

How To Build JackTrip from Source on Windows 10

Andrew Forsyth, Bonnie Kwong

20-08-25



Before we start...

This is for Windows 10 users who want to build JackTrip from the source code.

This allows testing out experimental features that might not be ready for widespread release.

This guide assumes you can already run JackTrip by using the installer. If you're looking for the latest JackTrip installers, go to:

<https://ccrma.stanford.edu/software/jacktrip/>

Instead of following these instructions.

Installing Git

Why do I need Git?

Git is a ubiquitous software in software development.


It allows lots of people to contribute the code they write to a project.

It allows people like us to copy the code (clone the repository).

We'll need the code in order to build the app, so we need git!

Download Git for Windows

<https://git-scm.com/downloads>



 **git** --fast-version-control


Search entire site...

About
Documentation
Downloads
GUI Clients
Logos
Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

Downloads

 **Mac OS X**  **Windows**

 **Linux/Unix**

Older releases are available and the Git source repository is on [GitHub](#).

GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

[View GUI Clients →](#)

Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.


[View Logos →](#)

Git via Git

If you already have Git installed, you can get the latest development version via Git itself:

```
git clone https://github.com/git/git
```

You can also always browse the current contents of the git repository using the [web interface](#).



Latest source Release
2.28.0
Release Notes (2020-07-27)
[Download 2.28.0 for Windows](#)

</> About this site
Patches, suggestions, and comments are welcome.

Git is a member of Software Freedom Conservancy



Git 2.28.0 Setup



Information

Please read the following important information before continuing.



When you are ready to continue with Setup, click Next.

GNU General Public License

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
License is intended to guarantee your freedom to share and change

<https://gitforwindows.org/>



Only show new options

Next >

Cancel

I'm sure you'll read all of this



Git 2.28.0 Setup



Select Components

Which components should be installed?



Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.

- ☐ Additional icons
 - ☐ On the Desktop
- ☒ Windows Explorer integration
 - ☒ Git Bash Here
 - ☒ Git GUI Here
- ☒ Git LFS (Large File Support)
- ☒ Associate .git* configuration files with the default text editor
- ☒ Associate .sh files to be run with Bash
- ☐ Use a TrueType font in all console windows
- ☐ Check daily for Git for Windows updates

Current selection requires at least 269.5 MB of disk space.

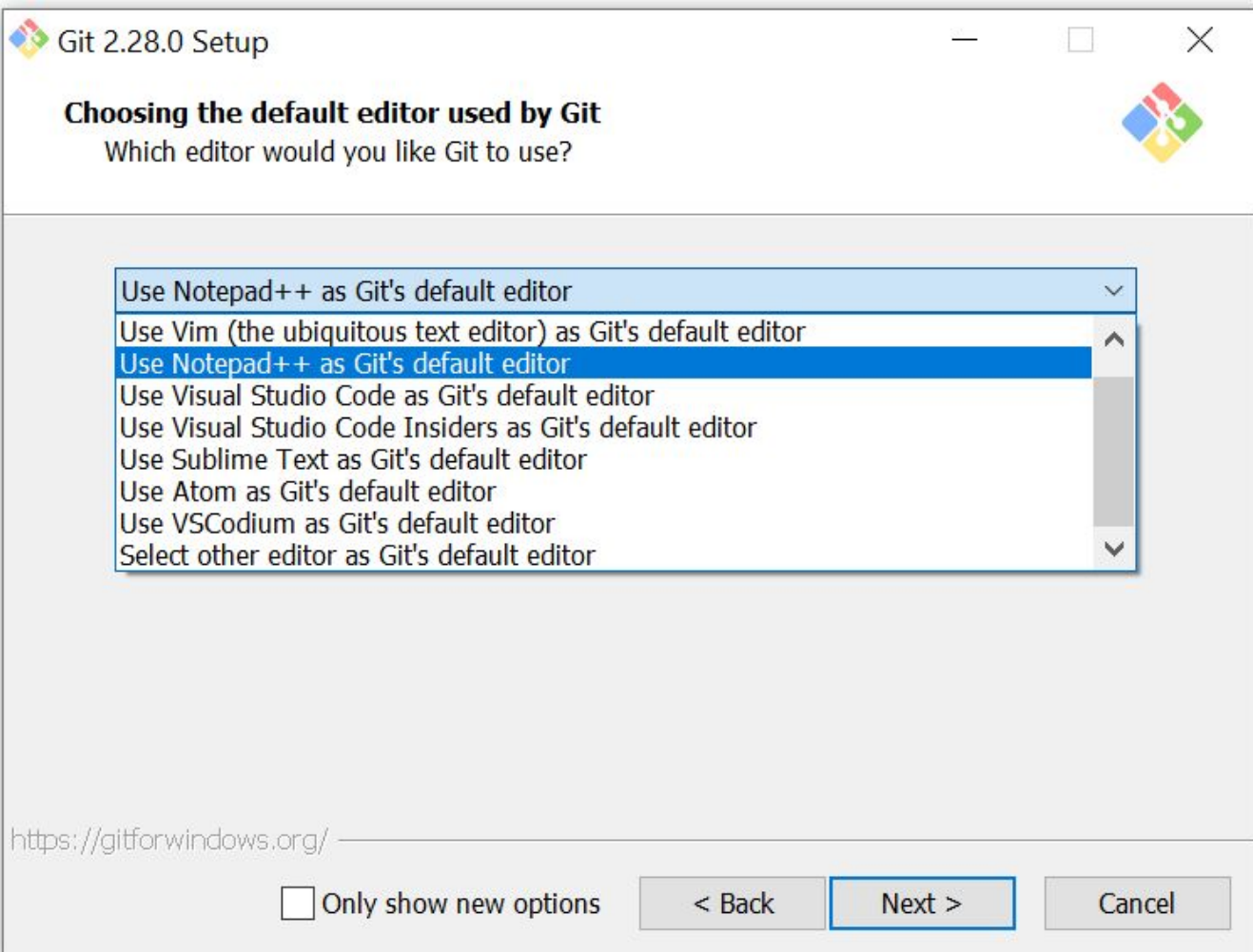
<https://gitforwindows.org/>

☐ Only show new options

< Back

Next >

Cancel



You can use whatever you want for an editor!

We won't do much editing unless we're contributing to the project.



