

# Building in windows

## Bash On Windows 10

A new feature in Windows 10 allows any developer to quickly and easily run an entire linux subsystem in windows and access it via a bash terminal. This gives developers full use of the entire linux OS and all of the great existing linux tools and programs. When Bash for Windows is up and running it feels like you sshed into a full linux box, except the linux distro is actually running alongside windows locally.

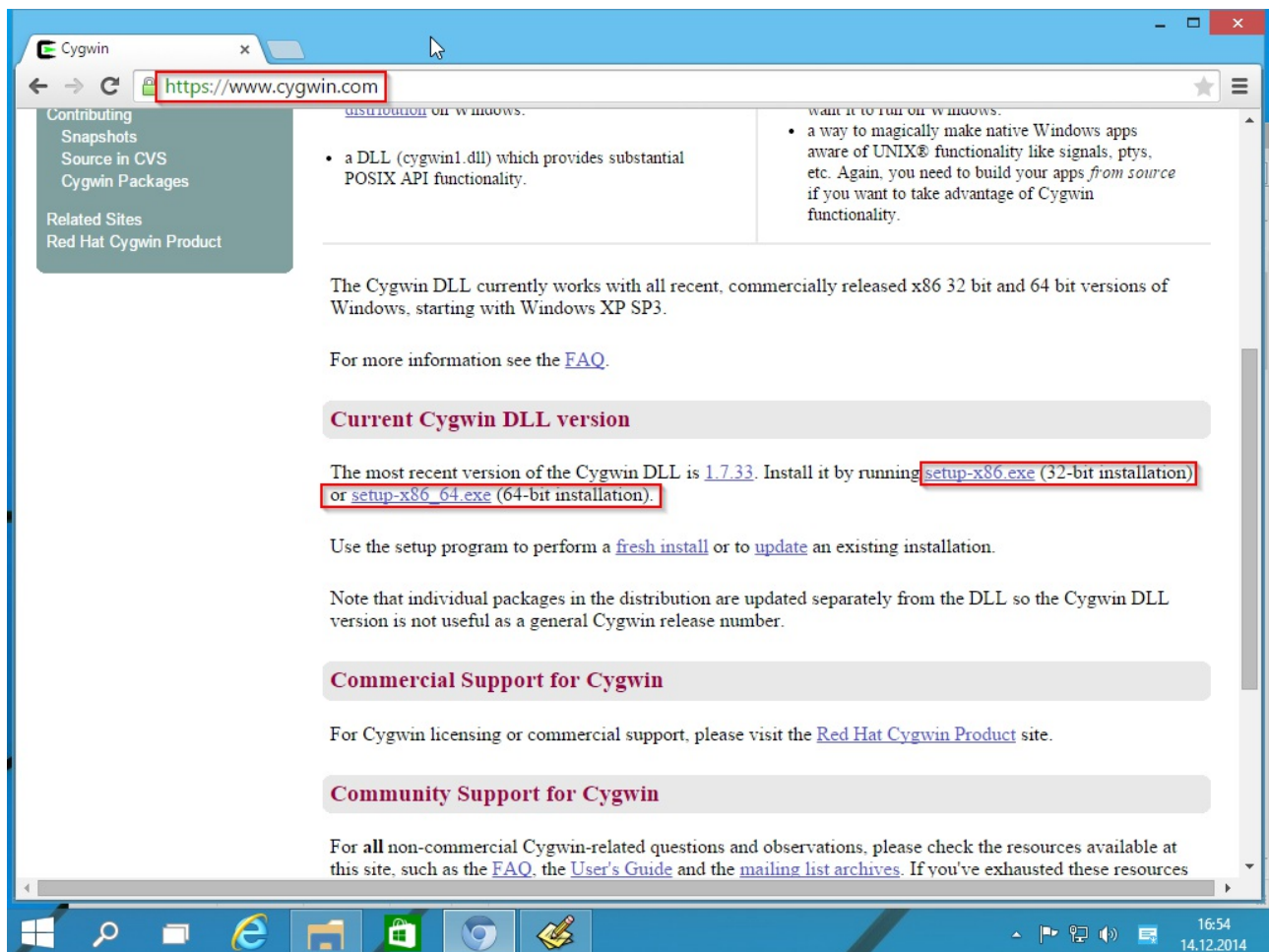
If you use Bash on Windows you can easily build cleanflight exactly as you would for Ubuntu. (the linux distro running on Windows is Ubuntu Trusty)

Setup for Bash on Windows is very easy and takes less than 5 minutes. [For instructions follow the official guide here. \(https://msdn.microsoft.com/commandline/wsl/install\\_guide\)](https://msdn.microsoft.com/commandline/wsl/install_guide)

Once you have Bash On Windows running you can follow the "Building in Ubuntu" instructions for building cleanflight.

