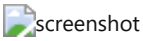
Solo se garantiza que el archivo README original esté actualizado.

scrcpy (v1.21)

Esta aplicación proporciona control e imagen de un dispositivo Android conectado por USB (o por TCP/IP). No requiere acceso *root*. Compatible con *GNU/Linux*, *Windows* y *macOS*.



Se enfoca en:

- **ser ligera**: aplicación nativa, solo muestra la imagen del dispositivo
- **rendimiento**: 30~120fps, dependiendo del dispositivo
- **calidad**: 1920×1080 o superior
- **baja latencia**: 35~70ms
- **inicio rápido**: ~1 segundo para mostrar la primera imagen
- **no intrusivo**: no deja nada instalado en el dispositivo
- **beneficios**: sin cuentas, sin anuncios, no requiere acceso a internet
- **libertad**: software gratis y de código abierto

Con la aplicación puede:

- grabar la pantalla
- duplicar la imagen con la pantalla apagada
- copiar y pegar en ambos sentidos
- configurar la calidad
- usar la pantalla del dispositivo como webcam (V4L2) (solo en Linux)
- emular un teclado físico (HID) (solo en Linux)
- y mucho más...

Requisitos

El dispositivo Android requiere como mínimo API 21 (Android 5.0).

Asegurate de habilitar el adb debugging en tu(s) dispositivo(s).

En algunos dispositivos, también necesitas habilitar una opción adicional para controlarlo con el teclado y ratón.

Consigue la app

Resumen

- Linux: `apt install scrcpy`
- Windows: download
- macOS: `brew install scrcpy`

Construir desde la fuente: BUILD (proceso simplificado)

Linux

En Debian y Ubuntu:

```
apt install scrcpy
```

Hay un paquete Snap: `scrcpy`.

Para Fedora, hay un paquete COPR: `scrcpy`.

Para Arch Linux, hay un paquete AUR: `scrcpy`.

Para Gentoo, hay un paquete Ebuild: `scrcpy/`.

También puedes construir la aplicación manualmente (proceso simplificado).

Windows

Para Windows, por simplicidad, hay un pre-compilado con todas las dependencias (incluyendo `adb`):

- README

También está disponible en Chocolatey:

```
choco install scrcpy
choco install adb    # si aún no está instalado
```

Y en Scoop:

```
scoop install scrcpy
scoop install adb    # si aún no está instalado
```

También puedes construir la aplicación manualmente.

macOS

La aplicación está disponible en Homebrew. Solo instalala:

```
brew install scrcpy
```

Necesitarás `adb`, accesible desde `PATH`. Si aún no lo tienes:

```
brew install android-platform-tools
```

También está disponible en MacPorts, que configura el adb automáticamente:

```
sudo port install scrcpy
```

También puedes construir la aplicación manualmente.

Ejecutar

Packaging status	
Alpine Linux 3.16	1.24
Alpine Linux Edge	1.24
ALT Linux p9	1.16
ALT Linux p10	1.21
ALT Sisyphus	1.21
antiX-19	1.12.1
AOSC	1.24
Arch	1.24
Arch Linux 32 i686	1.24
Arch Linux 32 pentium4	1.24
Arch Linux ARM aarch64	1.24
AUR	1.17.r3.ge...
Chocolatey	1.24
Debian 11	1.17
Debian 11 Backports	1.23
Debian 12	1.24
Debian Unstable	1.24
Devuan 4.0	1.17
Devuan Unstable	1.24
DPorts	1.9
FreeBSD Ports	1.24
Funtoo 1.4	1.24
Gentoo	1.24
Homebrew	1.24
Kali Linux Rolling	1.24
LiGurOS stable	1.24
LiGurOS develop	1.24
MacPorts	1.24
Manjaro Stable	1.24
Manjaro Testing	1.24
Manjaro Unstable	1.24
MPR	1.24
MSYS2 mingw	1.24
MX Linux MX-17	1.12.1
MX Linux MX-19	1.12.1
nixpkgs stable 21.05	1.17
nixpkgs stable 21.11	1.20
nixpkgs stable 22.05	1.24
nixpkgs unstable	1.24
OpenMandriva 4.1	1.12.1
OpenMandriva 4.2	1.17
OpenMandriva Rolling	1.24
OpenMandriva Cooker	1.24

