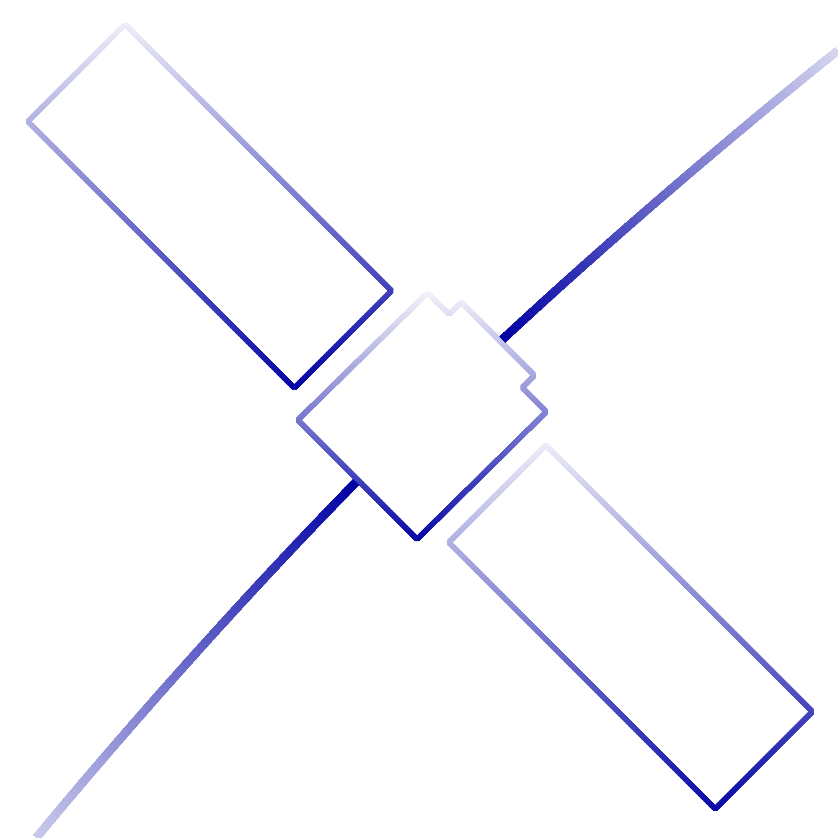


COMPASS

Developer’s Guide

Embry-Riddle Aeronautical University

Daytona Beach, FL

Table of Contents

1. Project Conventions
2. Editing Procedures
3. Deploying the Project
4. Resources

TODO:

Add QT and OpenGL Tutorials? Not like, *tutorial* tutorials, but have an outline of how they work and important information to know about each (and links to tutorials).

Project Conventions:

* Use Spaces instead of Tabs
  + QT replaces Tabs with spaces automatically
  + Notepad++ Setting: **settings**>**preferences**>**language**>**Replace by Space**
  + Convert Tabs to Spaces in Notepad++: **Edit**>**Blank Operations>TAB to Space**
* Use Standard Header Comment

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Computational Photometry Analyzer for Small Satellites

\* Embry-Riddle Aeronautical University

\*

\* File: filename.h

\*

\* @author John Doe

\* @version 8/11/2017

\*

\* Include a description of this file here.

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

* Use the prefix “m\_” for all member variables.
* Comment Everything everywhere.
* Keep a change log with each addition to the program and add it to the master change log when uploading to GitHub.

TODO: Example change Log entry. Create format.

* Make code as modular as possible. Keeps things organized and easy to find.
* Avoid using QT libraries where possible. (Example: Use STD::String instead of QString)

Procedures for editing COMPASS:

1. Download and install Git: <https://git-scm.com/downloads>

Download and install QT Creator Open Source: <https://www.qt.io/download-open-source/?hsCtaTracking=f977210e-de67-475f-a32b-65cec207fd03%7Cd62710cd-e1db-46aa-8d4d-2f1c1ffdacea>

1. Clone the COMPASS repository onto your machine using the Git Command Terminal

- CMD: git clone <https://github.com/henry-valentine/COMPASS>

2. Configure and build the project in QT

- Open COMPASS.pro in QT

- On the selected default compiler kit, go to the Details tab and deselect *profile* and *debug*

- Select the default compiler kit and continue

- Change the build directory by going to **Projects>General>Build Directory**

- Click **Browse** and select <Your Repo Copy Directory>/COMPASS/build

3. Create a new branch from master to add your changes

- Branch should be titled according to what is being added

- CMD: git branch origin <Branch Name>

4. Checkout branch and make all changes

- CMD: git checkout <Branch Name>

- Add all changes to the program that you intend to make and return to step 5.

5. Delete your build information before committing changes

- Delete COMPASS/COMPASS.pro.user

- Delete all files in COMPASS/build *except* for “ui\_CpsWindow.h”

6. Add changes to the branch

- CMD: git add -A

7. Commit changes to the branch and include a message describing the changes

- CMD: git commit -m "I changed stuff"

8. Make sure your copy of the repository is up-to-date (VERY IMPORTANT)

- CMD: git checkout master

- CMD: git pull

- If any files were updated after running the pull command, you need to merge master into your branch.

- checkout your branch again

- CMD: git checkout <Branch Name>

- Merge master into your branch

- CMD: git merge master

- You may end up with a merge conflict and the merge will fail

- If there is a conflict, find the conflict and resolve it (Ask if you're not sure)

- Add and commit changes

9. Once your version is up to date, push your branch up to GitHub to be reviewed

- Check out your branch

- CMD: git checkout <Branch Name>

- Push it up

- CMD: git push

10. Create a pull request so your changes are reviewed

- Go to repository on GitHub <https://github.com/henry-valentine/COMPASS>

- Select your branch

- Click Green Button (Compare, Review, Create Pull Request)

- Green Button at the top (Create Pull Request)

- Add a title and comments to the pull request describing what changes or additions were made

- Click Send Pull Request

11. Have a peer review your pull request, do not merge the pull request yourself.

Helpful Tips:

Accidently Overwrite Files on Local Repository before adding them to GitHub:

- CMD: git fsck −−lost-found

When promoting a Widget to a user defined class in QT Designer, make sure you add the entire path for the include file. E.g. include/cpsGUI/cpsGraph.h