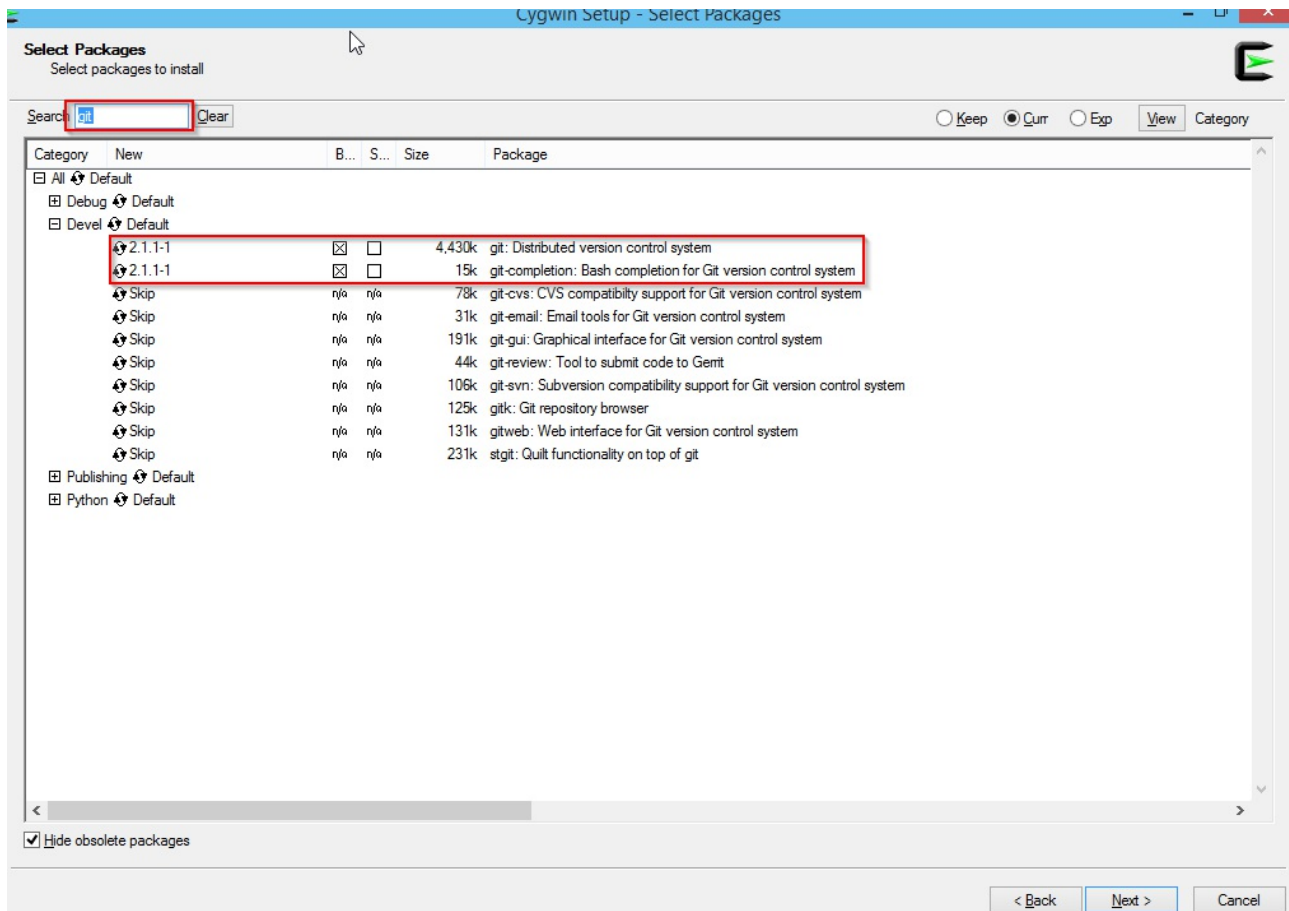
##Setup Cygwin

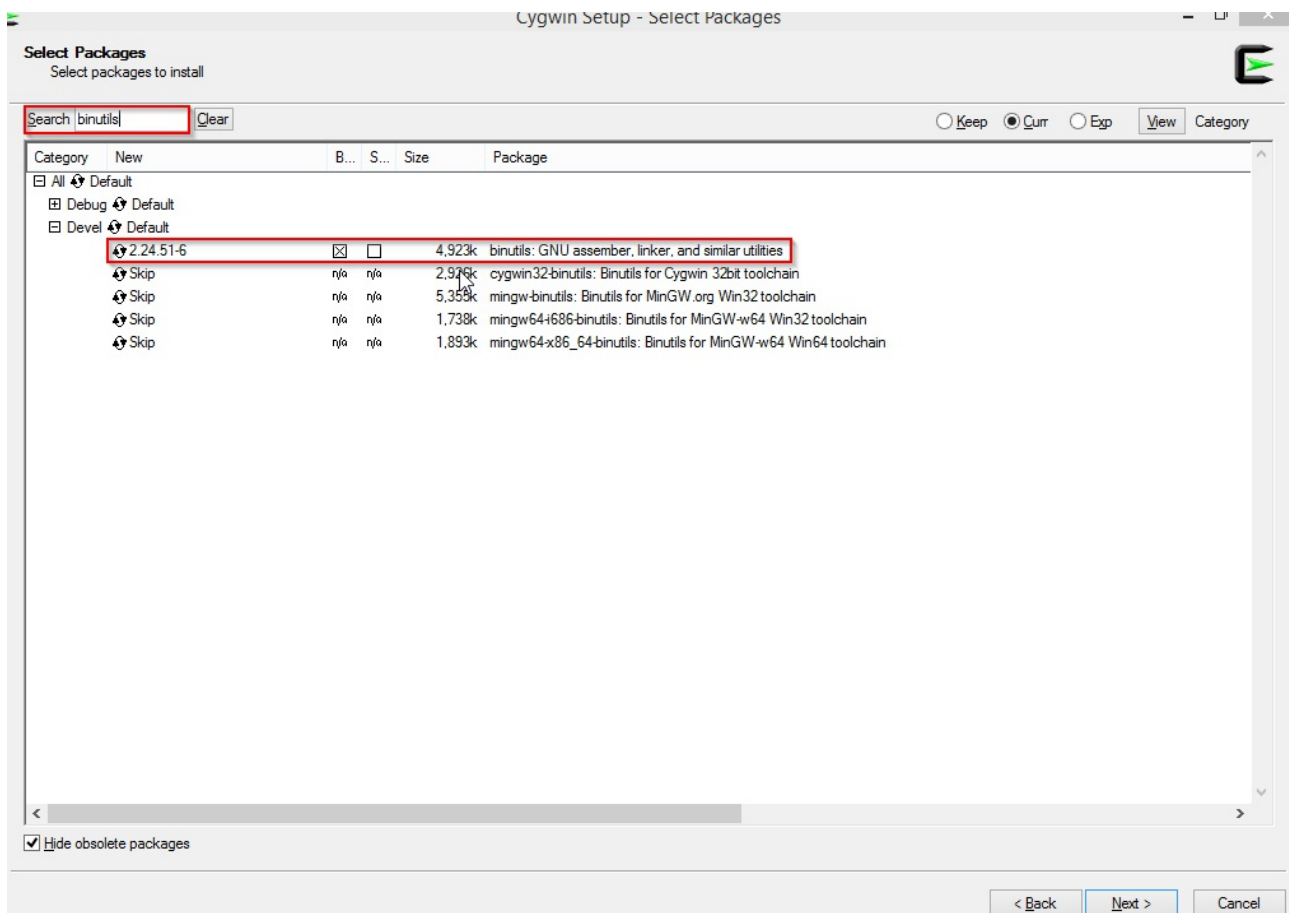
