

Ciontek Android API Introduction

V1.0.6

| | | |
|--------|----------------------------------|----|
| 1. | Overview | 1 |
| 2. | Import API library | 1 |
| 3. | All API methods | 2 |
| 3.1 | System utils | 2 |
| 3.1.1 | getOSVersion | 2 |
| 3.1.2 | getDeviceId | 2 |
| 3.1.3 | installRomPackage | 3 |
| 3.1.4 | enableDevelopmentMode | 3 |
| 3.1.5 | getDevelopmentModeEnabled | 3 |
| 3.1.6 | enableUsbAdb | 4 |
| 3.1.7 | getUsbAdbEnabled | 4 |
| 3.1.8 | setHomeActivity | 4 |
| 3.1.9 | resetDefaultHome | 5 |
| 3.1.10 | reboot | 5 |
| 3.1.11 | shutdown | 5 |
| 3.1.12 | StartScheduleReboot | 5 |
| 3.1.13 | cancelScheduledReboot | 6 |
| 3.1.14 | setAutoTimeSource | 6 |
| 3.1.15 | setSystemTimeZone | 6 |
| 3.1.16 | setSystemTime | 6 |
| 3.1.17 | setUsbMode | 7 |
| 3.1.18 | getUsbMode | 7 |
| 3.1.19 | enableStatusBar | 7 |
| 3.1.20 | getStatusBarEnabled | 8 |
| 3.1.21 | enableNavigationBar | 8 |
| 3.1.22 | getNavigationBarEnabled | 8 |
| 3.1.23 | enterKiosk | 9 |
| 3.1.24 | exitKiosk | 9 |
| 3.1.25 | screencap | 9 |
| 3.1.26 | enableFunctionKey | 10 |
| 3.1.27 | disableFunctionKey | 10 |
| 3.1.28 | showHomeKey | 10 |
| 3.1.29 | showRecentKey | 11 |
| 3.1.30 | showBackKey | 11 |
| 3.1.31 | isHomeKeyShown | 11 |
| 3.1.32 | isRecentKeyShown | 12 |
| 3.1.33 | isBackKeyShown | 12 |
| 3.2 | App Install/Uninstall/Permission | 12 |
| 3.2.1 | enableAppInstallWhiteList | 12 |

