



| python-gitlab

How to automate your job

Agenda

What is it and why use it?

Live Demo – Library

Live Demo – CLI

What is it?

The screenshot shows the GitHub repository page for `python-gitlab/python-gitlab`. The repository is public and has 1.8k stars, 555 forks, and 47 watchers. It is a Python wrapper for the GitLab API. The repository includes a file tree with folders like `.github`, `docs`, `gitlab`, and `tests`, and files like `.dockerignore`, `.gitignore`, `.gitlab-ci.yml`, `.pre-commit-config.yaml`, `.readthedocs.yml`, `.renovaterc.json`, `AUTHORS`, `CHANGELOG.md`, `CONTRIBUTING.rst`, `COPYING`, `Dockerfile`, `MANIFEST.in`, `README.rst`, and `codecov.yml`. The repository also has 27 branches and 72 tags. A merge pull request #2019 is visible, along with a list of recent commits and releases.

```
1 import gitlab
2
3
4 gl = gitlab.Gitlab(per_page=100)
5 for project in gl.projects.list(as_list=False):
6     print(f"{project.id} - {project.path_with_namespace}")
```

```
$ gitlab group-project list --group-id gitlab-org
id: 35337613
id: 35104827
id: 34979658
```

```
$ gitlab -o yaml -f name,id,visibility group-project list --group-id gitlab-org
- id: 35337613
  name: Release Sample Projects
  visibility: public
- id: 35104827
  name: gitlab-web-ide
  visibility: public
- id: 34979658
  name: ngalang-personal
  visibility: public
- id: 34936209
  name: labkit-python
  visibility: public
- id: 34770409
  name: GitLab Metrics Exporter
  visibility: public
```

What is it?

python-gitlab v3.4.0

Q Search

TABLE OF CONTENTS

- Getting started with the CLI
- Getting started with the API
- CLI examples
- API examples
- API reference (gitlab package)
- CLI reference (gitlab command)
- Changelog
- Release notes
- FAQ

python-gitlab

(*) Test passing pypi package 3.4.0 docs passing codecov 93% python 3.7 | 3.8 | 3.9 | 3.10 chat on gitter

code style: black

python-gitlab is a Python package providing access to the GitLab server API. It supports the v4 API of GitLab, and provides a CLI tool (`gitlab`).

Installation

As of 3.0.0, `python-gitlab` is compatible with Python 3.7+.

Use `pip` to install the latest stable version of `python-gitlab`:

```
$ pip install --upgrade python-gitlab
```

The current development version is available on both [GitHub.com](https://github.com) and [GitLab.com](https://gitlab.com), and can be installed directly from the git repository:

```
$ pip install git+https://github.com/python-gitlab/python-gitlab.git
```

From GitLab:

```
$ pip install git+https://gitlab.com/python-gitlab/python-gitlab.git
```

Using the docker image

- Python library and CLI tool
- ~1.4m monthly package installs
- Available from PyPI, source or as Docker image
- Powers IaC and automation tooling (Ansible modules, `python-semantic-release`, ..)

Why?

"It's just a few curl calls, I don't need a library"

Why?

"It's just a few curl calls, I don't need a library"

Two curl calls later:

```
# prepare urlencode <url> function, allowing converting provided string
```

```
urlencode: |
  urlencode() {
    # urlencode <string>

    old_lc_collate=$LC_COLLATE;
    LC_COLLATE=C;

    local length="${#1}"
    for i in `seq 1 $length`
    do
      local c=$(expr substr "$1" $i 1)
      case $c in
        [a-zA-Z0-9._~]) printf '%s' "$c" ;;
        *) printf '%%%02X' "$c" ;;
      esac
    done

    LC_COLLATE=$old_lc_collate
  }
```

jq doesn't fetch pages. You need to fetch pages with multiple curl calls, JSON outputs. For example you could get a JSON array of the combined

```
url=https://mywebsite.com/api/cars.json
jq -s '.[0].items + .[1].items' <(curl "$url&page=1" | jq .) <
```

▲ The following statement will retry 5 times or a maximum of 5 seconds, and no exponential backoff policy

128

```
▼
🕒
curl --connect-timeout 5 \
  --max-time 10 \
  --retry 5 \
  --retry-delay 0 \
  --retry-max-time 40 \
  'http://your_url'
```

Why?

"It's just a few curl calls, I don't need a library"

Two curl calls later:

```
# prepare urlencode <url> function, allowing converting provided string
```

```
urlencode: |
  urlencode() {
    # urlencode <string>

old_lc_collate=$LC_COLLATE;
LC_COLLATE=C;

local length="${#1}"
for i in `seq 1 $length`
do
  local c=$(expr substr "$1" $i 1)
  case $c in
    [a-zA-Z0-9._~]) printf '%s' "$c" ;;
    *) printf '%%%02X' "$c" ;;
  esac
done

LC_COLLATE=C
}
```

jq doesn't fetch pages. You need to fetch pages with multiple curl calls, JSON outputs. For example you could get a JSON array of the combined

```
url=https://mywebsite.com/api/cars.json
jq -s '.[0].items + .[1].items' <(curl "$url&page=1" | jq .) <
```

▲ The following statement will retry 5 times or a maximum of 5 seconds, and no exponential backoff policy

128

```
▼
🕒
curl --connect-timeout 5 \
  --max-time 10 \
  --retry 5 \
  --retry-delay 0 \
  --retry-max-time 40 \
  'http://your_url'
```

python-gitlab:

```
projects = gl.projects.list(all=True)
```

Why?

"It's just a few curl calls, I don't need a library"

Two curl calls later:

```
# prepare urlencode <url> function, allowing converting provided string
urlencode: |
  urlencode() {
    # urlencode <string>

    old_lc_collate=$LC_COLLATE;
    LC_COLLATE=C;

    local length="${#1}"
    for i in `seq 1 $length`
    do
      local c=$(expr substr "$1" $i 1)
      case $c in
        [a-zA-Z0-9._~_-]) printf '%s' "$c" ;;
        *) printf '%%%02X' "$c" ;;
      esac
    done

    LC_COLLATE=$old_lc_collate
  }
```

jq doesn't fetch pages. You need to fetch pages with multiple curl calls, JSON outputs. For example you could get a JSON array of the combined

```
url=https://mywebsite.com/api/cars.json
jq -s '.[0].items + .[1].items' <(curl "$url&page=1" | jq .) <
```

The following statement will retry 5 times or a maximum of 5 seconds, and no exponential backoff policy

128

```
curl --connect-timeout 5 \
  --max-time 10 \
  --retry 5 \
  --retry-delay 0 \
  --retry-max-time 40 \
  'http://your_url'
```

python-gitlab does:

- Proper URL encoding
- Pagination
- Rate limits
- Retries on network & server errors
- Complex API attributes (arrays, delimited strings)
- Generic methods for arbitrary endpoints
- Passing arbitrary API parameters (in most cases)

Live Demo - Library

Live Demo - CLI

python-gitlab without python

| Contacts

Nejc Habjan – nejc.habjan@siemens.com

Max Wittig – max.wittig@siemens.com

Issue tracker: <https://github.com/python-gitlab/python-gitlab>

Docs: <https://python-gitlab.readthedocs.io/en/stable/>

Chat: <https://gitter.im/python-gitlab/Lobby>