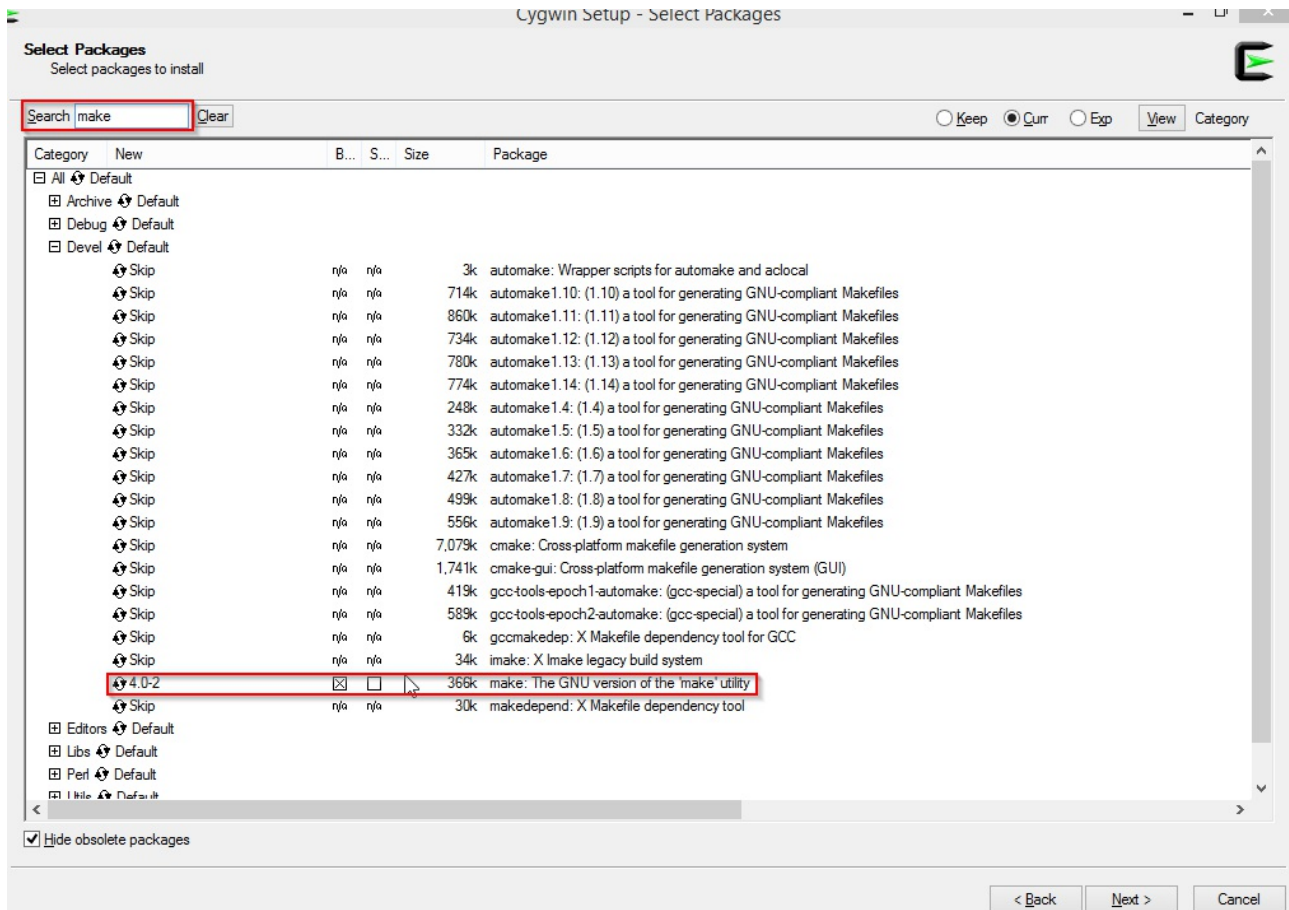
download the Setup\*.exe from <https://www.cygwin.com/>