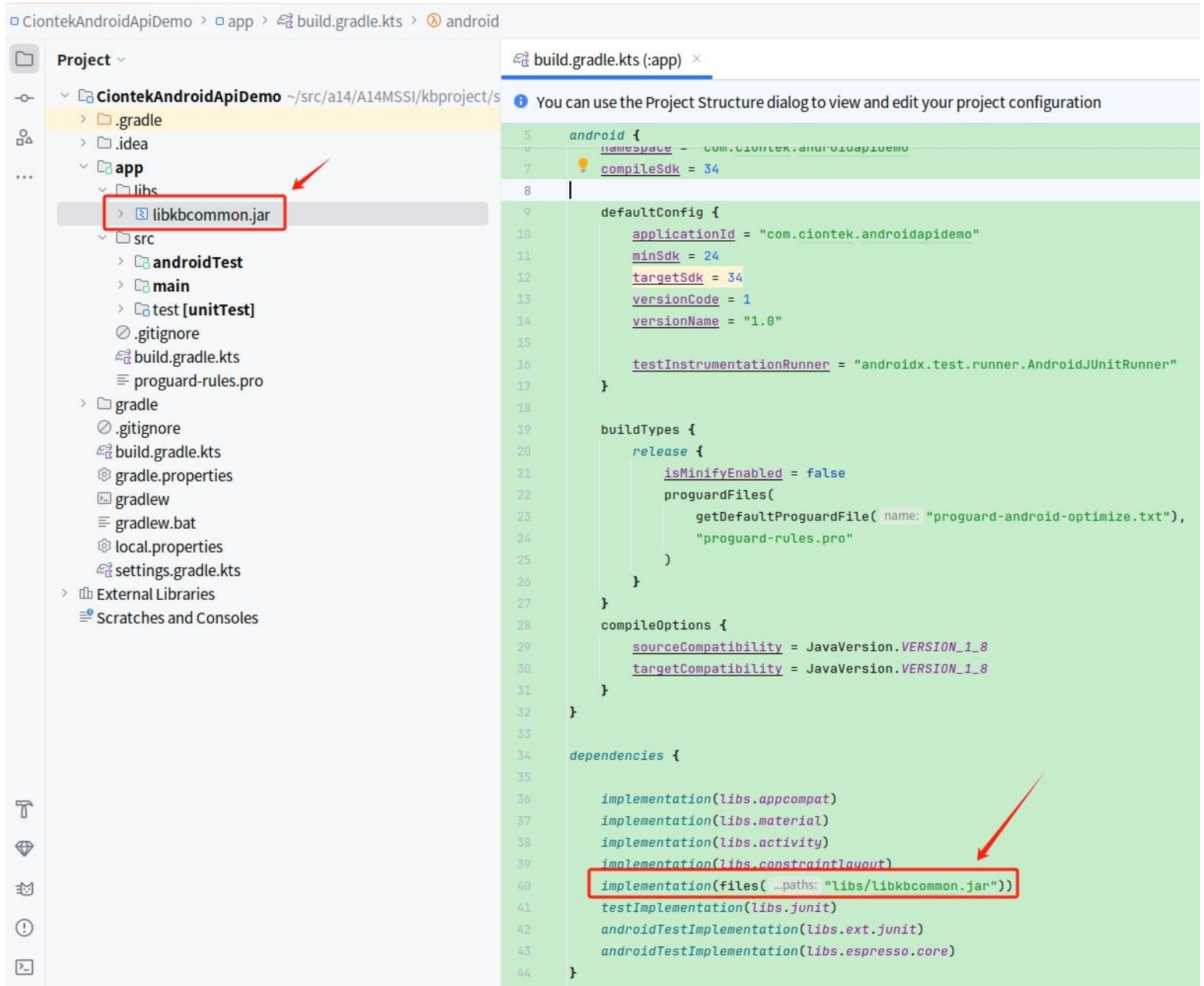
| | | |
|------------|--|-----------|
| 3.2.2 | disableAppInstallWhiteList | 13 |
| 3.2.3 | addAppToInstallWhiteList | 13 |
| 3.2.4 | delAppFromInstallWhiteList | 13 |
| 3.2.5 | getAppInstallWhiteList | 14 |
| 3.2.6 | enableAppUninstallBlackList | 14 |
| 3.2.7 | disableAppUninstallBlackList | 14 |
| 3.2.8 | addAppToUninstallBlackList | 15 |
| 3.2.9 | delAppFromUninstallBlackList | 15 |
| 3.2.10 | getAppUninstallBlackList | 16 |
| 3.2.11 | setAppSignatureVerifyEnabled | 16 |
| 3.2.12 | setAppSignatureVerifyPassword | 16 |
| 3.2.13 | addAppSignatureCertificate | 17 |
| 3.2.14 | delAppSignatureCertificate | 17 |
| 3.2.15 | getAppSignatureCertificate | 18 |
| 3.2.16 | installAppSilent | 18 |
| 3.2.17 | installApplication | 19 |
| 3.2.18 | uninstallAppSilent | 19 |
| 3.2.19 | setAppOpsMode | 20 |
| 3.3 | Network control/Network information | 20 |
| 3.3.1 | setWifiEnabled | 20 |
| 3.3.2 | setMobileNetworkEnabled | 21 |
| 3.3.3 | enableBluetooth | 21 |
| 3.3.4 | setAirplaneModeOn | 21 |
| 3.3.5 | getWlanMacAddress | 22 |
| 3.3.6 | getWlanIPAddress | 22 |
| 3.3.7 | getEth0MacAddress | 22 |
| 3.3.8 | getEth0IPAddress | 23 |
| 3.3.9 | getEth0IPMethod | 23 |
| 3.3.10 | setEth0Configuration | 23 |
| 3.3.11 | setDefaultDataSubId | 24 |
| 3.3.12 | getCellularIMEI | 24 |
| 3.3.13 | getCellularICCID | 24 |
| 3.3.14 | setPreferredAPN | 25 |
| 3.3.15 | insertAPN | 25 |
| 3.3.16 | deleteAPN | 25 |
| 3.3.17 | getCurrentPreferredApnId | 26 |

1. Overview

This document introduces what additional API methods we have at the Android OS, what they do, and how to invoke them.

2. Import API library

To use all of our additional API methods, you have to import our **libkbcommon.jar** to your module.



Once libkbcommon.jar has been successfully imported, you can use class **KbCommonManager** to invoke all of our additional API methods, for example:

```

10
11 import com.kingberry.common.api.KbCommonManager;
12
13 public class MainActivity extends AppCompatActivity {
14
15     @Override
16     protected void onCreate(Bundle savedInstanceState) {
17         super.onCreate(savedInstanceState);
18         EdgeToEdge.enable( $this$enableEdgeToEdge: this);
19         setContentView(R.layout.activity_main);
20         ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
21             Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
22             v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
23             return insets;
24         });
25
26         KbCommonManager kbCommonManager = KbCommonManager.getInstance();
27         kbCommonManager.reboot();
28     }
29 }

```

3. All API methods

3.1 System utils

3.1.1 getOSVersion

Prototype: `String getOSVersion()`

Function: Get the Android OS firmware version.

Sample code:

```

import com.kingberry.common.api.KbCommonManager;

String firmwareVer = KbCommonManager.getInstance().getOSVersion();

```

3.1.2 getDeviceId

Prototype: `String getDeviceId()`

Function: Get the device serial number. Return an empty string if the device number is not available.