Git 2.28.0 Setup



Adjusting your PATH environment

How would you like to use Git from the command line?



☐ **Use Git from Git Bash only**

This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.

☒ **Git from the command line and also from 3rd-party software**

(Recommended) This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools. You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.

☐ **Use Git and optional Unix tools from the Command Prompt**

Both Git and the optional Unix tools will be added to your PATH.

Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.

<https://gitforwindows.org/>



Only show new options

< Back

Next >

Cancel



Git 2.28.0 Setup



Choosing HTTPS transport backend



Which SSL/TLS library would you like Git to use for HTTPS connections?

☒ **Use the OpenSSL library**

Server certificates will be validated using the ca-bundle.crt file.

☐ **Use the native Windows Secure Channel library**

Server certificates will be validated using Windows Certificate Stores.
This option also allows you to use your company's internal Root CA certificates distributed e.g. via Active Directory Domain Services.

<https://gitforwindows.org/>



Only show new options

< Back

Next >

Cancel



Git 2.28.0 Setup



Configuring the line ending conversions

How should Git treat line endings in text files?



☒ **Checkout Windows-style, commit Unix-style line endings**

Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").

☐ **Checkout as-is, commit Unix-style line endings**

Git will not perform any conversion when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Unix ("core.autocrlf" is set to "input").

☐ **Checkout as-is, commit as-is**

Git will not perform any conversions when checking out or committing text files. Choosing this option is not recommended for cross-platform projects ("core.autocrlf" is set to "false").

<https://gitforwindows.org/>



Only show new options

< Back

Next >

Cancel

Git 2.28.0 Setup

Configuring the terminal emulator to use with Git Bash

Which terminal emulator do you want to use with your Git Bash?

☒ **Use MinTTY (the default terminal of MSYS2)**

Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `wintpy` to work in MinTTY.

☐ **Use Windows' default console window**

Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.

<https://gitforwindows.org/>

☐ Only show new options

< Back Next > Cancel

MinTTY is basically a wrapper for the command prompt (normal windows terminal)

It lets you use unix style commands and has other benefits. A lot of people are more familiar with unix style commands, because that's what Mac and Linux use.



Git 2.28.0 Setup



Choose the default behavior of `git pull`

What should `git pull` do by default?



☒ **Default (fast-forward or merge)**

This is the standard behavior of `git pull`: fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.

☐ **Rebase**

Rebase the current branch onto the fetched branch. If there are no local commits to rebase, this is equivalent to a fast-forward.

☐ **Only ever fast-forward**

Fast-forward to the fetched branch. Fail if that is not possible.

<https://gitforwindows.org/>

☐ Only show new options

< Back

Next >

Cancel



Git 2.28.0 Setup



Choose a credential helper

Which credential helper should be configured?



☐ **None**

Do not use a credential helper.

☒ **Git Credential Manager**

The [Git Credential Manager for Windows](#) handles credentials e.g. for Azure DevOps and GitHub (requires .NET framework v4.5.1 or later).

☐ **Git Credential Manager Core**

(NEW!) Use the new, [cross-platform version of the Git Credential Manager](#). See more information about the future of Git Credential Manager [here](#).

<https://gitforwindows.org/>

☐ Only show new options

< Back

Next >

Cancel

This will helpfully save your
credentials
(username and password)



Git 2.28.0 Setup



Configuring extra options



Which features would you like to enable?

☒ **Enable file system caching**

File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.

☐ **Enable symbolic links**

Enable [symbolic links](#) (requires the SeCreateSymbolicLink permission). Please note that existing repositories are unaffected by this setting.

<https://gitforwindows.org/>

☐ Only show new options

< Back

Next >

Cancel



Git 2.28.0 Setup



Configuring experimental options



Which bleeding-edge features would you like to enable?

☐ **Enable experimental support for pseudo consoles.**

(NEW!) This allows running native console programs like Node or Python in a Git Bash window without using winpty, but it still has known bugs.

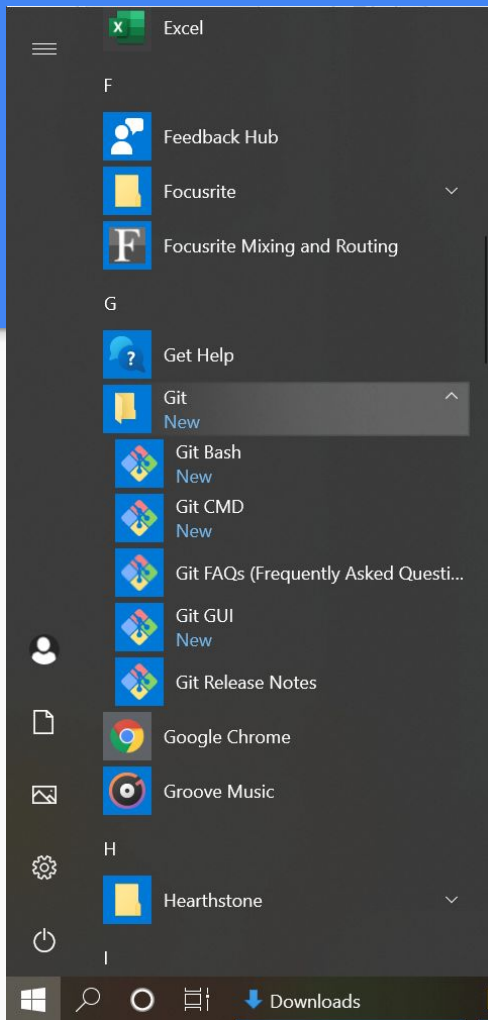
<https://gitforwindows.org/>

☐ Only show new options

< Back

Next >

Cancel



Install the thing, you'll see GIT show up in the start menu.
Running Git Bash opens the MinTTY application

MinTTY is basically a wrapper for the command prompt
(normal windows terminal)

It lets you use unix style commands and has other
benefits. A lot of people are more familiar with unix style
commands, because that's what Mac and Linux use.

Run Git Bash to open the unix style terminal
(You can also use command prompt if you'd like)

That's it!

You installed git!

