



KOZEN Component
SDK Development Documentation
V1.5

Shanghai Xiangcheng Communication Technology Co., Ltd.

Table of Contents

● 1. Revision History	4
● 2. Overview	5
2.1 Introduction	5
2.2 Android version and IDE version supported by the SDK	5
2.3 Feature Introduction	5
2.3.1 Component SDK Engine Module	5
2.3.2 SecondaryScreen Module	5
2.3.3 Keyboard Module	5
2.4 Importing the SDK	6
2.5 Initializing the SDK	6
● 3. API Interface Introduction	7
➤ 3.1 Component SDK initialization	7
3.1.1 Initialize the SDK	7
3.1.2 Reverse Initialization SDK	7
3.1.3 Get the secondary screen module	7
3.1.4 Get the keyboard module	8
-- ComponentService instance callback - InitListener --	8
3.1.5 Initialization results	8
➤ 3.2 Keyboard Operation module	8
-- Get keyboard operation module - getKeyboardManager --	8
3.2.1 Start listening to physical keys and return key values	9
3.2.2 Stop listening to physical keys	9
3.2.3 Enable or disable physical key press sound	9
3.2.4 Get physical key press sound status	9
-- Keyboard callback - InputCallback--	10
3.2.3 Initialization results	10
➤ 3.3 Secondary Screen Operation module	10
-- Get secondary screen operation module - getSecondaryScreenManager--	10
3.3.1 Display a single image	11
3.3.2 Image slideshow	11
3.3.3 Display a video	12
3.3.4 Power on/off the screen	12
3.3.5 Set the screen brightness	13
3.3.6 Set the boot logo image	13
3.3.7 Get the screen resolution of the secondary screen	13
3.3.8 Display a custom view	14
3.3.9 Display a custom view with optional image slideshow support	14
3.3.10 Display the default wallpaper	14
3.3.11 Get power-on status of the secondary screen	15
3.3.12 Display the default wallpaper	15
-- Secondary screen listener - IResultCallback --	15
3.3.13 View display success	15

3.3.14 View display failure	16
● 4. Error Code Definition	16
4.1 CommonError	16
4.2 KeyboardError	16
4.5 SecondaryScreenError	16
● 5. Entity Class Definition	17
5.1 KeyboardConstant.KeyCode	17
5.2 com.kozen.component.constant.KeyboardConstant.KeyAction	17

● 1. Revision History

Version	Release	Modify Record	Adapted SDK version	Author
1.5	2026/01/29	<ul style="list-style-type: none">● Add default wallpaper display on the secondary screen.● Add support for application paths for displaying images, videos, and logos.● Add support to get the secondary screen power-on status.● Add support to get the secondary screen brightness.● Save the secondary screen brightness after reboot.	ComponentService1.3.x	Johnny
1.4	2025/09/09	Add PCI Statement	ComponentService1.2.x	Johnny
1.3	2025/08/12	<ul style="list-style-type: none">● Add keyboard voice broadcast● Fixed documentation errors regarding CommonError issues	ComponentService1.2.x	Johnny
1.2	2025/07/03	Secondary screen image display with GIF support	ComponentService1.1.x	Johnny
1.1	2025/04/25	Switch to English version	ComponentService1.0.x	Johnny
1.0	2025/04/24	Renamed setDefaultBackground to setBootLogo	ComponentService1.0.x	Yanan.Zhu
0.4	2025/04/24	Updated key definitions in Key Module	ComponentService1.0.x	Yue.Cui
0.3	2025/04/22	Functionality adjustments and optimizations	ComponentService1.0.x	Yanan.Zhu
0.2	2025/04/21	Add Key Module	ComponentService1.0.x	Yue.Cui
0.1	2025/04/15	Added Secondary Screen Module	ComponentService1.0.x	Yue.Cui

● 2. Overview

2.1 Introduction

KozenComponentSDK Service is a hardware-based API SDK provided by KOZEN, specifically designed for Java and Android developers. This SDK enables developers to quickly access hardware operation interfaces, facilitating efficient business logic implementation.

This document serves as the Component SDK Service API reference.