Sample code:

```

import com.kingberry.common.api.KbCommonManager;

String sn = KbCommonManager.getInstance().getDeviceId();

```

3.1.3 installRomPackage

Prototype: `int installRomPackage(String absoluteUpdatePackagePath)`

Function: Use an OTA update package to update your system. Return 0 if the background updating task has started successfully. Please check the update notification in system notification panel for the update progress and result.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//OTA update system
int ret =
KbCommonManager.getInstance().installRomPackage("/storage/emulated/0/Download/update.zip");
if (ret == 0) {
    //The background updating system task has started successfully
} else {
    //Failed to start the background updating system task
}
```

3.1.4 enableDevelopmentMode

Prototype: `void enableDevelopmentMode(boolean enable)`

Function: Enable the developer mode or not.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Enable developer mode
KbCommonManager.getInstance().enableDevelopmentMode(true);

//Disable developer mode
KbCommonManager.getInstance().enableDevelopmentMode(false);
```

3.1.5 getDevelopmentModeEnabled

Prototype: `boolean getDevelopmentModeEnabled()`

Function: Return the developer mode is enabled or not.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;
```

```
//Return developer mode is enabled or not
```

```
boolean enabled = KbCommonManager.getInstance().getDevelopmentModeEnabled();
```

3.1.6 enableUsbAdb

Prototype: `void enableUsbAdb(boolean enable)`

Function: Enable USB debugging or not.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Enable USB debugging
KbCommonManager.getInstance().enableUsbAdb(true);

//Disable USB debugging
KbCommonManager.getInstance().enableUsbAdb(false);
```

3.1.7 getUsbAdbEnabled

Prototype: `boolean getUsbAdbEnabled()`

Function: Return USB debugging is enabled or not.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Return USB debugging is enabled or not
boolean enabled = KbCommonManager.getInstance().getUsbAdbEnabled();
```

3.1.8 setHomeActivity

Prototype: `boolean setHomeActivity(ComponentName activity)`

Function: Set the target app as default home. This `activity` must declare the "android.intent.category.HOME" category. Return true if set successfully, false otherwise.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Set com.teslacoilsw.launcher as default home
boolean set = KbCommonManager.getInstance().setHomeActivity(
    ComponentName.unflattenFromString("com.teslacoilsw.launcher/.NovaLauncher"));
```


3.1.9 resetDefaultHome

Prototype: `boolean resetDefaultHome()`

Function: Reset the default home app to the firmware default home. Return true if reset successfully, false otherwise.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Reset the default home app to the firmware default home
boolean reset = KbCommonManager.getInstance().resetDefaultHome();
```

3.1.10 reboot

Prototype: `void reboot()`

Function: Restart the device.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

KbCommonManager.getInstance().reboot();
```

3.1.11 shutdown

Prototype: `void shutdown()`

Function: Shutdown the device.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

KbCommonManager.getInstance().shutdown();
```

3.1.12 StartScheduleReboot

Prototype: `void StartScheduleReboot(long seconds)`

Function: Restart the device after specified seconds.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

KbCommonManager.getInstance().StartScheduleReboot(10);
```


3.1.13 cancelScheduledReboot

Prototype: `void cancelScheduledReboot()`

Function: If you post a delayed restart task through 'StartScheduleReboot' method, you can cancel the task through this method.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

KbCommonManager.getInstance().cancelScheduledReboot();
```

3.1.14 setAutoTimeSource

Prototype: `boolean setAutoTimeSource(int timeSource)`

Function: Set how you want to automatically update the system time. Return 0 if set successfully, otherwise false. If `timeSource` is 1, system will use network-provided time; if `timeSource` is 2, system will use GPS-provided time; and if `timeSource` is 0, system will not auto-update system time.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Use network-provided time to auto-update time
boolean set = KbCommonManager.getInstance().setAutoTimeSource(1);
//Use GPS-provided time to auto-update time
boolean set = KbCommonManager.getInstance().setAutoTimeSource(2);
//Disable auto-update time
boolean set = KbCommonManager.getInstance().setAutoTimeSource(0);
```

3.1.15 setSystemTimeZone

Prototype: `void setSystemTimeZone(String timeZone)`

Function: Set system time zone.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Set America/New_York as system time zone
KbCommonManager.getInstance().setSystemTimeZone("America/New_York");
```

3.1.16 setSystemTime

Prototype: `void setSystemTime(long timeInMilliseconds)`

Function: Set system time. [timeInMilliseconds](#) is the time in milliseconds since the Epoch.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Set system time to 2024-11-18 02:24:51 (New York time zone)
KbCommonManager.getInstance().setSystemTime(1731914691000L);
```

3.1.17 setUsbMode

Prototype: `boolean setUsbMode(int mode)`

Function: Set USB to host mode or device mode. Return true if set successfully. If mode is 0 will set to host mode, other values will set to device mode. Some projects don't support this API.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Set USB to host mode
boolean set = KbCommonManager.getInstance().setUsbMode(0);

