running a client or server from source

running from source

I write the client and server entirely in python, which can run straight from source. It is not simple to get hydrus running this way, but if none of the built packages work for you (for instance you use a non-Ubuntu-compatible flavour of Linux), it may be the only way you can get the program to run. Also, if you have a general interest in exploring the code or wish to otherwise modify the program, you will obviously need to do this stuff.

a quick note about Linux flavours

I often point people here when they are running non-Ubuntu flavours of Linux and cannot run my build. One Debian user mentioned that he had an error like this:

- *ImportError: /home/user/hydrus/libX11.so.6: undefined symbol: xcb_poll_for_reply64*

But that by simply deleting the *libX11.so.6* file in the hydrus install directory, he was able to boot. I presume this meant my hydrus build was then relying on his local libX11.so, which happened to have better API compatibility. If you receive a similar error, you might like to try the same sort of thing. Let me know if you discover anything!

building on windows

Installing some packages on windows with pip may need Visual Studio's C++ Build Tools for your version of python. Although these tools are free, it can be a pain to get them through the official (and often huge) downloader installer from Microsoft. Instead, install Chocolatey and use this one simple line:

> choco install -y vcbuildtools visualstudio2017buildtools

Trust me, this will save a ton of headaches!

what you will need

You will need basic python experience, python 3.x and a number of python modules. Most of it you can get through pip.

If you are on Linux or macOS, or if you are on Windows and have an existing python you do not want to stomp all over with new modules, I recommend you create a virtual environment:

Note, if you are on Linux, it may be easier to use your package manager instead of messing around with venv. A user has written a great summary with all needed packages here.

If you do want to create a new venv environment:

- (navigate to your hydrus extract folder)
- pip3 install virtualenv (if you need it)
- pip3 install wheel (if you need it)
- mkdir venv
- virtualenv --python=python3 venv
- . venv/bin/activate

That '. venv/bin/activate' line turns your venv on, and will be needed every time you run the client.pyw/server.py files. You can easily tuck it into a launch script.

On Windows, the path is venv\Scripts\activate, and the whole deal is done much easier in cmd than Powershell. If you get Powershell by default, just type 'cmd' to get an old fashioned command line. In cmd, the launch command is just 'venv\scripts\activate', no leading period.

After that, you can go nuts with pip. I think this will do for most systems:

- pip3 install beautifulsoup4 chardet html5lib lxml nose numpy opencv-python-headless six Pillow psutil PyYAML requests Send2Trash service_identity twisted

You may want to do all that in smaller batches.

You will also need Qt5. Either PySide2 (default) or PyQt5 are supported, through qtpy. You can install, again, with pip:

- pip3 install qtpy PySide2

-or-

- pip3 install qtpy PyQtChart PyQt5

Qt 5.15 currently seems to be working well, but 5.14 caused some trouble.

And optionally, you can add these packages:

- **python-mpv - to get nice video and audio support!**

  > If you are on Linux/macOS, you will likely need the mpv library installed to your system, *not just mpv*, which is often called 'libmpv1'. You can usually get it with *apt*.

- lz4 - for some memory compression in the client
- pylzma - for importing rare ZWS swf files
- cloudscraper - for attempting to solve CloudFlare check pages
- pysocks - for socks4/socks5 proxy support (although you may want to try "requests[socks]" instead)
- >PyOpenSSL - to generate a certificate if you want to run the server or the client api
- mock httmock pyinstaller - if you want to run test.py and make a build yourself
- PyWin32 pypiwin32 pywin32-ctypes - helpful to ensure you have if you want to make a build in Windows

Here is a masterline with everything for general use:

- pip3 install beautifulsoup4 chardet html5lib lxml nose numpy opencv-python-headless six Pillow psutil PyOpenSSL PyYAML requests Send2Trash service_identity twisted qtpy PySide2 python-mpv lz4 pylzma cloudscraper pysocks

For Windows, depending on which compiler you are using, pip can have problems building some modules like lz4 and lxml. This page has a lot of prebuilt binaries--I have found it very helpful many times. You may want to update python's sqlite3.dll as well--you can get it here, and just drop it in C:\Python37\DLLs or wherever you have python installed. I have a fair bit of experience with Windows python, so send me a mail if you need help.

If you don't have ffmpeg in your PATH and you want to import videos, you will need to put a static FFMPEG executable in the install_dir/bin directory. Have a look at how I do it in the extractable compiled releases if you can't figure it out. On Windows, you can copy the exe from one of those releases, or just download the latest static build right from the FFMPEG site.

Once you have everything set up, client.pyw and server.py should look for and run off client.db and server.db just like the executables. They will look in the 'db' directory by default, or anywhere you point them with the "-d" parameter, again just like the executables.

I develop hydrus on and am most experienced with Windows, so the program is more stable and reasonable on that. I do not have as much experience with Linux or macOS, so I would particularly

appreciate your Linux/macOS bug reports and any informed suggestions.

my code

Unlike most software people, I am more INFJ than INTP/J. My coding style is unusual and unprofessional, and everything is pretty much hacked together. Please look through the source if you are interested in how things work and ask me if you don't understand something. I'm constantly throwing new code together and then cleaning and overhauling it down the line.

I work strictly alone, so while I am very interested in detailed bug reports or suggestions for good libraries to use, I am not looking for pull requests. Everything I do is WTFPL, so feel free to fork and play around with things on your end as much as you like.

Revision #1
Created 12 March 2021 16:50:21 by CuddleBear
Updated 12 March 2021 18:24:07 by CuddleBear