gitlab-projects-settings

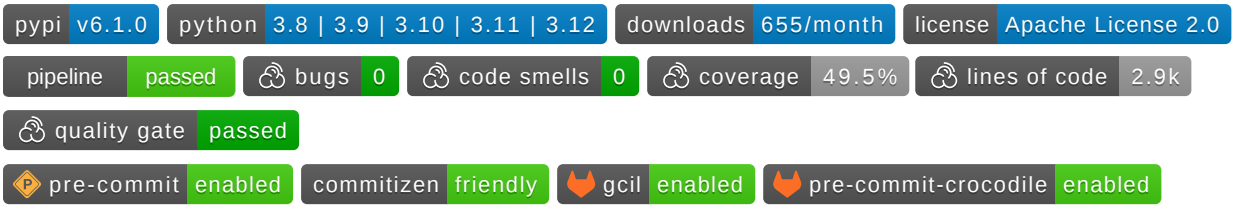
Author: Adrian DC

Description: Configure GitLab groups and projects settings automatically

Date: 01/03/2025

Version: `6.1.1`

▸ Usage



Configure GitLab groups and projects settings automatically

Documentation: https://radiandevcore.gitlab.io/tools/gitlab-projects-settings
Package: https://pypi.org/project/gitlab-projects-settings/

Table of contents

Purpose

This tool can automatically configure and update the GitLab settings
of groups, subgroups and projects, using multiple available options.

Repetitive tasks can be performed accross multiple projects at once,
for example protecting tags and branches, or setting a new avatar recursively.

The following step is required before using the tool:

- The GitLab user tokens must be created with an `api` scope (a short expiration date is recommended)

Examples

```
# Show the helper menu
gitlab-projects-settings

# Protect all projects' tags under a group
gitlab-projects-settings --protect-tags -- 'https://gitlab.com/group'

# Update all avatars and descriptions under a group
gitlab-projects-settings --set-avatar ./avatar.png --update-description 'https://gitlab.com/group'

# Automatically detect and reset features of projects based on usage
gitlab-projects-settings --reset-features -- 'https://gitlab.com/group/project'
```

Usage

```
usage: gitlab-projects-settings [-h] [--version] [--no-color] [--update-check] [--settings] [--set GROUP KEY VAL]
                                [-c FILES] [--confirm] [--dry-run] [--dump] [--exclude-group] [--exclude-subgroups]
                                [--exclude-projects] [--available-features] [--reset-features [KEEP_FEATURES]]
                                [--disable-features FEATURES] [--enable-features FEATURES] [--reset-members]
                                [--set-avatar FILE] [--set-description TEXT] [--update-descriptions]
                                [--set-roles-create-projects [ROLE]] [--set-roles-create-subgroups [ROLE]]
                                [--run-housekeeping] [--prune-unreachable-objects]
                                [--archive-projects | --unarchive-projects] [--delete-groups] [--delete-projects]
                                [--protect-branches] [--protect-tags [LEVEL]] [--set-merge-method [METHOD]]
                                [--set-merge-squash [SQUASH]] [--set-merge-pipelines [CHECK]]
                                [--set-merge-skipped [CHECK]] [--set-merge-resolved [CHECK]]
                                [--add-jobs-token-allowlist PATH] [--remove-jobs-token-allowlist PATH]
                                [--erase-jobs-artifacts | --erase-jobs-contents] [--get-project-issues-boards]
                                [--set-project-issues-boards JSON] [--get-group-labels | --set-group-labels JSON]
                                [--get-project-labels | --set-project-labels JSON] [--]
                                [url_path]

gitlab-projects-settings: Configure GitLab groups and projects settings automatically

internal arguments:
  -h, --help                          # Show this help message
  --version                           # Show the current version
  --no-color                          # Disable colors outputs with 'NO_COLOR=1'
                                      # (or default settings: [themes] > no_color)
  --update-check                      # Check for newer package updates
  --settings                          # Show the current settings path and contents
  --set GROUP KEY VAL                 # Set settings specific 'VAL' value to [GROUP] > KEY
                                      # or unset by using 'UNSET' as 'VAL'

credentials arguments:
  -c FILES, --config FILES            # Python GitLab configuration files (default: PYTHON_GITLAB_CFG environment)

common settings arguments:
  --confirm                           # Automatically confirm all removal and contents warnings
  --dry-run                           # Enable dry run mode to check without saving
  --dump                              # Dump Python objects of groups and projects
  --exclude-group                     # Exclude parent group settings
  --exclude-subgroups                 # Exclude children subgroups settings
  --exclude-projects                  # Exclude children projects settings

general settings arguments:
  --available-features                # List the available GitLab project features known by the tool
  --reset-features [KEEP_FEATURES]    # Reset features of GitLab projects based on usage
                                      # (Optionally keep features separated by ",")
  --disable-features FEATURES         # List of features to disable separated by ","
  --enable-features FEATURES          # List of features to enable separated by ","
  --reset-members                     # Reset members of GitLab projects and groups
  --set-avatar FILE                   # Set avatar of GitLab projects and groups
  --set-description TEXT              # Set description of GitLab projects and groups
  --update-descriptions               # Update description of GitLab projects and groups automatically

group settings arguments:
  --set-roles-create-projects [ROLE]   # Set roles allowed to create projects
[noone,owner,maintainer,developer,administrator] (default: developer)
  --set-roles-create-subgroups [ROLE]  # Set roles allowed to create subgroups [owner,maintainer] (default:
maintainer)

advanced settings arguments:
  --run-housekeeping                  # Run housekeeping of GitLab project or projects in groups
```

```
  --prune-unreachable-objects          # Prune unreachable objects of GitLab project or projects in groups
  --archive-projects                   # Archive project or projects in GitLab groups
  --unarchive-projects                 # Unarchive project or projects in GitLab groups
  --delete-groups                      # Delete group or groups in GitLab groups
  --delete-projects                    # Delete project or projects in GitLab groups

repository settings arguments:
  --protect-branches                   # Protect branches with default master/main, develop and staging
  --protect-tags [LEVEL]               # Protect tags at level [no-one,admins,maintainers,developers] (default: no-
one)

merge requests settings arguments:
  --set-merge-method [METHOD]          # Set project merge requests method (Merge, Semi-linear, Fast-forward,
default: Fast-forward)
  --set-merge-squash [SQUASH]          # Set project merge requests squashing (Do not allow, Allow, Encourage,
Require, default: Allow)
  --set-merge-pipelines [CHECK]        # Set project merge requests check for successful pipelines (true, false,
default: True)
  --set-merge-skipped [CHECK]          # Set project merge requests check for skipped pipelines (true, false,
default: True)
  --set-merge-resolved [CHECK]         # Set project merge requests check for resolved threads (true, false, default:
True)

ci/cd settings arguments:
  --add-jobs-token-allowlist PATH      # Add a group or project to CI/CD job token allowlist
  --remove-jobs-token-allowlist PATH   # Remove a group or project from CI/CD job token allowlist
  --erase-jobs-artifacts               # Erase all CI/CD jobs artifacts
  --erase-jobs-contents                # Erase all CI/CD jobs artifacts and traces

issues arguments:
  --get-project-issues-boards          # Get the GitLab project issues boards in JSON format
  --set-project-issues-boards JSON     # Set the GitLab project issues boards from JSON format

labels arguments:
  --get-group-labels                   # Get the GitLab group labels in JSON format
  --set-group-labels JSON              # Set the GitLab group labels from JSON format
  --get-project-labels                 # Get the GitLab project labels in JSON format
  --set-project-labels JSON            # Set the GitLab project labels from JSON format

positional arguments:
  --                                   # Positional arguments separator (recommended)
  url_path                             # GitLab group or project path URL

environment variables:
  GITLAB_TOKEN                         # GitLab API token environment variable
  CI_JOB_TOKEN                         # GitLab CI job token environment variable (CI only)
```