Installing Qt

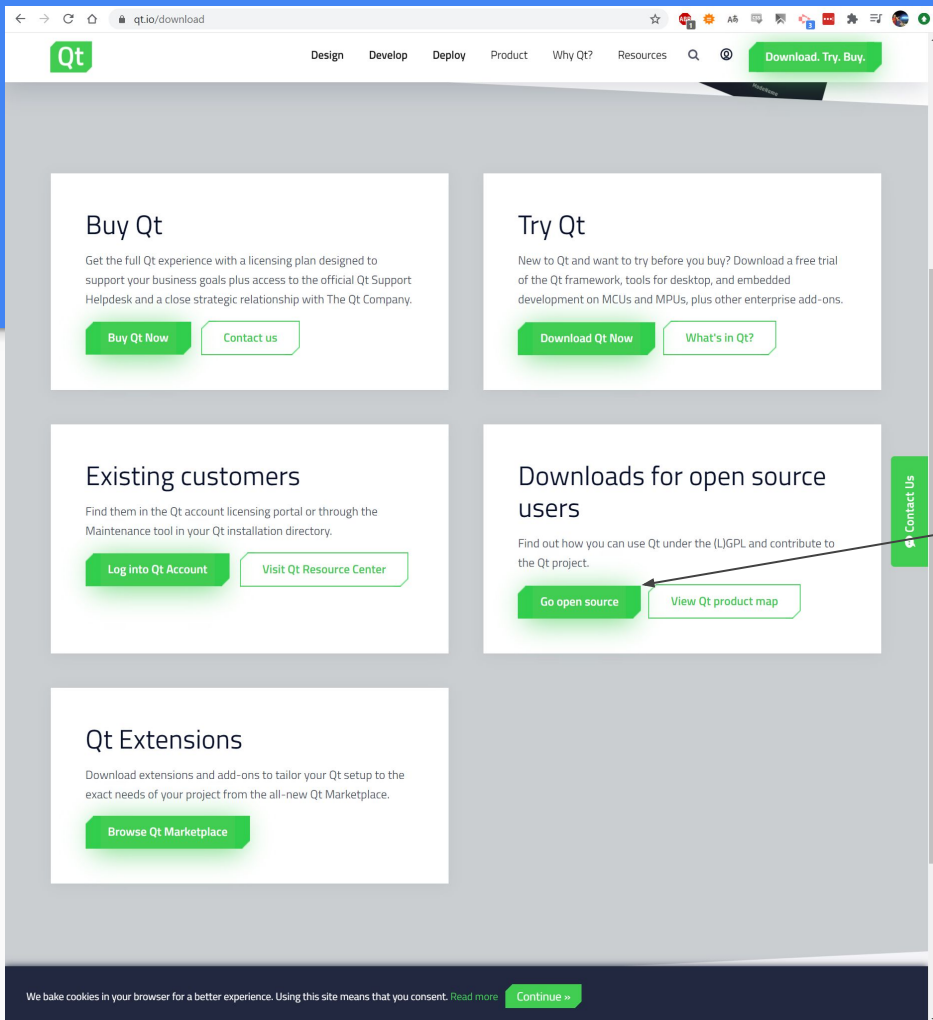
Why do I need Qt?

Qt is a compiler

It is what turns raw code into an app (AKA build from source).

It has a lot of helpful libraries and tools for making applications

Specifically, we need the apps qmake and g++ to build JackTrip



We're all good with this because it's an open source project we're working on. If you use this to develop an app that's NOT open source, you run into legal problems by not paying Qt for it

Due Diligence

Confirm that you can abide by the obligations of Qt open source licensing.

When deciding which license to use, please check your corporate open source policy or consult a software licensing legal expert.

Find more information on Qt
licensing and open source
obligations [here](#).

Source Code

Set up your local development environment, get the Qt source code from the repositories, and build the libraries on your machine.

Grab the code at <https://code.qt.io/> or if you need help, the community not only contributes to Qt, but also to the Qt Wiki where you can learn how to get started.

Start Contributing

Qt source code repositories are open to the public, which means that that you can immediately help guide and shape the future development of Qt by contributing code, translations, examples and more.

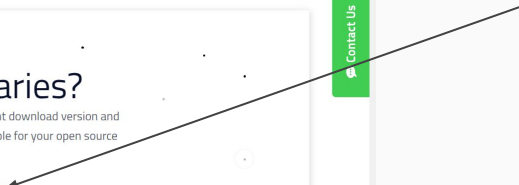
Learn what's involved when contributing to the Qt project [here](#).

Looking for Qt binaries?

Find them in the Qt Online Installer. It will steer you to the right download version and help you install tools and add-on components that are available for your open source license.

[Download the Qt Online Installer](#)

 [Contact Us](#)



Download

- Buy Qt
- Free trial
- Open source
- Product packaging
- Terms & Conditions

Resources

- Customer success stories
- Solutions by industry
- Events
- Educational programs
- Licensing info.
- Get Extensions

Professional Services

- Consulting
- Qt Support
- Qt Training
- Partner Directory

Developers

- Documentation
- Wiki
- Forums
- Contribute to Qt

Company

- Investors
- News
- Careers
- Office Locations

[Feedback](#) [Sign In](#)

We bake cookies in your browser for a better experience. Using this site means that you consent. [Read more](#)

Continue »



Qt Setup

Select Components

Please select the components you want to install.

Select Package Categories

- ☐ Archive
- ☐ LTS
- ☒ Latest releases
- ☐ Preview

Filter

Default

Select All

Deselect All

Qt

- > ☐ Qt 5.15.0
- > ☐ Qt 5.14.2
- > ☐ Qt 5.13.2
- > ☐ Qt 5.12.9
- > ☐ Qt 5.12.5
- > ☐ Qt 5.12.4
- > ☐ Qt 5.12.3
- > ☐ Qt 5.12.2
- > ☐ Qt 5.9.9
- > ☒ Developer and Designer Tools

Check the box for the latest *two* versions of Qt.

At the time of writing this, the latest version was 5.15.0.

This component will occupy approximately 646.90 MB on your hard disk drive.

Next

Cancel



Qt Setup

License Agreement

Please read the following license agreements. You must accept the terms contained in these agreements before continuing with the installation.

Qt Installer LGPL License Agreement

Python Software Foundation License Version 2

MICROSOFT SOFTWARE LICENSE TERMS MICROSOFT WINDOWS SOFTWARE DEVELOPMENT KIT (SDK) FOR WINDOWS 10

GENERAL

Qt is available under a commercial license with various pricing models and packages that meet a variety of needs. Commercial Qt license keeps your code proprietary where only you can control and monetize on your end product's development, user experience and distribution. You also get great perks like additional functionality, productivity enhancing tools, world-class support and a close strategic relationship with The Qt Company to make sure your product and development goals are met.