//Set USB to device mode
boolean set = KbCommonManager.getInstance().setUsbMode(1);
```

3.1.18 getUsbMode

Prototype: `int getUsbMode()`

Function: Get USB working mode. Return 0 is host mode, return 1 is device mode. Some projects don't support this API.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//0 is host mode, 1 is device mode
int mode = KbCommonManager.getInstance().getUsbMode();
```

3.1.19 enableStatusBar

Prototype: `void enableStatusBar(boolean enable)`

Function: Allow the status bar to expand or not.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;
```

```
//Allow the status bar to expand  
KbCommonManager.getInstance().enableStatusBar(true);  
  
//Don't allow the status bar to expand  
KbCommonManager.getInstance().enableStatusBar(false);
```

3.1.20 getStatusBarEnabled

Prototype: `boolean getStatusBarEnabled()`

Function: Return the status bar expansion is enabled or not. Return true the status bar can expand, return false the status bar can't expand.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;  
  
boolean enabled = KbCommonManager.getInstance().getStatusBarEnabled();  
if (enabled) {  
    //status bar can expand  
}else {  
    //status bar can't expand  
}
```

3.1.21 enableNavigationBar

Prototype: `void enableNavigationBar(boolean enable)`

Function: Hide the navigation bar or not.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;  
  
//Show navigation bar if it is hidden  
KbCommonManager.getInstance().enableNavigationBar(true);  
  
//Hide navigation bar  
KbCommonManager.getInstance().enableNavigationBar(false);
```

3.1.22 getNavigationBarEnabled

Prototype: `boolean getNavigationBarEnabled()`

Function: Return the navigation bar is shown or hidden. Return true is shown, return false is hidden.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;
```

```
//Is the navigation bar shown or not
```

```
boolean shown = KbCommonManager.getInstance().getNavigationBarEnabled();  
if (shown) {  
    //navigation bar is shown  
} else {  
    //navigation bar is hidden  
}
```

3.1.23 enterKiosk

Prototype: `boolean enterKiosk(ComponentName activity)`

Function:

Make the target app work in fake KIOSK mode. The target `activity` must declare the "android.intent.category.HOME" category. Return true if entered successfully, false otherwise.

Fake KIOSK mode is that the status bar can't expand, the navigation bar is hidden, and the target app will be set as default home, so end users can't switch to other apps manually.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;  
  
//Nova Launcher app enter fake KIOSK mode  
boolean entered = KbCommonManager.getInstance()  
    .enterKiosk(ComponentName.unflattenFromString("com.teslacoilsw.launcher/.NovaLauncher"));
```

3.1.24 exitKiosk

Prototype: `boolean exitKiosk()`

Function: Exit the fake KIOSK mode. The status bar will can expand, the hidden navigation bar will be restored, and the default home app will be reset to the firmware default home. Return true if exited successfully, false otherwise.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;  
  
//Exit fake KIOSK mode  
boolean exited = KbCommonManager.getInstance().exitKiosk();
```

3.1.25 screenshot

Prototype: `void screenshot(String path)`

Function: Screenshot and save to specified path.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Screenshot and save to /storage/emulated/0/Download/screenshot.png
KbCommonManager.getInstance().screencap("/storage/emulated/0/Download/screenshot.png");
```

3.1.26 enableFunctionKey

Prototype: `int enableFunctionKey(int keyCode)`

Function: Enable a specified key. Return 0 if enabled successfully, false otherwise.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Enable power key (hardware key)
KbCommonManager.getInstance().enableFunctionKey(KeyEvent.KEYCODE_POWER);

//Enable recent key (virtual key)
KbCommonManager.getInstance().enableFunctionKey(KeyEvent.KEYCODE_RECENT_APPS);
```

3.1.27 disableFunctionKey

Prototype: `int disableFunctionKey(int keyCode)`

Function: Disable a specified key. Return 0 if enabled successfully, false otherwise.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Disable power key (hardware key)
KbCommonManager.getInstance().disableFunctionKey(KeyEvent.KEYCODE_POWER);

//Disable recent key (virtual key)
KbCommonManager.getInstance().disableFunctionKey(KeyEvent.KEYCODE_RECENT_APPS);
```

3.1.28 showHomeKey

Prototype: `int showHomeKey(boolean show)`

Function: Hide/Show the HOME button in the navigation bar.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;
```

```
//Hide home button
KbCommonManager.getInstance().showHomeKey(false);

//Show home button
KbCommonManager.getInstance().showHomeKey(true);
```

3.1.29 showRecentKey

Prototype: `int showRecentKey(boolean show)`

Function: Hide/Show the RECENT button in the navigation bar.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Hide recent button
KbCommonManager.getInstance().showRecentKey(false);

//Show recent button
KbCommonManager.getInstance().showRecentKey(true);
```

3.1.30 showBackKey

Prototype: `int showBackKey(boolean show)`

Function: Hide/Show the BACK button in the navigation bar.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Hide back button
KbCommonManager.getInstance().showBackKey(false);

//Show back button
KbCommonManager.getInstance().showBackKey(true);
```

