



Nova 2020

Turning a static website more dynamic using GitHub Actions





Gina “foosel” Häußge

Project lead & main developer of OctoPrint



foosel / Turning a static website more dynamic using GitHub Actions

Nova 2020



Connection

State

State: **Printing**Resend ratio: **5 / 15.1K (0%)**File: **3DBenchy_.25.gcode**Uploaded: **2018-08-08 12:53:39**User: **test**

Timelapse: -

Filament (Tool 0): **1.74m**Approx. Total Print Time: **00:43:29**Print Time: **00:04:40**Print Time Left: **00:25:12** ●Printed: **462.7KB / 2.8MB**

15%

Print

Pause

Cancel

Files



Search...

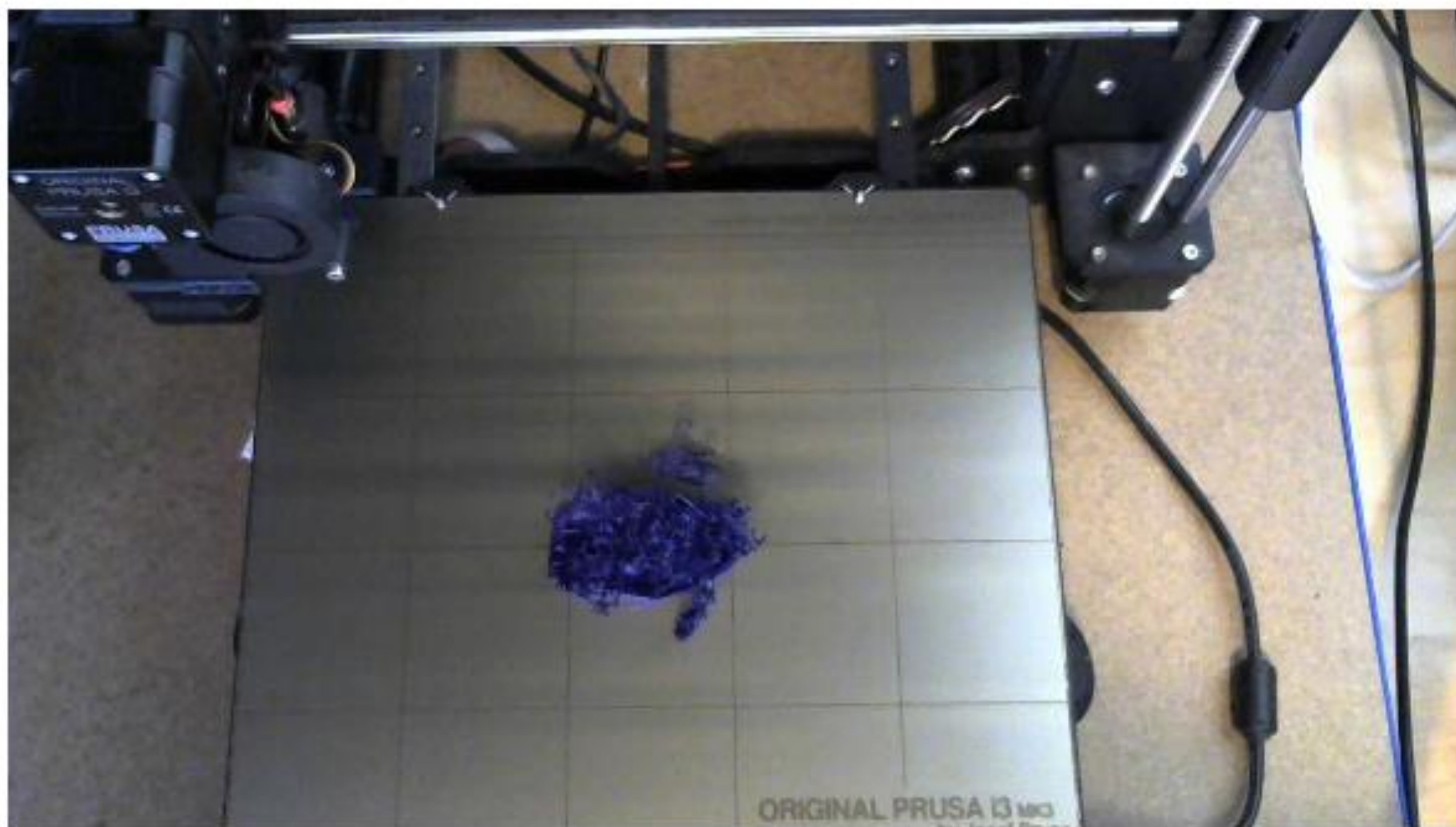
Temperature

Control

GCode Viewer

Terminal

Timelapse



X/Y

Z

Tool (E)

