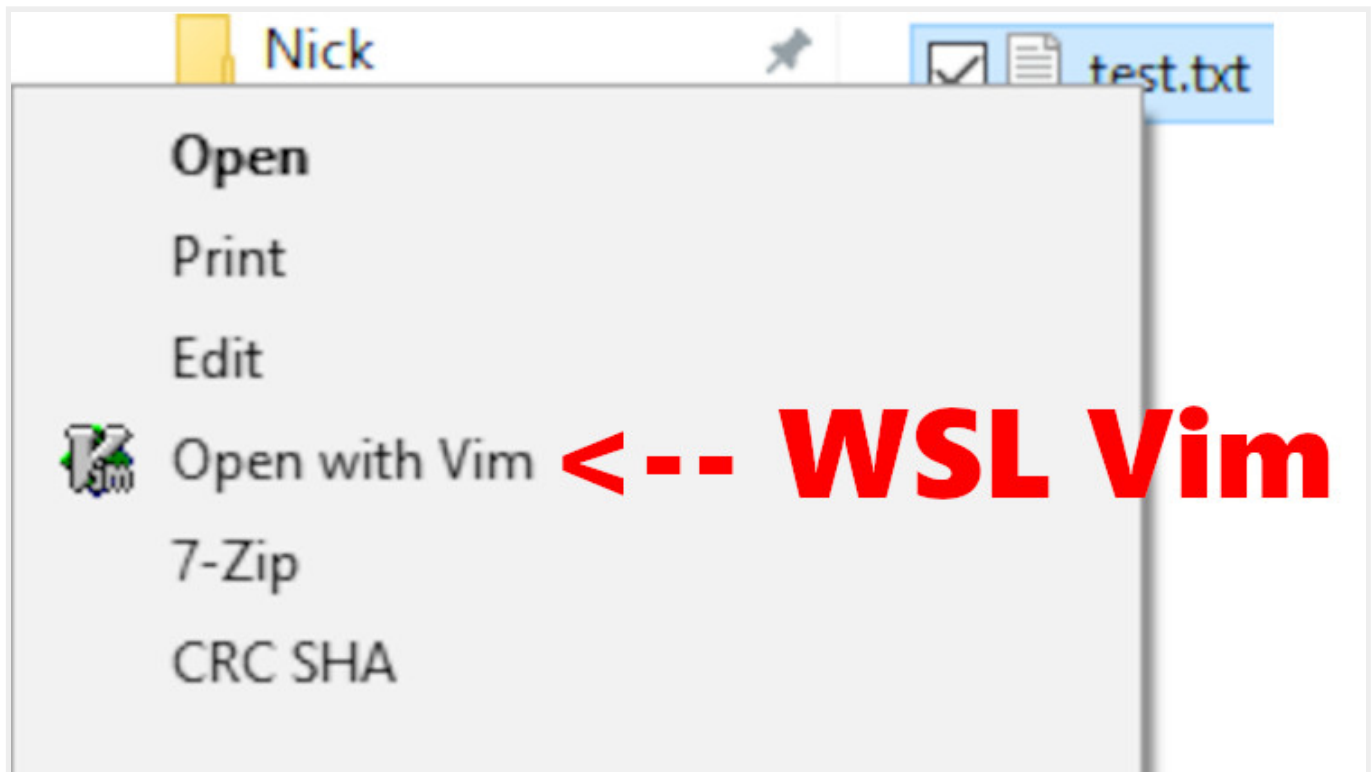


## Learn Docker With My Newest Course

Dive into Docker ([https://diveintodocker.com/?utm\\_source=nj&utm\\_medium=website-banner&utm\\_campaign=/blog/launching-wsl-programs-from-a-right-click-windows-menu](https://diveintodocker.com/?utm_source=nj&utm_medium=website-banner&utm_campaign=/blog/launching-wsl-programs-from-a-right-click-windows-menu)) takes you from "What is Docker?" to confidently applying Docker to your own projects. It's packed with best practices and examples. [Start Learning Docker → \(https://diveintodocker.com/?utm\\_source=nj&utm\\_medium=website-banner&utm\\_campaign=/blog/launching-wsl-programs-from-a-right-click-windows-menu\)](https://diveintodocker.com/?utm_source=nj&utm_medium=website-banner&utm_campaign=/blog/launching-wsl-programs-from-a-right-click-windows-menu)

Updated on February 26th, 2019 in #dev-environment (<https://nickjanetakis.com/blog/tag/dev-environment-tips-tricks-and-tutorials>)

## Launching WSL Programs from a Right Click Windows Menu



A common use case for this would be opening a specific file with terminal Vim by right clicking a file in Windows explorer.