Python GitLab configuration file

`gitlab-projects-settings` uses the same configuration files as the `python-gitlab` API, holding domains, URL and private tokens credentials for the GitLab instances.

The default user configuration file can be created at `~/.python-gitlab.cfg`.

The `-c` or `--config` parameters can provide specific configuration files, otherwise the `PYTHON_GITLAB_CFG` environment variable can be used.

Example `~/.python-gitlab.cfg` configuration file:

~/.python-gitlab.cfg

```
[global]
default = gitlab.com
ssl_verify = true
timeout = 5

[gitlab.com]
url = https://gitlab.com
private_token = glpat-...

[gitlab.local.dev]
url = https://gitlab.local.dev
private_token = glpat-...

[gitlab.private.dev:4243]
url = https://gitlab.private.dev:4243
private_token = glpat-...
ssl_verify = /usr/local/share/ca-certificates/gitlab.private.dev.crt
```

`python-gitlab` **configuration files documentation:** Getting started with the CLI / Configuration files

Userspace available settings

`gitlab-projects-settings` creates a `settings.ini` configuration file in a userspace folder.

For example, it allows to disable the automated updates daily check (`[updates] > enabled`)

The `settings.ini` file location and contents can be shown with the following command:

```
gitlab-projects-settings --settings
```

Environment available configurations

`gitlab-projects-settings` uses `colored` for colors outputs and `questionary` for interactive menus.

If colors of both outputs types do not match the terminal's theme,
an environment variable `NO_COLOR=1` can be defined to disable colors.

Dependencies

- colored: Terminal colors and styles
- python-gitlab: A python wrapper for the GitLab API
- questionary: Interactive terminal user interfaces
- setuptools: Build and manage Python packages
- update-checker: Check for package updates on PyPI

References

- commitizen: Simple commit conventions for internet citizens
- git-cliff: CHANGELOG generator
- gitlab-release: Utility for publishing on GitLab
- gcil: Launch .gitlab-ci.yml jobs locally
- mkdocs: Project documentation with Markdown
- mkdocs-exporter: Exporter plugin for mkdocs documentation
- mkdocs-material: Material theme for mkdocs documentation
- mypy: Optional static typing for Python
- pre-commit: A framework for managing and maintaining pre-commit hooks
- pre-commit-crocodile: Git hooks intended for developers using pre-commit
- PyPI: The Python Package Index
- twine: Utility for publishing on PyPI

gitlab-projects-settings