General



5

mm

Motors off

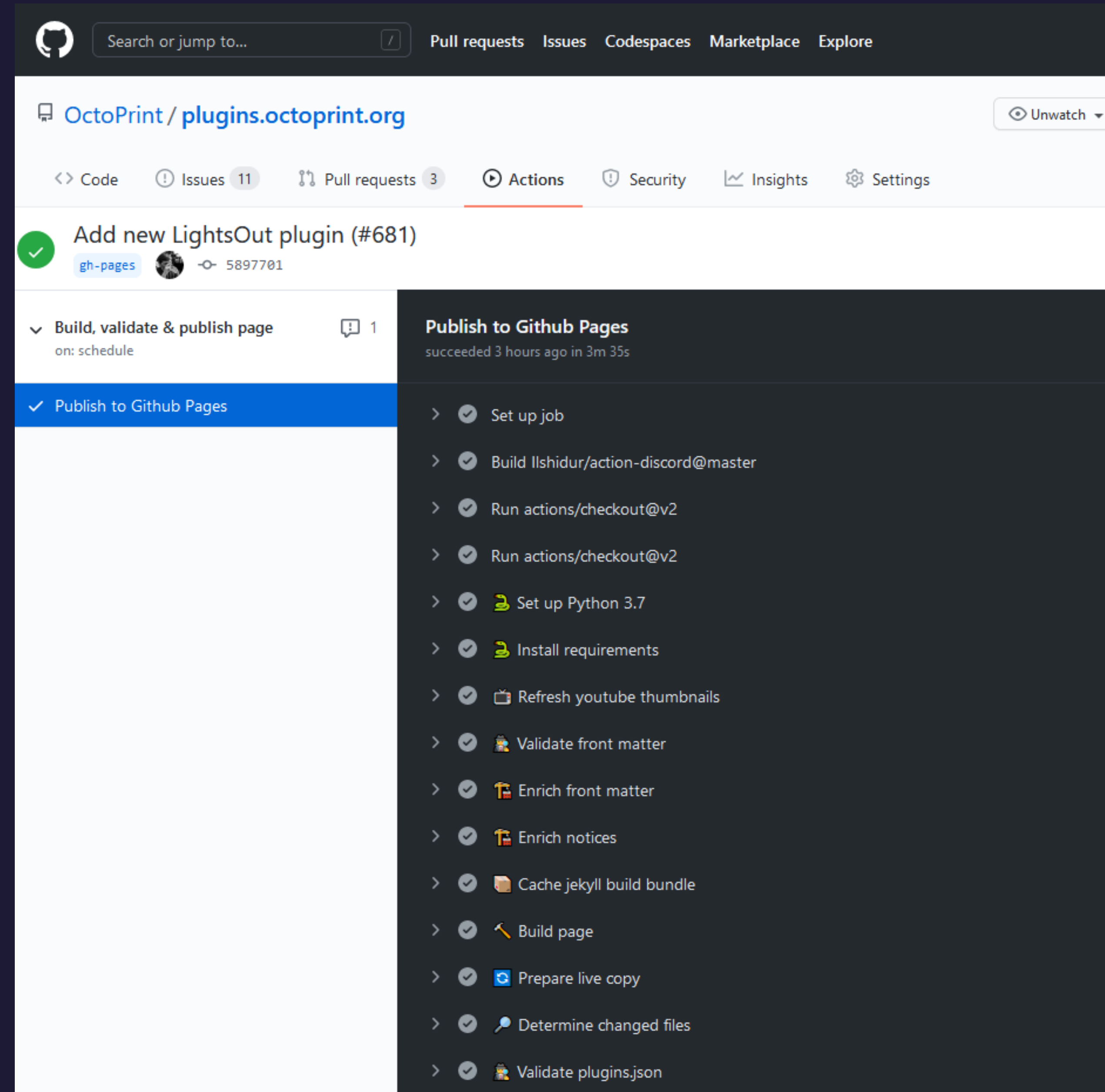
Starting point

- Plugin repository for OctoPrint at **plugins.octoprint.org**
- Over 260 plugins by 180 authors, most of them hosted on GitHub
- Static page: Jekyll (w/ collections) on GitHub Pages
- Problem: Which plugins are popular, hot, well maintained, ...
- Problem: Outdated compatibility information (Python 2 vs 3)



Enter: GH Actions

- Page build workflow
- Caching
- Data enrichment



The screenshot shows the GitHub Actions interface for the repository 'OctoPrint / plugins.octoprint.org'. The workflow 'Add new LightsOut plugin (#681)' is displayed, showing a successful run of the 'Publish to Github Pages' action. The workflow steps are listed on the right, including 'Set up job', 'Build llshidur/action-discord@master', 'Run actions/checkout@v2', 'Set up Python 3.7', 'Install requirements', 'Refresh youtube thumbnails', 'Validate front matter', 'Enrich front matter', 'Enrich notices', 'Cache jekyll build bundle', 'Build page', 'Prepare live copy', 'Determine changed files', and 'Validate plugins.json'.