Qt has been created under the belief of open development and providing freedom and choice to developers. To support that, The Qt Company also licenses Qt under open source licenses, where most of the functionality is available under LGPLv3. It should be noted that the tools as well as some add-on components are available only under GPLv3. In order to preserve the true meaning of open development and uphold the spirit of free software, it is imperative that the rules and regulations of open source licenses are followed. If you use Qt under open-source licenses, you need to make sure that you comply with all the licenses of the components you use.

Qt also contains some 3rd party components that are available under different open-source licenses. Please refer to the documentation for more details on 3rd party licenses used in Qt.

GPLv3 and LGPLv3

GNU LESSER GENERAL PUBLIC LICENSE

The Qt Toolkit is Copyright (C) 2017 The Qt Company Ltd.
Contact: <https://www.qt.io/licensing>

You may use, distribute and copy the Qt GUI Toolkit under the terms of GNU Lesser General Public License version 3, which supplements GNU General Public License Version 3. Both of the licenses are displayed below.

GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <<http://fsf.org/>>
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

- ☒ I have read and agree to the terms contained in the license agreements.
☐ I do not accept the terms and conditions of the above license agreements.

Next

Cancel



Qt Setup



Start Menu shortcuts

Select the Start Menu in which you would like to create the program's shortcuts. You can also enter a name to create a new directory.

Qt

Accessibility
Accessories
Administrative Tools
ASIO4ALL v2
Chrome Apps
Jack
Maintenance
ScopelImage 9.0
Startup
Steam
System Tools
Windows PowerShell
WinRAR
Zoom

Next

Cancel

← → ▾ ↑  > Control Panel > System and Security >



environment



Control Panel Home

System and Security

Network and Internet

Hardware and Sound

Programs

User Accounts

Appearance and
Personalization

Clock and Region

Ease of Access



System

Edit environment variables for your account

 [Edit the system environment variables](#)

 Search Windows Help and Support for "environment"

System Properties



Computer Name Hardware Advanced System Protection Remote

You must be logged on as an Administrator to make most of these changes.

Performance

Visual effects, processor scheduling, memory usage, and virtual memory

Settings...

User Profiles

Desktop settings related to your sign-in

Settings...

Startup and Recovery

System startup, system failure, and debugging information

Settings...

Environment Variables...

OK

Cancel

Apply

Environment Variables

User variables for forsy

Variable	Value
OneDrive	C:\Users\forsy\OneDrive
OneDriveConsumer	C:\Users\forsy\OneDrive
Path	c:\Qt\%VERSION%\bin
TEMP	C:\Users\forsy\AppData\Local\Temp
TMP	C:\Users\forsy\AppData\Local\Temp

New...

Edit...

Delete

System variables

Variable	Value
ComSpec	C:\WINDOWS\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS	6
OS	Windows_NT
Path	C:\Program Files (x86)\Intel\Intel(R) Management Engine Compone...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE	AMD64
PROCESSOR_IDENTIFIER	Intel64 Famliy 6 Model 158 Stepping 10. GenuineIntel

New...

Edit...

Delete

OK

Cancel

Edit environment variable

C:\Program Files (x86)\Intel\Intel(R) Management Engine Compone...
 C:\Program Files\Intel\Intel(R) Management Engine Components\iC...
 C:\Windows\system32
 C:\Windows
 C:\Windows\System32\Wbem
 C:\Windows\System32\WindowsPowerShell\v1.0\
 C:\Windows\System32\OpenSSH\
 C:\Program Files (x86)\Intel\Intel(R) Management Engine Compone...
 C:\Program Files\Intel\Intel(R) Management Engine Components\D...
 C:\Program Files (x86)\Intel\Intel(R) Management Engine Compone...
 C:\Program Files\Intel\Intel(R) Management Engine Components\IPT
 C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common
 %SystemRoot%\system32
 %SystemRoot%
 %SystemRoot%\System32\Wbem
 %SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
 %SYSTEMROOT%\System32\OpenSSH\
 C:\Program Files\Git\cmd
 c:\Qt\%VERSION%\bin

New

Edit

Browse...

Delete

Move Up

Move Down

Edit text...

OK

Cancel

Clone the Repo

Why do I want to clone the repo?

We need to copy the code in order to build it into an app

We can also contribute to the project after we do this

That's another presentation entirely :)

```
MINGW64:/c/Users/forsy/Experimental_JackTrip_Repos
'Saved Games' /
'Searches' /
'SendTo' @
'Start Menu' @
'Templates' @
'Videos' /

forsy@DESKTOP-1V9ENMM MINGW64 ~
$ cd Experimental_JackTrip_Repos/

forsy@DESKTOP-1V9ENMM MINGW64 ~/Experimental_JackTrip_Repos
$ ls

forsy@DESKTOP-1V9ENMM MINGW64 ~/Experimental_JackTrip_Repos
$ git clone https://github.com/antonrunov/jacktrip.git
Cloning into 'jacktrip'...
remote: Enumerating objects: 41, done.
remote: Counting objects: 100% (41/41), done.
remote: Compressing objects: 100% (38/38), done.
remote: Total 4678 (delta 15), reused 4 (delta 3), pack-reused 4637
Receiving objects: 100% (4678/4678), 14.59 MiB | 27.16 MiB/s, done.
Resolving deltas: 100% (3467/3467), done.

forsy@DESKTOP-1V9ENMM MINGW64 ~/Experimental_JackTrip_Repos
$ ls
jacktrip/

forsy@DESKTOP-1V9ENMM MINGW64 ~/Experimental_JackTrip_Repos
$
```