**Quick Jump:** [Putting Together a Command to Run](#) | [Creating the Right Click Menu Item](#) | [Launching WSL Programs as Custom Shortcuts](#)

I recently switched to using Vim (</blog/vim-is-saving-me-hours-of-work-when-writing-books-and-courses>) as my primary code editor, and it's currently running directly inside of WSL. It's the terminal version of Vim, rather than running GVim directly from within Windows.

Since I spend most of my time in a terminal, I do most of my file browsing there too with a tool called ranger, but every once in a while I find myself in Windows explorer browsing through files and I want to be able to open a text file with Vim.

VSCode has a menu item added where you can right click a file and then open it with VSCode and that's exactly what I wanted to replicate with the new set up.

With Vim, it's a little more tricky to set up because it involves launching your WSL terminal, running a bash command and also transforming the Windows file path so it works inside of WSL. That's what we're going to cover in this article.

The cool thing about this strategy is it will work for all terminals and all programs you want to launch. So if you don't use the same terminal as me, then all you have to do is adjust a few flags for your specific terminal, and when it comes to launching Vim, you could easily replace that with whatever app you want to launch instead.

## Putting Together a Command to Run

Before we create the right click menu item, we need to come up with the command to run. This would be the command to run after selecting the menu item.

### 3 components to launching and running a WSL command:

1. The absolute path to your terminal's binary
2. Determining which flags your terminal supports
3. The bash command you want to run

### 1. The Absolute Path to Your Terminal's Binary

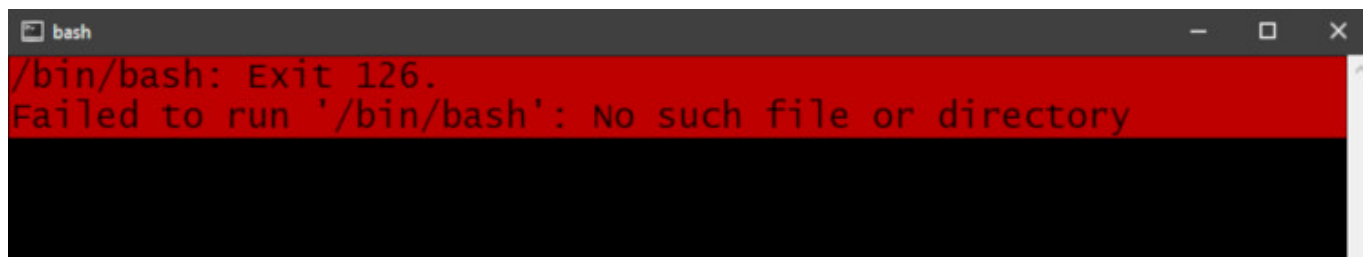
Nowadays I am using wsltty as my WSL terminal of choice. It doesn't have the lowest input latency like the default `ubuntu.exe` terminal, but the hot keys for adjusting font sizes on the fly makes it a win for me as someone who creates videos.

The path to this terminal can be found at

`C:\Users\Nick\AppData\Local\wsltty\bin\mintty.exe`. You will want to replace `Nick` with your Windows user name.

*That path is going to be different for each terminal so if you're using a different terminal you'll need to figure out where yours is located.*

You can verify it works by copying your path and then hitting `Win + R` to launch the Windows command runner, and then paste it in. After running it, you should see a terminal window pop open.

