Companion — PDF Presenter user guide

This is **a Bitfocus Companion connection module for PDF Presenter**. It lets a Stream Deck — or any other Companion surface — drive a running copy of the app: slides, sections, the output window, black/white screen, the laser pointer, and the watched-folder file list.

It talks straight to the app's own OSC listener over UDP and receives feedback on a local port. There is nothing to install on the app side beyond turning OSC on.

> **This module was built with AI assistance and reviewed by a human.** Review it before relying on it in production. It is **not in the official Companion module store** — it installs as a developer module, which is a slightly longer path and is covered below.

Setting it up

1. Install the module. It is not in the store, so Companion has to be pointed at it:

```
git clone https://github.com/stoatworks-labs/companion-module-pdf-presenter-lite
cd companion-module-pdf-presenter-lite
npm install
```

In Companion, go to **Settings → Developer modules path** and point it at **the parent directory containing this repo's folder** — not at the folder itself. Restart Companion, or use "Rescan for developer modules" if your version has it. "PDF Presenter" then appears as an installable connection.

2. Turn OSC on in the app. In PDF Presenter, click **Start OSC** in the titlebar. It is remembered across restarts once you have started it once.

3. Add the connection, and set three fields:

Field	Default	What it is
App host	`127.0.0.1`	The machine running PDF Presenter. The default assumes Companion is on the same machine.
App listen port	`35551`	Where the app listens. Matches the app's own default.
Local feedback port	`35550`	Where this module listens for the app's replies. Matches the app's own default.

If Companion and the app are on different machines, change **App host** to the app's real IP — *and* make sure the app's own OSC settings panel has its **Feedback host** pointing back at the

Companion machine's IP, not `127.0.0.1`. Miss that second half and actions will work while every variable stays empty, which is a confusing way to fail.

Actions

Slides

Action	What it does
Next slide / Previous slide	Step through the deck
Go to slide number	Jump to an absolute slide
Go to first slide / Go to last slide	Ends of the deck

Sections

Action	What it does
Go to first slide of section	Jump to a named section
Go to first slide of next section	Skip forward a section
Go to first slide of previous section	Skip back a section

The named-section action offers **a live dropdown of whatever the app last reported**, plus a variables-aware custom value for dynamic use.

Output window

Action	What it does
Start Output from first slide	Open the output window at the top of the deck
Start Output from current slide	Open it where you already are
Close Output	Shut the output window
Toggle black screen / Toggle white screen	Cover the output
Toggle laser pointer overlay	Show/hide the pointer
Set current slide as desktop wallpaper	Useful for a holding slide behind the output

Auto-advance

Action	What it does
Pause auto-advance / Resume auto-advance	Hold and release a timed run

Protocol control

Action	What it does
Enable / Disable OSC actions	Turn remote control off from the surface
Enable / Disable OSC feedback	Turn the app's replies off
Request feedback refresh	Ask the app to re-send everything
Set watched folder	Point the app at a different folder
Request watched-folder file list	Refresh the file dropdown
Open file from watched folder	Load a deck by name

Feedbacks

Two, and only two:

- **Slideshow state** — edit / running / running-with-auto-advance-paused.
- **OSC file access enabled**.

> **There is deliberately no feedback for "laser pointer on" or "auto-advance enabled".** The app's OSC protocol never broadcasts either as a standalone value — only the combined edit/running/paused state is sent — so such a feedback would have to fabricate data the app does not provide. If you need a button to *look* like it knows, drive it from the slideshow state instead.

Variables

Variable	Contents
`presentationName`	Presentation file name
`slideCount`	Total slide count
`slideCountVisible`	Total slide count, excluding hidden slides
`state`	Presentation state (edit / running / paused)
`currentSlide`	Current slide number
`slidesRemaining`	Slides left in the deck
`sectionIndex`	Current section index
`sectionName`	Current section name
`sectionSlidesRemaining`	Slides remaining in the current section
`previousSectionName`	Previous section name — "Start of deck" if this is the first
`previousSectionFirstSlide`	First slide of the previous section
`nextSectionName`	Next section name — "End of deck" if this is the last
`nextSectionFirstSlide`	First slide of the next section
`fileAccessEnabled`	Whether OSC file-open access is enabled
`activeFolder`	Watched folder, relative to the home directory
`activeFolderFullPath`	Watched folder, full path
`activeFolderFileCount`	Number of files in the watched folder
`activeFolderFileNames`	JSON array of file names in the watched folder

The section variables are the useful ones for building a presenter's surface: a button showing `$(<connection-label>:nextSectionName)` tells the operator what is coming without them reading the deck. The prefix is whatever you named the connection in Companion.

Troubleshooting

Symptom	Cause
Module doesn't appear in Companion	The developer modules path points at the repo folder rather than **its parent**.
Buttons work, all variables empty	Feedback isn't getting back. On separate machines, the app's **Feedback host** must be the Companion machine's IP, not `127.0.0.1`.
Nothing happens at all	OSC isn't started in the app — click **Start OSC** in its titlebar — or App host / App listen port is wrong.
Section dropdown is empty or stale	The list is whatever the app last reported. Fire **Request feedback refresh**.
File-open action does nothing	File access is disabled in the app; check the `fileAccessEnabled` variable.
No feedback for the laser pointer	Correct, and deliberate — the protocol doesn't broadcast it.

See also

- README — what it does, installation, and the prior art this module was structured against
- PDF Presenter — the application this controls, and its OSC protocol