2.2 Android version and IDE version supported by the SDK

System environment	Platform	Compile environment
Android 6.0 and above	ARM 64, ARM 32	Android Studio, IntelliJ

2.3 Feature Introduction

2.3.1 Component SDK Engine Module

- This module handles SDK initialization and provides access to various module operation classes.
- Operation class object: ComponentEngine

2.3.2 SecondaryScreen Module

- This module handles functions related to the secondary screen.
- Operation class object: IScreen
- Example to get the module operation class:

JAVA: ComponentEngine.INSTANCE.getScreenManager()

Kotlin: ComponentEngine.screenManager

2.3.3 Keyboard Module

- This module handles functions related to the keyboard.
- Operation class object: IKeyboard
- Example to get the module operation class:

JAVA: ComponentEngine.INSTANCE.getKeyboardManager()

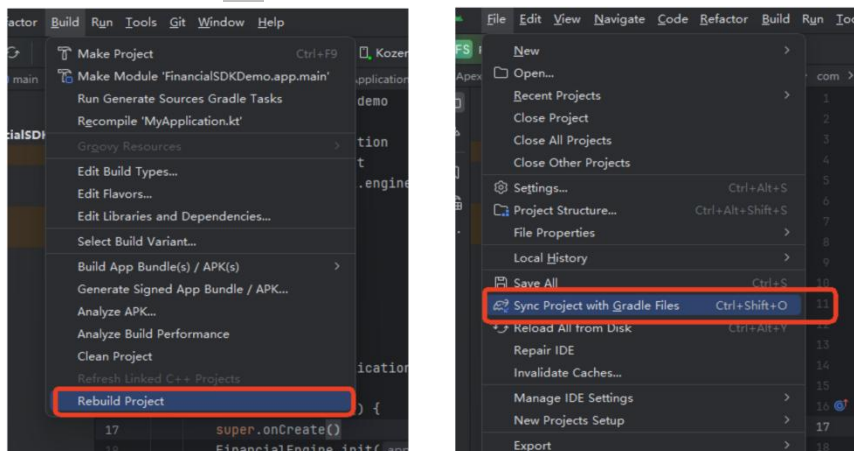
Kotlin: ComponentEngine.keyboardManager

2.4 Importing the SDK

Local Dependency: Place the `ComponentLib-xxx-release.aar` file in the `libs` directory of your Android Studio project. Add the following code to the `build.gradle` file:

```
kotlin
dependencies {
    implementation(files("/libs/ComponentLib_0.0.2_release.aar"))
}
```

After importing the `.aar` file, sync and rebuild the project.



2.5 Initializing the SDK

Please initialize the Component SDK in your Application. Example:

```
kotlin
ComponentEngine.init(this) { code, msg ->
    val message = if (code == 0) {
        "init success"
    } else {
        "init failed, $msg"
    }

    Toast.makeText(this, message, Toast.LENGTH_SHORT).show()
}
```

If initialization is successful, the callback will return `result == 0`.

If initialization fails, the callback will return `result == -1`.

After the Component SDK is successfully initialized, use ComponentEngine to obtain the operation objects for each module. Otherwise, an error code -10001 (service not connected) will be thrown.

If the component service is disconnected during use, the InitListener will be called again with result == -1. Upon receiving this callback, reinitialize the Component SDK.

If an interface throws the error code -10001 (service not connected) during use, reinitialize the Component SDK.

● 3. API Interface Introduction

void deInit()	Reverse Initialization SDK
IKeyboard getKeyboardManager()	Get the keyboard module
IScreen getScreenManager()	Get the secondary screen module
void init(android.content.Context application, InitListener callback)	Initialize the SDK

➤ 3.1 Component SDK initialization

3.1.1 Initialize the SDK

Prototype	void init(android.content.Context application, InitListener callback)
Function	Initialize the SDK
Parameters	Parameter: application - context callback - initialization callback
Return Value	
Notes	SDK initialization with the provided context and callback function.

3.1.2 Reverse Initialization SDK

Prototype	void deInit()
Function	Reverse Initialization SDK
Parameters	
Return Value	
Notes	This method is used to reverse initialization the SDK.