Enchufa el dispositivo Android, y ejecuta:

```
scrcpy
```

Acepta argumentos desde la línea de comandos, listados en:

```
scrcpy --help
```

Parabola	1.24
Pardus 21	1.17
Parrot	1.17
Pisi Linux	1.24
PureOS landing	1.17
Raspbian Stable	1.17
Raspbian Testing	1.24
RPM Sphere	1.24
Scoop	1.24
SlackBuilds	1.24
Solus	1.24
Trisquel 10.0	1.12.1
Ubuntu 20.04	1.12.1
Ubuntu 22.04	1.21
Ubuntu 22.10	1.24
Void Linux x86_64	1.24

Características

Configuración de captura

Reducir la definición

A veces es útil reducir la definición de la imagen del dispositivo Android para aumentar el desempeño.

Para limitar el ancho y la altura a un valor específico (ej. 1024):

```
scrcpy --max-size 1024
scrcpy -m 1024  # versión breve
```

La otra dimensión es calculada para conservar el aspect ratio del dispositivo. De esta forma, un dispositivo en 1920×1080 será transmitido a 1024×576.

Cambiar el bit-rate

El bit-rate por defecto es 8 Mbps. Para cambiar el bit-rate del video (ej. a 2 Mbps):

```
scrcpy --bit-rate 2M
scrcpy -b 2M  # versión breve
```

Limitar los fps

El fps puede ser limitado:

```
scrcpy --max-fps 15
```

Es oficialmente soportado desde Android 10, pero puede funcionar en versiones anteriores.

Recortar

La imagen del dispositivo puede ser recortada para transmitir solo una parte de la pantalla.

Por ejemplo, puede ser útil para transmitir la imagen de un solo ojo del Oculus Go:

```
scrcpy --crop 1224:1440:0:0   # 1224x1440 con coordenadas de origen en (0,0)
```

Si `--max-size` también está especificado, el cambio de tamaño es aplicado después de cortar.

Fijar la rotación del video

Para fijar la rotación de la transmisión:

```
scrcpy --lock-video-orientation     # orientación inicial
scrcpy --lock-video-orientation=0   # orientación normal
scrcpy --lock-video-orientation=1   # 90° contrarreloj
scrcpy --lock-video-orientation=2   # 180°
scrcpy --lock-video-orientation=3   # 90° sentido de las agujas del reloj
```

Esto afecta la rotación de la grabación.

La ventana también puede ser rotada independientemente.

Codificador

Algunos dispositivos pueden tener más de una rotación, y algunos pueden causar problemas o errores. Es posible seleccionar un codificador diferente:

```
scrcpy --encoder OMX.qcom.video.encoder.avc
```

Para listar los codificadores disponibles, puedes pasar un nombre de codificador inválido, el error te dará los codificadores disponibles:

```
scrcpy --encoder _
```

Capturas y grabaciones

Grabación

Es posible grabar la pantalla mientras se transmite:

```
scrcpy --record file.mp4
scrcpy -r file.mkv
```

Para grabar sin transmitir la pantalla:

```
scrcpy --no-display --record file.mp4
scrcpy -Nr file.mkv
# interrumpe la grabación con Ctrl+C
```

Los "skipped frames" son grabados, incluso si no se mostrados en tiempo real (por razones de desempeño). Los frames tienen *marcas de tiempo* en el dispositivo, por lo que el "packet delay variation" no impacta el archivo grabado.

v4l2loopback

En Linux se puede mandar el stream del video a un dispositivo loopback v4l2, por lo que se puede abrir el dispositivo Android como una webcam con cualquier programa compatible con v4l2.

Se debe instalar el modulo `v4l2loopback`:

```
sudo apt install v4l2loopback-dkms
```

Para crear un dispositivo v4l2:

```
sudo modprobe v4l2loopback
```

Esto va a crear un nuevo dispositivo de video en `/dev/videoN`, donde `N` es un número (hay más opciones disponibles para crear múltiples dispositivos o usar un ID en específico).