3.1.31 isHomeKeyShown

Prototype: `boolean isHomeKeyShown()`

Function: Get the HOME button in the navigation bar is shown or hidden.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;
```

```
boolean isHomeShown = KbCommonManager.getInstance().isHomeKeyShown();
```

3.1.32 isRecentKeyShown

Prototype: `boolean isRecentKeyShown()`

Function: Get the RECENT button in the navigation bar is shown or hidden.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

boolean isRecentShown = KbCommonManager.getInstance().isRecentKeyShown();
```

3.1.33 isBackKeyShown

Prototype: `boolean isBackKeyShown()`

Function: Get the BACK button in the navigation bar is shown or hidden.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

boolean isBackShown = KbCommonManager.getInstance().isBackKeyShown();
```

3.2 App Install/Uninstall/Permission

3.2.1 enableAppInstallWhiteList

Prototype: `boolean enableAppInstallWhiteList()`

Function:

Use the app installation whitelist to restrict app installations. Return true if enabled successfully, false otherwise.

The app installation whitelist stores the package names of all the apps that are allowed to be installed, any app that attempts to install will be rejected if system cannot find a matching package name in the whitelist.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

boolean enabled = KbCommonManager.getInstance().enableAppInstallWhiteList();
```


3.2.2 disableAppInstallWhiteList

Prototype: `boolean disableAppInstallWhiteList()`

Function:

Don't use the app installation whitelist to restrict app installations. Return true if disabled successfully, false otherwise.

The app installation whitelist stores the package names of all the apps that are allowed to be installed, any app that attempts to install will be rejected if system cannot find a matching package name in the whitelist.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

boolean disabled = KbCommonManager.getInstance().disableAppInstallWhiteList();
```

3.2.3 addAppToInstallWhiteList

Prototype: `boolean addAppToInstallWhiteList(String appPackageName)`

Function:

Add an app package name to the app installation whitelist. Return true if added successfully, false otherwise.

The app installation whitelist stores the package names of all the apps that are allowed to be installed, any app that attempts to install will be rejected if system cannot find a matching package name in the whitelist.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Add Skype to app installation whitelist.
boolean added = KbCommonManager.getInstance().addAppToInstallWhiteList("com.skype.raider");
```

3.2.4 delAppFromInstallWhiteList

Prototype: `boolean delAppFromInstallWhiteList(String appPackageName)`

Function:

Remove an app package name from the app installation whitelist. Return true if deleted successfully, false otherwise.

The app installation whitelist stores the package names of all the apps that are allowed to be installed, any app that attempts to install will be rejected if system cannot find a matching package name in the whitelist.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Remove Skype from app installation whitelist.

boolean deleted = KbCommonManager.getInstance().delAppFromInstallWhiteList("com.skype.raider");
```

3.2.5 getAppInstallWhiteList

Prototype: `List<String> getAppInstallWhiteList()`

Function:

Get all app package names stored in the app installation whitelist. Return null is the app installation whitelist is empty.

The app installation whitelist stores the package names of all the apps that are allowed to be installed, any app that attempts to install will be rejected if system cannot find a matching package name in the whitelist.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

List<String> whiteApps = KbCommonManager.getInstance().getAppInstallWhiteList();
```

3.2.6 enableAppUninstallBlackList

Prototype: `boolean enableAppUninstallBlackList()`

Function:

Use the app uninstallation blacklist to restrict app uninstallations. Return true if enabled successfully, false otherwise.

The app uninstallation blacklist stores the package names of all the apps that are not allowed to be uninstalled. If the package name of an installed app matches any of the package names in the app uninstallation blacklist, then this app cannot be uninstalled.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

boolean enabled = KbCommonManager.getInstance().enableAppUninstallBlackList();
```

3.2.7 disableAppUninstallBlackList

Prototype: `boolean disableAppUninstallBlackList()`

Function:

Don't use the app uninstallation blacklist to restrict app uninstallations. Return true if disabled successfully, false otherwise.

The app uninstallation blacklist stores the package names of all the apps that are not allowed to be

uninstalled. If the package name of an installed app matches any of the package names in the app uninstallation blacklist, then this app cannot be uninstalled.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

boolean disabled = KbCommonManager.getInstance().disableAppUninstallBlackList();
```

3.2.8 addAppToUninstallBlackList

Prototype: `boolean addAppToUninstallBlackList(String appPackageName)`

Function:

Add an app package name to the app uninstallation blacklist. Return true if added successfully, false otherwise.