3.1.3 Get the secondary screen module

Prototype	IScreen getScreenManager()
Function	Get the secondary screen module

Parameters	
Return Value	Return: IScreen
Notes	This method retrieves the secondary screen manager module.

3.1.4 Get the keyboard module

Prototype	IKeyboard getKeyboardManager()
Function	Get the keyboard module
Parameters	
Return Value	Return: IKeyboard
Notes	This method retrieves the keyboard manager module.

-- ComponentService instance callback - InitListener --

void onResult(int result, String errorMsg)	Initialization results
--	------------------------

3.1.5 Initialization results

Prototype	void onResult(int result, String errorMsg)
Function	Initialization result
Parameters	Parameters: result - Initialization result. 0: success; other values: failure errorMsg - Detailed error information, refer to CommonError
Return Value	
Notes	This method handles the initialization result, providing success or failure along with an error message.

➤ 3.2 Keyboard Operation module

Keyboard Interface Declaration

The keyboard is designed solely for general input and interaction purposes. It functions as a non-secure module and is not connected via a secure chip, thus it does not have the capability to process or display sensitive information.

To ensure user data security and comply with information security standards and regulatory requirements, it is strictly prohibited to use the keyboard to input any cardholder information, transaction PINs, or other sensitive content, in order to prevent unauthorized interception, tampering, or misuse of such data.

-- Get keyboard operation module - getKeyboardManager --

int startPhysicalKeyboard(InputCallback callback)	Start listening to physical key presses, return key value
---	---

int stopPhysicalKeyboard()	Stop listening to physical key presses
int isKeyButtonVoiceEnable()	Check if the physical key press sound is enabled
int switchKeyButtonVoiceEnable(boolean enable)	Enable or disable physical key press sound

3.2.1 Start listening to physical keys and return key values

Prototype	int startPhysicalKeyboard(InputCallback callback)
Function	Start listening to physical keys and return key values
Parameters	Parameters: callback – Callback triggered by physical key events
Return Value	Return: 0 – Operation succeeded; Others – Operation failed. Specific error codes can be found in CommonError and KeyboardError
Notes	This method starts the listener for physical key events and returns the key values.

3.2.2 Stop listening to physical keys

Prototype	int stopPhysicalKeyboard()
Function	Stop listening to physical keys
Parameters	
Return Value	Return: 0 – Operation succeeded; Others – Operation failed. Specific error codes can be found in CommonError and KeyboardError
Notes	This method stops the listener for physical key events.

3.2.3 Enable or disable physical key press sound

Prototype	int switchKeyButtonVoiceEnable(boolean enable)
Function	Enable or disable physical key press sound
Parameters	Parameters: enable – true: Enable sound; false: Disable sound
Return Value	Return: 0 – Operation succeeded; Others – Operation failed. See CommonError and KeyboardError for details
Notes	Controls whether the device plays a sound when a physical key is pressed.

3.2.4 Get physical key press sound status

Prototype	int isKeyButtonVoiceEnable()
Function	Get physical key press sound status
Parameters	

Return Value	Return: 0 – Sound enabled 1 – Sound disabled Others – Operation failed. See <code>CommonError</code> and <code>KeyboardError</code> for details
Notes	Queries whether the physical key press sound is currently enabled.

-- Keyboard callback – `InputCallback`--

<code>void onKey(KeyboardConstant.KeyCode keyCode, KeyboardConstant.KeyAction action)</code>	Callback for physical keyboard
--	--------------------------------

3.2.3 Initialization results

Prototype	<code>void onKey(KeyboardConstant.KeyCode keyCode, KeyboardConstant.KeyAction action)</code>
Function	Physical key callback
Parameters	Parameters: keyCode – Key value from the <code>KeyCode</code> enum action – Key action from the <code>KeyAction</code> enum. 0: Pressed, 1: Released
Return Value	
Notes	This method handles the callback for physical key events based on the key code and action.

➤ 3.3 Secondary Screen Operation module

Secondary Display Interface Declaration

The secondary display is designed solely for the presentation and interaction of non-critical data. Its interface is a non-secure module and is not connected via a secure chip, thus it does not have the capability to process or display sensitive information.

