

Resources

GitLab Repository

URL: http://gitlab01.local/csavugot/GitSrcCtrl_GUI

This repository contains all documentation and files for this project.

Terminology

Term	Definition
Repository (repo)	Folder uploaded to GitLab containing all project files. Also supports docs and non-code files.
Remote repo	Version stored on GitLab, accessed via URL. Opposite of Local.
Local repo	Version stored on your computer. Opposite of Remote.
Branch	A separate version of your code for developing/testing without affecting the main code.
GUI	Graphical user interface — buttons and windows instead of typed commands.
Tag	Snapshot marking an official release, like v1.0 .
Commit Message	Short summary of what changed in a commit. Helps others understand your work.
Commit Hash	Unique ID for a commit. Used to track and reference specific changes.
Stage (add)	Selects which files to include in your next commit. Like packing for a shipment.
Commit	Saves a snapshot of your changes locally with a message.
Push	Uploads local commits to the remote repository (GitLab).
Pull	Downloads and applies new commits from the remote repo.
Fetch	Downloads new commits from the remote repo but doesn't apply them yet.
Checkout	Switches to another branch or version of the project.

Git Flow Branch Summary

Branch	Purpose	From	To	Tag?	Summary
main	Stable, production-ready code	release/* , hotfix/*	—	Yes	Final code. Merge via --no-ff . Tag on merge.

Branch	Purpose	From	To	Tag?	Summary
develop	Integrates all completed features	main	release/*	No	Base for feature and release branches.
feature/<name>	Develop new features	develop	develop	No	Short-lived. Delete after merge.
hardware/<name>	Hardware-specific testing/debugging	develop	varies	No	Used for real hardware work. May be deleted or rebased.
release/v<version>	Final cleanup before release	develop	main, develop	Yes	Tag on merge to main, then merge into develop.
hotfix/v<version>	Emergency fixes to production	main	main, develop	Yes	Urgent patch. Tag after fix.

Tip: Always merge with `--no-ff` to keep clear branch history.
This ensures Git creates a visible merge commit, not a silent pointer move.

Branch Naming Conventions

Branch Type	Format Example	Description
Feature	feature/export-csv	For new features or experiments.
Hardware	hardware/spi-check	For hardware testing or debug work.
Release	release/v1.2	For prepping production versions.
Hotfix	hotfix/v1.3	For urgent patches to production code.
Tag	v1.0, v2.5	Used on main to mark official release points.

Note: `feature/*` and `hardware/*` branches are short-lived and always based on `develop`.