ArchPlug

DETAILED INSTRUCTIONS

Fast Script Launcher

Instantly search and run any Revit script with keyboard shortcuts.

For Autodesk Revit 2022–2026 · Windows

archplug.com · July 2026

Overview

Fast Script Launcher (FSL) is a command-palette-style launcher for your Revit automation scripts. Click one button on the ribbon (or press its keyboard shortcut), type a few characters, hit **Enter** — and your script runs immediately inside the live Revit session.

FSL turns a folder of scripts into an instantly searchable library:

- It scans a scripts folder of your choice — including all subfolders — for **Python scripts (`.py`)** and **Dynamo graphs (`.dyn`)**.
- A fast, keyboard-driven search window lets you find any script by name or by a short **keyword shortcut** you embed in the file name.
- Python scripts run **in-process** in Revit's IronPython engine with the standard `__revit__`, `doc`, `uidoc`, and `app` variables already available — pyRevit-style scripts work as-is.
- Dynamo graphs run **headless** through Revit's own Dynamo automation engine — no Dynamo window opens, the graph just executes, exactly like Dynamo Player but from your keyboard.
- Your most recently run scripts appear at the top of the list, marked with a ★, so repeat tasks are always one keystroke away.

There is nothing to configure inside your scripts and no special file format to learn. If it runs in Revit's Python environment or in Dynamo, FSL can launch it.

Key Benefits & Use Cases

For Python / pyRevit-style script users

- **Zero-friction execution.** Instead of opening a Python console, browsing to a file, and running it, you type two or three letters and press Enter. The launcher window remembers its position and opens focused on the search box — your hands never leave the keyboard.
- **pyRevit-style scripts work unchanged.** FSL injects `__revit__`, `doc`, `uidoc`, and `app` into every script's namespace *and* into Python builtins, so scripts (and modules they import) that reference these names run without modification.
- **Sibling imports just work.** The folder containing the script you run is temporarily added to `sys.path`, so a script can `import` helper modules that sit next to it.
- **Live iteration.** Because scripts are read from disk each time you run them, you can edit a script in your editor, switch to Revit, and re-run it immediately — no restart, no reload step.

For BIM managers distributing scripts to teams

- **One shared folder = one team toolkit.** Point FSL at a network or synced folder and every script you drop into it (or into any subfolder) is instantly available to the whole team — no per-user installation of individual tools, no ribbon customization.
- **Keyword shortcuts standardize usage.** Add a short keyword in parentheses to a file name — e.g. `(wt) Tag All Walls.py` — and everyone on the team can run the tool by typing `wt` + Enter. You define the shortcut vocabulary once, in the file names themselves.
- **Mixed Python + Dynamo libraries.** Teams that maintain both Python scripts and Dynamo graphs can serve both from the same launcher. Dynamo graphs are clearly marked with a teal **DYN** badge in the results list, and they run without opening the Dynamo editor — end users never see a node canvas.
- **Organize freely.** Subfolders are scanned recursively, so you can keep your library organized by discipline, project, or author without affecting how users find and run scripts.

Requirements

- **Autodesk Revit 2022 – 2026** (Windows). Revit is Windows-only, and so is Fast Script Launcher.
- **Windows 10 / 11.**
- Fast Script Launcher is part of the **ArchPlug suite**. The **ArchPlug Settings** plugin (included with the suite) is where you set the scripts folder that FSL scans.
- To run **Dynamo graphs (`.dyn`)**, Dynamo for Revit must be present on the machine. Dynamo ships with the standard Revit installation, so on a normal install this is already satisfied — nothing extra is downloaded or installed.

Installation

1. Download the Fast Script Launcher `.msi` installer from your account at archplug.com.
2. Close Revit and run the installer. It installs the plugin for the Revit versions it supports.
3. Start Revit. FSL appears on the **ArchPlug** ribbon tab alongside your other ArchPlug tools.
4. Click the Fast Script Launcher button to open the search window. On first run, use **ArchPlug Settings** to point FSL at your scripts folder (see the workflow below).

> Tip: Revit lets you assign a native keyboard shortcut to any ribbon button (**View ▸ User Interface ▸ Keyboard Shortcuts**, or type `KS`). Assigning one to Fast Script Launcher gives you a true "press key → type → Enter" workflow without touching the mouse.