To ensure user data security and comply with information security standards and regulatory requirements, it is strictly prohibited to display any cardholder information, transaction data, or other sensitive content on the secondary display, in order to prevent unauthorized interception, tampering, or misuse of such data.

-- Get secondary screen operation module – `getSecondaryScreenManager`--

<code>int[] getScreenResolution()</code>	Get the resolution of the secondary screen
<code>int power(boolean on)</code>	Power on/off the screen
<code>int setBrightness(int value)</code>	Set the screen brightness
<code>int setBootLogo(String filePath)</code>	Set the boot logo image
<code>void show(android.view.View view,</code>	Display a custom View

IResultCallback resultCallback)	
int showPic(String picPath)	Display a single image
int showPic(ArrayList<String> picPathList, int intervalSeconds)	Image slideshow
int showVideo(String videoPath)	Display a video
void show(android.view.View view, ArrayList<PicData> picDataList, IResultCallback resultCallback)	Display a custom View with support for both static and animated images
int showWallpaper()	Display the default wallpaper
int getPowerOnStatus()	Get power-on status of the secondary screen
int getBrightness()	Get brightness level of the secondary screen

3.3.1 Display a single image

Prototype	int showPic(String picPath)
Function	Display a single image
Parameters	<p>Parameters:</p> <p>picPath – The image path. Supported image types are bmp, png, jpeg, jpg, gif. Supports APP resources or TF card.</p> <ol style="list-style-type: none"> 1. APP resources <p>Supports drawable and mipmap directories.</p> <p>GIF format images must be placed in the drawable directory.</p> <p>Before using this API, you need to manually place the image into the APP project directory res/drawable or res/mipmap, and pass it as a parameter according to the specified path format.</p> <p>Path format examples: res://drawable/xxx.png, res://mipmap/xxx.png, res://drawable/xxx.gif</p> <ol style="list-style-type: none"> 2. TF card <p>Before using this API, you need to manually place the image into the TF card, and then pass the absolute path of the resource as the parameter, for example: sdcard/xxx/xxx.png</p>
Return Value	<p>Return:</p> <p>0 – Operation succeeded;</p> <p>Others – Operation failed.</p> <p>Specific error codes can be found in SecondaryScreenError and CommonError</p>
Notes	This method displays a single image on the screen.

3.3.2 Image slideshow

Prototype	int showPic(ArrayList<String> picPathList, int intervalSeconds)
Function	Image slideshow
Parameters	<p>Parameters:</p> <p>picPathList – An array of image paths. Supported image types are bmp, png, jpeg, jpg, gif. Supports APP resources or TF card.</p> <ol style="list-style-type: none"> 1. APP resources

	<p>Supports drawable and mipmap directories.</p> <p>GIF format images must be placed in the drawable directory.</p> <p>Before using this API, you need to manually place the images into the APP project directory res/drawable or res/mipmap, and pass them as parameters according to the specified path format.</p> <p>Path format examples: res://drawable/xxx.png, res://mipmap/xxx.png, res://drawable/xxx.gif</p> <p>2. TF card</p> <p>Before using this API, you need to manually place the images into the TF card, and then pass the absolute paths of the resources as parameters.</p> <p>intervalSeconds – Interval in seconds between image transitions</p>
Return Value	<p>Return:</p> <p>0 – Operation succeeded (only if all image types are valid);</p> <p>Others – Operation failed.</p> <p>Specific error codes can be found in SecondaryScreenError and CommonError</p>
Notes	This method starts an image slideshow with the given list of images and interval.

3.3.3 Display a video

Prototype	int showVideo(String videoPath)
Function	Display a video
Parameters	<p>Parameters:</p> <p>videoPath – The video path. The supported video type is mp4. Supports APP resources or TF card.</p> <p>1. APP resources</p> <p>Supports the raw directory.</p> <p>Video resources must be placed in the raw directory.</p> <p>Before using this API, you need to manually place the video resource into the APP project directory res/raw, and pass it as a parameter according to the specified path format.</p> <p>Path format example: res://raw/xxx.mp4</p> <p>2. TF card</p> <p>Before using this API, you need to manually place the video resource into the TF card, and then pass the absolute path of the resource as the parameter</p>
Return Value	<p>Return:</p> <p>0 – Operation succeeded;</p> <p>Others – Operation failed.</p> <p>Specific error codes can be found in SecondaryScreenError and CommonError</p>
Notes	This method plays a video on the screen.