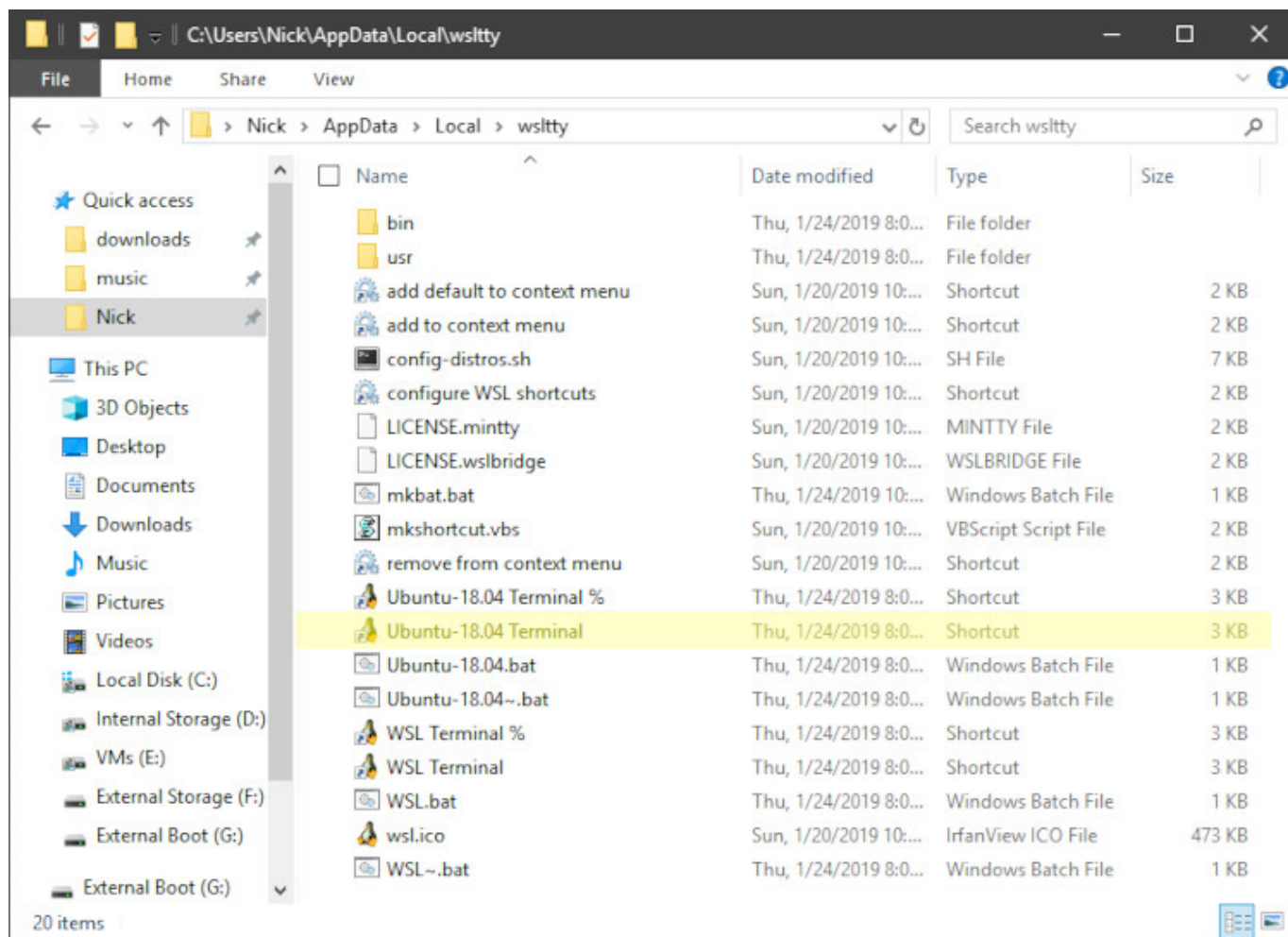


In the case of wsltty, the terminal we're actually running is mintty, but it has a WSL mode. We haven't enabled that mode yet so you'll get an error. **That error is normal.**

## 2. Determining Which Flags Your Terminal Supports

When it comes to wsltty you can go to `C:\Users\Nick\AppData\Local\wsltty` in Windows explorer and see a bunch of files related to this terminal.

Some of the files you'll see are shortcuts with a Linux penguin icon.



Right click the one that says `Ubuntu-18.04 Terminal` and go to properties.