The app uninstallation blacklist stores the package names of all the apps that are not allowed to be uninstalled. If the package name of an installed app matches any of the package names in the app uninstallation blacklist, then this app cannot be uninstalled.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Add Skype to app uninstallation blacklist.
boolean added = KbCommonManager.getInstance().addAppToUninstallBlackList("com.skype.raider");
```

3.2.9 delAppFromUninstallBlackList

Prototype: `boolean delAppFromUninstallBlackList(String appPackageName)`

Function:

Remove an app package name from the app uninstallation blacklist. Return true if deleted successfully, false otherwise.

The app uninstallation blacklist stores the package names of all the apps that are not allowed to be uninstalled. If the package name of an installed app matches any of the package names in the app uninstallation blacklist, then this app cannot be uninstalled.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Remove Skype from app uninstallation blacklist.
boolean deleted = KbCommonManager.getInstance().delAppFromUninstallBlackList("com.skype.raider");
```

3.2.10 getAppUninstallBlackList

Prototype: `List<String> getAppUninstallBlackList()`

Function:

Get all app package names stored in the app uninstallation blacklist. Return null is the app uninstallation blacklist is empty.

The app uninstallation blacklist stores the package names of all the apps that are not allowed to be uninstalled. If the package name of an installed app matches any of the package names in the app uninstallation blacklist, then this app cannot be uninstalled.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

List<String> uninstallBlackApps = KbCommonManager.getInstance().getAppUninstallBlackList();
```

3.2.11 setAppSignatureVerifyEnabled

Prototype: `boolean setAppSignatureVerifyEnabled(boolean enable, String password)`

Function:

App installation requires verification of app's SHA1 signature or not. Return true if enabled successfully, false otherwise.

This operation requires a password, please contact us for the default password.

System stores a list of app installation trusted SHA1 signatures, only apps that signed by one of these stored SHA1 signatures can be installed if app installation requires verification of app's SHA1 signature feature is enabled.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

boolean enabled = KbCommonManager.getInstance().setAppSignatureVerifyEnabled(true, "ctk123456");
```

3.2.12 setAppSignatureVerifyPassword

Prototype:

`boolean setAppSignatureVerifyPassword(String oldPassword, String newPassword)`

Function:

Change the password required by `setAppSignatureVerifyEnabled`, `addAppSignatureCertificate`, `delAppSignatureCertificate` and `getAppSignatureCertificate`. Return true if changed successfully, false otherwise.

This operation requires a password, please contact us for the default password.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;
```

```
boolean changed = KbCommonManager.getInstance().setAppSignatureVerifyPassword("ctk123456",  
"new123456");
```

3.2.13 addAppSignatureCertificate

Prototype: `boolean addAppSignatureCertificate(String sha1Signature, String password)`

Function:

Add a SHA1 signature to the app installation trusted SHA1 signatures list. Return true if added successfully, false otherwise.

This operation requires a password, please contact us for the default password.

System stores a list of app installation trusted SHA1 signatures, only apps that signed by one of these stored SHA1 signatures can be installed if app installation requires verification of app's SHA1 signature feature is enabled.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;
```

```
//Add Skype SHA1 signature to the app installation trusted SHA1 signatures list
```

```
//The SHA1 signature of Skype is 77:18:07:D1:B8:41:4D:69:89:E7:D8:EF:0B:97:97:24:3B:93:1F:95
```

```
//You should delete all ":" while calling addAppSignatureCertificate
```

```
boolean added = KbCommonManager.getInstance()
```

```
.addAppSignatureCertificate("771807D1B8414D6989E7D8EF0B9797243B931F95", "ctk123456");
```

3.2.14 delAppSignatureCertificate

Prototype: `boolean delAppSignatureCertificate(String sha1Signature, String password)`

Function:

Remove a SHA1 signature you added through `addAppSignatureCertificate` from the app installation trusted SHA1 signatures list. Return true if deleted successfully, false otherwise.

This operation requires a password, please contact us for the default password.

System stores a list of app installation trusted SHA1 signatures, only apps that signed by one of these stored SHA1 signatures can be installed if app installation requires verification of app's SHA1 signature feature is enabled.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;
```

```
//Remove Skype SHA1 signature from the app installation trusted SHA1 signatures list
```

```
boolean deleted = KbCommonManager.getInstance()
```

```
.delAppSignatureCertificate("771807D1B8414D6989E7D8EF0B9797243B931F95", "ctk123456");
```

3.2.15 getAppSignatureCertificate

Prototype: `List<String> getAppSignatureCertificate(String password)`

Function:

Get all app installation trusted SHA1 signatures. Return null is the app installation trusted SHA1 signatures list is empty.

This operation requires a password, please contact us for the default password.

System stores a list of app installation trusted SHA1 signatures, only apps that signed by one of these stored SHA1 signatures can be installed if app installation requires verification of app's SHA1 signature feature is enabled.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

List<String> trustedSigns = KbCommonManager.getInstance().getAppSignatureCertificate("ctk123456");
```

3.2.16 installAppSilent

Prototype: `void installAppSilent(String apkAbsolutePath, IInstallerListener callback)`

Function:

Install an app in background, you can get the install result from the 'callback'.

For more details, please refer to the sample code.

