Post-processing scripts

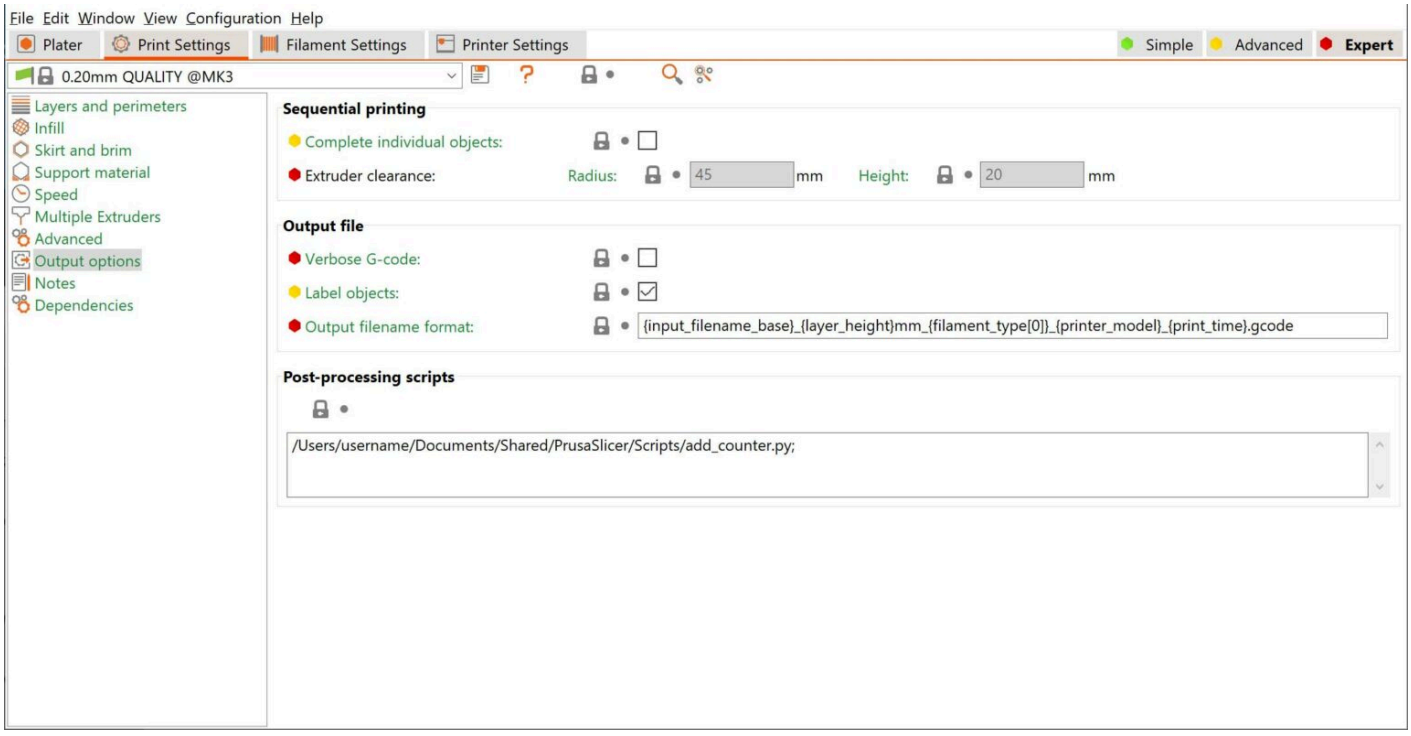
Relevant for PRUSASLICER

1 comment

Article is also available in following languages

There are some things that PrusaSlicer simply doesn't do. However, using post-processing scripts you can automatically modify the generated G-code to do (almost) anything you want.

You can specify the path to the script in **Print settings - Output options - Post-processing scripts**.



Script setup

Post-processing scripts can be written in **any programming language** (Perl, Python, Ruby, Bash etc.). They just need to be recognized by your system as an executable and accept the path to the G-code file as the only argument.

If you want to run several scripts, put each script invocation into its own line.

Script execution

Each script will be passed the absolute path of a **temporary** G-code file that PrusaSlicer generates. This file is stored in a temporary folder on your drive (typically your fast system drive). The script is then executed to modify the G-code in place and the resulting G-code file is written to your selected target folder or sent to a print host, such as PrusaConnect or Octoprint.

> Please note that the G-code viewer still visualizes G-code before post-processing.

PrusaSlicer automatically converts all environment variables used in post-processing scripts to uppercase. To ensure compatibility, always reference them using fully capitalized names, such as "SLIC3R_FILL_DENSITY".

Two additional environment variables are passed to post processing scripts: The environment variable SLIC3R_PP_HOST provides the host specification, where "File" means copying the G-code to a local hard drive or removable media, while the other values ("PrusaLink", "Repetier", "SL1Host", "OctoPrint", "FlashAir", "Duet", "AstroBox" ...) specify the print host type the G-code will be sent to.

The environment variable SLIC3R_PP_OUTPUT_NAME contains the name of the G-code file including path (for SLIC3R_PP_HOST == "File") or a name that will be given to the file after it is uploaded to the host (PrusaLink, Octoprint ...)

The post-processing script may suggest a new output file name (likely based on SLIC3R_PP_OUTPUT_NAME) by saving it as a single line into a new "output name" temp file, for example to add time stamps or sequence numbers to the final G-codes.The "output name" file name is to be created by suffixing the input G-code file name with ".output_name". PrusaSlicer will read the new name of the file and handle it properly, for example, when sending it to Octoprint.