Inside of the target box you'll see a path that looks like this:

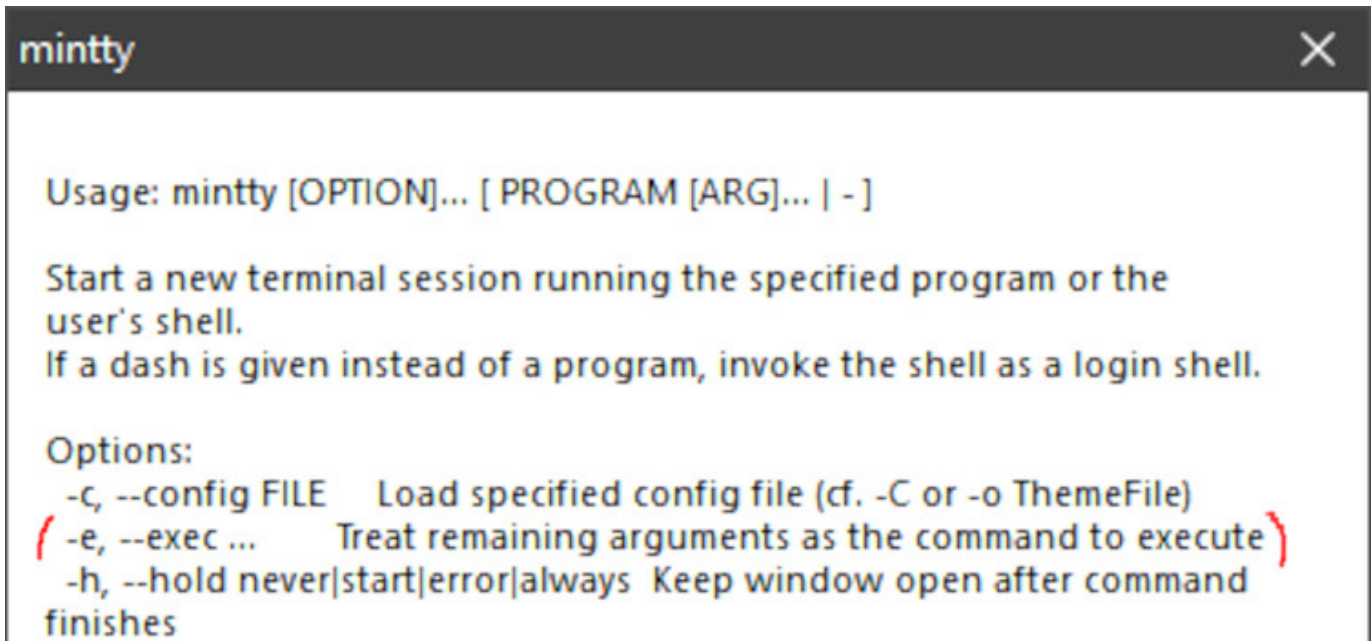
```
C:\Users\Nick\AppData\Local\wsltty\bin\mintty.exe --WSL="Ubuntu-18.04" --
configdir="C:\Users\Nick\AppData\Roaming\wsltty" --~
```

This shortcut launches the terminal and places you in your WSL home directory. All we have to do now is run a specific bash command instead of loading the home directory.

But before we get into that, let's talk a little more about these flags. The `--WSL` flag loads a specific WSL distro and the `--configdir` is where you can supply your config file. Those are both pretty self explanatory and are specific to mintty.

However, the `--~` looks a little magical to me, so I wanted to see what's available with mintty, so I hit `Win + R` and ran `C:\Users\Nick\AppData\Local\wsltty\bin\mintty.exe --help`.

That loads up the help menu:



```
mintty

Usage: mintty [OPTION]... [ PROGRAM [ARG]... | - ]

Start a new terminal session running the specified program or the
user's shell.
If a dash is given instead of a program, invoke the shell as a login shell.

Options:
  -c, --config FILE    Load specified config file (cf. -C or -o ThemeFile)
  -e, --exec ...       Treat remaining arguments as the command to execute
  -h, --hold never|start|error|always Keep window open after command
                        finishes
```

One of the options that caught my eye was `-e` which states that every argument after `-e` will be treated as a command to execute. That is exactly what we want.

What we want to do is spawn a new Bash session and then launch Vim. The reason I wanted to load Bash is due to having a `~/.profile` file which sources a little script to make my Vim colors more accurate in 256 color mode. This is part of the gruvbox theme (<https://github.com/morhetz/gruvbox>) I use.

### 3. The Bash Command You Want to Run

Before we get into launching Vim, you can test out how to run any Bash command by running this from your existing WSL terminal: `bash --login -c "vim --version"`.

You should see version information about Vim, or whatever command you decided to run if you're applying this to a different application.

So that takes care of running any Bash command. The only thing that's left to do is instruct Vim to open a specific file instead of getting the version back.