User Interface Walkthrough

FSL has a single, deliberately minimal window titled **"Fast Script Launcher"** — a dark-themed search palette (620 × 460 px by default, freely resizable, minimum 400 × 300). It opens centered on screen the first time; after that it reopens exactly where you last left it. The window always stays on top of Revit but does not appear in the Windows taskbar.

From top to bottom:

Search box

A large text input at the top-left. It receives keyboard focus automatically the moment the window opens, so you can start typing immediately. Typing filters the results list live with every keystroke.

"shortcuts" toggle

To the right of the search box sits a small switch labeled **shortcuts**. It controls how FSL searches:

- **Off (default):** all scripts are listed, and your query matches against the **file name** *or* the **keyword** (the text in parentheses in the file name).
- **On (switch shows the orange accent color):** the list is restricted to scripts that *have* a keyword, and your query matches against the **keyword only**. This is the power-user mode — with well-chosen keywords, two or three characters uniquely identify any tool.

The toggle's state is remembered between sessions.

Hint line

Directly under the search box, a small gray reminder of the keyboard controls:

```
Enter = run | Esc = close | Up/Down = navigate
```

Results list

The main body of the window. Each row shows:

- **Keyword** (if the script has one) in bold orange on the left, displayed in parentheses — e.g. **(wt)**.
- **Script name** (the file name without extension) in white. Long names are trimmed with an ellipsis; hover over a row to see a tooltip with the full file name (Dynamo graphs additionally show **[Dynamo graph]** in the tooltip).
- **DYN badge** — Dynamo graphs carry a bold teal **DYN** tag on the right edge of the row so you can tell them apart from Python scripts at a glance.
- **★ (star)** — when the search box is empty, your recently run scripts are listed first, each marked with a star instead of its keyword.

The currently selected row is highlighted in orange; hovering highlights rows in gray. The first result is always pre-selected, so a single **Enter** runs the top match.

Status bar

The bottom line of the window shows, in small gray text:

- **"N scripts"** — total number of scripts found in your folder (shown when the search box is empty).
- **"N results"** — number of matches for the current query.
- Error and guidance messages when something needs attention, for example:
- `Set the scripts folder in ArchPlug Settings and try again.` — no scripts folder is configured yet.
- `ERROR: Folder not found: <path>` — the configured folder no longer exists (renamed, moved, or a disconnected network drive).
- `ERROR scanning folder: …` — the folder could not be read.

Dialogs you may see

- **"Script Error"** — shown if a script you ran raised an error. It displays the error type and message plus the file name of the script (`In: <script>.py`), so you know exactly which tool failed and why.
- **"Dynamo Not Found"** — shown if you run a `.dyn` graph on a machine where Dynamo for Revit is missing; it explains that Dynamo normally ships with Revit and can be restored from the Revit installer.

Step-by-Step Workflow

1. **Set your scripts folder.** Open **ArchPlug Settings** on the ArchPlug ribbon tab and set the scripts folder. This is the folder (with all its subfolders) that FSL will scan. It can be local, on a network share, or in a synced cloud folder.
2. **Add scripts.** Drop `.py` scripts and/or `.dyn` Dynamo graphs into the folder. No registration step — files on disk *are* the library.
3. **(Optional) Add keyword shortcuts.** Rename files to include a short keyword in parentheses, e.g. `(wt) Tag All Walls.py` or `(sh) Sheet Setup.dyn`. The first parenthesized text in the file name becomes that script's shortcut keyword.
4. **Open the launcher.** Click **Fast Script Launcher** on the ArchPlug ribbon (or press the Revit keyboard shortcut you assigned to it). The window opens with the cursor already in the search box; with an empty query you see your ★ recent scripts on top, followed by the full library.
5. **Search.** Start typing. Matching is case-insensitive substring matching — `wall` finds `Tag All Walls.py`. With the **shortcuts** toggle on, type the keyword instead — `wt`.

