playerctl Media player remote

START HERE

`playerctl` **`play-pause`**
Toggle playback

`playerctl` **`metadata title`**
Current title

`playerctl` **`next`**
Skip track

`playerctl` **`-l`**
List players

`playerctl` **`previous`**
Previous track

`playerctl` **`-p`** *`<name>`* **`play`**
Target one player

`playerctl` **`status`**
Playing or paused?

`playerctl` **`volume 0.1+`**
Louder

playerctl command → D-Bus → **MPRIS** session bus → selects → **Player** Spotify, VLC... → plays → **Audio** output

CONCEPTS

playerctl talks to any MPRIS media player over D-Bus (Spotify, VLC, mpv with a plugin, browsers), so one command controls them all.

Without -p, the command acts on the first player that supports it; playerctld (below) makes that the most recently active one.

Volume is a fraction from 0.0 to 1.0, not a percentage, and position is in seconds.

PLAYBACK

`playerctl` **`play`**
Start playing

`playerctl` **`pause`**
Pause playback

`playerctl` **`play-pause`**
Toggle play and pause

`playerctl` **`stop`**
Stop playback

`playerctl` **`next`**
Next track

`playerctl` **`previous`**
Previous track

`playerctl` **`position 30+`**
Seek 30 seconds forward
A trailing - seeks back: position 30-

`playerctl` **`position 90`**
Jump to 90 seconds

`playerctl` **`position`**
Print the position in seconds

`playerctl` **`open`** *`<uri>`*
Open a file or URL in the player

`playerctl` **`loop None|Track|Playlist`**
Set the loop mode

PLAYBACK (cont.)

`playerctl` **`shuffle On|Off|Toggle`**
Set or toggle shuffle

VOLUME

`playerctl` **`volume`**
Print the volume (0.0 to 1.0)

`playerctl` **`volume 0.5`**
Set volume to 50%

`playerctl` **`volume 0.05+`**
Raise volume by 5%

`playerctl` **`volume 0.05-`**
Lower volume by 5%

Not every player accepts volume changes; use pactl for the system volume.

CHOOSING PLAYERS

`playerctl` **`-l`**
List running players

`playerctl` **`-p`** *`<name> <cmd>`*
Control a specific player

`playerctl` **`-p vlc,spotify`** *`<cmd>`*
Try players in order
Acts on the first listed one that supports the command

`playerctl` **`-a`** *`<cmd>`*
Apply to all players

`playerctl` **`-i`** *`<name> <cmd>`*
Ignore a player

`playerctl` **`-p %any,`***`<name> <cmd>`*
Use %any as a fallback slot

`playerctl` **`-s`** *`<cmd>`*
Suppress diagnostic messages

`playerctl` **`-v`**
Print the version

CHOOSING PLAYERS (cont.)

`playerctld daemon`
Start the activity-tracking daemon
Orders players by last activity, so the default target is the one you used last

METADATA

`playerctl` **`status`**
Playing, Paused or Stopped

`playerctl` **`metadata`**
Print all track metadata

`playerctl` **`metadata`** *`<key>`*
Print one value
Keys: artist, title, album, or any raw key such as mpris:length

`playerctl` **`metadata artist`**
Current artist

`playerctl` **`metadata mpris:artUrl`**
Album art URL

`playerctl` **`-a status`**
Status of every player

FORMAT AND FOLLOW

`playerctl` **`metadata -f "{{artist}} - {{title}}"`**
Custom output line

`playerctl` **`-F metadata -f "{{artist}} - {{title}}"`**
Print on every change
Blocks and keeps running; ideal for status bars

`{{ playerName }}`
`{{ status }}`
Player name and status variables

`{{ position }}`
`{{ volume }}`
Position (microseconds) and volume

FORMAT AND FOLLOW (cont.)

`{{ duration(mpris:length) }}`
Format microseconds as mm:ss

`{{ duration(mpris:length - position) }}`
Time remaining

`{{ uc(status) }} / {{ lc(artist) }}`
Change case

`{{ trunc(title, 30) }}`
Cut text to 30 characters

`{{ default(artist, "unknown") }}`
Fallback for a missing value

`{{ emoji(status) }}`
Show status as an emoji

`{{ markup_escape(title) }}`
Escape markup for bars like polybar

Quote the format string so the shell leaves the braces alone.

RECIPES

Status bar line

```
playerctl -F metadata \
  -f '{{ emoji(status) }}
{{ artist }} - {{ title }}'
```

Pause everything, then play one

```
playerctl -a pause
playerctl -p spotify play
```

Keyboard media keys (i3 config)

```
bindsym XF86AudioPlay exec
playerctl play-pause
bindsym XF86AudioNext exec
playerctl next
bindsym XF86AudioPrev exec
playerctl previous
```

Notify the current track

```
notify-send "$(playerctl
metadata -f '{{artist}} -
{{title}}')"
```