



# QGROUNDCONTROL: RAMPART EDITION

User Manual

Team Rampart: Nate Zeleny, Keller Mikkelson, Eric Gault, Samuel Gilb  
Team Mentor: Mahsa Keshavarz  
Client: Dr. Michael Shafer

## Table of Contents

1. Introduction .....	2
2. Installation .....	2
For General Use .....	2
For Development .....	2
Downloading and Installing Visual Studio Community 2017 .....	2
Downloading and Installing QtCreator .....	4
Cloning the Repository off GitHub .....	6
Running the program for the first time .....	8
3. Configuration and Daily Operation .....	10
To run the program for daily operation.....	10
To Connect to the FTP Server .....	11
To Disconnect from the FTP Server .....	12
To Upload or Create a Configuration File .....	12
To Send Start and Stop Commands to the Software Defined Radio (SDR).....	13
4. Maintenance .....	14
5. Troubleshooting.....	14
6. Code Layout .....	16
System Diagrams.....	16
Code Files Diagram.....	17
Visual Layout Diagram .....	17
Interpreting the diagrams .....	17
7. Future Development .....	18
8. Conclusion .....	18

## 1. Introduction

Thank you so much for choosing Team RAMPART for your business needs. The additional radio telemetry tab we've added to the QGroundControl flight planning software is sure to save you time and has been custom designed to meet your needs. Some of the key highlights include:

- The ability to quickly change radio telemetry settings via FTP connection.
- The ability to conveniently reuse radio telemetry settings.
- The ability to easily download flight data via FTP connection.
- The ability to clearly view the UDP-based heartbeat and terminal messages sent by the drone's companion computer in real time.
- The ability to simply send radio start and stop commands to the drone's companion computer via UDP.

The purpose of this user manual is to help you, the client, successfully install, administer, and maintain the product in your actual business context going forward.

## 2. Installation<sup>1</sup>

### For General Use

We have given you a thumb drive with a Zip file called RampartQGCRRelease. Unzip the file and open the folder that you unzip. In this folder there is another folder titled "release" open it and scroll to the bottom to find an application name "QGroundControl" this is the executable that will run our program. We also recommend creating a shortcut on the desktop ". To create the shortcut right-click "QGroundControl" and then click the "create shortcut" button. A shortcut will be created in the folder, move it to your desktop and you will be good to go.

### For Development

We will first discuss how to install all the necessary components for developing the project. There are 3 steps to installing the product for use:

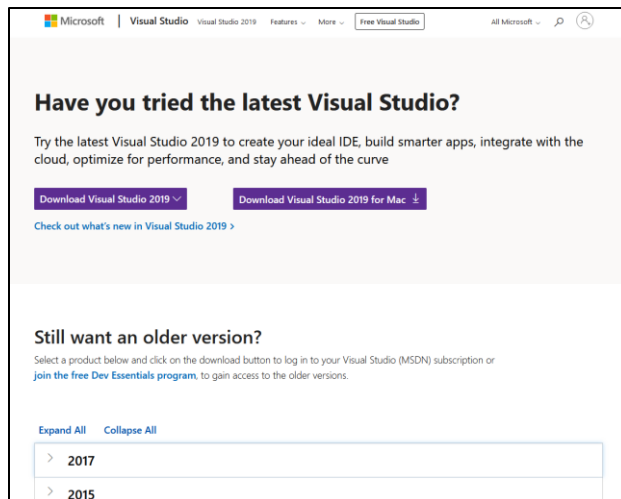
1. Downloading & Installing Visual Studio 2017
2. Downloading & Installing QtCreator
3. Cloning the RAMPART UAV-RT branch of QGroundControl

### Downloading and Installing Visual Studio Community 2017

-First follow this link: [Visual Studio Community 2017](#) which will bring you to a page that looks like this:

---

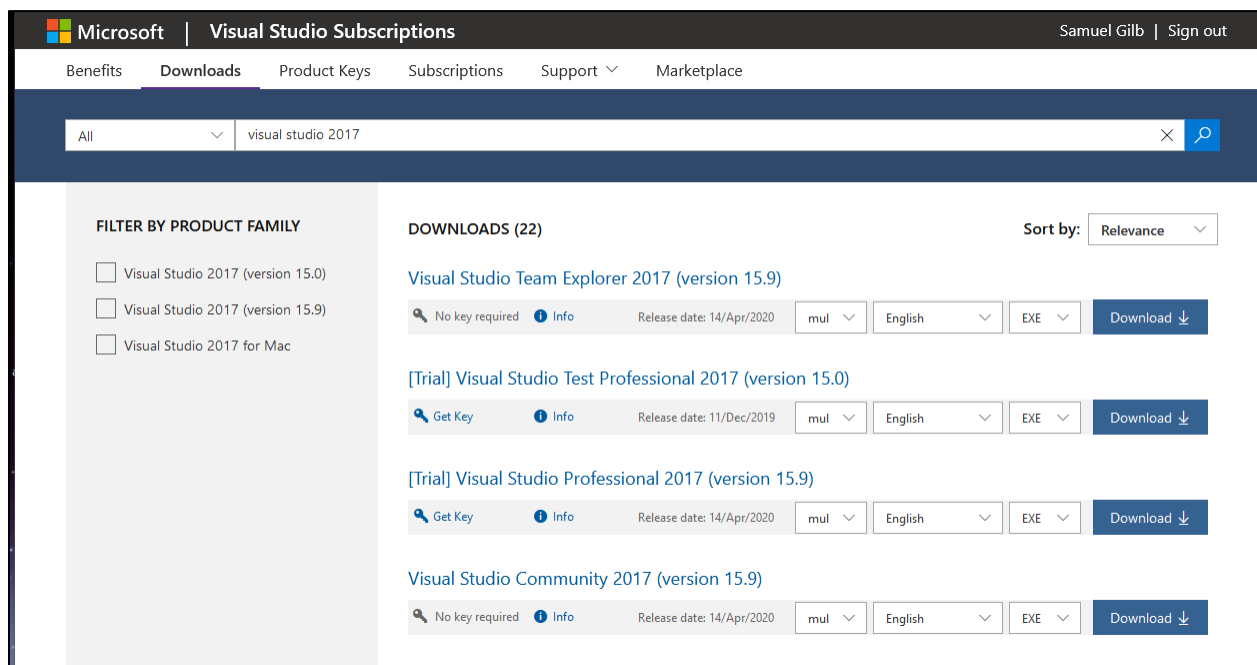
<sup>1</sup> These instructions are for machines running the Windows operating system version: Vista or higher