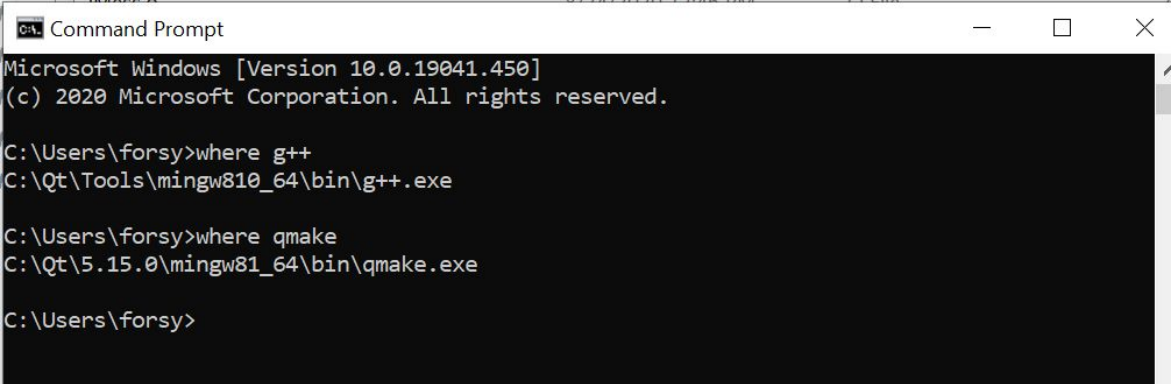
Make a folder where you'd like to clone the repo.

Open Git Bash to open the terminal

Navigate to the folder you made with the command "cd" (change directory)

Enter "git clone <url of the project you want to clone>"

In the example, I cloned Anton's fork of the JackTrip repository, located at the URL pictured

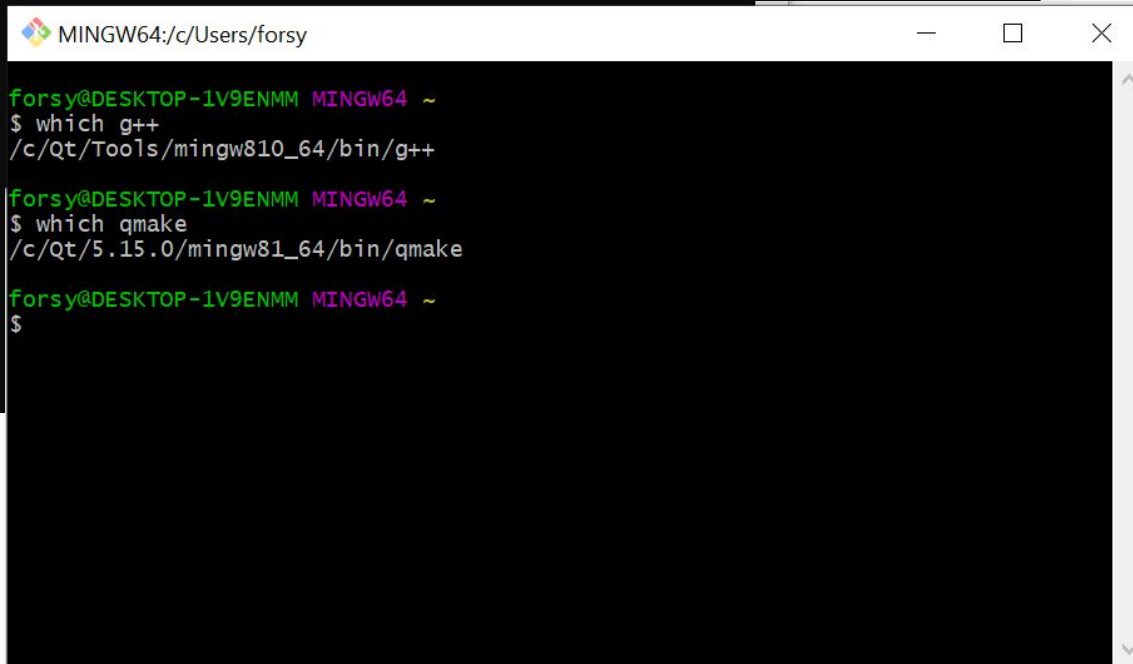


```
CA: Command Prompt
Microsoft Windows [Version 10.0.19041.450]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\forsy>where g++
C:\Qt\Tools\mingw810_64\bin\g++.exe

C:\Users\forsy>where qmake
C:\Qt\5.15.0\mingw81_64\bin\qmake.exe

C:\Users\forsy>
```



```
MINGW64:/c/Users/forsy

forsy@DESKTOP-1V9ENMM MINGW64 ~
$ which g++
/c/Qt/Tools/mingw810_64/bin/g++

forsy@DESKTOP-1V9ENMM MINGW64 ~
$ which qmake
/c/Qt/5.15.0/mingw81_64/bin/qmake

forsy@DESKTOP-1V9ENMM MINGW64 ~
$
```

In order to build JackTrip from source, we need the help of two applications, g++ and qmake.

Those applications should have come with your installation of Qt.

In order to test if your system can find those applications, you can use the command “where” (CMD language) or “which” (unix based language)