6. **Pick a result.** The top match is pre-selected. Use **Up/Down** to move through the list (the selection wraps around at both ends), or point with the mouse.
7. **Run.** Press **Enter** (or double-click a row). The window closes and the script executes immediately in the current Revit session — Python scripts in-process, Dynamo graphs headless. Press **Esc** at any time to close without running anything.
8. **Repeat faster next time.** The script you just ran moves to the top of the ★ recent list, so re-running it is: open launcher → Enter.

Feature Reference

Folder scanning

- FSL scans the configured scripts folder **recursively** (all subdirectories) every time the window opens, so newly added or renamed files appear without any refresh action.
- Two file types are collected: `*.py` (Python) and `*.dyn` (Dynamo graphs). Everything else is ignored.
- The total count is reported in the status bar as **"N scripts"**.

Keyword shortcuts

- A keyword is declared directly in the file name: the **first text in parentheses** is extracted as the shortcut, e.g. `(dim) Renumber Dimensions.py` → keyword `dim`.
- Keywords are shown in bold orange in the results list.
- With the **shortcuts** toggle **on**, only scripts with keywords are listed and search matches keywords exclusively — the fastest possible lookup.
- The toggle state persists across Revit sessions.

Search matching

- Matching is **case-insensitive substring** matching — no need to type from the beginning of a name.
- Toggle **off**: a query matches if it appears in the file name *or* in the keyword.
- Toggle **on**: a query matches only against keywords.
- Results update on every keystroke, and the first result is auto-selected so **Enter** always runs the best match.

Recent scripts

- FSL remembers your **8 most recently run scripts** (per user).
- With an empty search box, recents are listed first, each marked ★, followed by the full library.

- The list is self-cleaning: entries whose files have been deleted or moved simply disappear.
- Recents respect the shortcuts toggle — with the toggle on, only recents that have keywords are shown.

Keyboard controls

Key	Action
Enter	Run the selected script (works from the search box or the list)
Esc	Close the window without running anything
Down / Up	Move the selection; wraps from last to first and vice versa
(typing)	Filters the list live
Double-click	Runs the clicked script (mouse alternative to Enter)

Python script execution

- Scripts run **in-process** inside Revit's IronPython engine — the same live session, the same open document, no external interpreter.
- Each script gets a fresh namespace pre-loaded with:
- `__revit__` and `__uiapp__` — the Revit `UIApplication`
- `doc` — the active `Document`
- `uidoc` — the active `UIDocument`
- `app` — the `Application`
- plus `__doc__`, `__uidoc__`, `__app__` aliases, `__file__` set to the script's path, and `__name__ = "__main__"`
- The same variables are also placed into Python **builtins**, so helper modules your script imports can access `__revit__` / `doc` / `uidoc` / `app` too — matching pyRevit conventions.
- The script's own directory is temporarily added to `sys.path` during execution, so `import my_helpers` works for modules stored next to the script.
- A built-in compatibility shim provides a minimal `traceback` module if the engine doesn't have one, so scripts that call `traceback.format_exc()` in their error handlers won't crash on that call.
- If the script raises an error, FSL shows a **"Script Error"** dialog with a cleaned-up error type (e.g. `ArgumentError` rather than a raw .NET exception name), the message, and the script's file name.

Dynamo graph execution

- `.dyn` files run **headless** through the DynamoRevit automation interface — the same mechanism Dynamo Player uses. No Dynamo window opens; the graph is loaded, executed

synchronously against the current document, and shut down afterwards.

- Graphs saved in **manual run mode** are executed anyway — you don't need to re-save graphs as automatic.
- FSL uses the Dynamo binaries that ship with your Revit version. If Dynamo has already been opened in the current session, that loaded instance's location is used; otherwise FSL finds the standard Dynamo for Revit installation for your running Revit version automatically.
- Dynamo rows are visually distinct: a teal **DYN** badge in the list and a **[Dynamo graph]** note in the row tooltip.

Window memory

- **Position:** the window's on-screen position is saved when it closes and restored next time. If your monitor layout changed and the saved position would land off-screen (e.g. a disconnected second monitor), FSL detects this and falls back to opening centered — the window can never come back "lost" outside the visible desktop.
- **Size:** the window is freely resizable to fit long script names or small screens.
- **Focus and stacking:** the window opens as a dialog owned by the Revit main window, so it stays on top of Revit while you search, and the search box is focused instantly.
- The dark title bar and orange accent follow the shared ArchPlug visual theme.

