SDK MANUAL

PTA0157 SDK

3nStar
ONE BRAND, ONE SOLUTION

Disclaimer of Warranty

This document is provided “as is”. 3nStar Inc. (the “company”) does not provide any express or implied representations or warranties as to the accuracy, reliability, completeness, marketability, specific purpose and non-infringement of any statements, information, and content in this document. This document is only used as a reference for usage guidance.

Due to product version upgrades or other reasons, this document may be updated or modified from time to time without any notice.

Contents

1. Overview

This document mainly introduces the different display schemes supported on the Rockchip SDK platform to facilitate customer development. It is suitable for the Android version of Android11.0 and is suitable for the chip RK3399/RK3288/RK3326/PX30/RK3568.

2. Introduction to the abnormal display scheme

There are currently two different display schemes: Android Presentation and Android Activity specify screen startup.

Android presentation requires the corresponding interface to be called in APP development to make the specified view (Presentation view is a special dialog type view) displayed on the secondary screen.

Android Activity specifies the screen to start, and the APP can use the display id parameter to display directly on the corresponding screen when starting the activity.

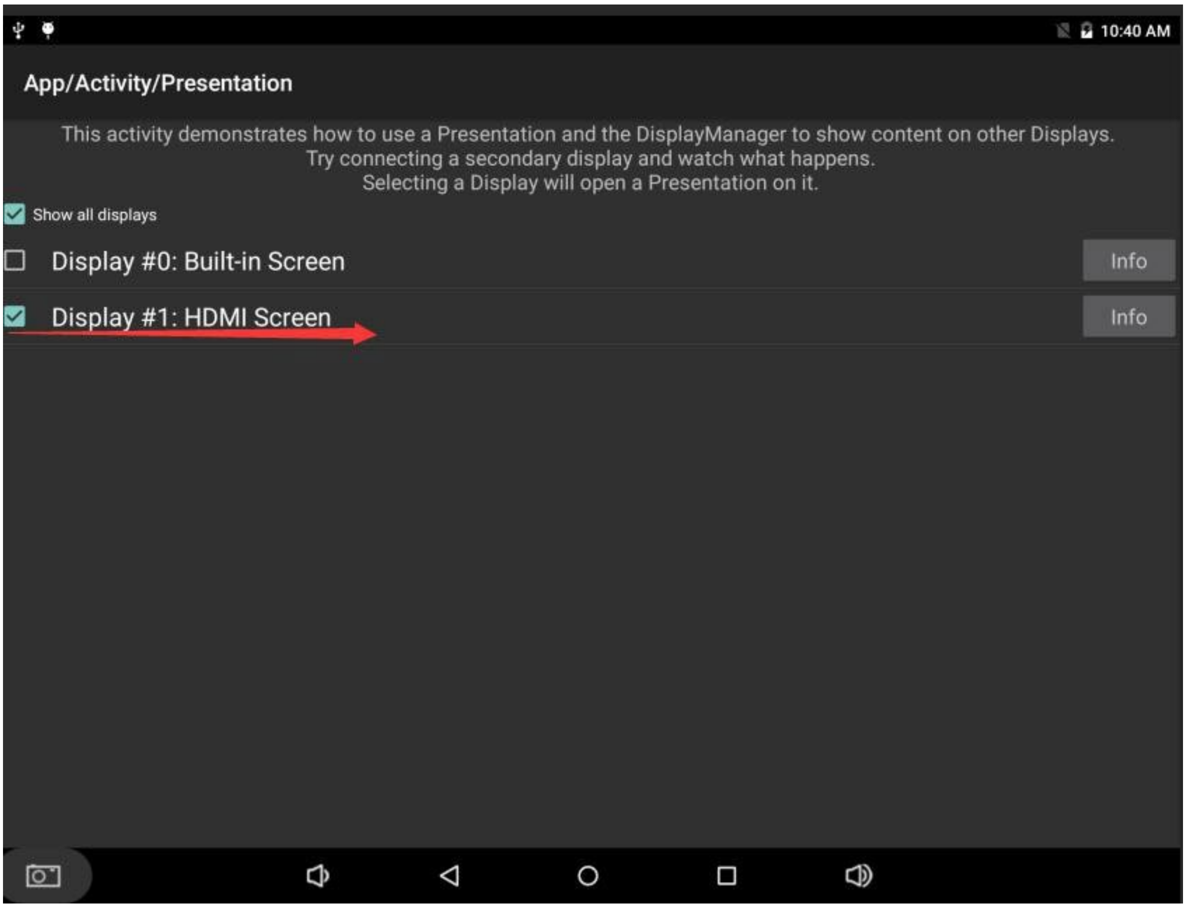
The main difference between the two is：

1. The activity of the former needs to be developed independently and the content that needs to be displayed is cast to the secondary screen. The latter can call the command line or system interface to cast the activity of the third-party app to the secondary screen without the source code.