Here are pictures of both types of terminals, doing the same thing

```

MINGW64:/c/Users/forsy
_c64/bin:/c/Qt/5.15.0/mingw81_64/bin:/usr/bin/vendor_perl:/usr/bin/core_perl)

forsy@DESKTOP-1V9ENMM MINGW64 ~
$ which g--
which: no g-- in (C:/c/Users/forsy/bin:/mingw64/bin:/usr/local/bin:/usr/bin:/bin:/mingw64/bin:/usr/bin:/c/Users/forsy/bin:/c/Program Files (C
x86)/Intel/Intel(R) Management Engine Components/iCLS:/c/Program Files/Intel/Intel(R) Management Engine Components/iCLS:/c/windows/system3
2:/c/windows:/c/windows/System32/wbem:/c/windows/System32/windowsPowerShell/v1.0:/c/windows/System32/OpenSSH:/c/Program Files (x86)/Intel/
Intel(R) Management Engine Components/DAL:/c/Program Files/Intel/Intel(R) Management Engine Components/DAL:/c/Program Files (x86)/Intel/In
tel(R) Management Engine Components/IPT:/c/Program Files/Intel/Intel(R) Management Engine Components/IPT:/c/Program Files (x86)/NVIDIA Cor
poration/PhysX/Common:/c/WINDOWS/system32:/c/WINDOWS:/c/WINDOWS/System32/wbem:/c/WINDOWS/System32/windowsPowerShell/v1.0:/c/WINDOWS/System
32/OpenSSH:/cmd:/c/Qt/%VERSION%/bin:/c/Qt/Tools/mingw810_64/bin:/c/Qt/5.15.0/mingw81_64/bin:/c/Qt/%VERSION%/bin:/c/Qt/Tools/mingw810_64/bi
n:/c/Qt/5.15.0/mingw81_64/bin:/usr/bin/vendor_perl:/usr/bin/core_perl)

forsy@DESKTOP-1V9ENMM MINGW64 ~
$

forsy@DESKTOP-1V9ENMM MINGW64 ~
$

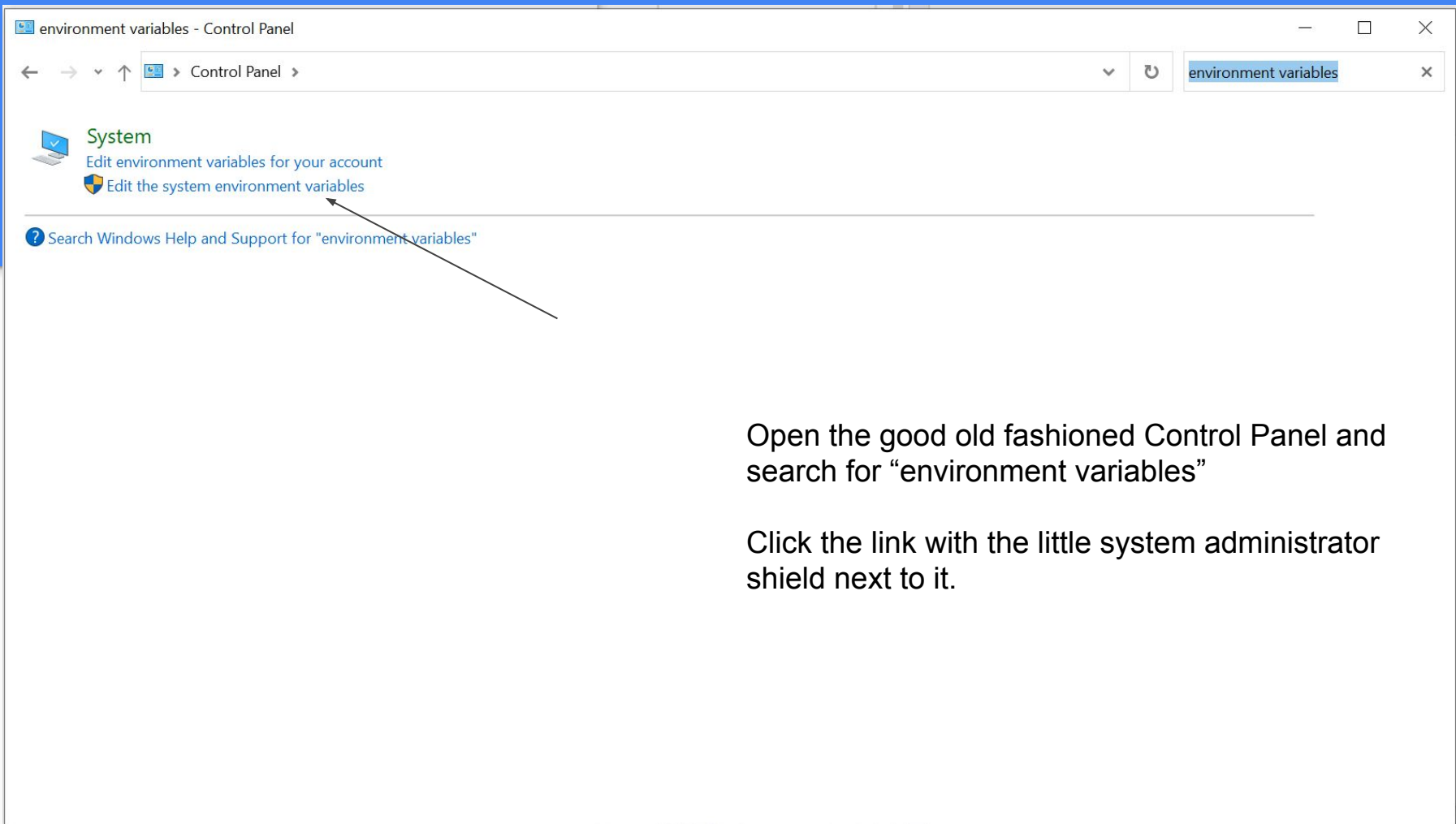
forsy@DESKTOP-1V9ENMM MINGW64 ~
$

forsy@DESKTOP-1V9ENMM MINGW64 ~
$

```

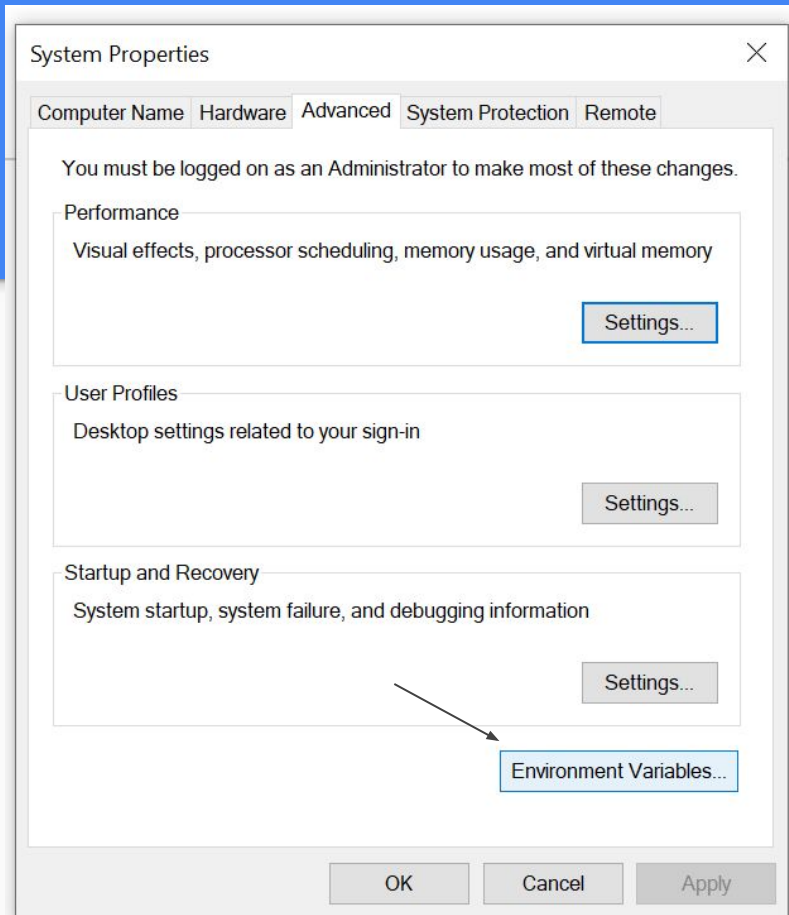
If your terminal didn't spit back the file path when you look for "which g++" or "where g++", you might get a message like this. (I used the nonexistent application "g--" for this example, since my system already knows where "g++" is.) My system is telling me that it doesn't know where "g--" is. It's also telling me all the places it's looking for "g--"

