或输出
    * @date 20200213
    * @author sky
     * @param gpio，所要操作的 gpio 值  1 代表 gpio1。arg，1 代表设置为输入口，0 代表设置为输出口。
    * @return  设置成功返回 true，失败返回 false
   */
```

参数名/返回值	类型	说明	举例
返回值	Boolean	设置是否成功	成功-true　失败-false
参数值	Gpio	Gpio 的值	1
参数值	Arg	输入还是输出	输入口-1　输出口-0

范例：

```
MyManager manager = MyManager.getInstance(this);
manager. setGpioDirection(1, 0);
```

9.2 获取当前 gpio 是输入口还是输出口

函数：public String getGpioDirection(int gpio)

描述：根据具体的 gpio 值获取当前 gpio 的状态是输入还是输出。该方法也可判断该 gpio 是否有效，如果没有返回值，就说明该 gpio 不可用

实现此接口的 API 版本：V4.1-20200213

```
/**
    * @method getGpioDirection(int gpio)
    * @description 根据具体的 gpio 值获取当前 gpio 的状态是输入还是输出
    * @date 20200213
    * @author sky
    * @param gpio，gpio 的值
    * @return 输入口返回 in，输出口返回 out
   */
```

参数名/返回值	类型	说明	举例
返回值	String	输入还是输出	输入-in 输出-out
参数值	Int	Gpio 值	1

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.getGpioDirection(1);
```

9.3 设置 gpio 的电平

函数：public boolean writeGpioValue(int gpio, String arg)

描述：根据具体的 gpio 值控制该 gpio 的电平

实现此接口的 API 版本：V4.1-20200213

```
/**
    * @method writeGpioValue(int gpio, String arg)
    * @description 控制 gpio 电平，只针对输出口
    * @date 20200213
    * @author sky
    * @param gpio，gpio 的值。Arg，高电平--1，低电平--0
    * @return 写入成功返回 true，失败返回 false
   */
```

参数名/返回值	类型	说明	举例
返回值	Boolean	写入是否成功	成功--true 失败--false
参数名	Int	Gpio 值	1
参数名	String	高低电平	高--1 低--0

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.writeGpioValue(1, 1);
```

9.4 获取当前 gpio 的电平

函数：public String getGpioValue(int gpio)

描述：获取当前 gpio 的电平

实现此接口的 API 版本：V4.1-20200213

```
/**
    * @method getGpioValue(int gpio)
    * @description 获取当前 gpio 的电平
```

```
     * @date 20200213
     * @author sky
     * @param gpio，gpio 的值
     * @return  高电平返回 1，低电平返回 0
    */
```

参数名/返回值	类型	说明	举例
返回值	String	高电平或低电平	高电平--1　低电平--0
参数名	Int	Gpio 值	1

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.getGpioValue(1)；
```

第十章 其他

10.1 设置并保存系统时间

函数：public void setTime(int year, int month, int day, int hour, int minute,int sec)

描述：设置并保存系统时间

实现此接口的 API 版本：V1.0-20180602

```
/**
    * @method setTime(int year, int month, int day, int hour, int minute, int sec)
    * @description 设置系统时间
    * @date  20180602
    * @author sky
    * @param 传入需要设置的年月日时分秒，其中月份从 1-12，时间按照 24 小时制
   */
```

参数名/返回值	类型	说明	举例
year	int	年	
month	int	月（1 到 12）	
day	int	日	
hour	int	（24 小时制）小时	
minute	int	（24 小时制）分钟	
sec	int	秒	
返回值	void		

范例：

//设置 2016-03-16, 13:44:20

MyManager manager = MyManager.getInstance(this);

manager. setTime (2016, 3, 16, 13, 44,20);

10.2 打开或关闭自动确定时间

函数：public void switchAutoTime(boolean open)

描述：设置打开或关闭自动确定时间

实现此接口的 API 版本：V1.0-20180602

```
/**
    * @method switchAutoTime(boolean open)
    * @description  控制自动确定日期和时间的开关
    * @date 20180602
    * @author sky
    * @param open，传入 true 就是打开开关，false 关闭开关
   */
```

参数名/返回值	类型	说明	举例
参数名	boolean	True 打开自动确定时间 False 关闭自动确定时间	

范例：

MyManager manager = MyManager.getInstance(this);

manager.switchAutoTime(false);//关闭自动确定日期和时间

10.3 控制软键盘是否能弹出

函数：public void setSoftKeyboardHidden(boolean hidden)

