

Guide: record the screen and capture snapshots

Use DjMirror Pulse's recording API or MCP tools to save the main application window as an MP4 video or a PNG snapshot. This guide covers a short, safe recording workflow, where files are saved, and what to check if a recording has no audio or does not start.

Recording is controlled through the local HTTP API or a compatible MCP client. There is **no record button** in the desktop interface. For API startup and authentication, see [HTTP & WebSocket API](#). For AI-client setup, see [MCP Server for Users](#).

1. Before you record

1. Start DjMirror Pulse and prepare the visual output you intend to capture.
2. Close settings and control panels that should not appear in the video. The recorder captures the main application window, including any control or settings overlay that is open.
3. If the video should contain music, configure the audio input first and play a short test. Recording uses the selected audio-analysis input, or the system default input if no selected input is available.
4. Check that the destination drive has enough free space. Video file size depends on resolution, frame rate, content and recording duration; the app does not provide a storage estimate here.

The recorder supports one active recording at a time. Its default maximum duration is 30 minutes; the recorder stops automatically when that limit is reached.

2. Record a short test with the HTTP API

The desktop HTTP server is enabled by default and normally listens on port 8080. First confirm that the local server responds:

```
curl.exe http://127.0.0.1:8080/api/v1/health
```

If the server is disabled, enable it in the app configuration or start DjMirror Pulse with `--server`. If you configured an API key, add `-H "X-API-Key: YOUR_KEY"` to each request below. Do not put a real key in a shared script, screenshot or support message.

Without an API key, the local API is open by default. Do not expose it to untrusted devices or networks; see the [Mobile Remote guide](#) for API key settings.

Start with a short recording. The defaults are 20 frames per second and a 30-minute maximum; this example limits the test to 60 seconds:

```
curl.exe -X POST "http://127.0.0.1:8080/api/v1/recording/start" `
-H "Content-Type: application/json" `
-d '{"maxDurationSec":60,"fileName":"soundcheck.mp4"}'
```

The start request answers with the settings in use:

```
{ "data": { "fileName": "soundcheck.mp4", "fps": 20, "hasAudio": true,
"maxDurationSec": 60, "recording": true }, "ok": true }
```

Check the current state:

```
curl.exe "http://127.0.0.1:8080/api/v1/recording/status"
```

```
{ "data": { "elapsedSec": 1.08, "fileName": "soundcheck.mp4", "fps": 20,
"hasAudio": true, "recording": true }, "ok": true }
```

The response includes the file name and hasAudio. This flag reports whether an audio input device was attached; it does not confirm that useful audio was captured. Stop the recording to finalize the MP4:

```
curl.exe -X POST "http://127.0.0.1:8080/api/v1/recording/stop"
```

The stop request waits briefly for the file to finish writing and then reports the result:

```
{ "data": { "durationSec": 3.18, "fileName": "soundcheck.mp4", "hasAudio":
false, "recording": false, "sizeBytes": 1404043 }, "ok": true }
```

Note

hasAudio in the **stop** answer can differ from the value reported while recording (in a short test it changed from true to false). Treat the flag as a hint only: open the MP4 in a media player and check both picture and sound before relying on it for an event.

Record a fixed length with one script

This PowerShell example records 60 seconds, stops, and copies the file to your desktop. Adjust the file name and add -H "X-API-Key: YOUR_KEY" to each call if you set an API key.

```
$api = "http://127.0.0.1:8080/api/v1/recording"
curl.exe -s -X POST "$api/start" -H "Content-Type: application/json" -d
'{"maxDurationSec":120,"fileName":"set.mp4"}'
Start-Sleep -Seconds 60
curl.exe -s -X POST "$api/stop"
curl.exe -s -o "$env:USERPROFILE\Desktop\set.mp4" "$api/download/set.mp4"
```

The recorder also stops by itself when maxDurationSec is reached, so a forgotten recording cannot run forever.