Sample code:

```
import com.kingberry.common.IInstallerListener;
import com.kingberry.common.api.KbCommonManager;

//Install /sdcard/Download/Skype.apk in background
KbCommonManager.getInstance().installAppSilent("/sdcard/Download/Skype.apk",
    new IInstallerListener.Stub() {
        @Override
        public void onInstallComplete(String packageName, int returnCode, String message) {
            if (returnCode == 0) {
                //Install successfully
            } else {
                //Failed to install, get more information from returnCode and message
            }
        }

        @Override
        public void onUninstallComplete(String packageName, int returnCode, String message) {
            //nothing to do
        }
    }
```

```
}  
});
```

3.2.17 installApplication

Prototype:

```
void installApplication(String apkAbsolutePath, IInstallerListener callback, boolean open)
```

Function:

Install an app in background, you can get the install result from the 'callback'. If the 'open' is set to true, system will open the installed app after install completed if the app contains an {Intent#CATEGORY_INFO} activity or an {Intent#CATEGORY_LAUNCHER} activity.

For more details, please refer to the sample code.

Sample code:

```
import com.kingberry.common.IInstallerListener;  
import com.kingberry.common.api.KbCommonManager;  
  
//Install /sdcard/Download/Skype.apk in background and then open it  
KbCommonManager.getInstance().installApplication("/sdcard/Download/Skype.apk",  
    new IInstallerListener.Stub() {  
        @Override  
        public void onInstallComplete(String packageName, int returnCode, String message) {  
            if (returnCode == 0) {  
                //Install successfully  
            } else {  
                //Failed to install, get more information fro returnCode and message  
            }  
        }  
  
        @Override  
        public void onUninstallComplete(String packageName, int returnCode, String message) {  
            //nothing to do  
        }  
    }, true);
```

3.2.18 uninstallAppSilent

Prototype:

```
void uninstallAppSilent(String packageName, IInstallerListener callback)
```

Function:

Install an app in background, you can get the install result from the 'callback'.

For more details, please refer to the sample code.

Sample code:

```
import com.kingberry.common.IInstallerListener;
import com.kingberry.common.api.KbCommonManager;

//Uninstall Skype in background
KbCommonManager.getInstance().uninstallAppSilent("com.skype.raider",
    new IInstallerListener.Stub() {
        @Override
        public void onInstallComplete(String packageName, int returnCode, String message) {
            //nothing to do
        }

        @Override
        public void onUninstallComplete(String packageName, int returnCode, String message) {
            if (returnCode == 0) {
                //Uninstall successfully
            } else {
                //Failed to uninstall, get more information fro returnCode and message
            }
        }
    });
```

3.2.19 setAppOpsMode

Prototype: `void setAppOpsMode(String opStr, String packageName, boolean grant)`

Function: Grant some special accesses by default to app. For example, allow app to display over other apps. There are more usages in Chapter 4.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Allow WeChat to display over other apps
KbCommonManager.getInstance()
    .setAppOpsMode(AppOpsManager.OPSTR_SYSTEM_ALERT_WINDOW, "com.tencent.mm", true);
```

3.3 Network control/Network information

3.3.1 setWifiEnabled

Prototype: `void setWifiEnabled(boolean enable)`

Function: Turn on/off the WiFi. 'enable' #true to turn on, #false to turn off.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Turn on WiFi
KbCommonManager.getInstance().setWifiEnabled(true);

//Turn off WiFi
KbCommonManager.getInstance().setWifiEnabled(false);
```

3.3.2 setMobileNetworkEnabled

Prototype: void setMobileNetworkEnabled(boolean enable)

Function: Turn on/off the mobile network. 'enable' #true to turn on, #false to turn off.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Turn on mobile network
KbCommonManager.getInstance().setMobileNetworkEnabled(true);

//Turn off mobile network
KbCommonManager.getInstance().setMobileNetworkEnabled(false);
```

3.3.3 enableBluetooth

Prototype: void enableBluetooth(boolean enable)

Function: Turn on/off the Bluetooth. 'enable' #true to turn on, #false to turn off.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Turn on Bluetooth
KbCommonManager.getInstance().enableBluetooth(true);

//Turn off Bluetooth
KbCommonManager.getInstance().enableBluetooth(false);
```

3.3.4 setAirplaneModeOn

Prototype: void setAirplaneModeOn(boolean on)

Function: Turn on/off the airplane mode. 'on' #true to turn on, #false to turn off.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Turn on airplane mode
KbCommonManager.getInstance().setAirplaneModeOn(true);

//Turn off airplane mode
KbCommonManager.getInstance().setAirplaneModeOn(false);
```

3.3.5 getWlanMacAddress

Prototype: `String getWlanMacAddress()`

Function: Get the device WiFi MAC address. An empty string will be returned if the WiFi is not turned on or the device WiFi MAC address is unavailable.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

String wifiMacAddr = KbCommonManager.getInstance().getWlanMacAddress();
```