Para ver los dispositivos disponibles:

```
# requiere el paquete v4l-utils
v4l2-ctl --list-devices
# simple pero generalmente suficiente
ls /dev/video*
```

Para iniciar scrcpy usando una fuente v4l2:

```
scrcpy --v4l2-sink=/dev/videoN
scrcpy --v4l2-sink=/dev/videoN --no-display  # deshabilita la transmisión de imagen
scrcpy --v4l2-sink=/dev/videoN -N            # más corto
```

(reemplace `N` con el ID del dispositivo, compruebe con `ls /dev/video*`)

Una vez habilitado, podés abrir el stream del video con una herramienta compatible con v4l2:

```
ffplay -i /dev/videoN
vlc v4l2:///dev/videoN   # VLC puede agregar un delay por buffering
```

Por ejemplo, podrías capturar el video usando OBS.

Buffering

Es posible agregar buffering al video. Esto reduce el ruido en la imagen ("jitter") pero aumenta la latencia (vea #2464).

La opción de buffering está disponible para la transmisión de imagen:

```
scrcpy --display-buffer=50  # agrega 50 ms de buffering a la imagen
```

y las fuentes V4L2:

```
scrcpy --v4l2-buffer=500    # agrega 500 ms de buffering a la fuente v4l2
```

Conexión

TCP/IP (Inalámbrica)

Scrcpy usa `adb` para comunicarse con el dispositivo, y `adb` puede conectarse vía TCP/IP. El dispositivo debe estar conectado a la misma red que la computadora:

Automático

La opción `--tcpip` permite configurar la conexión automáticamente. Hay 2 variables.

Si el dispositivo (accesible en 192.168.1.1 para este ejemplo) ya está escuchando en un puerto (generalmente 5555) esperando una conexión adb entrante, entonces corré:

```
scrcpy --tcpip=192.168.1.1       # el puerto default es 5555
scrcpy --tcpip=192.168.1.1:5555
```

Si el dispositivo no tiene habilitado el modo adb TCP/IP (o si no sabés la dirección IP), entonces conectá el dispositivo por USB y corré:

```
scrcpy --tcpip    # sin argumentos
```

El programa buscará automáticamente la IP del dispositivo, habilitará el modo TCP/IP, y se conectará al dispositivo antes de comenzar a transmitir la imagen.

Manual

Como alternativa, se puede habilitar la conexión TCP/IP manualmente usando `adb`:

1. Conecta el dispositivo al mismo Wi-Fi que tu computadora.
2. Obtén la dirección IP del dispositivo, en Ajustes → Acerca del dispositivo → Estado, o ejecutando este comando:

   ```
   adb shell ip route | awk '{print $9}'
   ```

3. Habilita adb vía TCP/IP en el dispositivo: `adb tcpip 5555`.
4. Desenchufa el dispositivo.
5. Conéctate a tu dispositivo: `adb connect IP_DEL_DISPOSITIVO:5555` *(reemplaza `IP_DEL_DISPOSITIVO`)*.
6. Ejecuta `scrcpy` con normalidad.

Podría resultar útil reducir el bit-rate y la definición:

```
scrcpy --bit-rate 2M --max-size 800
scrcpy -b2M -m800  # versión breve
```

Múltiples dispositivos

Si hay muchos dispositivos listados en `adb devices`, será necesario especificar el *número de serie*:

```
scrcpy --serial 0123456789abcdef
scrcpy -s 0123456789abcdef  # versión breve
```

Si el dispositivo está conectado por TCP/IP:

```
scrcpy --serial 192.168.0.1:5555
scrcpy -s 192.168.0.1:5555  # versión breve
```

Puedes iniciar múltiples instancias de *scrcpy* para múltiples dispositivos.

Iniciar automáticamente al detectar dispositivo

Puedes utilizar AutoAdb:

```
autoadb scrcpy -s '{}'
```

Túneles

Para conectarse a un dispositivo remoto, es posible conectar un cliente local `adb` a un servidor remoto `adb` (siempre y cuando utilicen la misma versión de protocolos *adb*).

Servidor ADB remoto

Para conectarse a un servidor ADB remoto, haz que el servidor escuche en todas las interfaces:

```
adb kill-server
adb -a nodaemon server start
# conserva este servidor abierto
```

Advertencia: todas las comunicaciones entre los clientes y el servidor ADB están desencriptadas.