#### Opening files with Vim:

For example, if you wanted to open your `bashrc` file from within WSL you would run `vim $HOME/.bashrc`. The takeaway here is you supply the file path to `vim`.

But let's say you have a file in `C:\Users\Nick\test.txt` which is outside of WSL. We can't do `vim C:\Users\Nick\test.txt`.

Under normal circumstances, if you were running Vim manually inside of an existing WSL terminal you would do `vim /c/Users/Nick/test.txt`, or perhaps you would start it off with `/mnt/c` depending on where you mounted your C drive.

## Converting Windows paths to WSL paths:

It just so happens when you install WSL, it comes with a tool called `wslpath`.

If you launch a WSL terminal and run `wslpath` you will get back the help menu.

```
$ wslpath
wslpath: Invalid argument
Usage:
  -a    force result to absolute path format
  -u    translate from a Windows path to a WSL path (default)
  -w    translate from a WSL path to a Windows path
  -m    translate from a WSL path to a Windows path, with '/' instead of '\'

EX: wslpath 'c:\users'
```

The main thing to note is the `-u` flag. It's the default option and it will translate a Windows path to a WSL path. That means we can run `wslpath 'C:\Users\Nick\test.txt'`.

```
$ wslpath 'C:\Users\Nick\test.txt'
/c/Users/Nick/test.txt
```

The quotes are necessary. Otherwise the path will be malformed, but you should get back a properly formatted WSL path that looks like `/c/Users/Nick/test.txt`.

## Putting it all together to create our command:

Now we know how to run any Bash commands we want, and we also know how to automatically convert file paths. Let's put them together.

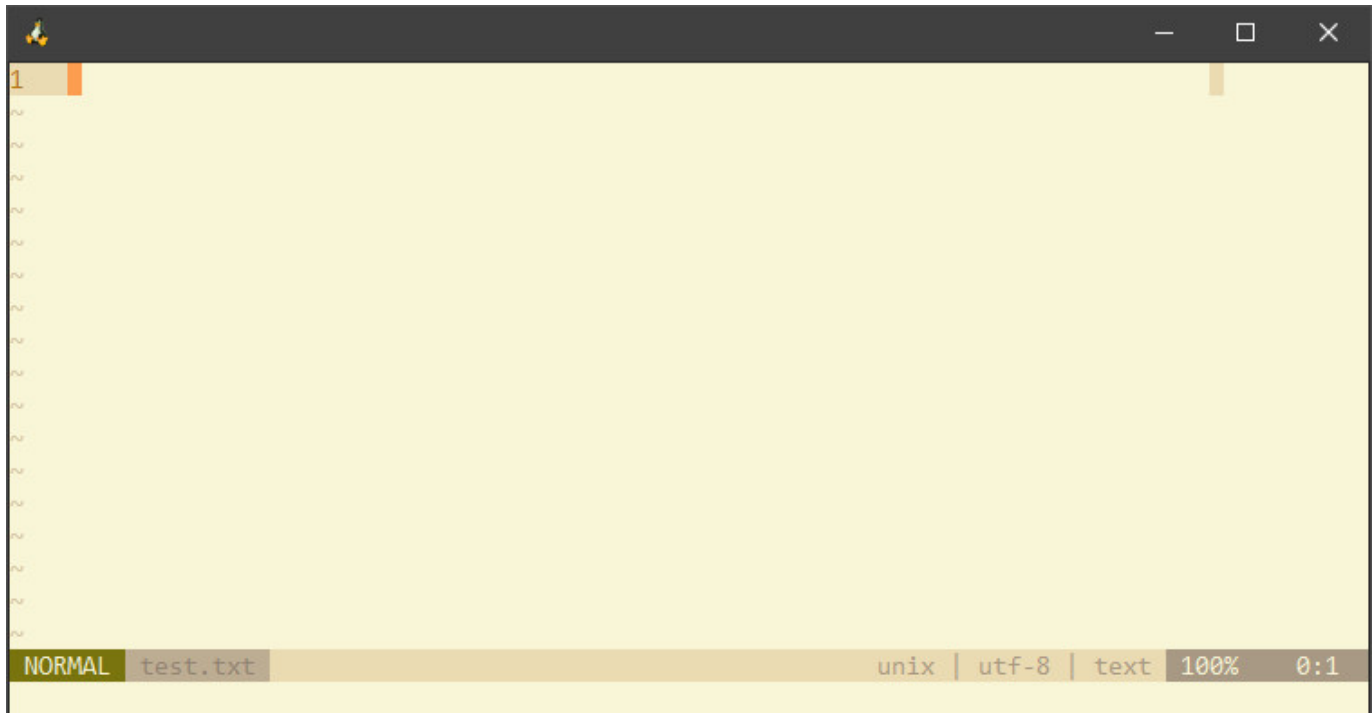
Our command will look like this: `bash --login -c "vim $(wslpath 'C:\Users\Nick\test.txt')"`

Which we can run in a WSL terminal and it should open that file. Feel free to close it.