2. The former has only one activity at the top level, and the specified content is displayed on the secondary screen through a special dialog. The latter is that two activities are displayed on the main and secondary screens.

2.1 Android Presentation

There are already related demos for this interface in the SDK. Please enter the development/samples/ApiDemos directory and compile and generate the corresponding apk. After installing the apk, click the App->Activity->Presentation option to enter the Presentation calling interface. In this interface, you need to click the checkbox option on the secondary screen below to display the corresponding picture on the secondary screen.



App/Activity/Presentation

This activity demonstrates how to use a Presentation and the DisplayManager to show content on other Displays.
Try connecting a secondary display and watch what happens.
Selecting a Display will open a Presentation on it.

☑ Show all displays

☐ Display #0: Built-in Screen

☑ Display #1: HDMI Screen

The specific code is located in the following path：

development/samples/ApiDemos/src/com/example/android/apis/app/PresentationActivity.java

2.2 Android Activity specifies the screen to start

Set the display id of the specified screen in the startActivity interface parameter, and the activity will start the display directly on the specified screen.

Activity's multi-monitor support requires device support <feature name="android.software.activities_ on_secondary_displays"/>.At the same time, the application or activity needs to support the split-screen attribute, that is, set the new attribute android: resizeableActivity="true" under the <applicat ion> or <activity> tags. After you switch to Android N, the default value of android: resizeableActivity is true.

ActivityOptions provides two new functions to support multiple monitors：

setLaunchDisplayId() specifies which display the activity should be displayed on after it starts.

getLaunchDisplayId() returns the current startup display of the operation component.

setLaunchDisplayId() usage example：

In the example, the MediaRouter interface is used to obtain the display id of the secondary screen. Similarly, the display id of the response can also be obtained using the DisplayManager interface.

```
private void showSecondByActivity(Context context){
ActivityOptions options = ActivityOptions.makeBasic();
MediaRouter mediaRouter = (MediaRouter)
                      context.getSystemService(Context.MEDIA_ROUTER_SERVICE);
          MediaRouter.RouteInfo route =
                        mediaRouter.getSelectedRoute(MediaRouter.ROUTE_TYPE_LIVE_VIDE
          O); if (route != null) {
              Display presentationDisplay = route.getPresentationDisplay();
              options.setLaunchDisplayId(presentationDisplay.getDisplayId());
              //options.
              Intent intent = new
              Intent(“android.intent.action.MUSIC_PLAYER”);
              intent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
              startActivity(intent, options.toBundle());
          }
  }
```

At the same time, the adb shell has also been extended to support multiple monitors.The shell start command can now be used to start the operation component and specify the target display of the operation component：

`adb shell am start --display <display_id> <activity_name>`

For example: adb shell am start --display 1 com.android.settings/.Settings (Start the settings interface to the specified secondary screen)

3 Dual-screen hetero-touch configuration

3.1 Principle description

From the following Montion Event, you can see that the event reported by the system contains a variable of Display ID, and the system will distribute the event to the corresponding main screen or secondary screen based on the value of this display ID.

```
MotionEvent(deviceId=2,source=0x00001002,action=1,actionButton=0x00000000,flags=0x000
00000,metaState=0x00000000,buttonState=0x00000000,edgeFlags=0x00000000,xPrecision=8.
5,yPrecision=8.9,displayId=0, pointers=[0: (496.1, 1029.6)]), policyFlags=0x62000000,
age=13734.7ms
```

The Display-ID value of the touch event is frameworks/native/services/inputflinger/Eventhub.cpp of openDeviceLocked In, the system determines that if it is the TP of the secondary screen, the touch device will be set

With the INPUT_DEVICE_CLASS_EXTUAL attribute, the corresponding touch event will be passed to the secondary screen window.Whether it is an I2C, USB or Bluetooth interface, you only need to set the touch device of the secondary screen to this attribute to realize the touch of the secondary screen.

3.2 How to configure the secondary screen TP

There are currently two ways to configure the secondary screen TP: 1 Modify EventHub.cpp code. 2 Configure the IDC file of the touch screen.