-If you do not have a Dev Essentials account, click the hyperlink titled “join the free Dev Essentials program”. After creating an account return to the page by clicking the hyperlink in this document again. If you already have an account ignore this and continue to the next step.

-Next expand the 2017 tab by clicking the arrow to the left of 2017 and then click the button labeled “download” in the expanded tab.

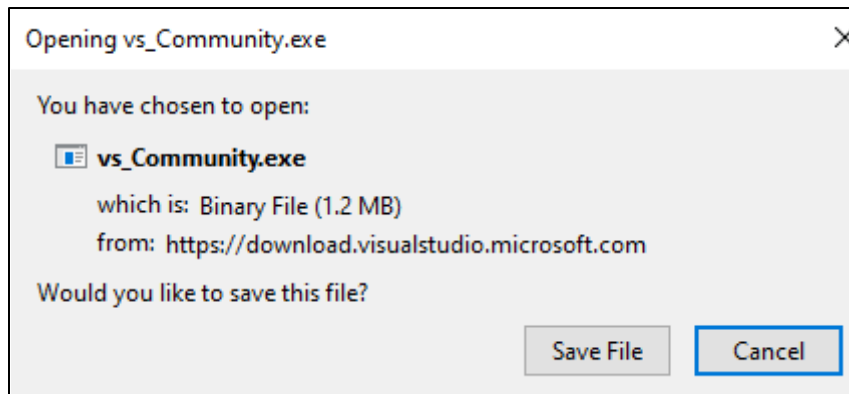
-This will bring you to a page that looks like this:



-In the Filter by product family area click the box for “Visual Studio 2017 (version 15.9)”.

-After that in the downloads area locate and click the download button for the item titled “Visual Studio Community 2017 (version 15.9)”.

-At this point a pop-up should appear that looks like this.



-Click "Save File".

-Locate the file in your downloads folder or wherever you download files to and click on the file titled vs\_Community.exe to run the installer.

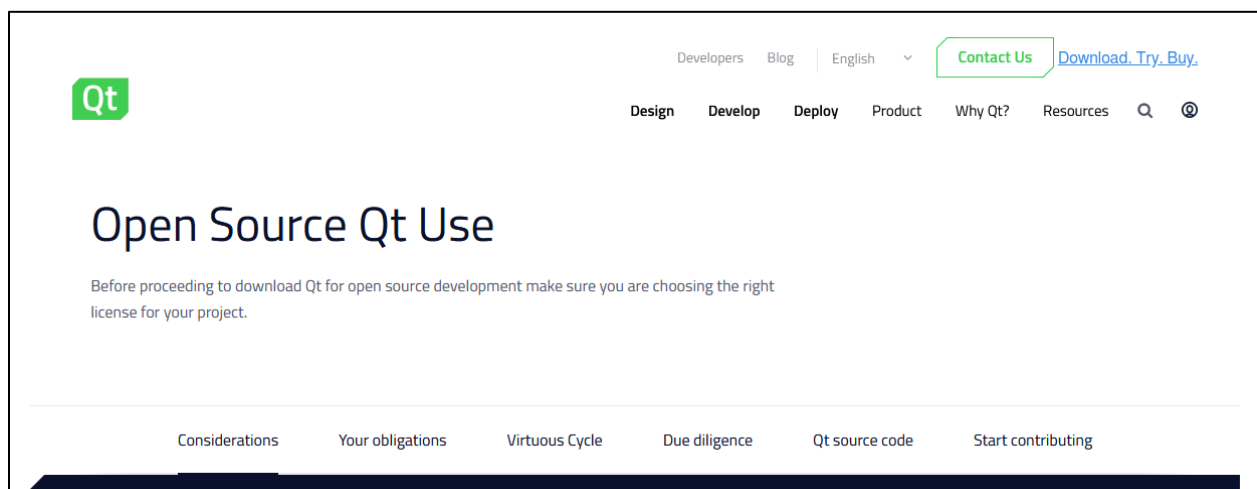
-Follow the instructions in the wizard and wait for the installation to complete.

-Congratulations you have installed the necessary version of Visual Studio 2017.

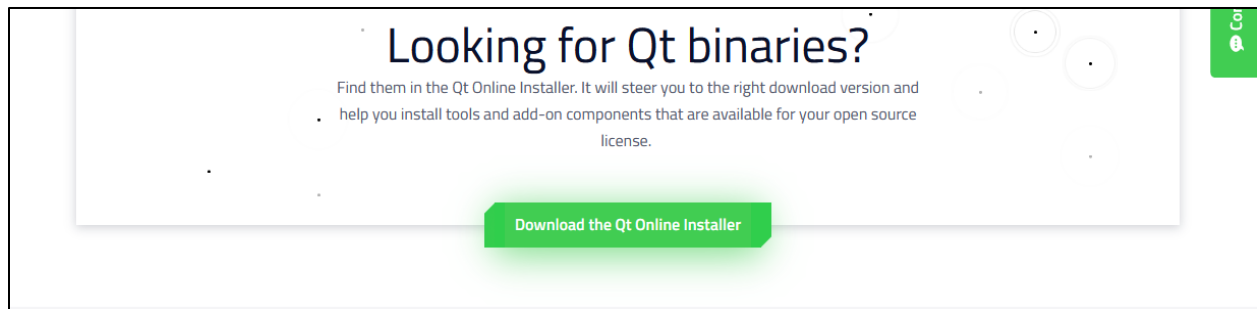
#### Downloading and Installing QtCreator

-To start open this link: [Qt Online Installer Download](#) in your web browser.