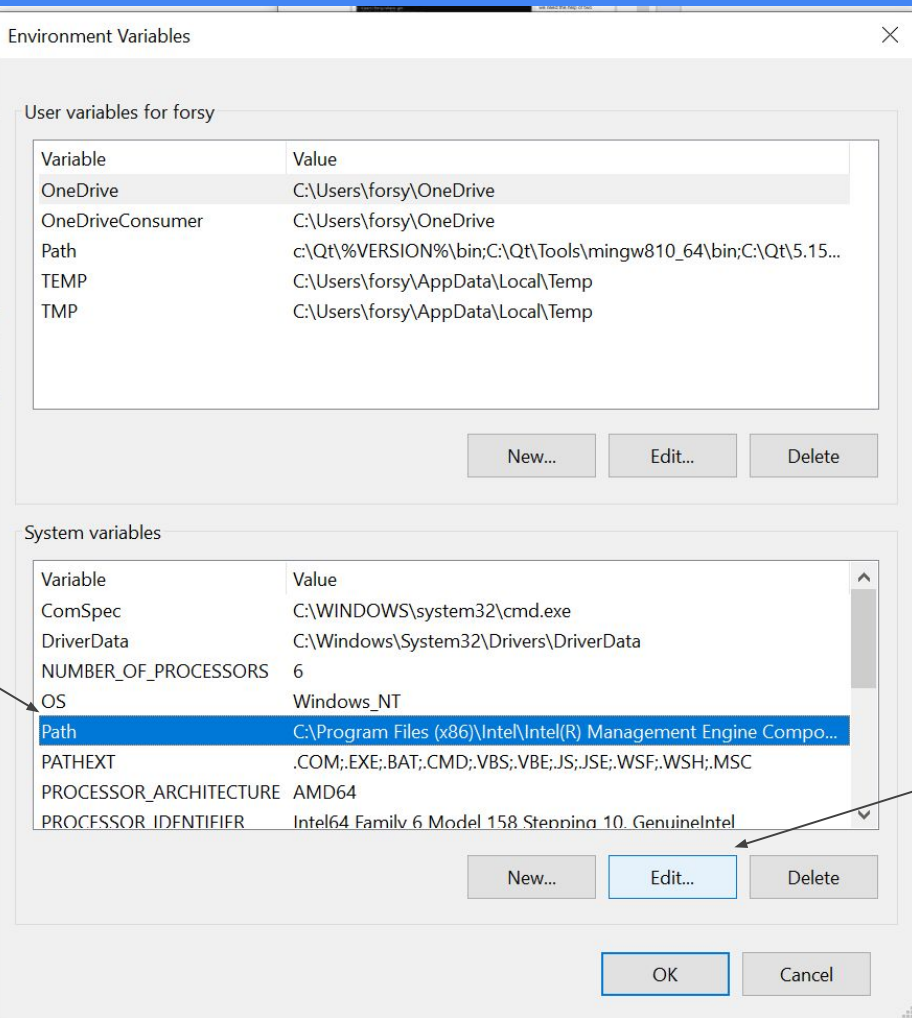
If you get a message like this, you need to tell your system where it can find those applications. We do this by adding some folders to the “Path” system environment variable



Open the good old fashioned Control Panel and search for “environment variables”

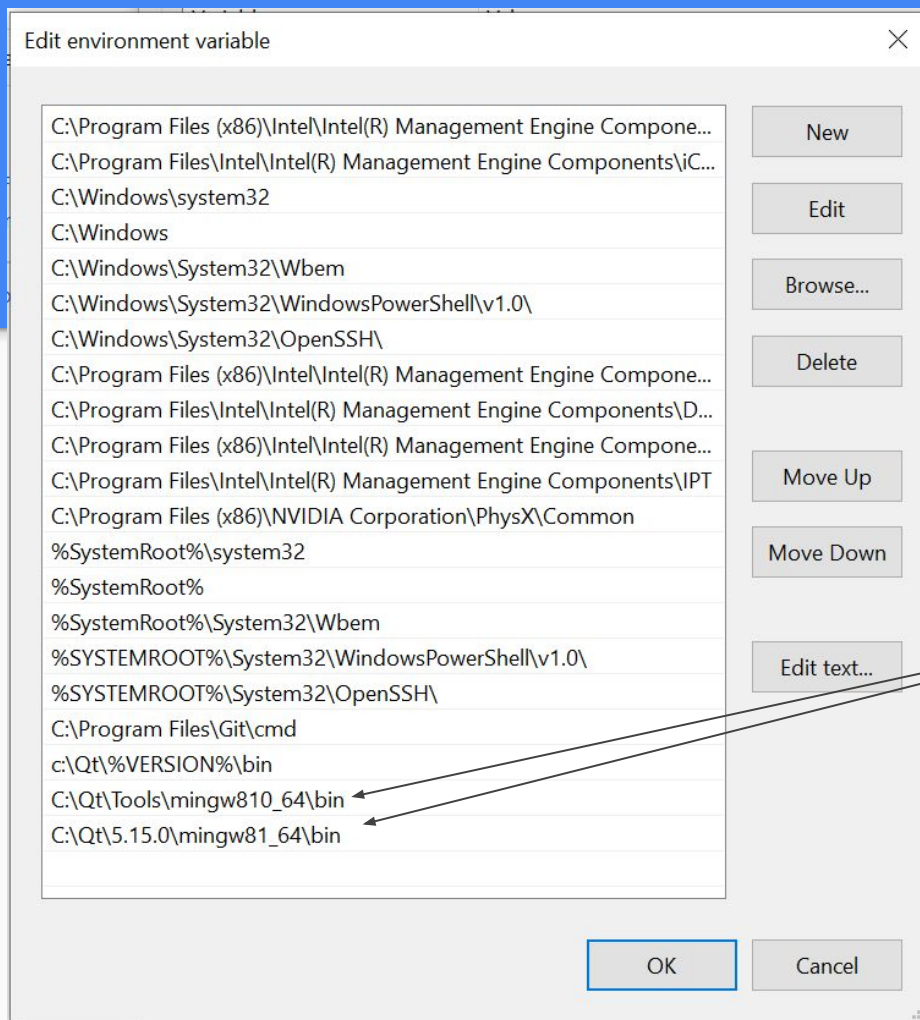
Click the link with the little system administrator shield next to it.