Now we're ready to put everything together to create our command:

```
C:\Users\Nick\AppData\Local\wsltty\bin\mintty.exe --WSL="Ubuntu-18.04" --  
configdir="C:\Users\Nick\AppData\Roaming\wsltty" -e bash --login -c "vim $(wslpath  
'C:\Users\Nick\test.txt')
```

If you copy / paste that and run it from **Win + R** it should open wsltty, load Vim and then display the empty `test.txt` file.



I've edited out the title of the terminal window but yours should look a little crazy. It likely has information about `wslbridge` and other things that have no use to us.

### Adding finishing touches by customizing the terminal window's title:

If you look back at the wsltty help menu, it has another flag called `-t` which lets you set the title of the terminal window.

We can add that to our command before the `-e` flag like so:

```
C:\Users\Nick\AppData\Local\wsltty\bin\mintty.exe --WSL="Ubuntu-18.04" --  
configdir="C:\Users\Nick\AppData\Roaming\wsltty" -t "C:\Users\Nick\test.txt" -e bash  
--login -c "vim $(wslpath 'C:\Users\Nick\test.txt')
```

Now if we run the command again, it will show the file path in the title bar.



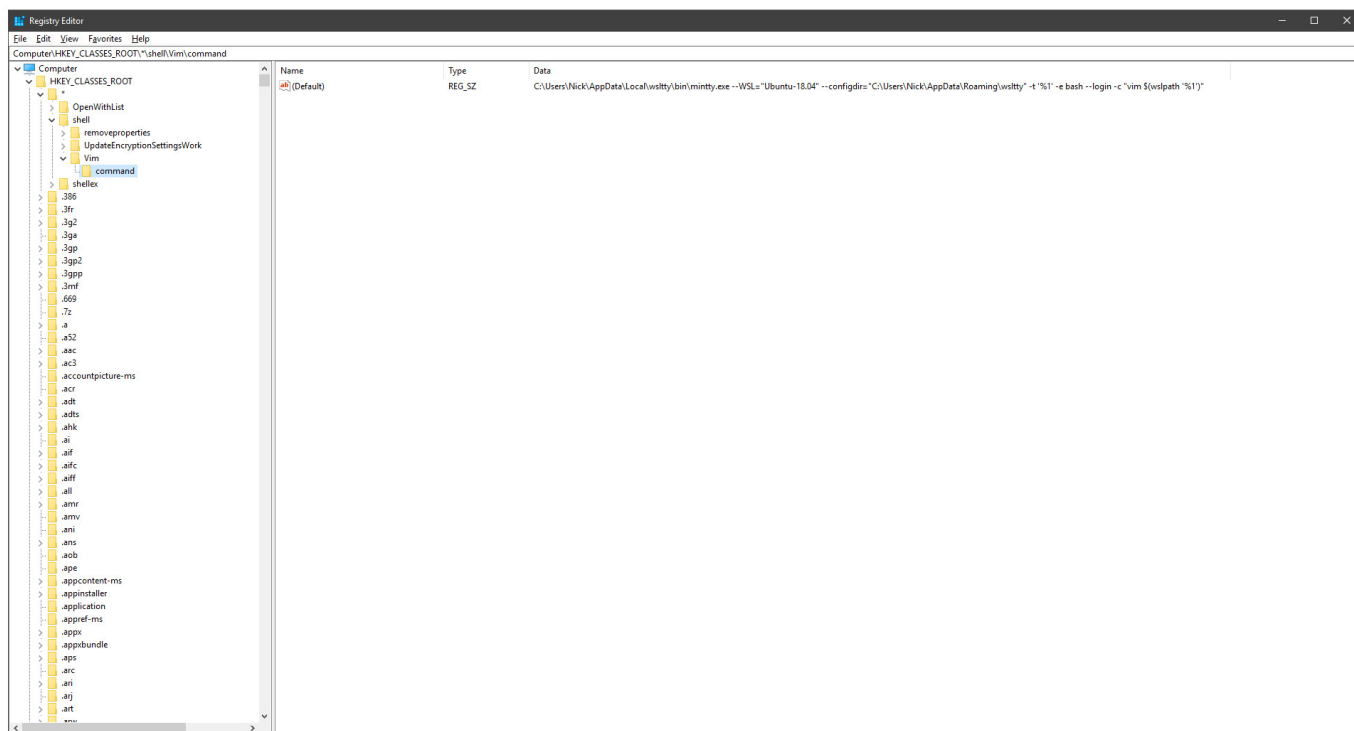
This is great. We have our command. The only problem now is it's hard coded to a specific file, but we'll see how to make that dynamic based on what file was right clicked.