-You will be directed to a page that looks like this:

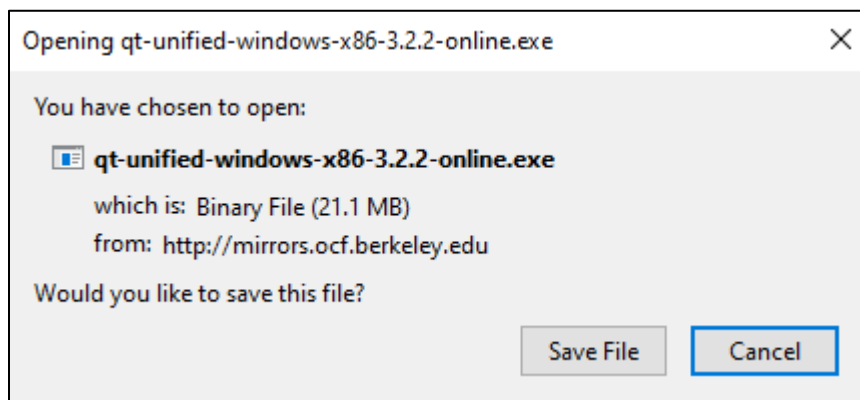


-Scroll to the bottom and click on the button which reads "Download the Qt Online Installer" as seen below:



-This will take you to another page with the title “Install Qt”. Make sure the operating system detected is windows and click the “Download” button.

-You will see a window that looks like the image below, click the “Save File” button.



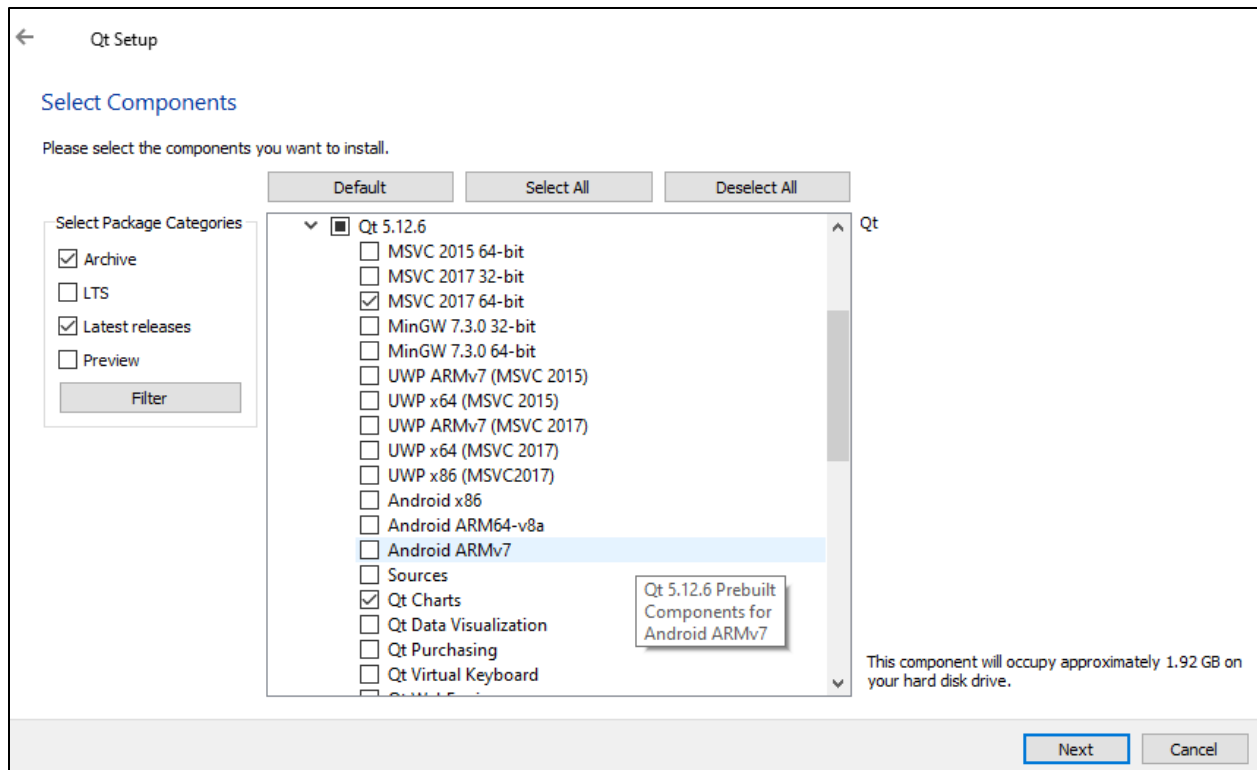
-Locate the file in your download directory or wherever you downloaded the file to and click it to begin the Qt Installation process.

-You will be asked to enter Qt Account credentials either enter your preexisting credentials or to make a free Qt account.

-After this progress through the next steps in the wizard and then set up the installation directory to where you want on your device.

-Upon reaching the select components dialog in the leftmost box click the square next to “archive” and then click the “filter” button, this will prompt another meta information download.