Supondremos que este servidor se puede acceder desde 192.168.1.2. Entonces, desde otra terminal, corré scrcpy:

```
export ADB_SERVER_SOCKET=tcp:192.168.1.2:5037
scrcpy --tunnel-host=192.168.1.2
```

Por default, scrcpy usa el puerto local que se usó para establecer el tunel `adb forward` (típicamente `27183`, vea `--port`). También es posible forzar un puerto diferente (puede resultar útil en situaciones más complejas, donde haya múltiples redirecciones):

```
scrcpy --tunnel-port=1234
```

Túnel SSH

Para comunicarse con un servidor ADB remoto de forma segura, es preferible usar un túnel SSH.

Primero, asegurate que el servidor ADB está corriendo en la computadora remota:

```
adb start-server
```

Después, establecé el túnel SSH:

```
# local  5038 --> remoto  5037
# local 27183 <-- remoto 27183
ssh -CN -L5038:localhost:5037 -R27183:localhost:27183 your_remote_computer
# conserva este servidor abierto
```

Desde otra terminal, corré scrcpy:

```
export ADB_SERVER_SOCKET=tcp:localhost:5038
scrcpy
```

Para evitar habilitar "remote port forwarding", puedes forzar una "forward connection" (nótese el argumento `-L` en vez de `-R`):

```
# local  5038 --> remoto  5037
# local 27183 --> remoto 27183
ssh -CN -L5038:localhost:5037 -L27183:localhost:27183 your_remote_computer
# conserva este servidor abierto
```

Desde otra terminal, corré scrcpy:

```
export ADB_SERVER_SOCKET=tcp:localhost:5038
scrcpy --force-adb-forward
```

Al igual que las conexiones inalámbricas, puede resultar útil reducir la calidad:

```
scrcpy -b2M -m800 --max-fps 15
```

Configuración de la ventana

Título

Por defecto, el título de la ventana es el modelo del dispositivo. Puede ser modificado:

```
scrcpy --window-title 'My device'
```

Posición y tamaño

La posición y tamaño inicial de la ventana puede ser especificado:

```
scrcpy --window-x 100 --window-y 100 --window-width 800 --window-height 600
```

Sin bordes

Para deshabilitar el diseño de la ventana:

```
scrcpy --window-borderless
```

Siempre adelante

Para mantener la ventana de scrcpy siempre adelante:

```
scrcpy --always-on-top
```

Pantalla completa

La aplicación puede ser iniciada en pantalla completa:

```
scrcpy --fullscreen
scrcpy -f  # versión breve
```

Puede entrar y salir de la pantalla completa con la combinación MOD+f.

Rotación

Se puede rotar la ventana:

```
scrcpy --rotation 1
```

Los posibles valores son:

- `0` : sin rotación
- `1` : 90 grados contrarreloj
- `2` : 180 grados
- `3` : 90 grados en sentido de las agujas del reloj

La rotación también puede ser modificada con la combinación de teclas MOD+← *(izquierda)* y MOD+→ *(derecha)*.

Nótese que *scrcpy* maneja 3 diferentes rotaciones:

- MOD+r solicita al dispositivo cambiar entre vertical y horizontal (la aplicación en uso puede rechazarlo si no soporta la orientación solicitada).
- `--lock-video-orientation` cambia la rotación de la transmisión (la orientación del video enviado a la PC). Esto afecta a la grabación.
- `--rotation` (o MOD+←/MOD+→) rota solo el contenido de la imagen. Esto solo afecta a la imagen mostrada, no a la grabación.

Otras opciones

Solo lectura ("Read-only")

Para deshabilitar los controles (todo lo que interactúe con el dispositivo: eventos del teclado, eventos del mouse, arrastrar y soltar archivos):

```
scrcpy --no-control
scrcpy -n  # versión breve
```

Pantalla

Si múltiples pantallas están disponibles, es posible elegir cual transmitir:

```
scrcpy --display 1
```

Los ids de las pantallas se pueden obtener con el siguiente comando:

```
adb shell dumpsys display   # busque "mDisplayId=" en la respuesta
```

La segunda pantalla solo puede ser manejada si el dispositivo cuenta con Android 10 (en caso contrario será transmitida en el modo solo lectura).