Search or jump to... Pull requests Issues Codespaces Marketplace Explore

OctoPrint / plugins.octoprint.org Unwatch

<> Code Issues 11 Pull requests 3 Actions Security Insights Settings

✓ Add new LightsOut plugin (#681)
gh-pages 5897701

✓ Build, validate & publish page
on: schedule 1

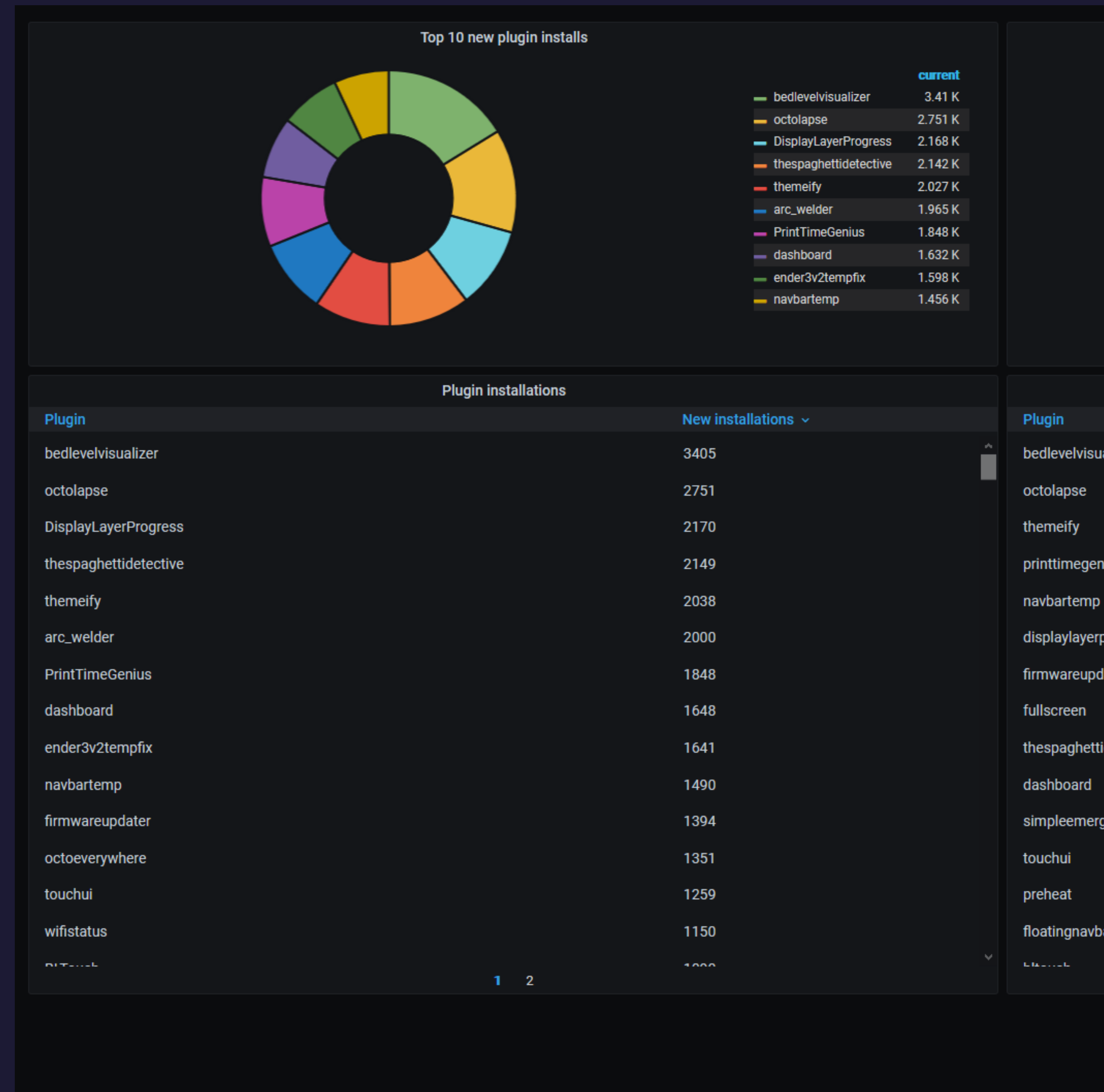
✓ Publish to Github Pages

Publish to Github Pages
succeeded 3 hours ago in 3m 35s

- > ✓ Set up job
- > ✓ Build llshidur/action-discord@master
- > ✓ Run actions/checkout@v2
- > ✓ Run actions/checkout@v2
- > ✓ 🐍 Set up Python 3.7
- > ✓ 📦 Install requirements
- > ✓ 📺 Refresh youtube thumbnails
- > ✓ 📄 Validate front matter
- > ✓ 📄 Enrich front matter
- > ✓ 📄 Enrich notices
- > ✓ 📦 Cache jekyll build bundle
- > ✓ 🔨 Build page
- > ✓ 🔄 Prepare live copy
- > ✓ 🔍 Determine changed files
- > ✓ 📄 Validate plugins.json

Usage data

- JSON export from ElasticSearch
- Python build step populates Jekyll collection prior to page build
- frontmatter, requests



GitHub Metadata & raw files

- Detect GitHub hosted plugins (source URL)
- Fetch latest release, number of releases, number of open/closed issues, number of stars, last push
- Fetch main plugin file, parse into AST & extract compatibility information
- Long runtime: Parallelization & not on pull requests



Challenge: API rate limiting