-After this download is completed click the arrow next to the Qt name and click the arrow next to “Qt 5.12.6” to expand the installation options, you will see a screen like the one as shown below. Make sure to check the MSVC 2017 64-bit box and the Qt Charts box 3.



-Proceed through the rest of the installation following the steps in the wizard.

-Congratulations you have installed the necessary version of Qt Creator and all required plugins.

### Cloning the Repository off GitHub

-To start navigate to the GitHub repository [right here](#). Keep this page open for either cloning case as you will need it for both.

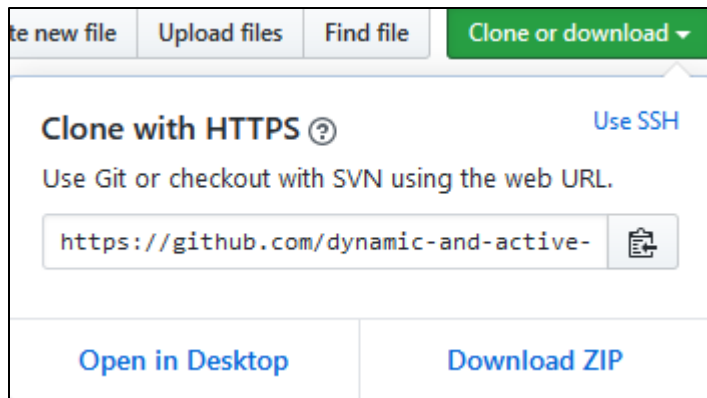
If you have a version of Git for Windows feel free to utilize the “To Clone Using Command Line interface” section otherwise scroll down to the “To Clone with GitHub Desktop “ section instead.

### To Clone using Command Line Interface (CLI)

-First open your command line interface of choice.

-Next navigate to the desired location of the cloned directory.

-At this point head back to the repository website and click the green button “Clone or download” and copy the URL by either clicking the clipboard button or by highlighting the URL and copying it. An example of the screen you will see can be seen below.



-In the Command Line Interface type the following commands (they are in italics):

*git pull (the URL copied goes here) uavrt*

at this point the repository should download.

*cd ./uavrt/*

*git submodule update*

-After running those command, the repository should be good to go.

-Congratulations you have downloaded the repository and are ready to build and run the product.

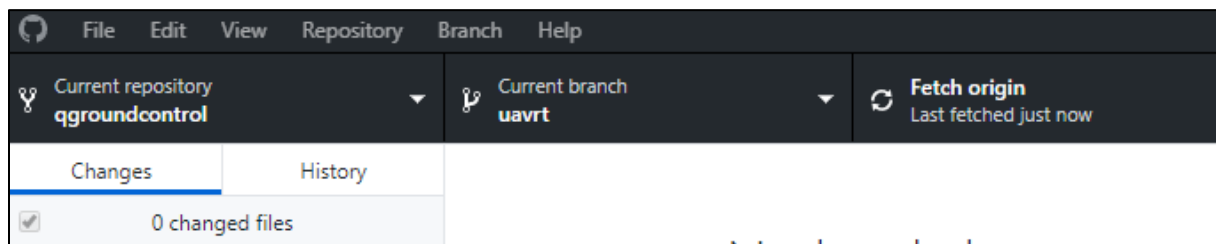
### To Clone with GitHub Desktop

-First if you do not have GitHub desktop follow this link to download it <https://desktop.github.com/>.

-Next open the webpage with the repository and click the clone or download button and then click on the "Open in Desktop" button on the bottom left. The window can be seen above in the command line interface set of instructions.

-This should open a dialog in the GitHub desktop app, make sure the local path is set to where you want to store the repository and click clone. This process may take a couple of minutes to complete.

-After the download has completed make sure to change the branch to uavrt (if you are having trouble locating the branch type "uavrt" in the filter area). Then press the fetch origin button to make sure the files have downloaded properly. The fetch origin button looks like the image below.



-Congratulations you have downloaded the repository once the fetch origin bar says: "Last fetched just now" and are ready to build and run the product.



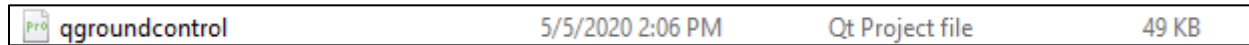
## Running the program for the first time

-After installing all the components as instructed above you should be able to build and run the project.

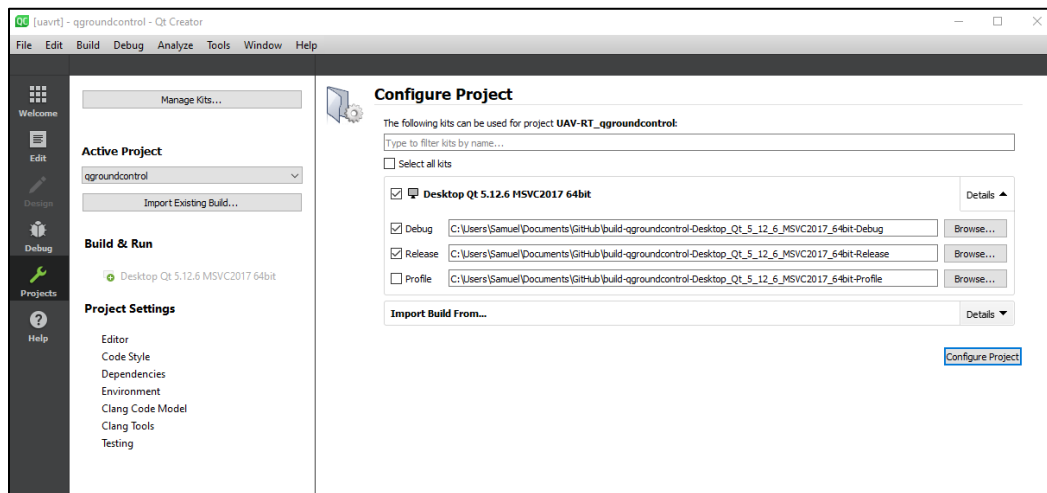
-First start by opening Qt Creator.