Tips & Best Practices

- **Design a keyword vocabulary.** Keep keywords short (2–4 characters), unique, and memorable — `wt` for wall tagging, `sh` for sheets, `qa` for model checks. With the shortcuts toggle on, every tool becomes a two-keystroke command.
- **Assign a Revit keyboard shortcut to the FSL ribbon button** (`KS` in Revit). The full loop then becomes: shortcut → type keyword → Enter — comparable to a command palette in a code editor.
- **Leave the search box empty to reuse recent tools.** During repetitive phases (tagging, sheet setup, QA passes) the ★ recents section means your working set is always pre-listed — just Enter, or one Down-arrow away.
- **Use subfolders for organization, names for retrieval.** Folder structure doesn't affect search, so organize the library however suits the team; make the *file names* descriptive, because that's what search matches.
- **For teams, host the folder on a share or synced drive.** Everyone points ArchPlug Settings at the same folder; updating a script in one place updates it for everybody the next time they open the launcher.
- **Name Dynamo graphs like tools, not like files.** Since users run graphs without ever seeing the canvas, a name like `(fin) Floor Finish Areas.dyn` communicates everything they need.

- **Iterate live.** When developing a script, keep your editor and Revit side by side — save in the editor, re-run from FSL. The file is re-read from disk on every run.

Troubleshooting & FAQ

The status bar says "Set the scripts folder in ArchPlug Settings and try again." No scripts folder has been configured yet. Open **ArchPlug Settings** on the ribbon, set the scripts folder, then reopen Fast Script Launcher.

The status bar says "ERROR: Folder not found: ..." The configured folder no longer exists at that path — it was moved, renamed, or is on a network/removable drive that isn't currently connected. Reconnect the drive or update the folder in ArchPlug Settings.

My script doesn't appear in the list. Check that: (1) the file extension is `.py` or `.dyn` — other types are ignored; (2) the file is inside the configured scripts folder or one of its subfolders; (3) the **shortcuts** toggle isn't on — with it on, only scripts that have a `(keyword)` in their file name are listed.

Searching finds nothing, but I know the script is there. If the **shortcuts** toggle is on, search matches *keywords only*, not file names. Switch the toggle off to search by name, or type the script's keyword instead.

I get a "Script Error" dialog when running my script. The dialog shows the error type, message, and which script failed. This is an error raised by the script itself while running — the same one you'd get in any Python console. Fix the script, save, and simply run it again from FSL (no restart needed).

I get "Dynamo Not Found" when running a .dyn graph. Dynamo for Revit isn't present on this machine. It normally ships with Revit — re-run the Revit installer and include Dynamo, then try again. Python scripts are unaffected.

My Dynamo graph is saved in Manual run mode — will it run? Yes. FSL forces execution even for graphs saved as manual, so you don't need to change the graph's run mode.

Does my script need to be written specially for FSL? No. Any script that expects the standard `__revit__`, `doc`, `uidoc`, `app` variables (the pyRevit/RevitPythonShell convention) runs as-is. Scripts also receive `__file__` and run with `__name__ == "__main__"`.

Can my script import helper modules from its own folder? Yes. The script's directory is added to the Python path for the duration of the run, so sibling imports work. The path is removed again afterwards to keep the environment clean.

The launcher window opened in an odd place / on a monitor I no longer have. FSL validates the saved window position against the currently visible screens and re-centers automatically if the saved spot is off-screen. If you simply want it elsewhere, drag it — the new position is remembered when the window closes.

How many recent scripts are kept? The last **8** scripts you ran, most recent first. Deleted or moved files drop off the list automatically.

I renamed a script's keyword but the old one still shows. The folder is re-scanned every time the launcher window opens, so simply close and reopen the launcher — the new file name (and keyword) will be picked up. Note that a rename also changes the file's identity for the ★ recents list.

Do new files require any kind of refresh or re-index? No. Every launch of the window performs a fresh scan of the folder, so anything added, removed, or renamed is reflected immediately.