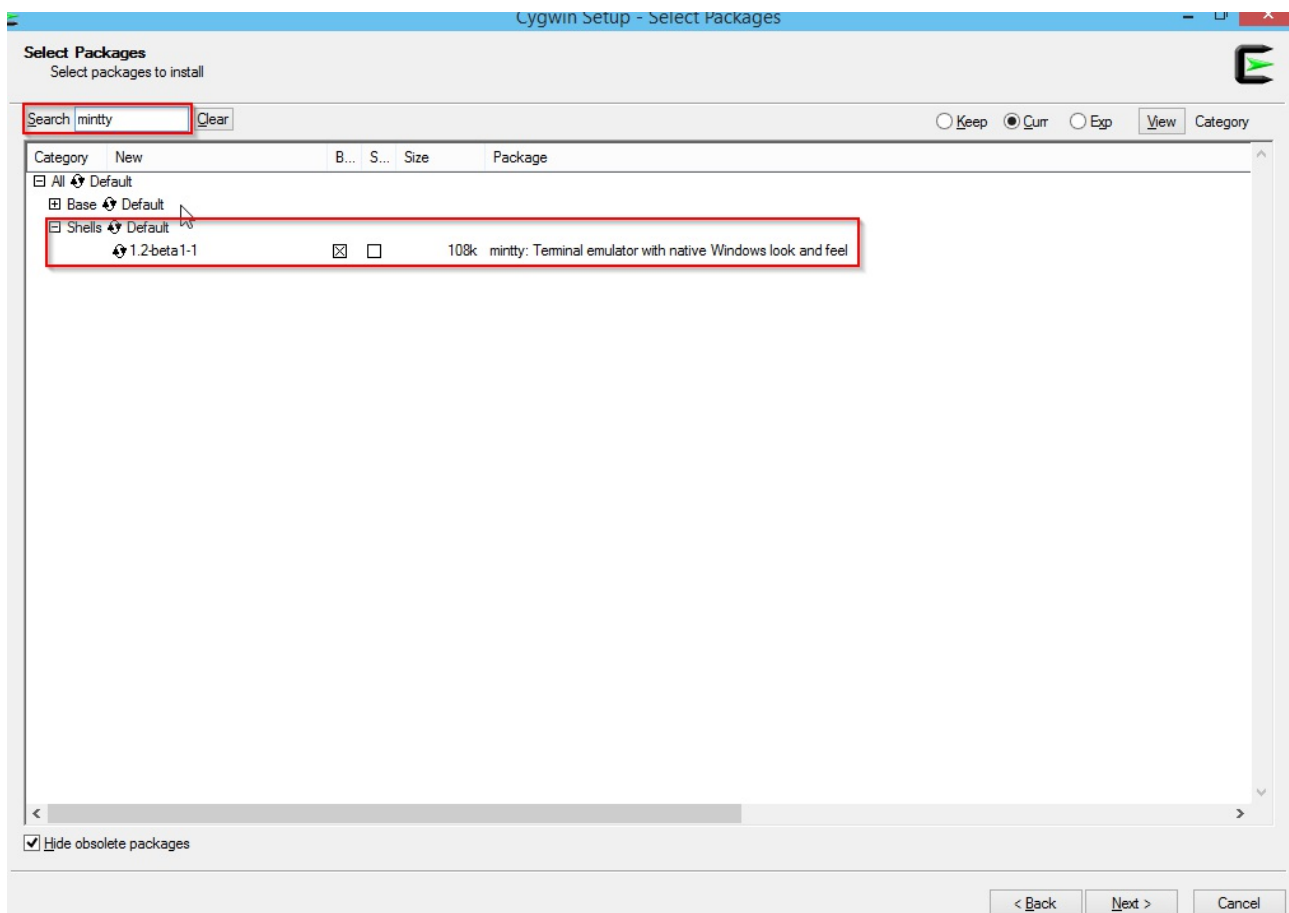
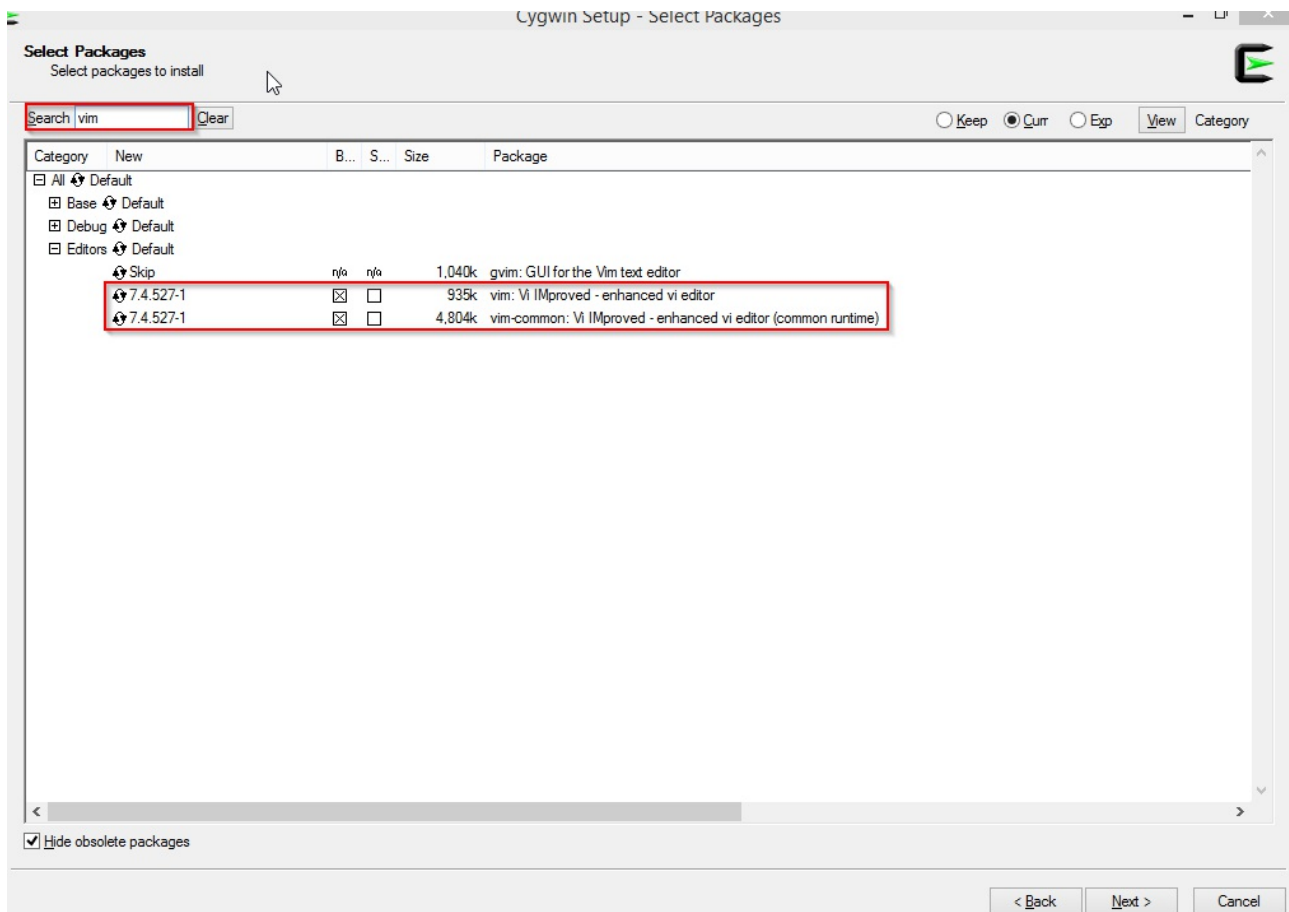