Permanecer activo

Para evitar que el dispositivo descanse después de un tiempo mientras está conectado:

```
scrcpy --stay-awake
scrcpy -w  # versión breve
```

La configuración original se restaura al cerrar scrcpy.

Apagar la pantalla

Es posible apagar la pantalla mientras se transmite al iniciar con el siguiente comando:

```
scrcpy --turn-screen-off
scrcpy -S  # versión breve
```

O presionando `MOD+o` en cualquier momento.

Para volver a prenderla, presione `MOD+Shift+o`.

En Android, el botón de `POWER` siempre prende la pantalla. Por conveniencia, si `POWER` es enviado vía scrcpy (con click-derecho o `MOD+p`), esto forzará a apagar la pantalla con un poco de atraso (en la mejor de las situaciones). El botón físico `POWER` seguirá prendiendo la pantalla.

También puede resultar útil para evitar que el dispositivo entre en inactividad:

```
scrcpy --turn-screen-off --stay-awake
scrcpy -Sw  # versión breve
```

Apagar al cerrar la aplicación

Para apagar la pantalla del dispositivo al cerrar scrcpy:

```
scrcpy --power-off-on-close
```

Mostrar clicks

Para presentaciones, puede resultar útil mostrar los clicks físicos (en el dispositivo físicamente).

Android provee esta opción en *Opciones para desarrolladores*.

Scrcpy provee una opción para habilitar esta función al iniciar la aplicación y restaurar el valor original al salir:

```
scrcpy --show-touches
scrcpy -t  # versión breve
```

Nótese que solo muestra los clicks *físicos* (con el dedo en el dispositivo).

Desactivar protector de pantalla

Por defecto, scrcpy no evita que el protector de pantalla se active en la computadora.

Para deshabilitarlo:

```
scrcpy --disable-screensaver
```

Control

Rotar pantalla del dispositivo

Presione `MOD`+`r` para cambiar entre posición vertical y horizontal.

Nótese que solo rotará si la aplicación activa soporta la orientación solicitada.

Copiar y pegar

Cuando que el portapapeles de Android cambia, automáticamente se sincroniza al portapapeles de la computadora.

Cualquier shortcut con `Ctrl` es enviado al dispositivo. En particular:

- `Ctrl`+`c` normalmente copia
- `Ctrl`+`x` normalmente corta
- `Ctrl`+`v` normalmente pega (después de la sincronización de portapapeles entre la computadora y el dispositivo)

Esto normalmente funciona como es esperado.

Sin embargo, este comportamiento depende de la aplicación en uso. Por ejemplo, *Termux* envía SIGINT con `Ctrl`+`c`, y *K-9 Mail* crea un nuevo mensaje.

Para copiar, cortar y pegar, en tales casos (solo soportado en Android >= 7):

- `MOD`+`c` inyecta `COPY`
- `MOD`+`x` inyecta `CUT`
- `MOD`+`v` inyecta `PASTE` (después de la sincronización de portapapeles entre la computadora y el dispositivo)

Además, `MOD`+`Shift`+`v` permite inyectar el texto en el portapapeles de la computadora como una secuencia de teclas. Esto es útil cuando el componente no acepta pegado de texto (por ejemplo en *Termux*), pero puede romper caracteres no pertenecientes a ASCII.

AVISO: Pegar de la computadora al dispositivo (tanto con `Ctrl`+`v` o `MOD`+`v`) copia el contenido al portapapeles del dispositivo. Como consecuencia, cualquier aplicación de Android puede leer su contenido. Debería evitar pegar contenido sensible (como contraseñas) de esta forma.

Algunos dispositivos no se comportan como es esperado al establecer el portapapeles programáticamente. La opción `--legacy-paste` está disponible para cambiar el comportamiento de `Ctrl`+`v` y `MOD`+`v` para que también inyecten el texto del portapapeles de la computadora como una secuencia de teclas (de la misma forma que `MOD`+`Shift`+`v`).

Para deshabilitar la auto-sincronización del portapapeles, use `--no-clipboard-autosync`.

Pellizcar para zoom

Para simular "pinch-to-zoom": `Ctrl`+*click-y-mover*.