描述：控制软键盘是否能弹出

实现此接口的 API 版本：V1.0-20181126

```
/**
    * @method setSoftKeyboardHidden(boolean hidden)
    * @description 控制软键盘是否弹出（主要是 EditText 控件）
    * @date 20181126
    * @author sky
    * @param hidden，传入 true，说明隐藏软键盘，传入 false 就是显示软键盘
   */
```

参数名/返回值	类型	说明	举例
参数名	boolean	True 软键盘可以弹出 False 软键盘不能弹出	

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.setSoftKeyboardHidden(false);
```

10.4 休眠时间的控制

函数：public void setDormantInterval(Context context,long time)

描述：休眠时间的控制

实现此接口的 API 版本：V2.2-20181129

```
/**
    * @method setDormantInterval(Context context, long time)
    * @description 休眠时间的控制
    * @date 20181129
    * @author sky
```

```
     * @param context，上下文对象。time，间隔时间（在间隔时间内没操作系统进入休眠）
    */
```

参数名/返回值	类型	说明	举例
返回值	long	间隔时间（如果关闭休眠，传的值是 2147483647）	如果间隔 15s 休眠，传的参数是 15000

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.setDormantInterval(context,15000);
```

10.5 查询自动确定网络时间状态

函数：public boolean isAutoSyncTime()

描述：获取自动确定网络时间的开关是否开启

实现此接口的 API 版本：V2.0-20180806

```
  /**
     * @method isAutoSyncTime()
     * @description  获取自动确定网络时间的开关状态
     * @date    20180806
     * @author sky
     * @return  返回 true 说明开启了自动确定网络时间，返回 false 说明关闭了此开关
    */
```

参数名/返回值	类型	说明	举例
返回值	boolean	True 打开自动确定时间 False 关闭自动确定时间	

范例：

MyManager manager = MyManager.getInstance(this);

manager.isAutoSyncTime();

10.6 以 ROOT 权限运行 shell 命令

函数：public void execSuCmd (String command)

描述：将以 ROOT 权限运行 shell 命令

实现此接口的 API 版本：V1.0-20180602

```
/**
     * @method execSuCmd(String command)
     * @description  在 root 权限下运行 shell 命令
     * @date 20180602
     * @author sky
     * @param command，传入需要执行的 shell 命令，比如 reboot
    */
```

参数名/返回值	类型	说明	举例
command	String	Shell 命令	“reboot”
返回值	void		

范例：

//查看目录并用 shell 命令安装 APK

MyManager manager = MyManager.getInstance(this);

manager.execSuCmd ("ls /mnt/sdcard/ ");

manager.execSuCmd ("system/bin/pm -install -r " + "mnt/sdcard/Update.apk ");

10.7 查询设备上网方式

函数：public int getCurrentNetType()

描述：获取当前的上网类型

实现此接口的 API 版本：V1.0-20180602

```
/**
    * @method getCurrentNetType()
    * @description 获取当前网络类型
    * @date 20180602
    * @author sky
    * @return 返回 int 值，0 表示以太网，1 表示 WIFI，2 表示移动网络，-100 表示未知网络
    */
```

参数名/返回值	类型	说明	举例
返回值	int	0 以太网， 1 wifi， 2 移动网络 -100 未知网络	

范例：

```
MyManager manager = MyManager.getInstance(this);
Int type = manager.getCurrentNetType();
```

10.8 查询 HDMI 输入状态

函数：public int getHdmiinStatus()

描述：HDMI 输入状态

实现此接口的 API 版本：V1.0-20180602

```
/**
    * @method getHdmiinStatus()
    * @description 获取 HDMI 连接情况
    * @date 20180602
```

```
     * @author sky
     * @return   HDMI 有输出返回 1，否则返回 0
    */
```

参数名/返回值	类型	说明	举例
返回值	int	返回 0 没有 HDMI 输入， 1 有 HDMI 输入	

范例：

```
MyManager manager = MyManager.getInstance(this);
Toast.makeText(getApplicationContext(),”HDMI” +
manager.getHdmiinStatus(),Toast.LENGTH_SHORT).show();
```

10.9 设置默认输入法

函数：public boolean isSetDefaultInputMethodSuccess(String defaultInputMethod)