Execute the download Setup and step through the installation wizard (no need to customize the settings here). Stop at the "Select Packages" Screen and select the following Packages for Installation:

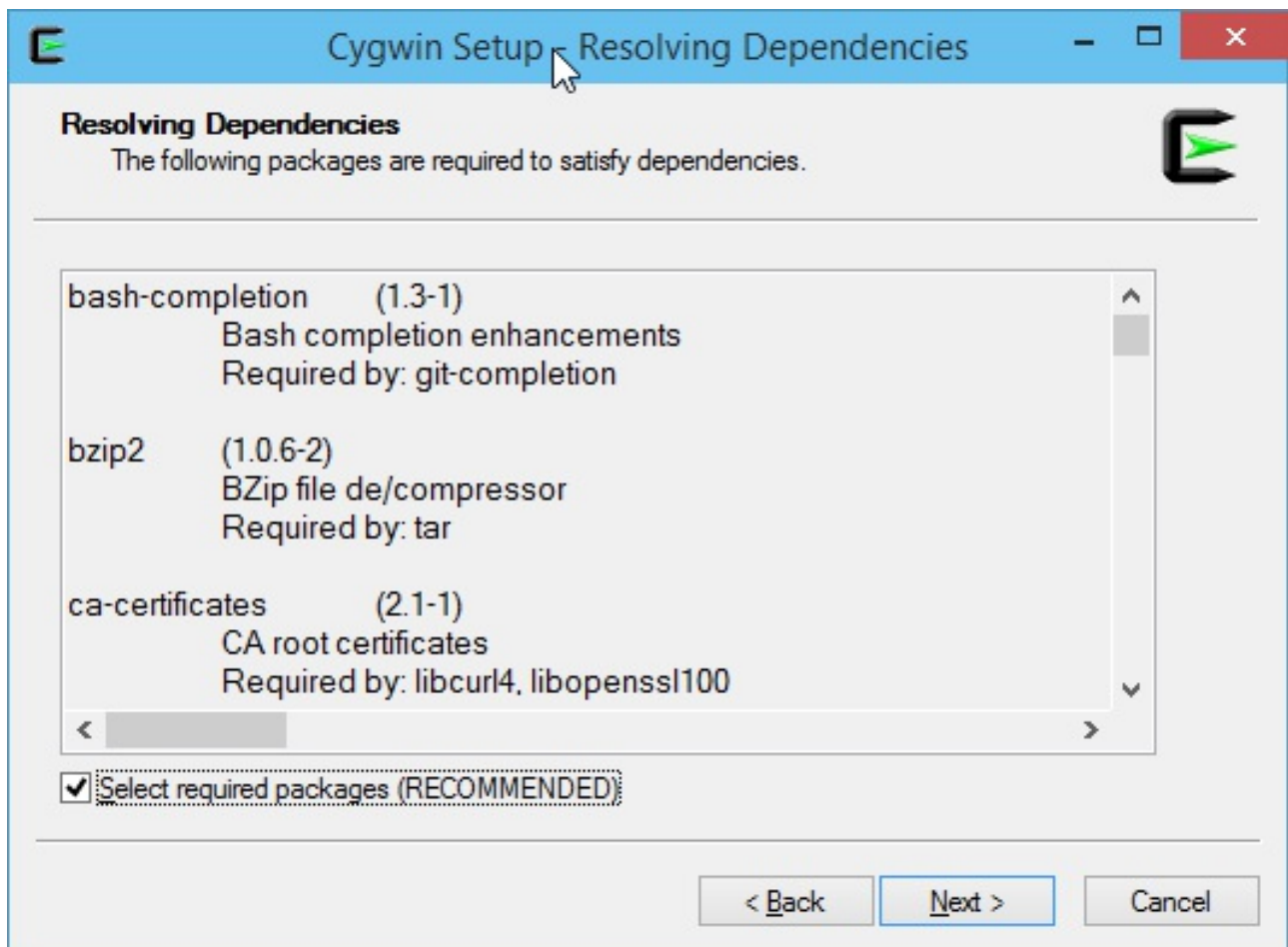
- Devel/git
- Devel/bash-completion (was git-completion, Optional)
- Devel/make
- Devel/binutils
- Editors/vim
- Editors/vim-common (Optional)
- Shells/mintty (should be already selected)







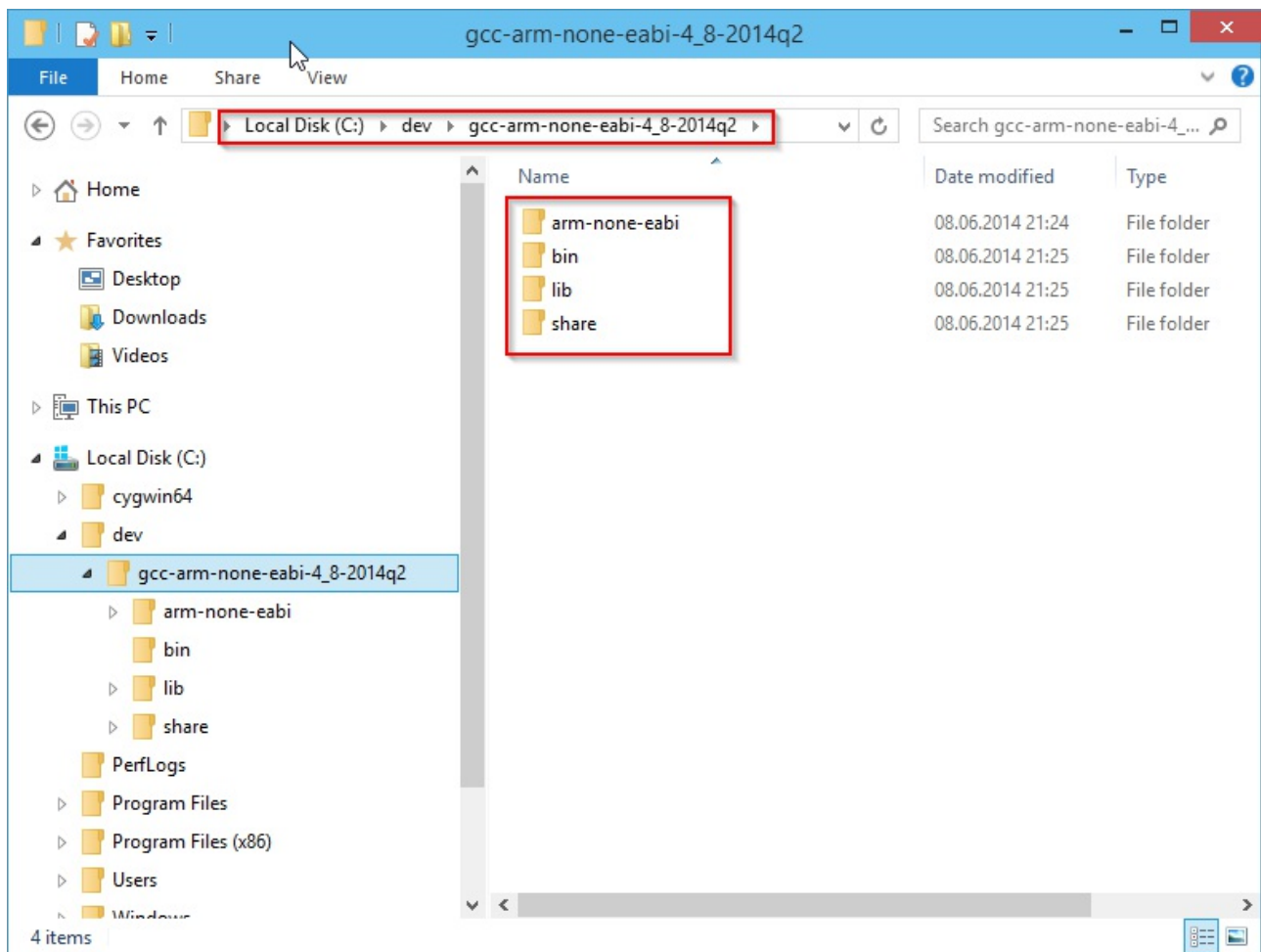
Continue with the Installation and accept all autodetected dependencies.