-Then on the projects tab click the open button.

-Then navigate to the folder where QGroundControl is stored. Navigate into the qgroundcontrol folder and locate the file named qgroundcontrol which should be of the type "Qt Project File". The file should look like the image below.

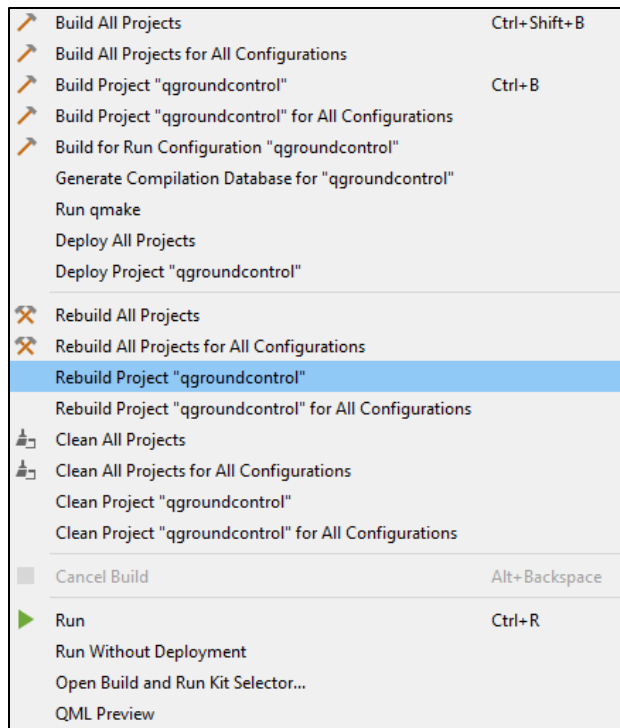


-Click this file and then click open, the project should open to a screen like the image on the next page. Click the details button on "Desktop Qt 5.12.6 MSVC2017 64bit" and uncheck the profile option as seen in the image below



If you see an error stating something similar to<sup>2</sup> "Project ERROR: MAVLink folder does not exist at 'libs/mavlink/include/mavlink/v2.0'! Run 'git submodule init && git submodule update'" on the command line." Open the build tab on the right and select "Rebuild project qgroundcontrol" option as seen in the image on the next page. The option is highlighted in blue.

<sup>2</sup> An error in this case is a red octagon, a warning which is a yellow triangle is not applicable and the program has built correctly. Our program doesn't introduce any new warnings and there should be 17 warnings in total on a successful build.

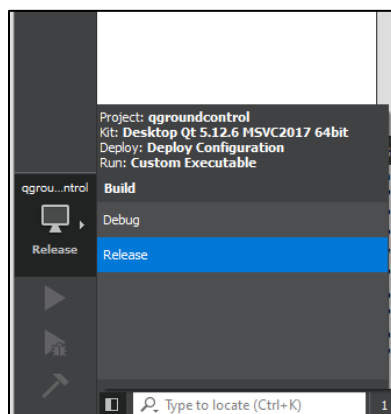


-The build should take roughly 5-15 minutes to complete.

-After the initial build completes select run on the build menu and the program should run.

TO INSTEAD MAKE AN EXECUTABLE FILE:

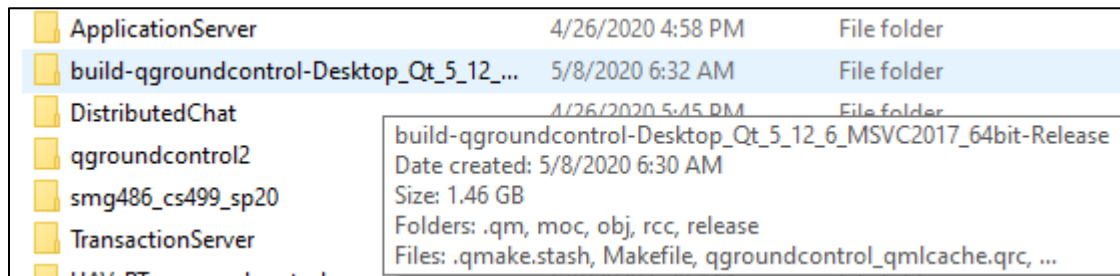
-Instead of building using the build menu click the computer icon in the bottom left of the Qt Creator interface and then make sure the word **release** is highlighted in blue as seen in the image below



-Then open the build menu and select the "Run qmake" option", wait for the completion of the command before going to the next step.

-Next click the arrow below the computer icon which should now be green. WARNING: This will take about 10 minutes to compile and take up some more space than before. You will only need to rebuild if you change program files. The .exe file takes about 10 seconds to start otherwise

-After the build finishes the executable and all needed libraries will be installed. To locate open the directory you cloned the GitHub repository into and there will be a new folder named something similar to the image below. The folder is highlighted in blue.



-Make sure the end of the folder's name is **-Release** otherwise you will need to redo the steps from selecting the computer icon

-Next navigate into the folder and then enter the folder named "release", scroll to the bottom and you will see a .exe file titled "QGroundControl". To create a shortcut on your desktop right-click the file and then click the "create shortcut" button. A shortcut will be created in the folder, move it to your desktop and you will be good to go.

-Congratulations you have created a .exe file of the project and are able to run it as needed.