## Creating the Right Click Menu Item

We'll have to open up the Windows registry, which you can do by hitting `Win + R` and then running `regedit.exe`. You'll also need to become an admin for this.

1. Navigate to `HKEY_CLASSES_ROOT\*\shell` by expanding the folders on the left
2. Right click `shell` and go to New, then select Key
3. Name your key `vim` or whatever makes sense for your app
4. Left click your new `vim` key then click the right hand side's `(Default)` item
5. Right click the `(Default)` item and go to Modify
6. Enter in `Open with Vim` for the `Value data` field and hit OK
7. Right click your new `vim` key and go to New, then select Key
8. Name your key `command`
9. Left click `command` then click the right hand side's `(Default)` item
10. Right click the `(Default)` item and go to Modify
11. Paste your final command in the `Value data` field
12. Change `C:\Users\Nick\test.txt` to `%1` in **both spots** and click OK

After all of those steps, it should look like this:





Now if you open Windows explorer and right click any file, you should see an “Open with Vim” menu item. If you click it, it will open that file in Vim with the correct file path and terminal title. The `%1` gets the current path of the file you right clicked.

If you want to change the wording of the menu item, change step 6.

### Adding finishing touches by showing a Vim icon:

The icon you choose to use will depend on what you’re running. You can download the Vim icon file from Github at

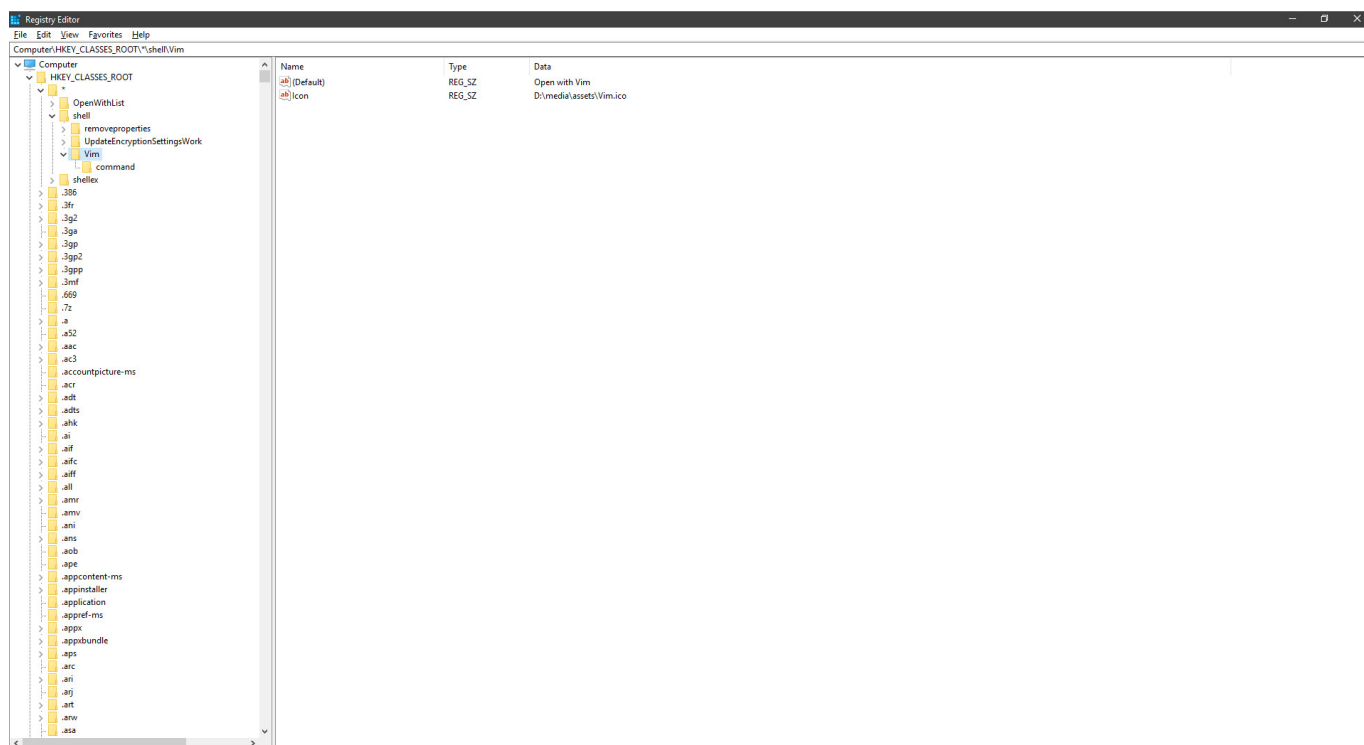
<https://raw.githubusercontent.com/vim/vim/master/src/vim.ico>