- API calls with personal access token
- Use GraphQL where possible, all in one query
 - Problem: releases (no prerelease filter)
 - Fetch via GraphQL, if prerelease fetch via API
- Refresh page only twice per day

```
16 PLUGINSTATS_500_URL = "https://data.octoprint.org/export/plugin_stats_500.json"
17
18 GITHUB_PREFIX = "https://github.com/"
19 GITHUB_GRAPHQL_URL = "https://api.github.com/graphql"
20 GITHUB_GRAPHQL_QUERY = ""
21 query {
22   repository(owner: \"{{user}}\", name: \"{{repo}}\") {
23     openIssues: issues(states: OPEN) {
24       totalCount
25     },
26     closedIssues: issues(states: CLOSED) {
27       totalCount
28     },
29     releasesCount: releases(last: 100){
30       totalCount
31     },
32     lastRelease: releases(last:1){
33       nodes {
34         name,
35         publishedAt,
36         url,
37         tagName,
38         isPrerelease
39       }
40     },
41     lastPush: defaultBranchRef {
42       target {
43         ... on Commit {
44           history(first: 1){
45             edges{
46               node {
47                 committedDate
48               }
49             }
50           }
51         }
52       }
53     },
54     watchers(last:100){
55       totalCount
56     },
57     stargazers(last:100){
58       totalCount
59     }
60   }
61 }""
62
63 GITHUB_REST_URL = "https://api.github.com"
```