3.3.4 Power on/off the screen

Prototype	int power(boolean on)
Function	Power on/off the screen
Parameters	<p>Parameters:</p> <p>on – true: Power on; false: Power off</p>

Return Value	Return: 0 – Operation succeeded; Others – Operation failed. Specific error codes can be found in <code>SecondaryScreenError</code> and <code>CommonError</code>
Notes	This method controls the power state of the screen.

3.3.5 Set the screen brightness

Prototype	<code>int setBrightness(int value)</code>
Function	Set the screen brightness
Parameters	Parameters: value – Integer value between 0 and 100
Return Value	Return: 0 – Operation succeeded; Others – Operation failed. Specific error codes can be found in <code>SecondaryScreenError</code> and <code>CommonError</code>
Notes	This method adjusts the brightness of the screen.

3.3.6 Set the boot logo image

Prototype	<code>int setBootLogo(String filePath)</code>
Function	Set the boot logo image
Parameters	Parameters: filePath – The path of the replaced image. Supported image types are bmp, png, jpeg, jpg. Supports APP resources or TF card. 1. APP resources Supports drawable and mipmap directories. Before using this API, you need to manually place the image into the APP project directory <code>res/drawable</code> or <code>res/mipmap</code> , and pass it as a parameter according to the specified path format. Path format examples: <code>res://drawable/xxx.png</code> , <code>res://mipmap/xxx.png</code> 2. TF card Before using this API, you need to manually place the image into the TF card, and then pass the absolute path of the resource as the parameter.
Return Value	Return: 0 – Operation succeeded; Others – Operation failed. Specific error codes can be found in <code>SecondaryScreenError</code> and <code>CommonError</code>
Notes	This method sets the default background image on the screen.

3.3.7 Get the screen resolution of the secondary screen

Prototype	<code>int[] getScreenResolution()</code>
Function	Get the screen resolution of the secondary screen
Parameters	

Return Value	Return: int[0] – Horizontal resolution (width); int[1] – Vertical resolution (height)
Notes	This method returns the screen resolution of the secondary screen.

3.3.8 Display a custom view

Prototype	void show(android.view.View view, IResultCallback resultCallback)
Function	Display a custom view
Parameters	Parameters: view – Custom view resultCallback – Callback for the result of showing the custom view
Return Value	
Notes	<ol style="list-style-type: none"> 1. The view size must match the secondary screen resolution, otherwise, the view may display incorrectly and touch events may fail. 2. Scrolling layouts are not supported. 3. On models with touch, only click events are supported; long press, double-click, or other gesture events are not supported 4. After modifying the view, the API needs to be called again to avoid.

3.3.9 Display a custom view with optional image slideshow support

Prototype	void show(android.view.View view, ArrayList<PicData> picDataList, IResultCallback resultCallback)
Function	Display a custom View with optional image slideshow support
Parameters	Parameters: view – Custom view to be displayed picDataList – List of images for slideshow (supports PNG, JPEG, or GIF; one at a time) resultCallback – Callback for view display result
Return Value	
Notes	<ol style="list-style-type: none"> 1. The view size must match the secondary screen resolution, otherwise, the view may display incorrectly and touch events may fail. 2. Scrolling layouts are not supported. 3. On models with touch, only click events are supported; long press, double-click, or other gesture events are not supported 4. After modifying the view, the API needs to be called again to avoid.

3.3.10 Display the default wallpaper

Prototype	int showWallpaper()
Function	Display the default wallpaper
Parameters	
Return Value	Return:

	0 – Operation succeeded Others – Operation failed See SecondaryScreenError and CommonError for details
Notes	This method displays the default wallpaper on the secondary screen.

3.3.11 Get power-on status of the secondary screen

Prototype	int getPowerOnStatus()
Function	Get the power-on status of the secondary screen
Parameters	
Return Value	Return: 0 – Powered on 1 – Powered off Others – Operation failed. See SecondaryScreenError and CommonError for details
Notes	This method checks whether the secondary screen is currently powered on or off.