##Setup GNU ARM Toolchain

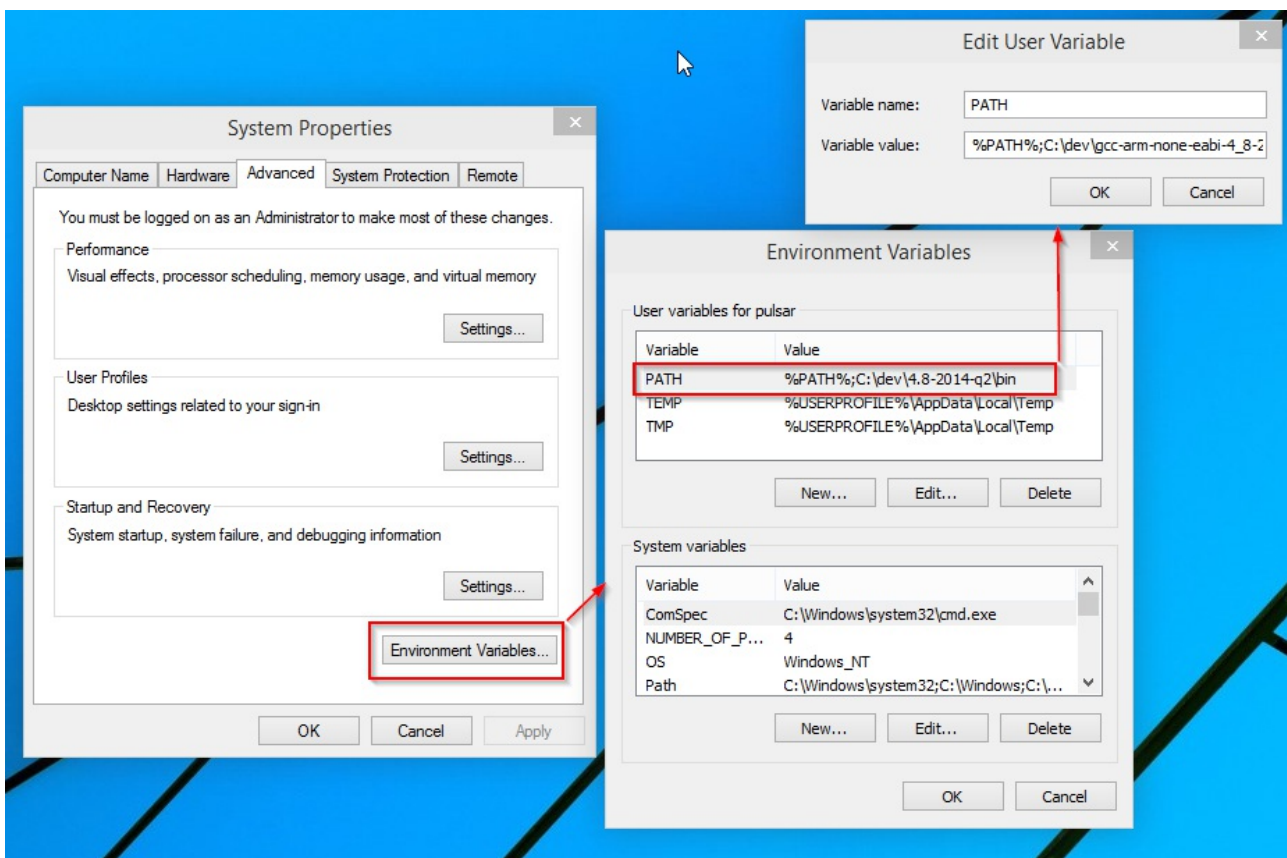
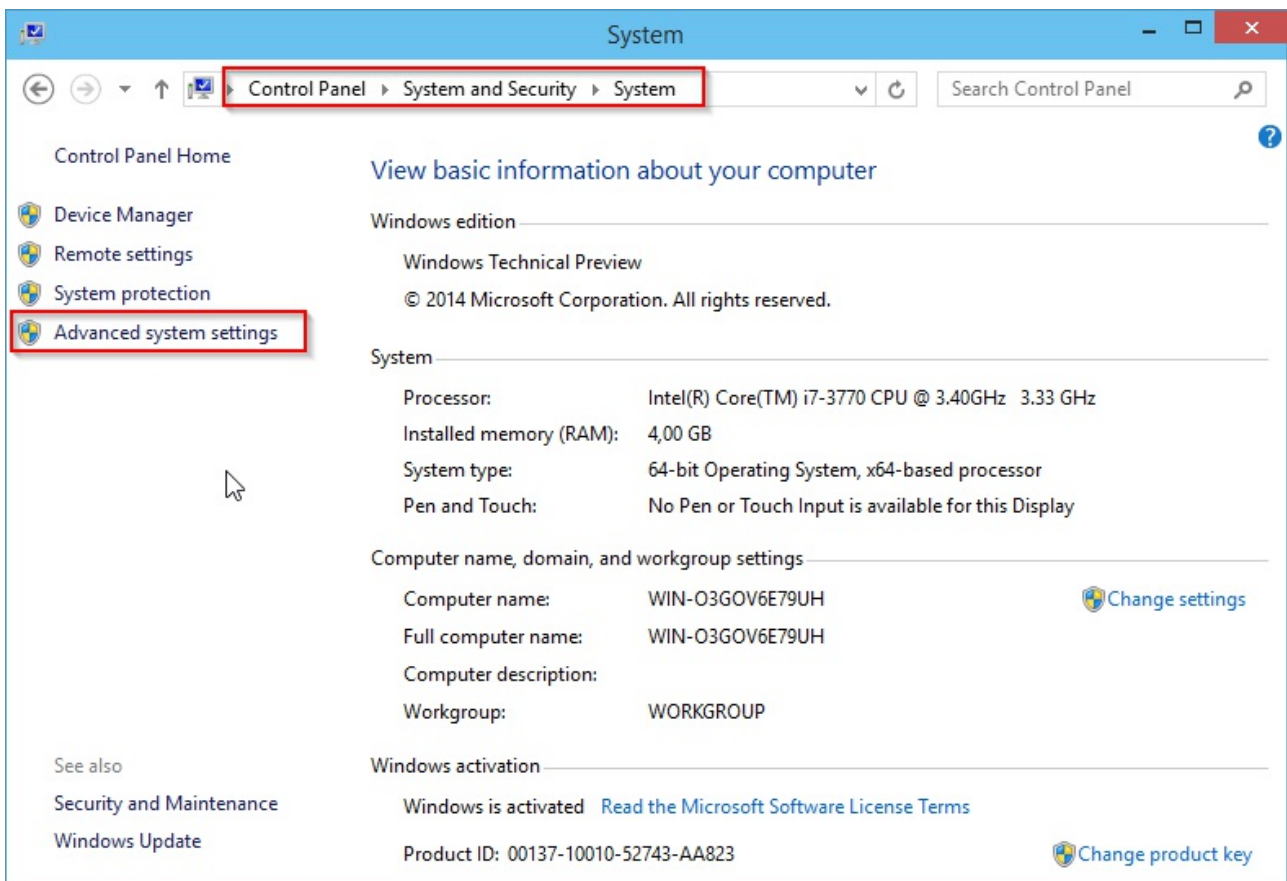
versions do matter, 5.4 is known to work well. Download this version from [https://launchpad.net/gcc-arm-embedded/5.0/5-2016-q2-update/+download/gcc-arm-none-eabi-5\\_4-2016q2-20160622-win32.zip](https://launchpad.net/gcc-arm-embedded/5.0/5-2016-q2-update/+download/gcc-arm-none-eabi-5_4-2016q2-20160622-win32.zip)