Star gazing ☆

- OAuth starring proxy
 - Python + flask + flask-dance
 - Docker
- Some JS magic allows starring right from the repository
- Challenge: OAuth Scope needed for starring unclear



Current Run Configuration

Plugin Status: Enabled

Pre-Processing Types: Automatic and Manual Processing Enabled

Resolution: 0.05mm

Maximum Arc Radius: 9999mm

Output File: A new file with a postfix of '.aw' will be added after pre-processing is completed

G90 Influences Extruder: false (using octoprint settings)

Source File Deletion: The source file will not be deleted

Show Stats

Edit Settings

[About](#) [Github](#) [YouTube](#) [PayPal](#) [Patreon](#)

Arc Welder v0.1.0rc1.dev6+u.62a38d3

The Arc Welder Tab

How To Use Arc Welder

Please read the [readme file](#) in the Github Repository for installation and usage instructions. I am planning to add a complete wiki for the plugin in the near future and will link to that here when it is complete.

Please note that if you are using Python 3, you may need to install the *python3-dev* package before *Arc Welder* will install. This is detailed in the [prerequisites section of the readme file](#) linked to above.

Registration date

24 Oct 2020

Active instances the past month

Installed on at least **2.04k** instances

New installs the past week

At least **489** new installations

GitHub stats [?](#) [\(Logout\)](#)

- Latest Release: **1.0.0**
released on 24 Oct 2020
- Releases: 7
- Stars: 142 
- Issues: 22 open, 49 closed
- Last push: 25 Oct 2020

Tags

[arc](#) [compression](#) [g2](#) [g3](#) [gcode](#)
[preprocessing](#) [stutter](#)

Compatibility Information

- OctoPrint: 1.4.0+
- Operating Systems: Linux, Windows, FreeBSD
- Python: >=2.7,<4

Repository...

- Users can easily find plugins that are popular, trending, ...
- Users get an indicator of the development activity on a plugin
- Users get more accurate compatibility information
- Users can show appreciation by starring plugins



like BigTreeTech Smart Filament Sensor to

th our powerful cloud management

onitoring task queue webcam

Print navigation bar.

OS Platform API.

ion 27 Sep 2020

Code commands in the Terminal tab

Sep 2020

n

020

sier to see where a message is from.

There are currently **262** plugins listed in this repository of which **199 (75%)** are marked as Python 3 compatible.

6 plugins have been marked as abandoned and are looking for a new maintainer.

All in all **178** plugin authors have spent time and effort to bring you these plugins.