Más precisamente, mantén `Ctrl` mientras presionas botón izquierdo. Hasta que no se suelte el botón, todos los movimientos del mouse cambiarán el tamaño y rotación del contenido (si es soportado por la app en uso) respecto al centro de la pantalla.

Concretamente, scrcpy genera clicks adicionales con un "dedo virtual" en la posición invertida respecto al centro de la pantalla.

Emular teclado físico (HID)

Por default, scrcpy usa el sistema de Android para la injección de teclas o texto: funciona en todas partes, pero está limitado a ASCII.

En Linux, scrcpy puede emular un teclado USB físico en Android para proveer una mejor experiencia al enviar *inputs* (usando USB HID vía AOAv2): deshabilita el teclado virtual y funciona para todos los caracteres y IME.

Sin embargo, solo funciona si el dispositivo está conectado por USB, y por ahora solo funciona en Linux.

Para habilitar este modo:

```
scrcpy --hid-keyboard
scrcpy -K  # más corto
```

Si por alguna razón falla (por ejemplo si el dispositivo no está conectado vía USB), automáticamente vuelve al modo default (un mensaje se escribirá en la consola). Se puede usar los mismos argumentos en la línea de comandos tanto si se conecta con USB o vía TCP/IP.

En este modo, los *raw key events* (*scancodes*) se envían al dispositivo, independientemente del mapeo del teclado en el host. Por eso, si el diseño de tu teclado no concuerda, debe ser configurado en el dispositivo Android, en Ajustes → Sistema → Idioma y Entrada de Texto → Teclado Físico.

Se puede iniciar automáticamente en esta página de ajustes:

```
adb shell am start -a android.settings.HARD_KEYBOARD_SETTINGS
```

Sin embargo, la opción solo está disponible cuando el teclado HID está activo (o cuando se conecta un teclado físico).

Preferencias de inyección de texto

Existen dos tipos de eventos generados al escribir texto:

- *key events*, marcando si la tecla es presionada o soltada;
- *text events*, marcando si un texto fue introducido.

Por defecto, las letras son inyectadas usando *key events*, para que el teclado funcione como es esperado en juegos (típicamente las teclas WASD).

Pero esto puede causar problemas. Si encuentras tales problemas, los puedes evitar con:

```
scrcpy --prefer-text
```

(Pero esto romperá el comportamiento del teclado en los juegos)

Por el contrario, se puede forzar scrcpy para siempre injectar *raw key events*:

```
scrcpy --raw-key-events
```

Estas opciones no tienen efecto en los teclados HID (todos los *key events* son enviados como *scancodes* en este modo).

Repetir tecla

Por defecto, mantener una tecla presionada genera múltiples *key events*. Esto puede causar problemas de desempeño en algunos juegos, donde estos eventos no tienen sentido de todos modos.

Para evitar enviar *key events* repetidos:

```
scrcpy --no-key-repeat
```

Estas opciones no tienen efecto en los teclados HID (Android maneja directamente las repeticiones de teclas en este modo)

Botón derecho y botón del medio

Por defecto, botón derecho ejecuta RETROCEDER (o ENCENDIDO) y botón del medio INICIO. Para inhabilitar estos atajos y enviar los clicks al dispositivo:

```
scrcpy --forward-all-clicks
```

Arrastrar y soltar archivos

Instalar APKs

Para instalar un APK, arrastre y suelte el archivo APK (terminado en `.apk`) a la ventana de *scrcpy*.

No hay respuesta visual, un mensaje se escribirá en la consola.

Enviar archivos al dispositivo

Para enviar un archivo a `/sdcard/Download/` en el dispositivo, arrastre y suelte un archivo (no APK) a la ventana de *scrcpy*.

No hay ninguna respuesta visual, un mensaje se escribirá en la consola.

El directorio de destino puede ser modificado al iniciar:

```
scrcpy --push-target=/sdcard/Movies/
```

Envío de Audio

Scrcpy no envía el audio. Use sndcpy.

También lea issue #14.

Atajos

En la siguiente lista, `MOD` es el atajo modificador. Por defecto es `Alt` (izquierdo) o `Super` (izquierdo).