描述：设置默认输入法

实现此接口的 API 版本：V3.1-20190228

```
 /**
     * @method isSetDefaultInputMethodSuccess(String defaultInputMethod)
     * @description  设置默认输入法，并判断是否设置成功
     * @date 20190228
     * @author sky
     * @param defaultInputMethod，需要设置的输入法的包名，例如谷歌拼音输入法
"com.google.android.inputmethod.pinyin/.PinyinIME"
     * @return  设置成功返回 true，失败返回 false
    */
```

参数名/返回值	类型	说明	举例
参数名	String	默认输入法 apk 的包名	谷歌拼音

			com.google.android.inputmethod.pinyin/.PinyinIME
返回值	Boolean	是否设置成功	成功返回 true　失败返回 false

范例：

MyManager manager = MyManager.getInstance(this);

manager.isSetDefaultInputMethodSuccess(“com.google.android.inputmethod.pinyin/.PinyinIME”);

10.10 获取默认输入法

函数：public String getDefaultInputMethod()

描述：获取默认输入法

实现此接口的 API 版本：V3.1-20190228

```
/**
    * @method getDefaultInputMethod()
    * @description 获取当前系统输入法
    * @date 20190228
    * @author sky
    * @return 返回当前输入法的包名，例如 com.google.android.inputmethod.pinyin/.PinyinIME
    */
```

参数名/返回值	类型	说明	举例
参数名	String	当前输入法 apk 的包名	谷歌拼音 com.google.android.inputmethod.pinyin

范例：

MyManager manager = MyManager.getInstance(this);

manager.getDefaultInputMethod();

10.11 设置系统语言

函数：public void setLanguage(String language,String country)

描述：设置系统语言

实现此接口的 API 版本：V3.2-20190513

```
/**
     * @method setLanguage(String language, String country)
     * @description 设置当前系统默认语言
     * @date 20190513
     * @author sky
     * @param language，语言，如 zh。country，国家，如 CN
    */
```

参数名/返回值	类型	说明	举例
参数名	String	设置的语言（小写字母）和国家（大写字母）	简体中文 ch CN 中文台湾 zh TW 加拿大英语 en CA

范例：

MyManager manager = MyManager.getInstance(this);

manager.setLanguage(“zh”,”CN”);

10.12 获取设备 CPU 温度

函数：public String getCPUTemperature()

描述：获取设备 CPU 温度。

实现此接口的 API 版本：V3.3-20190619

```
/**
    * @method getCPUTemperature()
    * @description 获取当前系统 CPU 温度
    * @date   20190619
    * @author sky
    * @return 返回 int 值，单位是摄氏度
   */
```

参数名/返回值	类型	说明	举例
参数名	String	设备的 CPU 温度	

范例：

```
MyManager manager = MyManager.getInstance(this);
 Log.d("API 显示","设备 CPU 温度: "+ manager.getCPUTemperature());
```

10.13 打开或关闭 adb 连接

函数：public void setADBOpen(boolean open)

描述：打开或关闭 adb

实现此接口的 API 版本：V3.4-20190628

```
/**
    * @method setADBOpen(boolean open)
    * @description 打开或关闭 adb 连接
    * @date 20190628
```

```
     * @author sky
     * @param open，true，打开 adb 连接开关。false，关闭 adb 连接
    */
```

参数名/返回值	类型	说明	举例
参数	Boolean	是否打开 adb，true 打开，false 关闭	

范例：

```
MyManager manager = MyManager.getInstance(this);
 manager.setADBOpen(true);
```

10.14 替换开机动画

函数：public void replaceBootanimation(String path)

描述：替换开机动画

实现此接口的 API 版本：V3.4-20190628

```
  /**
     * @method replaceBootanimation(String path)
     * @description  替换开机动画
     * @date    20190628
     * @author sky
     * @param path，开机动画 bootanimation.zip 所在的绝对路径
    */
```

参数名/返回值	类型	说明	举例
参数	String	开机动画所在路径	/storage/emulated/0/bootanimation.zip

范例：

```
MyManager manager = MyManager.getInstance(this);
```

manager.replaceBootanimation(“/storage/emulated/0/bootanimation.zip”);

10.15 设置默认 Launcher

函数：public void setDefaultLauncher(String packageAndClassName)

描述：系统有一个以上的 Launcher，可设置默认 Launcher

实现此接口的 API 版本：V3.5-20190719

```
/**
    * @method setDefaultLauncher(String packageAndClassName)
    * @description 设置系统默认 Launcher
    * @date 20190719
    * @author sky
    * @param packageAndClassName，设置的 Launcher 的包名和启动类名，例如
"com.android.launcher3/com.android.launcher3.Launcher"
    */
```