Script Parameters

You can use parameters with your script as such:

/path/to/executable becomes /path/to/executable with the arg outputfilename.gcode

/path/to/executable -arg -arg2 becomes /path/to/executable with args -arg, -arg2, and outputfilename.gcode

If path to executable or arguments contain spaces, these arguments need to be escaped using the escaping style common to the platform (shell escaping style on Linux and OSX, Windows command line escaping).

Examples

Python

Adding counter to the file name example

Path to the final exported file (or its name for Octoprint)

Python script that modifies the output path

Historical changes and breaking backward compatibility

Before version 2.4, PrusaSlicer used to execute the script on the final G-code exported to your desired target medium. Which was very often a removable drive, such as an SD card. SD cards are slow and wear out, so this was not ideal. There were also problems when sending the G-code to Octoprint. The new approach with the temporary file on your system drive prevents these problems and extends the functionality of post-processing scripts. However, it might break some of your existing scripts.

This documentation page includes some texts from the original Slic3r documentation and from Bob's Project Notebook.

Was this article helpful?

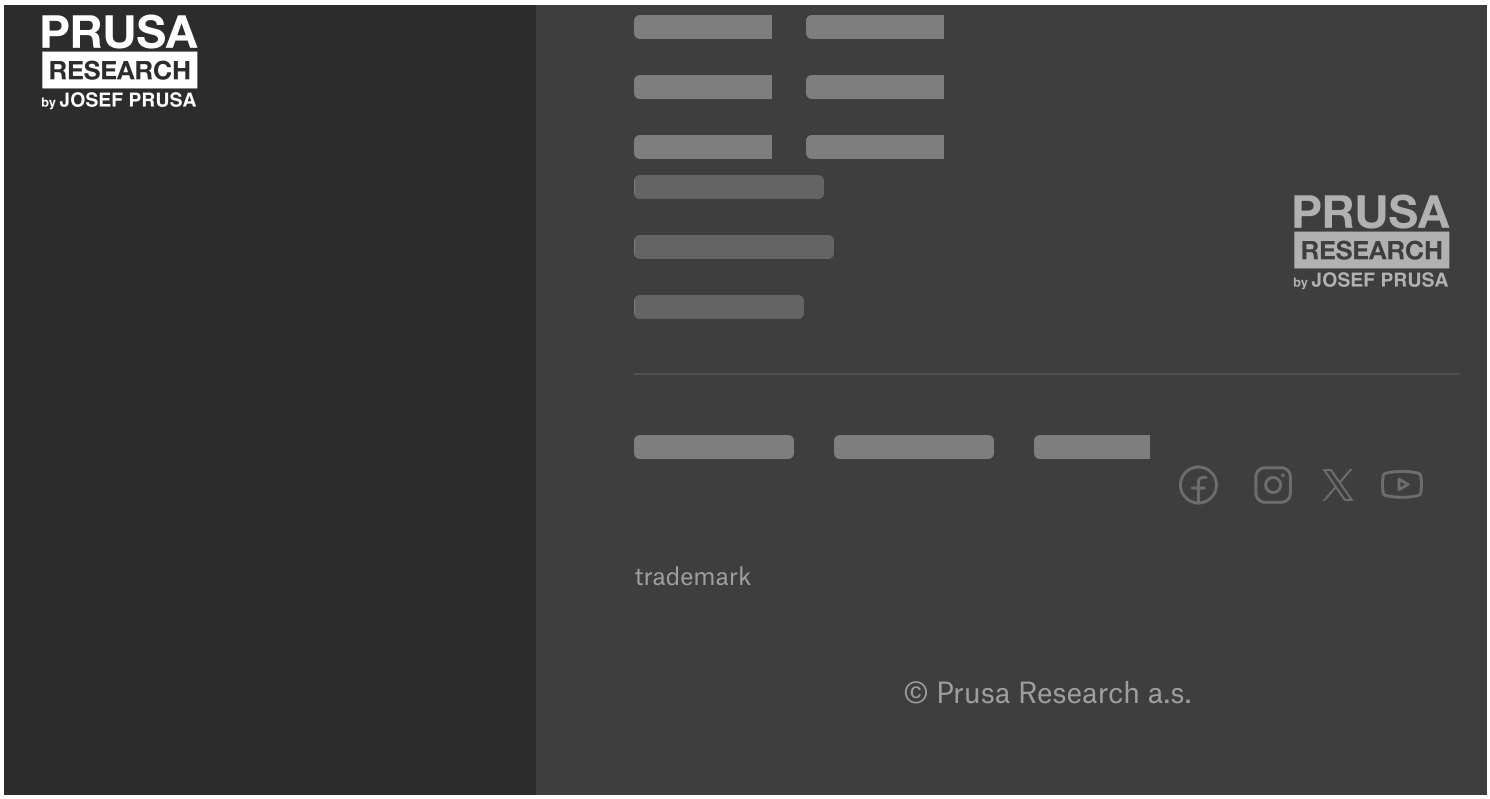
Log in to post a comment

aczekajski •

TLDR:
Your program must read the gcode from file passed as last arg and write results to the same file.

Each line of the "Post-processing scripts" will be run as a terminal command with the gcode filepath added at the end (as a last arg):
"node foo.js --bar" will be run as "node foo.js --bar /path/to/the/input.gcode".

Reply


PRUSA
RESEARCH
by JOSEF PRUSA
PRUSA
RESEARCH
by JOSEF PRUSA
trademark
© Prusa Research a.s.