3.3.12 Display the default wallpaper

Prototype	int getBrightness()
Function	Get the brightness level of the secondary screen
Parameters	
Return Value	Return: Brightness value (0 – 100) if successful Others – Operation failed. See SecondaryScreenError and CommonError for details
Notes	This method retrieves the current brightness setting of the secondary screen

-- Secondary screen listener – IResultCallback --

void onFailure(int errorCode, String msg)	View display failed
void onSuccess()	View display succeeded

3.3.13 View display success

Prototype	void onSuccess()
Function	View display success
Parameters	
Return Value	
Notes	This method is called when the view is successfully displayed.

3.3.14 View display failure

Prototype	void onFailure(int errorCode, String msg)
Function	View display failure
Parameters	Parameters: errorCode – Error code, refer to SecondaryScreenError for meanings msg – Error details
Return Value	
Notes	This method is called when the view fails to display, providing error code and details.

● 4. Error Code Definition

4.1 CommonError

Error Code	Error Description	Error Value
FEATURE_NOPERMISSION	Feature no-permission	-10011
DEVICE_BUSY	Device is busy	-10005
FEATURE_UNSUPPORTED	Feature unsupported	-10003
PARAMETERS_INVALID	Parameters invalid	-10002
SERVICE_DISCONNECT	Services are not connected, please initialize financial services	-10001
VERSION_NOT_MATCH	Version mismatch	-10004

4.2 KeyboardError

Error Code	Error Description	Error Value
KEYBOARD_UNKNOWN	Unknown error	-30001
KEYCODE_UNDEFINED	Key not defined	-30002

4.5 SecondaryScreenError

Error Code	Error Description	Error Value
VICE_SCREEN_CANCELED	Operation terminated	-20005
VICE_SCREEN_DECODE_FAILED	Decoding failed	-20003
VICE_SCREEN_INVALID_DATA	Invalid data	-20002
VICE_SCREEN_TIMEOUT	Interface call timeout	-20001
VICE_SCREEN_UNKNOWN	Unknown error	-20004

● 5. Entity Class Definition

5.1 KeyboardConstant.KeyCode

Constant Name	Description	Type	HexValue
BUTTON_0	Number key 0	ENUM	0x30
BUTTON_1	Number key 1	ENUM	0x31
BUTTON_2	Number key 2	ENUM	0x32
BUTTON_3	Number key 3	ENUM	0x33
BUTTON_4	Number key 4	ENUM	0x34
BUTTON_5	Number key 5	ENUM	0x35
BUTTON_6	Number key 6	ENUM	0x36
BUTTON_7	Number key 7	ENUM	0x37
BUTTON_8	Number key 8	ENUM	0x38
BUTTON_9	Number key 9	ENUM	0x39
BUTTON_ASTERISK	Asterisk key (*) or Multiply key (x)	ENUM	0x2A
BUTTON_PLUS	Plus key (+)	ENUM	0x2B
BUTTON_MINUS	Minus key (-)	ENUM	0x2D
BUTTON_DIVIDE	Divide key (/)	ENUM	0x2F
BUTTON_POUND	Pound key (#)	ENUM	0x23
BUTTON_DOT	Dot key (.)	ENUM	0x2E
BUTTON_ENTER	Enter key	ENUM	0x0D
BUTTON_BACKSPACE	Backspace key	ENUM	0x08
BUTTON_ESC	ESC key	ENUM	0x1B
BUTTON_FN	FN key (keyboard extension)	ENUM	0xE0
BUTTON_HORN	Horn key (for communication)	ENUM	0x98
BUTTON_PAGEDOWN	Page Down key	ENUM	0x22
BUTTON_PAGEUP	Page Up key	ENUM	0x21
BUTTON_USER_DEFINED	User-defined key (custom key value, Kozen button)	ENUM	0x99

5.2 com.kozen.component.constant.KeyboardConstant.KeyAction

Constant Name	Description	Type	Value
ACTION_DOWN	Key press down	int	0
ACTION_UP	Key release	int	1