参数名/返回值	类型	说明	举例
参数	String	默认 Launcher 的包名和启动类名	Launcher3 的包名和启动类名 "com.android.launcher3/com.android.launcher3.Launcher"

范例：

MyManager manager = MyManager.getInstance(this);

manager.setDefaultLauncher("com.android.launcher3/com.android.launcher3.Launcher");

10.16 执行关机操作

函数：public void shutdown()

描述：休眠之后唤醒系统的方法

实现此接口的 API 版本：V4.3-20200521

```
/**
     * @method shutdown()
     * @description 执行关机操作，走安卓标准关机流程
     * @date 20180602
     * @author sky
    */
```

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.shutdown();
```

10.17 设置开机自启

函数：public void selfStart(String packagname)

描述：设置开机自启 app

实现此接口的 API 版本：V5.1-20210716

```
 /**
     * @method selfStart(String packagname)
     * @description 设置开机自启
     * @date    2021-07-16
     * @author zyh
     * @param packagname，包名
    */
```

参数名/返回值	类型	说明	举例
参数	String	应用包名	rkandroidapi.ys.com.rkandroidapi

范例：

MyManager manager = MyManager.getInstance(this);

manager.selfStart("rkandroidapi.ys.com.rkandroidapi");

10.18 设置守护进程应用

函数：daemon(String packageName, int value)

描述：设置守护进程应用

实现此接口的 API 版本：V5.1-20210716

```
/**
    * @param packageName  需要守护应用的包名  none 代表无
    * @param value  守护的时间  0 = 30 秒 ， 1 = 60 秒  2=180 秒默认 30 秒
    * @return
    * @method daemon
    * @description  设置守护进程
    * @date: 2021/7/16
    * @author: zouyuanhang
    */
```

参数名/返回值	类型	说明	举例
参数	String	需要守护应用的包名	rkandroidapi.ys.com.rkandroidapi
参数	int	守护的时间	0

范例：

MyManager manager = MyManager.getInstance(this);

manager.daemon("rkandroidapi.ys.com.rkandroidapi",1);

10.19 应用安装白名单

函数：public void setAppInstallWhitelist(String isopen, String packageName)

描述：应用安装白名单

实现此接口的 API 版本：V5.1-20210716

```
/**
    * @param isopen true 启用白名单 false 不启用
    * @param packageName 需要增加白名单的应用包名，如需删除应用包名可用
Utils.removeFileData(packageName,false)
    * @return
    * @method setAppInstallWhitelist
    * @description 添加应用安装白名单
    * @date: 2021/1/12
    * @author: zouyuanhang
    */
```

参数名/返回值	类型	说明	举例
参数	String	true 启用白名单 false 不启用	true
参数	String	需要增加白名单的应用包名	rkandroidapi.ys.com.rkandroidapi

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.setAppInstallWhitelist("true","rkandroidapi.ys.com.rkandroidapi");
```

10.20 应用安装黑名单

函数：public void setAppInstallBlacklist(String isopen, String packageName)

描述：删除应用安装白名单

实现此接口的 API 版本：V5.1-20210716

```
 /**
        * @param isopen true  启用黑名单  false  不启用
        * @param packageName  需要增加黑名单的应用包名，如需删除应用包名可用Utils.removeFileData(packageName,true)
        * @return
        * @method setAppInstallBlacklist
        * @description  添加应用安装黑名单
        * @date: 2021/1/12
        * @author: zouyuanhang
        */
```

参数名/返回值	类型	说明	举例
参数	String	true 启用黑名单 false 不启用	true
参数	String	需要增加黑名单的应用包名	rkandroidapi.ys.com.rkandroidapi

范例：

```
MyManager manager = MyManager.getInstance(this);
manager.setAppInstallBlacklist("true","rkandroidapi.ys.com.rkandroidapi");
```

10.21 打开或关闭网络 adb 连接

函数：public void setADBOpen(boolean open)

描述：打开或关闭 adb

实现此接口的 API 版本：V3.4-20190628

```
/**
    * @method setNetworkAdb(boolean open)
    * @description  打开或关闭网络 adb 连接
    * @date 20190628
    * @author sky
    * @param open，true，打开网络 adb 连接开关。false，关闭网络 adb 连接
   */
```

参数名/返回值	类型	说明	举例
参数	Boolean	是否打开网络 adb，true 打开，false 关闭	

范例：

```
MyManager manager = MyManager.getInstance(this);
 manager.setNetworkAdb(true);
```