### 3. Configuration and Daily Operation

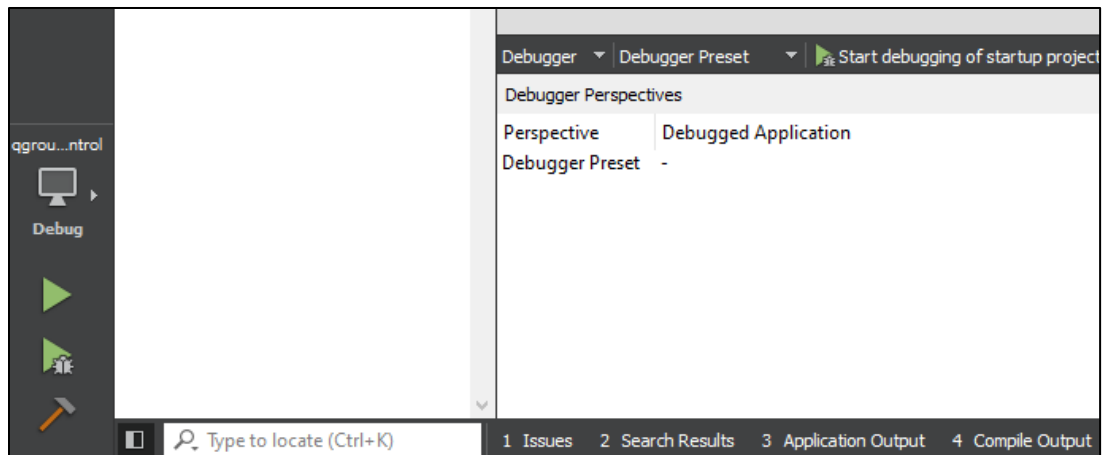
This section is dedicated to the configuration and daily operation of the system. A couple notes about the system. With the current implementation, many specific values such as the File Transfer Protocol IP and UDP IP are hard coded into the program.

The FTP IP is set to autofill to the IP at the time of this document (10.42.0.1:9090). If this is ever needed to be changed it can be written into the IP field within the user interface located at the top right, or the hardcoded value can be changed within the test.qml file.

For the UDP IP, it is written within the myudp.cpp file and cannot be changed from within the user interface. It is a global variable that can easily be changed at the top of the .cpp file without any problems.

#### To run the program for daily operation

Assuming you have completed the initial build as dictated in the installation section you should be able to use the "Build Project qgroundcontrol" & "Run" commands in the build menu. The build should take approximately 15-30 seconds and the run command should take approximately 15-25 seconds as well to launch the UI. After both commands have been ran you are good to go. For extra verbosity build using the debug mode in the bottom left this is displayed as a green arrow with a grey bug icon in front of it. The image below should illustrate the location.



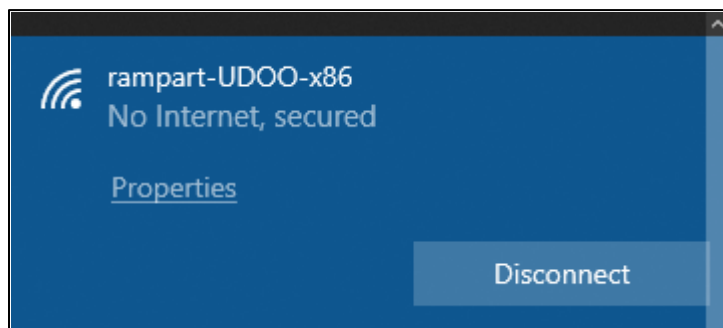
Our user interface is located in the top menu denoted by this icon as seen in the image below.



**For the next set of use cases we assume you have the project open and running.**

To Connect to the FTP Server

-First make sure you are on the companion computer's network which will look something like the image below in your network connections menu.



-After in our user interface which is depicted on the next page make sure that your IP and the IP in the box where 10.42.0.1, match. Then enter your username and password for the FTP server (it should be the login to your Udo). After, you should be able to press the "connect" button and be connected to the companion computer.

The image shows a web-based FTP client interface. At the top, it is titled "FTP". Below the title, there are three input fields: the first contains the IP address "10.42.0.1", the second is labeled "Username", and the third is labeled "Password". Below these fields are five buttons: "Connect" and "Disconnect" are in the first row, and "Download", "Upload", and "Clear Terminal" are in the second row. The "Clear Terminal" button is positioned to the right of the "Upload" button. Below the buttons is a large, empty rectangular area, likely a terminal or file list display.

To Disconnect from the FTP Server

Instead of clicking the “connect” button click the disconnect button as illustrated in the image above.

To Upload or Create a Configuration File

Press the Upload button and navigate to the desired file on your local computer. Then when you choose to upload it, a prompt will ask if you are sure you want to overwrite your file, hit yes. This is a glitch where the computer thinks it’s saving a file to your desired files current location, but in reality, it is being sent to the FTP server.

### Flight Settings

Tag Frequency (MHz)	<input type="text" value="Text Field"/>
RF Gain (dB)	<input type="text" value="Text Field"/>
IF Gain (dB)	<input type="text" value="Text Field"/>
BB Gain (dB)	<input type="text" value="Text Field"/>
Radio Sampling Rate (HZ)	<input type="text" value="Text Field"/>
Pulse Duration (s)	<input type="text" value="Text Field"/>
Pulse Repetition Rate (s)	<input type="text" value="Text Field"/>
Uav Telem. Sample Rate (Hz)	<input type="text" value="Text Field"/>

### Flight Notes

To upload a config file, click the import settings button and select the desired file.

Or to create a new file, fill in the fields all of which are required except flight notes which is optional. Then press the “Save Settings” button if you wish to use these settings again.