Select "Path"

Then click Edit...



I added these 2 entries, because these folders are where qmake and g++ were

```
Command Prompt
Microsoft Windows [Version 10.0.19041.450]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\forsy>where g++
C:\Qt\Tools\mingw810_64\bin\g++.exe

C:\Users\forsy>where qmake
C:\Qt\5.15.0\mingw81_64\bin\qmake.exe

C:\Users\forsy>
```

```
MINGW64:/c/Users/forsy

forsy@DESKTOP-1V9ENMM MINGW64 ~
$ which g++
/c/Qt/Tools/mingw810_64/bin/g++

forsy@DESKTOP-1V9ENMM MINGW64 ~
$ which qmake
/c/Qt/5.15.0/mingw81_64/bin/qmake

forsy@DESKTOP-1V9ENMM MINGW64 ~
$
```

Now, you should be able to which/where and get info about location of those applications. Now we're all set to build!

Build the project

Why do I need to build the project?

We have all the tools we need

Time to make an app that we can run!

```
MINGW64:/c/Users/forsy/Experimental_JackTrip_Repos/jacktrip/builddir
forsy@DESKTOP-1V9ENMM MINGW64 ~
$ cd Experimental_JackTrip_Repos/jacktrip/src
forsy@DESKTOP-1V9ENMM MINGW64 ~/Experimental_JackTrip_Repos/jacktrip/src (master)
$ mkdir ../builddir; cd ../builddir
forsy@DESKTOP-1V9ENMM MINGW64 ~/Experimental_JackTrip_Repos/jacktrip/builddir (master)
$ qmake -spec win32-g++ ../src/jacktrip.pro
```

in the terminal, navigate to the jacktrip/src dir with
`cd {yourfolder}/jacktrip/src`

type: `mkdir ../builddir; cd ../builddir`

`qmake -spec win32-g++ ../src/jacktrip.pro`

`mingw32-make clean`

`qmake -spec win32-g++ ../src/jacktrip.pro`

`mingw32-make release`

A lot of text should show up. You've made it!

Now, all there is to do is run the App we made.

Run JackTrip

```
MINGW64~/Users/forsy/Experimental_JackTrip_Repos/jacktrip/builddir/release
forsy@DESKTOP-1V9ENMM MINGW64 ~/Experimental_JackTrip_Repos/jacktrip/builddir (master)
$ cd release/
forsy@DESKTOP-1V9ENMM MINGW64 ~/Experimental_JackTrip_Repos/jacktrip/builddir/release (master)
$ ./jacktrip.exe

JackTrip: A System for High-Quality Audio Network Performance
over the Internet
Copyright (c) 2008-2018 Juan-Pablo Caceres, Chris Chafe.
SoundWIRE group at CCRMA, Stanford University
VERSION: 1.2beta2

Usage: jacktrip [-s|-c host] [options]

Options:
REQUIRED ARGUMENTS:
-s, --server                Run in Server Mode
-c, --client <peer_hostname_or_IP_num> Run in Client Mode
-S, --jacktripserver        Run in Hub Server Mode
-C, --pingtoserver <peer_name_or_IP>   Run in Hub Client Mode

OPTIONAL ARGUMENTS:
-n, --numchannels #         Number of Input and Output Channels (default: 2)
-q, --queue # (2 or more)   Queue Buffer Length, in Packet Size (default: 4)
-r, --redundancy # (1 or more) Packet Redundancy to avoid glitches with packet losses (default: 1)
-o, --portoffset #          Receiving port offset from base port 4464
--bindport #                Set only the bind port number (default: 4464)
--peerport #                Set only the Peer port number (default: 4464)
-b, --bitres # (8, 16, 24, 32) Audio Bit Rate Resolutions (default: 16)
-p, --hubpatch # (0, 1, 2, 3, 4) Hub auto audio patch, only has effect if running HUB SERVER
mode, 0=server-to-clients, 1=client loopback, 2=client fan out/in but not loopback, 3=reserved for TUB
, 4=full mix (default: 0)
-z, --zerounderrun          Set buffer to zeros when underrun occurs (default: wavetable)
-l, --loopback              Run in Loop-Back Mode
-j, --jamlink               Run in JamLink Mode (Connect to a JamLink Box)
--clientname                Change default client name (default: JackTrip)
--localaddress              Change default local host IP address (default: 127.0.0.1)
--nojackportsconnect        Don't connect default audio ports in jack

ARGUMENTS TO USE JACKTRIP WITHOUT JACK:
--rtaudio                  Use system's default sound system instead of Jack
--srate #                  Set the sampling rate, works on --rtaudio mode only (default: 48000)
--bufsize #                Set the buffer size, works on --rtaudio mode only (default: 128)
--deviceid #               The rtaudio device id --rtaudio mode only (default: 0)

ARGUMENTS TO DISPLAY IO STATISTICS:
--iostat <time_in_secs>    Turn on IO stat reporting with specified interval (in second)
--iostatlog <log_file>     Save stat log into a file (default: print in stdout)

HELP ARGUMENTS:
-v, --version              Prints Version Number
-V, --verbose              Verbose mode, prints debug messages
-h, --help                 Prints this Help

forsy@DESKTOP-1V9ENMM MINGW64 ~/Experimental_JackTrip_Repos/jacktrip/builddir/release (master)
$
```

The app should be in the “release” folder within builddir

Cd to release

Type: “./jacktrip” to print out the manual of parameters you can tack on when running it.

You’re done! Congrats!