Top 10 of the month

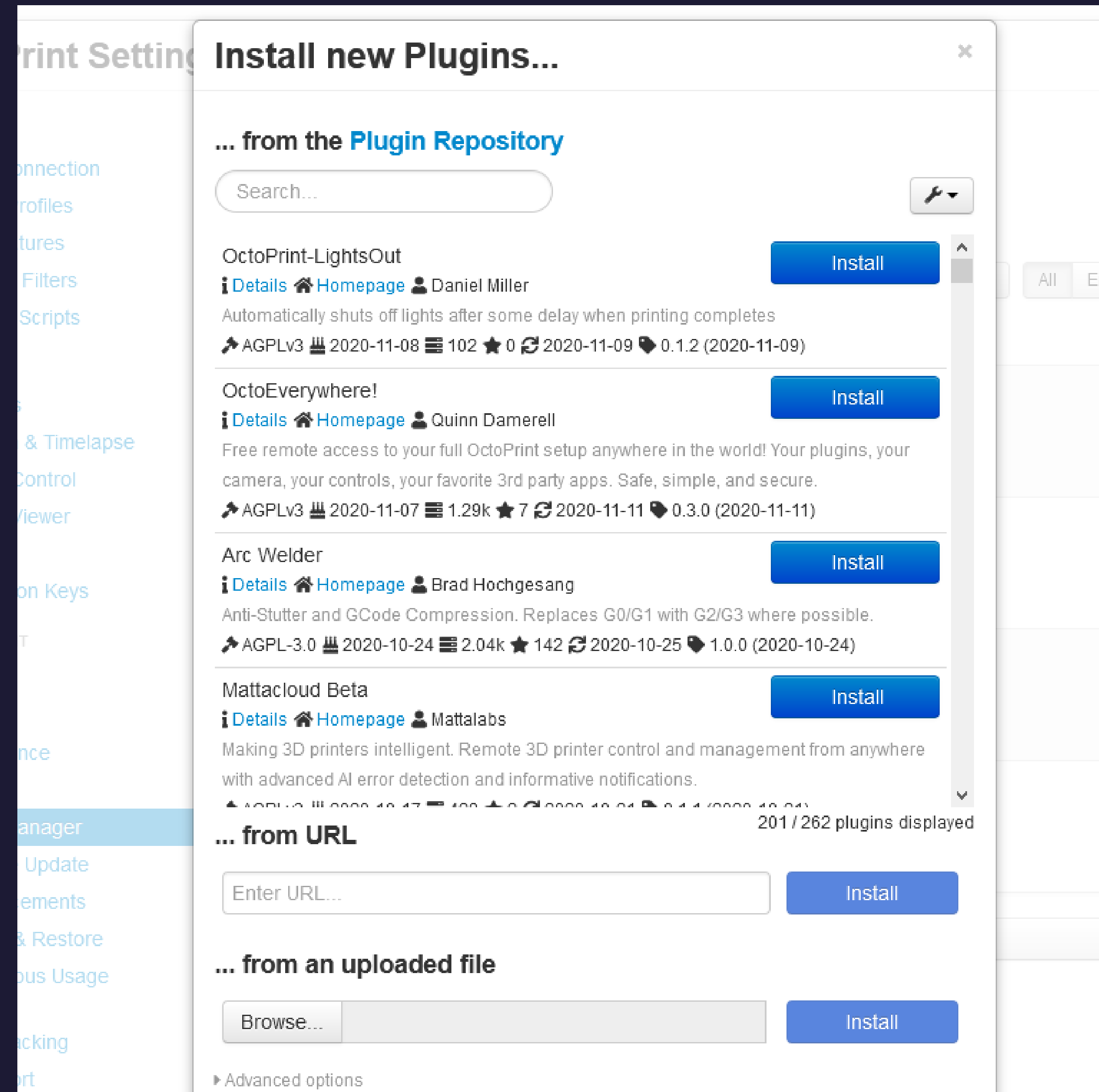
1. **Bed Level Visualizer**
over 19.86k instances
2. **Octolapse**
over 16.97k instances
3. **Themeify**
over 16.24k instances
4. **OctoPrint-PrintTimeGenius**
over 15.65k instances
5. **Navbar Temp**
over 14.18k instances
6. **DisplayLayerProgress**
over 13.99k instances
7. **Firmware Updater**
over 12.25k instances
8. **Access Anywhere - The Spaghetti Detective**
over 10.55k instances
9. **OctoPrint-Dashboard**
over 9.81k instances
10. **Simple Emergency Stop**
over 7.65k instances

Trending this week

1. **OctoEverywhere!**

... & OctoPrint

- Added to export to OctoPrint
- Availability in built-in repository browser
- Sorting, filtering, ...
- No starring (yet?)



Code?

Workflow, enrichment & validation scripts, page

Page source

OctoPrint/plugins.octoprint.org

Usage data JSON

Data exports

data.octoprint.org/export/

Star gazing

Star proxy

OctoPrint/github-star-proxy

Live in action

Plugin repo

plugins.octoprint.org