3.3.6 getWlanIPAddress

Prototype: `String getWlanIPAddress()`

Function: Get the device WiFi IPv4 address. An empty string will be returned if the device WiFi IPv4 address is unavailable.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

String wifiIpv4Addr = KbCommonManager.getInstance().getWlanIPAddress();
```

3.3.7 getEth0MacAddress

Prototype: `String getEth0MacAddress()`

Function: Get the device Ethernet MAC address. An empty string will be returned if the device Ethernet MAC address is unavailable.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

String ethMacAddr = KbCommonManager.getInstance().getEth0MacAddress();
```

3.3.8 getEth0IPAddress

Prototype: `String getEth0IPAddress()`

Function: Get the device Ethernet IPv4 address. An empty string will be returned if the device Ethernet IPv4 address is unavailable.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

String ethIpv4Addr = KbCommonManager.getInstance().getEth0IPAddress();
```

3.3.9 getEth0IPMethod

Prototype: `int getEth0IPMethod()`

Function: Get Ethernet IP assignment method. If return 0, it's DHCP; if return 1, it's STATIC

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Get Ethernet IP assignment method
int ethIpAssignment = KbCommonManager.getInstance().getEth0IPMethod();
if (ethIpAssignment == 0) {
    //IpAssignment.DHCP
}
if (ethIpAssignment == 1) {
    //IpAssignment.STATIC
}
```

3.3.10 setEth0Configuration

Prototype: `boolean setEth0Configuration(int ipAssignmentMethod, String ipAddr, String gatewayAddr, String dns1, String dns2)`

Function:

Configure the Ethernet. Return true if configured successfully, false otherwise.

`ipAssignmentMethod`: 0(DHCP) or 1(STATIC)

`ipAddr`: IPv4 format address string for IP(e.g. 192.168.1.128)

`gatewayAddr`: IPv4 format address string for Gateway(e.g. 192.168.1.1)

`dns1, dns2`: IPv4 format address string for DNS(e.g. 8.8.8.8)

Sample code:

```
import com.kingberry.common.api.KbCommonManager;
```

```
//Assign a static IP to the Ethernet
boolean staticConfigured = KbCommonManager.getInstance()
    .setEth0Configuration(1, "192.168.1.128", "192.168.1.1", "8.8.8.8", "8.8.4.4");

//DHCP assigns a IP for the Ethernet
boolean dhcpConfigured = KbCommonManager.getInstance()
    .setEth0Configuration(0, null, null, null, null);
```

3.3.11 setDefaultDataSubId

Prototype: `void setDefaultDataSubId(int subId)`

Function: Set the default mobile data SIM card with subscription ID.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

KbCommonManager.getInstance().setDefaultDataSubId(1);
```

3.3.12 getCellularIMEI

Prototype: `String getCellularIMEI(int slotIndex)`

Function: Get the IMEI string of specified SIM slot. slotIndex is 0 to get SIM 1 slot IMEI, slotIndex is 1 to get SIM 2 slot IMEI. Return the IMEI, or an empty string if the ICC ID is not available.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

//Get the IMEI of SIM 1 slot
String sim1IMEI = KbCommonManager.getInstance().getCellularIMEI(0);
if (TextUtils.isEmpty(sim1IMEI)) {
    //IMEI of SIM 1 slot is not available
}
```

3.3.13 getCellularICCID

Prototype: `String getCellularICCID(int slotIndex)`

Function: Get the ICC ID string of specified SIM card. slotIndex is 0 to get SIM card 1 ICC ID, slotIndex is 1 to get SIM card 2 ICC ID. Return the ICC ID, or an empty string if the ICC ID is not available.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;
```

```
//Get the ICC ID of SIM card 1
String sim1ICCID = KbCommonManager.getInstance().getCellularICCID(0);
if (TextUtils.isEmpty(sim1ICCID)) {
    //ICC ID of SIM card 1 is not available
}
```

3.3.14 setPreferredAPN

Prototype: `boolean setPreferredAPN(int apnId)`

Function: Set the preferred APN. Return true if set successfully, false otherwise. This method is usually used in conjunction with the `insertAPN` method.

Sample code:

```
import com.kingberry.common.api.KbCommonManager;

boolean set = KbCommonManager.getInstance().setPreferredAPN(1);
```

3.3.15 insertAPN

Prototype: `int insertAPN(`
 `String name,`
 `String apn,`
 `String type,`
 `String proxy,`
 `String port,`
 `String server,`
 `String mcc,`
 `String mnc,`
 `String numeric,`
 `String mmsc,`
 `String mmsproxy,`
 `String mmsport,`
 `String user,`
 `String password)`

Function: Add an APN configuration. Return the ID if added successfully, or -1 if failed to add.

3.3.16 deleteAPN

Prototype: `void deleteAPN(int apnId)`

Function: Delete an APN configuration with its ID.

3.3.17 **getCurrentPreferredApnId**

Prototype: `int getCurrentPreferredApnId(int subId)`

Function: Get the preferred APN ID of specified subscription ID.