Frequently Asked Questions

Read in another language

Here are the common reported problems and their status.

If you encounter any error, the first step is to upgrade to the latest version.

`adb` issues

`scrcpy` execute `adb` commands to initialize the connection with the device. If `adb` fails, then scrcpy will not work.

This is typically not a bug in *scrcpy*, but a problem in your environment.

`adb` not found

You need `adb` accessible from your `PATH`.

On Windows, the current directory is in your `PATH`, and `adb.exe` is included in the release, so it should work out-of-the-box.

Device not detected

> ```
> ERROR: Could not find any ADB device
> ```

Check that you correctly enabled adb debugging.

Your device must be detected by `adb`:

```
adb devices
```

If your device is not detected, you may need some drivers (on Windows). There is a separate USB driver for Google devices.

Device unauthorized

> *ERROR: Device is unauthorized: ERROR: --> (usb) 0123456789abcdef unauthorized ERROR: A popup should open on the device to request authorization.*

When connecting, a popup should open on the device. You must authorize USB debugging.

If it does not open, check stackoverflow.

Several devices connected

If several devices are connected, you will encounter this error:

ERROR: Multiple (2) ADB devices: ERROR: --> (usb) 0123456789abcdef device Nexus_5 ERROR: --> (tcpip) 192.168.1.5:5555 device GM1913 ERROR: Select a device via -s (--serial), -d (--select-usb) or -e (--select-tcpip)

In that case, you can either provide the identifier of the device you want to mirror:

```
scrcpy -s 0123456789abcdef
```

Or request the single USB (or TCP/IP) device:

```
scrcpy -d  # USB device
scrcpy -e  # TCP/IP device
```

Note that if your device is connected over TCP/IP, you might get this message:

> ```
> adb: error: more than one device/emulator
> ERROR: "adb reverse" returned with value 1
> WARN: 'adb reverse' failed, fallback to 'adb forward'
> ```

This is expected (due to a bug on old Android versions, see #5), but in that case, scrcpy fallbacks to a different method, which should work.

Conflicts between adb versions

> ```
> adb server version (41) doesn't match this client (39); killing...
> ```

This error occurs when you use several `adb` versions simultaneously. You must find the program using a different `adb` version, and use the same `adb` version everywhere.

You could overwrite the `adb` binary in the other program, or ask *scrcpy* to use a specific `adb` binary, by setting the `ADB` environment variable:

```
set ADB=/path/to/your/adb
scrcpy
```

Device disconnected

If *scrcpy* stops itself with the warning "Device disconnected", then the `adb` connection has been closed.

Try with another USB cable or plug it into another USB port. See #281 and #283.

Control issues

Mouse and keyboard do not work

On some devices, you may need to enable an option to allow simulating input. In developer options, enable:

> ***USB debugging (Security settings)***
> *Allow granting permissions and simulating input via USB debugging*

Special characters do not work

The default text injection method is limited to ASCII characters. A trick allows to also inject some accented characters, but that's all. See #37.

Since scrcpy v1.20 on Linux, it is possible to simulate a physical keyboard (HID).

Client issues

The quality is low

If the definition of your client window is smaller than that of your device screen, then you might get poor quality, especially visible on text (see #40).

This problem should be fixed in scrcpy v1.22: **update to the latest version**.

On older versions, you must configure the scaling behavior:

> `scrcpy.exe` > *Properties* > *Compatibility* > *Change high DPI settings* > *Override high DPI scaling behavior* > *Scaling performed by: Application*.

Also, to improve downscaling quality, trilinear filtering is enabled automatically if the renderer is OpenGL and if it supports mipmapping.

On Windows, you might want to force OpenGL to enable mipmapping:

```
scrcpy --render-driver=opengl
```

Issue with Wayland

By default, SDL uses x11 on Linux. The video driver can be changed via the `SDL_VIDEODRIVER` environment variable:

```bash
export SDL_VIDEODRIVER=wayland
scrcpy
```

On some distributions (at least Fedora), the package `libdecor` must be installed manually.

See issues #2554 and #2559.

KWin compositor crashes

On Plasma Desktop, compositor is disabled while *scrcpy* is running.

As a workaround, disable "Block compositing".

Crashes

Exception

There may be many reasons. One common cause is that the hardware encoder of your device is not able to encode at the given definition:

> ```
> ERROR: Exception on thread Thread[main,5,main]
> android.media.MediaCodec$CodecException: Error 0xfffffc0e
> ...
> Exit due to uncaughtException in main thread:
> ERROR: Could not open video stream
> INFO: Initial texture: 1080x2336
> ```

or

> ```
> ERROR: Exception on thread Thread[main,5,main]
> java.lang.IllegalStateException
>         at android.media.MediaCodec.native_dequeueOutputBuffer(Native Method)
> ```

Just try with a lower definition:

```
scrcpy -m 1920
scrcpy -m 1024
scrcpy -m 800
```

Since scrcpy v1.22, scrcpy automatically tries again with a lower definition before failing. This behavior can be disabled with `--no-downsize-on-error`.

You could also try another [encoder](#).

If you encounter this exception on Android 12, then just upgrade to scrcpy >= 1.18 (see [#2129](#)):

```
> ERROR: Exception on thread Thread[main,5,main]
java.lang.AssertionError: java.lang.reflect.InvocationTargetException
    at
com.genymobile.scrcpy.wrappers.SurfaceControl.setDisplaySurface(SurfaceControl.java:75)

    ...
Caused by: java.lang.reflect.InvocationTargetException
    at java.lang.reflect.Method.invoke(Native Method)
    at
com.genymobile.scrcpy.wrappers.SurfaceControl.setDisplaySurface(SurfaceControl.java:73)

    ... 7 more
Caused by: java.lang.IllegalArgumentException: displayToken must not be null
    at
android.view.SurfaceControl$Transaction.setDisplaySurface(SurfaceControl.java:3067)
    at android.view.SurfaceControl.setDisplaySurface(SurfaceControl.java:2147)
    ... 9 more
```

Command line on Windows

Since v1.22, a "shortcut" has been added to directly open a terminal in the scrcpy directory. Double-click on `open_a_terminal_here.bat`, then type your command. For example:

```
scrcpy --record file.mkv
```

You could also open a terminal and go to the scrcpy folder manually:

1. Press `Windows`+`r`, this opens a dialog box.

2. Type `cmd` and press `Enter`, this opens a terminal.

3. Go to your *scrcpy* directory, by typing (adapt the path):

   ```
   cd C:\Users\user\Downloads\scrcpy-win64-xxx
   ```

and press `Enter`

4. Type your command. For example:

   ```
   scrcpy --record file.mkv
   ```

If you plan to always use the same arguments, create a file `myscrcpy.bat` (enable show file extensions to avoid confusion) in the `scrcpy` directory, containing your command. For example:

```
scrcpy --prefer-text --turn-screen-off --stay-awake
```

Then just double-click on that file.

You could also edit (a copy of) `scrcpy-console.bat` or `scrcpy-noconsole.vbs` to add some arguments.

Translations

This FAQ is available in other languages:

- Italiano (Italiano, `it`) - v1.19
- 한국어 (Korean, `ko`) - v1.11
- 简体中文 (Simplified Chinese, `zh-Hans`) - v1.22