Extract the contents of this archive to any folder of your choice, for instance C:\dev\gcc-arm.



add the "bin" subdirectory to the PATH Windows environment variable: %PATH%;C:\dev\gcc-arm\bin



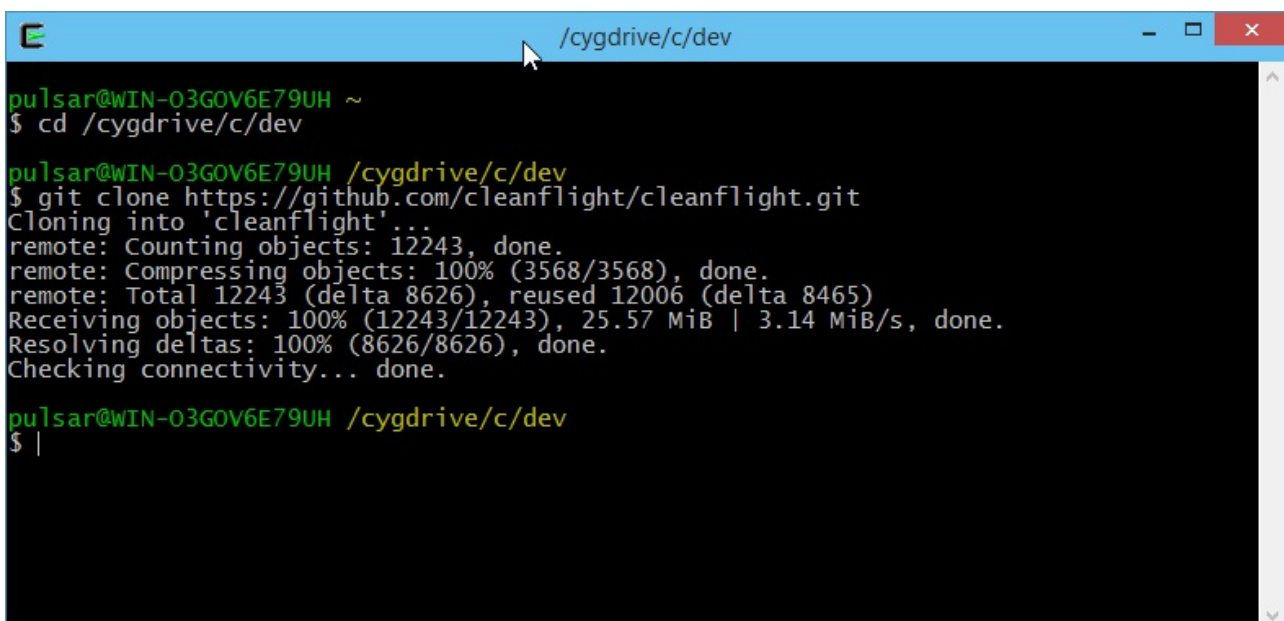
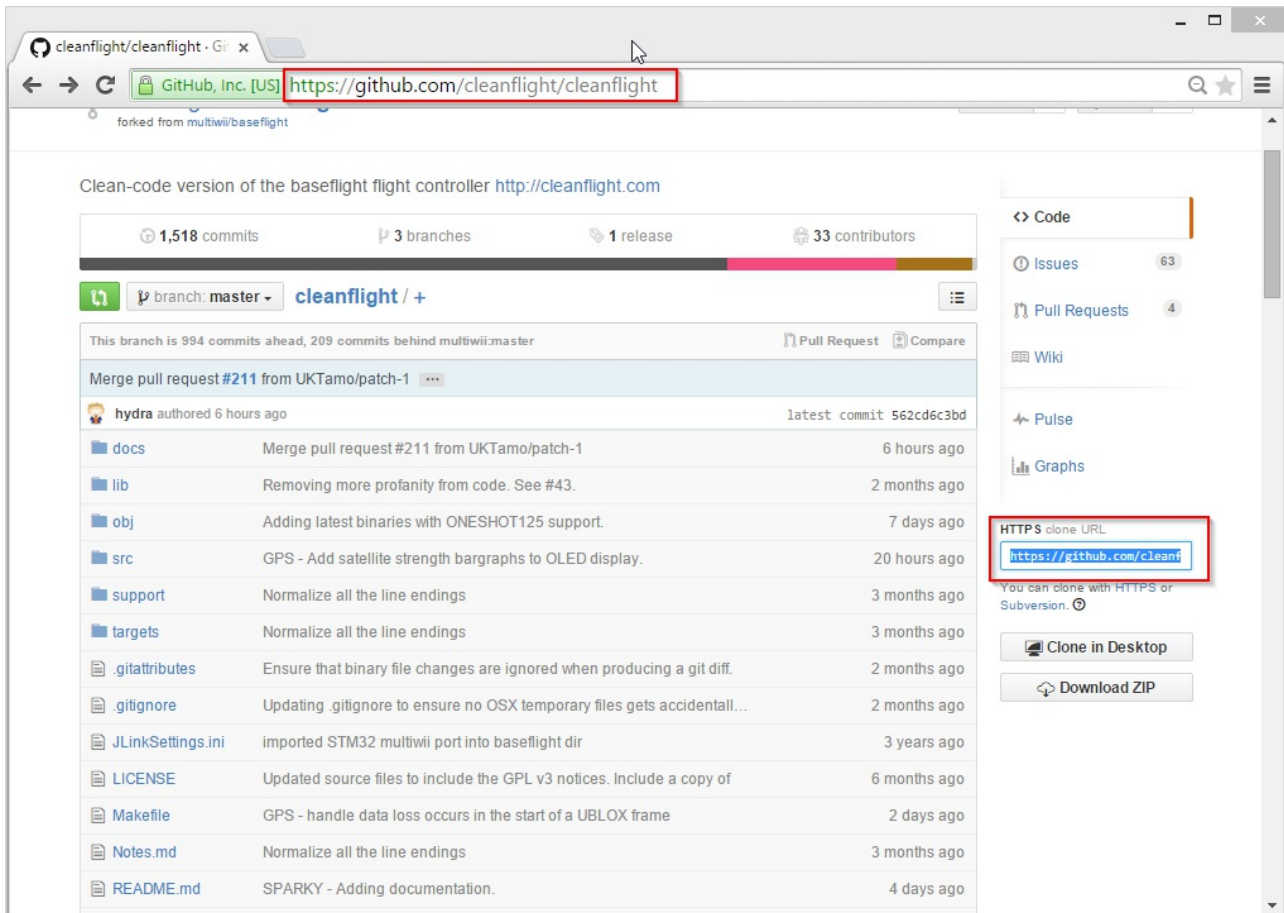