Se puede modificar usando `--shortcut-mod`. Las posibles teclas son `lctrl` (izquierdo), `rctrl` (derecho), `lalt` (izquierdo), `ralt` (derecho), `lsuper` (izquierdo) y `rsuper` (derecho). Por ejemplo:

```
# use RCtrl para los atajos
scrcpy --shortcut-mod=rctrl

# use tanto LCtrl+LAlt o LSuper para los atajos
scrcpy --shortcut-mod=lctrl+lalt,lsuper
```

`Super` es generalmente la tecla `Windows` o `Cmd`.

Acción	Atajo
Alterne entre pantalla compelta	`MOD`+`f`
Rotar pantalla hacia la izquierda	`MOD`+`←` *(izquierda)*
Rotar pantalla hacia la derecha	`MOD`+`→` *(derecha)*
Ajustar ventana a 1:1 ("pixel-perfect")	`MOD`+`g`
Ajustar ventana para quitar los bordes negros	`MOD`+`w` | *Doble click izquierdo*[1]
Click en `INICIO`	`MOD`+`h` | *Click medio*
Click en `RETROCEDER`	`MOD`+`b` | *Click derecho*[2]
Click en `CAMBIAR APLICACIÓN`	`MOD`+`s` | *Cuarto botón*[3]
Click en `MENÚ` (desbloquear pantalla)[4]	`MOD`+`m`
Click en `SUBIR VOLUMEN`	`MOD`+`↑` *(arriba)*
Click en `BAJAR VOLUME`	`MOD`+`↓` *(abajo)*
Click en `ENCENDIDO`	`MOD`+`p`
Encendido	*Botón derecho*[2]
Apagar pantalla (manteniendo la transmisión)	`MOD`+`o`
Encender pantalla	`MOD`+`Shift`+`o`
Rotar pantalla del dispositivo	`MOD`+`r`
Abrir panel de notificaciones	`MOD`+`n` | *Quinto botón*[3]
Abrir panel de configuración	`MOD`+`n`+`n` | *Doble quinto botón*[3]
Cerrar paneles	`MOD`+`Shift`+`n`
Copiar al portapapeles[5]	`MOD`+`c`
Cortar al portapapeles[5]	`MOD`+`x`
Synchronizar portapapeles y pegar[5]	`MOD`+`v`
Inyectar texto del portapapeles de la PC	`MOD`+`Shift`+`v`

Habilitar/Deshabilitar contador de FPS (en stdout)	`MOD`+`i`
Pellizcar para zoom	`Ctrl`+*click-y-mover*
Arrastrar y soltar un archivo (APK)	Instalar APK desde la computadora
Arrastrar y soltar un archivo (no APK)	Mover archivo al dispositivo

[1]*Doble click en los bordes negros para eliminarlos.*
[2]*Botón derecho enciende la pantalla si estaba apagada, sino ejecuta RETROCEDER.*
[3]*Cuarto y quinto botón del mouse, si tu mouse los tiene.*
[4]*Para las apps react-native en desarrollo,* `MENU` *activa el menú de desarrollo.*
[5]*Solo en Android >= 7.*

Los shortcuts con teclas repetidas se ejecutan soltando y volviendo a apretar la tecla por segunda vez. Por ejemplo, para ejecutar "Abrir panel de configuración":

1. Apretá y mantené apretado `MOD`.
2. Después apretá dos veces la tecla `n`.
3. Por último, soltá la tecla `MOD`.

Todos los atajos `Ctrl`+*tecla* son enviados al dispositivo para que sean manejados por la aplicación activa.

Path personalizado

Para usar un binario de *adb* en particular, configure el path `ADB` en las variables de entorno:

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

Para sobreescribir el path del archivo `scrcpy-server`, configure el path en `SCRCPY_SERVER_PATH`.

Para sobreescribir el ícono, configure el path en `SCRCPY_ICON_PATH`.

¿Por qué *scrcpy*?

Un colega me retó a encontrar un nombre tan impronunciable como gnirehtet.

`strcpy` copia un **str**ing; `scrcpy` copia un **scr**een.

¿Cómo construir (BUILD)?

Véase BUILD (en inglés).

Problemas generales

Vea las preguntas frecuentes (en inglés).

Desarrolladores

Lea la hoja de desarrolladores (en inglés).

Licencia

Artículos

- Introducing scrcpy (en inglés)
- Scrcpy now works wirelessly (en inglés)