To send the configuration settings, do not use the Send Settings button, which is a stub button currently not in use. Instead, use the Upload button on the FTP control panel to upload the desired file.

To Send Start and Stop Commands to the Software Defined Radio (SDR)

To send a start command use the “start radio” button and to send a stop command “stop radio” button as shown in the image below.

### UDP

Last Heartbeat

## 4. Maintenance

Over time the QGroundControl main branch will continue to be updated, however own forked version of this software will not get any of these changes. If QGroundControl has an update that you would like to have included within our version, there are a few steps needed so that you can merge the new version with the main software. However, due to the unknown nature of these changes we cannot be certain that merging both branches will not cause any problems with our software.

1. Using the command line, navigate to the main folder containing the modified QGroundControl
2. Checkout the branch you would like to merge with via:
  - a. `git checkout master`
    - i. "master" refers to the branch you are currently working on and wish to merge to, if you are not working on the master branch than this word must be replaced with the branch you are currently working on.
3. Pull the desired branch from the upstream repository
  - a. `git pull https://github.com/mavlink/qgroundcontrol.git BRANCH_NAME`
    - i. BRANCH\_NAME refers to the branch you would like to take to merge with your changes.
    - ii. The .git link is based on the current link to the QGroundControl repository and this may change in the future.
4. If there are any merge problems they must be fixed here, GitHub has well written documentation on how to solve this that can be located at:  
<https://help.github.com/en/github/collaborating-with-issues-and-pull-requests/addressing-merge-conflicts>


There is no saving of debug logs, so there is no maintenance needed to ensure a debug output file does not get too large.

The current code can be found at: [https://github.com/dynamic-and-active-systems-lab/UAV-RT\\_qgroundcontrol](https://github.com/dynamic-and-active-systems-lab/UAV-RT_qgroundcontrol)

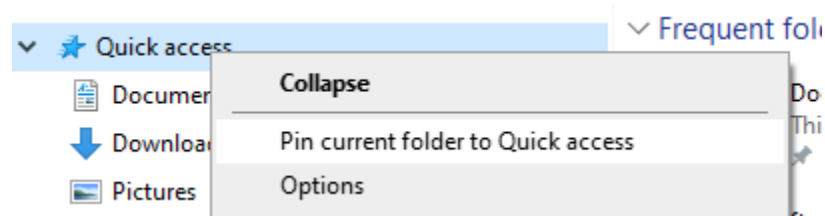
## 5. Troubleshooting

If there is a problem with QGroundControl itself, there are multiple forums that can help with the issue. Some of these websites include: <https://px4.slack.com/#/> and <https://discuss.px4.io/>.

- The 17 warnings given at compilation are inherently included with QGroundControl. This is because they use specific variable types in their custom UIs, but it doesn't seem to have any adverse effect on the program.
- When trying to download a file from the FTP and your local files are shown in the browser instead, follow this:
  - Type the following into a File Explorer address bar:

 `ftp://username:password@10.42.0.1/`

- Then right click the Quick access (located on the left of the File Explorer), and choose “Pin current folder to Quick access”. This way, you can select the FTP in the file browser in QGroundControl whenever you want.



- Windows does not inherently recognize an ftp URL as a possible scheme to be used for a `WindowsFileDialog`.

```
void __cdecl FtpDialog::my_stateChanged(int) "FTP STATUS changed: 'QFtp::LoggingIn'"
"myUrl OUTPUT" QUrl("ftp://rampart@10.42.0.1")
QWindowsNativeFileDialogBase::shellItem : Unhandled scheme: "ftp"
```

- Leave in code, there is a bug report for it so support for it may come soon.
  - Be sure to log into the FTP through the file dialog by typing  
“ftp://username:password@<ip address>”
- Project ERROR: MAVLink folder does not exist at 'libs/mavlink/include/mavlink/v2.0!' Run 'git submodule init && git submodule update' on the command line.
  - Run “Git submodule init && git submodule update”
  - Clean and rebuild project.
- qrc:/qml/test.qml:14:1: module “com.myself” is not installed

```
QML debugging is enabled. Only use this in a safe environment.
Settings location "C:/Users/Keller/AppData/Roaming/QGroundControl.org/QGroundControl.ini" Is writable?: true
Filter rules "*Log.debug=false\n"
serialnmea: No serial ports found
QGCPositionManager error 0
Location access failed.
MAVLinkLogManagerLog: MAVLink logs directory: "C:/Users/Keller/Documents/QGroundControl/Logs"
System reported locale: QLocale(English, Default, United States) "en_US"
Loading localization for "en_US"
Map Cache in: "C:/Users/Keller/AppData/Local/cache/QGCMapCache300" / "qgcMapCache.db"
qrc:/qml/test.qml:14:1: module "com.myself" is not installed
```

- Be sure that you have added the custom module as a  
`qmlRegisterType<packageName>("module_name", 1, 0, "packageName")` in the `QGCAApplication.cc` file. This is located around line 500.
  - This pertains to any custom module.
- If Application Output says, “No Internet Access”, that is because the local wifi network you connected to is not connected to the Internet.

```
No Internet Access
No Internet Access
No Internet Access
No Internet Access
No Internet Access
No Internet Access
No Internet Access
No Internet Access
```



- Ignore, this does not affect you during use of the system.

## 6. Code Layout

To better understand our code structure, we will first discuss which files we have uniquely added and the files which we have modified.