**Checkout and compile Cleanflight**

Head over to the Cleanflight Github page and grab the URL of the GIT Repository:  
"https://github.com/cleanflight/cleanflight.git"

Open the Cygwin-Terminal, navigate to your development folder and use the git commandline to checkout the repository:

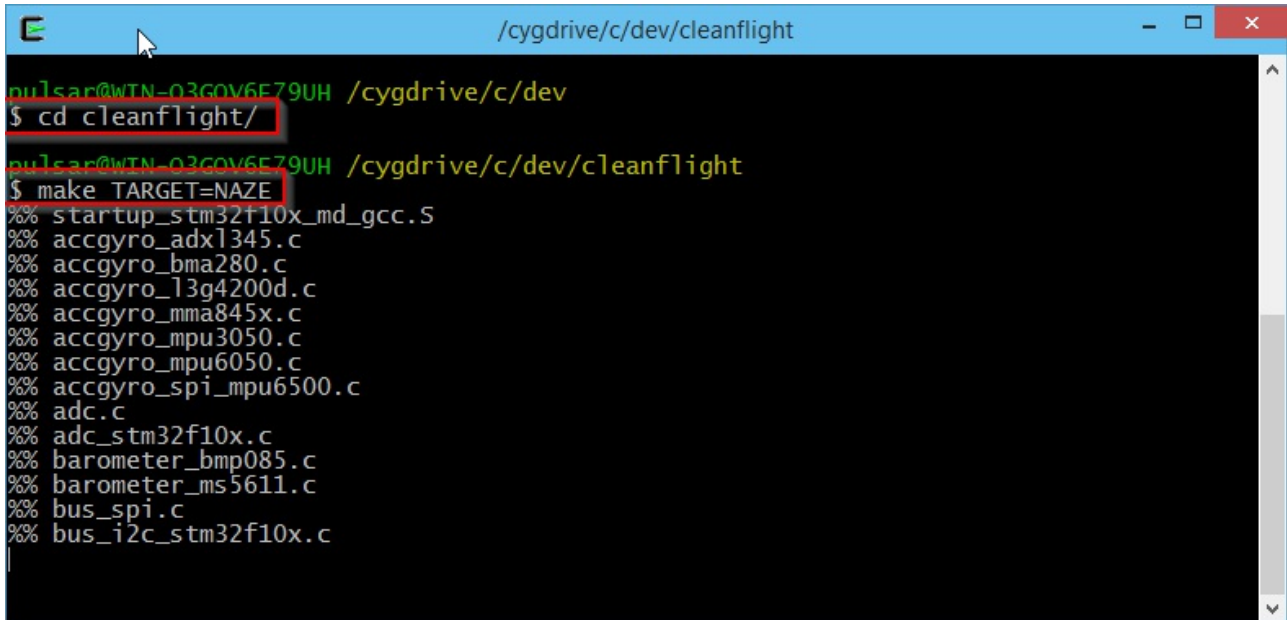
```
cd /cygdrive/c/dev
git clone https://github.com/cleanflight/cleanflight.git
```





To compile your Cleanflight binaries, enter the cleanflight directory and build the project using the make command. You can append TARGET=[HARDWARE] if you want to build anything other than the default NAZE target:

```
cd cleanflight
make TARGET=NAZE
```

A screenshot of a terminal window titled "/cygdrive/c/dev/cleanflight". The prompt is "pulsar@WTN-03GOV6E79UH /cygdrive/c/dev". The user enters "\$ cd cleanflight/" and the prompt changes to "pulsar@WTN-03GOV6E79UH /cygdrive/c/dev/cleanflight". The user then enters "\$ make TARGET=NAZE". The terminal shows a list of files being compiled, each preceded by "%%". The files listed are: startup\_stm32f10x\_md\_gcc.S, accgyro\_adx1345.c, accgyro\_bma280.c, accgyro\_l3g4200d.c, accgyro\_mma845x.c, accgyro\_mpu3050.c, accgyro\_mpu6050.c, accgyro\_spi\_mpu6500.c, adc.c, adc\_stm32f10x.c, barometer\_bmp085.c, barometer\_ms5611.c, bus\_spi.c, and bus\_i2c\_stm32f10x.c.

```
pulsar@WTN-03GOV6E79UH /cygdrive/c/dev
$ cd cleanflight/
pulsar@WTN-03GOV6E79UH /cygdrive/c/dev/cleanflight
$ make TARGET=NAZE
%% startup_stm32f10x_md_gcc.S
%% accgyro_adx1345.c
%% accgyro_bma280.c
%% accgyro_l3g4200d.c
%% accgyro_mma845x.c
%% accgyro_mpu3050.c
%% accgyro_mpu6050.c
%% accgyro_spi_mpu6500.c
%% adc.c
%% adc_stm32f10x.c
%% barometer_bmp085.c
%% barometer_ms5611.c
%% bus_spi.c
%% bus_i2c_stm32f10x.c
```

within few moments you should have your binary ready:

```
(...)
arm-none-eabi-size ./obj/main/cleanflight_NAZE.elf
  text    data    bss     dec     hex filename
  95388    308   10980  106676  1a0b4 ./obj/main/cleanflight_NAZE.elf
arm-none-eabi-objcopy -O ihex --set-start 0x8000000
obj/main/cleanflight_NAZE.elf obj/cleanflight_NAZE.hex
```

You can use the Cleanflight-Configurator to flash the obj/cleanflight\_NAZE.hex file.

## Updating and rebuilding

Navigate to the local cleanflight repository and use the following steps to pull the latest changes and rebuild your version of cleanflight:

```
cd /cygdrive/c/dev/cleanflight
git reset --hard
git pull
make clean TARGET=NAZE -j16 -l
make
```

You may want to remove -j16 -l if your having a hard time narrowing down errors. It does multithreaded make, however it makes it harder to know which warning or error comes from which file.