3. Capture a PNG snapshot

A snapshot can be captured independently of video recording. The main window must be visible.

```
curl.exe -X POST "http://127.0.0.1:8080/api/v1/recording/snapshot"
```

The response includes the PNG file name, its size in bytes and its dimensions:

```
{ "data": { "fileName": "snapshot_20261013_075259_999.png", "height": 1080, "sizeBytes": 148116, "width": 1920 }, "ok": true }
```

To list saved snapshots (each entry has the file name, modified time and size):

```
curl.exe "http://127.0.0.1:8080/api/v1/recording/snapshots"
```

To record at a different frame rate, add `fps` to the start request (for example `{"fps":30,"maxDurationSec":60}`); a lower value reduces CPU and disk use on weaker machines.

4. Find and manage saved files

By default, files are saved here:

File type	Default folder
MP4 recordings	Documents/Visuals/Recordings/
PNG snapshots	Documents/Visuals/Snapshots/

If `documentsPath` was changed in the effective `config.json`, use that location's `Visuals/Recordings/` and `Visuals/Snapshots/` folders instead. See [Config & CLI](#) for the configuration path.

The API can list MP4 files with `GET /api/v1/recording/list` and download an MP4 using `GET /api/v1/recording/download/<fileName>`. Snapshots can be listed with `GET /api/v1/recording/snapshots` and downloaded with `GET /api/v1/recording/snapshot/download/<fileName>`. For example, to copy a finished recording to another folder through the API:

```
curl.exe -o soundcheck.mp4 "http://127.0.0.1:8080/api/v1/recording/download/soundcheck.mp4"
```

Keep the original files in the folders if you want them available to the app; move or delete them using the operating system's file manager when the app is not recording.

The app has a maximum recording duration, but no automatic storage quota is documented. Check free disk space regularly and keep a separate backup of recordings you need to retain.

5. Use the MCP tools

If MCP is configured and available for your license, a compatible assistant can use these recording tools:

- `capture_screen_snapshot` to capture the visible main window.
- `start_screen_recording` to start an MP4 recording.
- `download_screen_recording` to retrieve a saved recording.

Ask for a short test first, then verify the returned file and audio yourself. Remote recording can be started without a prominent on-screen indicator, so do not leave the MCP/API accessible to untrusted users.

6. What each call returns

Call	Returns
POST <code>/recording/start</code>	<code>fileName</code> , <code>fps</code> , <code>hasAudio</code> , <code>maxDurationSec</code> , <code>recording: true</code>
GET <code>/recording/status</code>	<code>recording</code> , <code>elapsedSec</code> , <code>fileName</code> , <code>fps</code> , <code>hasAudio</code>
POST <code>/recording/stop</code>	<code>fileName</code> , <code>durationSec</code> , <code>sizeBytes</code> , <code>hasAudio</code> , <code>recording: false</code>
GET <code>/recording/list</code>	<code>recordings</code> : a list of <code>fileName</code> , <code>modified</code> , <code>sizeBytes</code>
POST <code>/recording/snapshot</code>	<code>fileName</code> , <code>width</code> , <code>height</code> , <code>sizeBytes</code>
GET <code>/recording/snapshots</code>	<code>snapshots</code> : a list of <code>fileName</code> , <code>modified</code> , <code>sizeBytes</code>

All answers use the standard envelope (`ok`, `data`, `error`, `meta`) described in the [HTTP & WebSocket API](#) guide.

7. Troubleshooting

Symptom	Check
Request cannot connect	Confirm DjMirror Pulse is running, the HTTP server is enabled, and the port is correct.
401 or 403 response	Add the configured X-API-Key header; confirm the active license allows the requested feature.
Recording already active	Check GET <code>/api/v1/recording/status</code> , then stop the active recording before starting another.
Snapshot fails	Make sure the main window is visible and the destination drive is writable.

Symptom	Check
MP4 has no sound	Check hasAudio, select a working audio input in Settings → AUDIO Device , and test its meters.
Recording stutters or misses frames	Try a shorter test, lower output resolution or frame rate, and close other GPU-intensive applications.
File is missing or incomplete	Stop the recording through the API, wait for completion, then check the configured Visuals/Recordings/ path.

For more endpoint details and error responses, consult the recording section in [docs_http_ws/README_HTTP_API_REFERENCE.md](#).

8. Quick checklist

- Close any overlays that should not appear in the capture.
- Confirm the audio input and test meters if sound is required.
- Start a short recording and check its status.
- Stop it explicitly and play the resulting MP4.
- Confirm the file is saved in the expected Visuals/Recordings/ or Visuals/Snapshots/ folder.