3.2.1 Modify the code

By modifying EventHub.The CPP code is as follows. In the isExternalDeviceLocked function, it is determined that the device name is the specified secondary screen device (for example, the secondary screen TP name is gsl3673), then true is returned.

```
   bool EventHub::isExternalDeviceLocked(Device* device) {
+    //example for set external device.
+     const char *USB_DEVICE_NAME ="gsl3673";
+     if(strcmp(device->identifier.name.string(),USB_DEVICE_NAME)==0){
+     ALOGD(" %d   name:   \"%s\"\n",__LINE__, device->identifier.name.string());
+       return true;
+     }
        if (device-
            >configuration
            ) {bool value;
            if (device->configuration-
    >tryGetProperty(String8("device.internal"), value)) {return
    !value;
            }
      }
      return device->identifier.bus == BUS_USB || device->identifier.bus == BUS_BLUETOOTH;
  }
```

3.2.2 Configure the IDC file of the specified touch screen device

Set device in the IDC file corresponding to the touch screen device.internal=0

For specific information, please refer to Google's official website ：
https://source.android.com/devices/input/input-device-configuration-files

3.3 Check whether the configuration is successful

You can use the dumpsys input command to check whether INPUT_DEVICE_CLASS_EXTERNAL is successfully set. In the following dumpsys information, it is determined whether the touch device Device 3: Nuvoton HID Transfer has been set to a secondary screen based on IsExternal.

```
Device 2: ilitek_ts
Generation: 14
IsExternal: false
HasMic:false
Sources:0x0000
1002
KeyboardType: 0
Device 3: Nuvoton
    HID Transfer
    Generation: 17
IsExternal: true
HasMic:false
Sources:0x0000
1002
KeyboardType: 0
```

4 sub-screen direction settings

4.1Rk3568 sub-screen direction configuration

RK3568 has dual-screen and three-screen usage scenarios：

In a dual-screen scene, the secondary screen passes persist.sys.rotation.einit-1 (attribute value is 0,1,2,3) The attribute is set in different directions, such as setprop persist.sys.rotation.einit-1 1, set the secondary screen to rotate 90 degrees. After setting this property, you need to restart the machine for verification.

In a three-screen scenario, there is one screen as the main screen and the other two screens as the secondary screen.Taking the following dumpsys input information as an example, the viewports corresponding to the three displays in the system are as follows. Among them, the Viewport INTERNAL is the main screen and the two Viewport EXTERNAL are the secondary screens. Each Viewport has its own unique ID and port value. The two external secondary screens can be distinguished according to the value of the Unique ID or port.

```
Viewport INTERNAL: displayId=0, uniqueId=local:0, port=0, orientation=0, logicalFrame=[0, 0, 1080, 1920], physicalFrame=[0, 0, 1080, 1920], deviceSize=[1080, 1920], isActive=[1]

Viewport EXTERNAL: displayId=0, uniqueId=local:1, port=1, orientation=1, logicalFrame=[0, 0, 1080, 1920], physicalFrame=[0, 0, 720, 1280], deviceSize=[720, 1280], isActive=[1]

Viewport EXTERNAL: displayId=0, uniqueId=local:2, port=2, orientation=0, logicalFrame=[0, 0, 1080, 1920], physicalFrame=[0, 0, 1440, 900], deviceSize=[1440, 900], isActive=[1]
```

The two secondary screens need to be set to persist separately.sys.rotation.einit-1，persist.sys.rotation.einit-2 (the attribute value is 0,1,2,3) These two attributes control the direction of the corresponding screen.(These two attributes correspond to the two secondary screens of port1 and port2 respectively).

4.2 RK3288, RK3399, PX30, RK3326 sub-screen direction configuration

Set the property persist.sys.rotation.einit (0,1,2,3) controls the direction of the secondary screen.

For example, setprop persist.sys.rotation.einit 1, set the secondary screen to rotate 90 degrees. After setting this property, you need to restart the machine for verification.

5 Other development configurations

5.1 Support input method display on the secondary screen

device/rockchip/common/display_settings.xml
Set the corresponding screen shouldShowIme to true, and the following configuration supports the input method to be displayed on the secondary screen of local:1.

```
    <?xml version='1.0' encoding='utf-8' standalone='yes' ?>
    <display-settings>
    <config identifier="0" />
    <display
    name="local:1"
    shouldShowIme="tru
    e"
    forcedDensity="240"/
    >
</display-settings>
```

5.2 Secondary screen DPI setting

in device/rockchip/common/display_settings.xml In the setting forcedDensity， For example

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<display-settings>
<config identifier="0" />
<display
    name="local:1"
    shouldShowIme="tru
    e"
    forcedDensity="240"/
    >
</display-settings>
```

5.3 Switch the display of the main and secondary screens of the mouse

Set up sys.mouse.Presentation is 1, turn on this function.When the abnormal display state is displayed, the boot mouse is displayed on the main screen by default. When the mouse moves to the edge of the screen, it will automatically switch to the center of the secondary screen to display

ONE
BRAND
ONE
SOLUTION

3nStar
ONE BRAND, ONE SOLUTION

www.3nstar.com | 3nstar