### Files we have added

1. qftp( .cpp, &.h )
2. myudp( .cpp & .h )
3. FtpDialog( .cpp & .h )

### Files which we have modified

1. QGCAApplication.cc
2. qgroundcontrol.pro
3. test.qml
4. MainToolbar.qml
5. MainRootWindow.qml

To distinguish our code in the changed files we have put comments above the changed code that look like this:

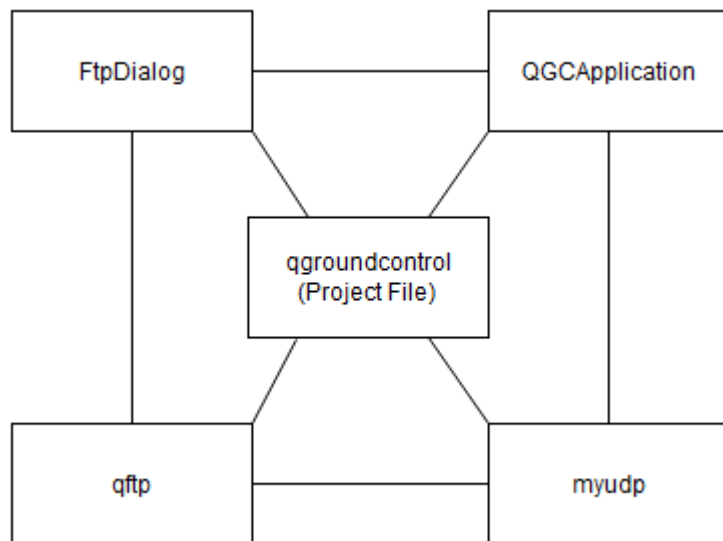
Above: -----UAVRT EDIT-----

With 8 hyphens on each side, to easily locate our changes we recommend using ctrl + f and typing the comment header as seen above.

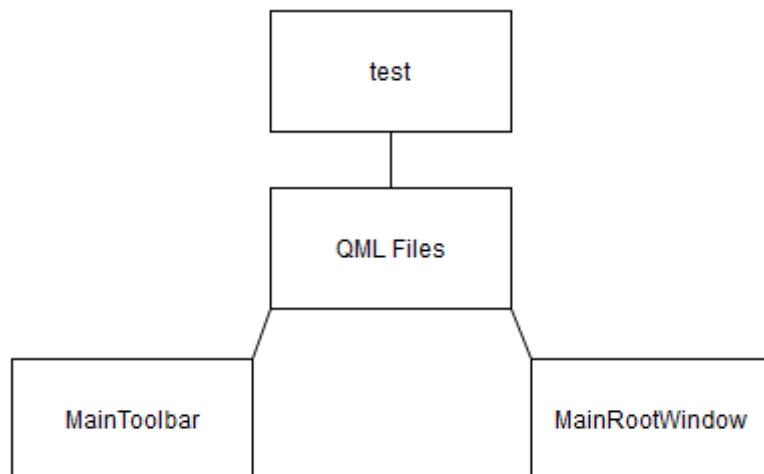
### System Diagrams

For the sake of simplicity, we will diagram our system using the file names but will not reference the file type. To make it clear however .cpp & .cc files contain C++ code, .h files are headers for .cpp & .cc files, .qml files are code for appearance written in the language QML, and the .pro file is a Qt project file. We will also only show the associations that are related to the files that we have changed or made. Finally, we will present this data in two diagrams, first the code files diagram which includes files that end with .cpp, .cc, .h, and .pro. the second diagram is for the visual layout and contains files ending in .qml

Code Files Diagram



Visual Layout Diagram



#### Interpreting the diagrams

Where a line is drawn the file is associated in some way to the end point's file. Normally this is through a header's inclusion in that file and some sort of usage of the functions. For the QML files they aren't associated to each other, so we added a blanket category to provide structure to the diagram.

## 7. Future Development

If we were able to continue development of this product, we would next focus on these important features

**Cleaning up the UI and changing the colors to better match the theme.** In order to make a more seamless integration into QGroundControl(QGC) we would need to spend some time color matching and UI work to make our tab seem like it is “supposed” to exist within QGC regularly. Certainly not required for the product but a nice touch, nonetheless.

**Adding a unique logo for our tab such as the DASL logo.** Being able to quickly ascertain which tab represents the special functions we have implemented is a very important part of user experience for us. We were focused on developing a robust backend in this phase, but UI development would be a part of the next development cycle for sure.

**Changing the UDP section to instead utilize 915MHz custom mavlink messages** to further expand the range that messages from the drone can be received and to be able to use start/stop commands in a larger radius.

**Automatically downloading data upon flight completion.** This streamlines our clients process even further and allows for quick deployments and clean-ups.

**Developing live data visualizations on the flight map based on UDP messages received.** We think this would provide for a useful user interface as well as elevate the usefulness of the overall product by allowing for better data visualization. This would be a development cycle all its own as there are a lot of moving parts and would provide a challenge to many teams.

## 8. Conclusion

From all the members of Team RAMPART, we would like to express our utmost thanks to our client, Dr. Michael Shafer as well as NAU’s cutting-edge capstone team for giving us the opportunity to work on this project. Contributing to the development of this UAV-RT system has given each member of this team invaluable experience in modifying and understanding existing programs as well as experience working with various wireless file transfer protocols. While all our team members will be moving on to professional careers, we would be happy to answer any short questions in the coming months to help get our software deployed and operating optimally in your organization. Simply email any or all the developers listed below if you need any help. We hope that everyone who uses this software will find it useful and will continue to enjoy its benefits for years to come.

With best wishes from your developers:

Keller Mikkelson -- [kjm496@nau.edu](mailto:kjm496@nau.edu) (Personal: [kjmikkelson05@gmail.com](mailto:kjmikkelson05@gmail.com))

Samuel Gilb -- [smg486@nau.edu](mailto:smg486@nau.edu)

Eric Gault -- [emg347@nau.edu](mailto:emg347@nau.edu)

Nathaniel Zeleny -- [nlz6@nau.edu](mailto:nlz6@nau.edu)