(<https://raw.githubusercontent.com/vim/vim/master/src/vim.ico>)

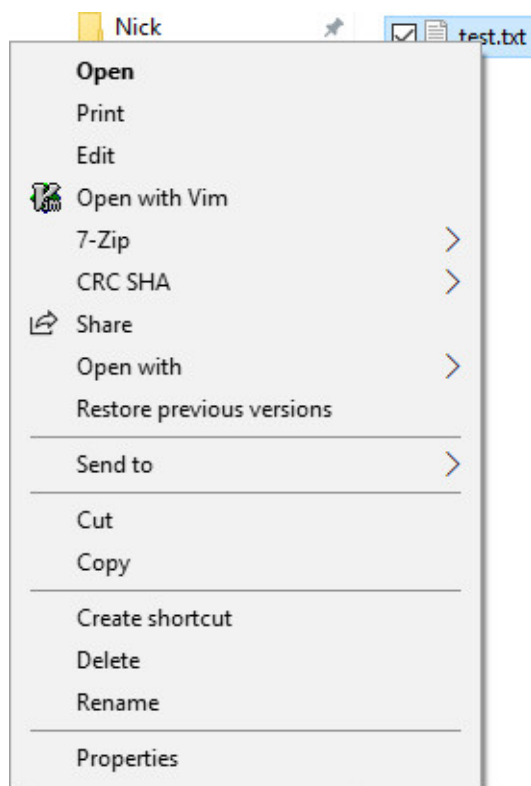
Save it somewhere on your computer. I put mine in `D:\media\assets\Vim.ico`.

Now we’ll have to go back to the Windows registry to set the icon.

1. Navigate to `HKEY_CLASSES_ROOT\*\shell` by expanding the folders on the left
2. Right click `Vim` and go to New, then select String Value
3. Enter in `Icon` as the Name and hit enter
4. Right click the `Icon` item and go to Modify
5. Paste the path to your icon in the `Value data` field and click OK



The next time you right click a file in Windows explorer you should see your icon. Done!



So that's how you can run any WSL tool or program directly from right clicking a file.

## Launching WSL Programs as Custom Shortcuts

Since we've gone through all of that, it's also worth mentioning you can do what we just did with custom commands to create shortcuts to any WSL tool or program.

For example, instead of opening a terminal that loads Bash (the default), you can create a new shortcut for your terminal (similar to the wsltty Ubuntu 18.04 shortcut from before), goto the properties and adjust the target to run something like a Python interpreter.

You could even change the icon from that same menu, no `regedit.exe` required.

Now you have a custom shortcut that you can click to directly open a Python interpreter.

You can even pin these shortcuts to your task bar.


**Which programs are you going to run this way? Let me know below!**


## Never Miss a Tip, Trick or Tutorial


[Get Updates](#)

Like you, I'm super protective of my inbox, so don't worry about getting spammed. You can expect a few emails per month (at most), and you can 1-click unsubscribe at any time. See what else you'll get (<https://nickjanetakis.com/newsletter>) too.

---

 [Tweet \(https://twitter.com/intent/tweet?url=https%3A%2F%2Fnickjanetakis.com%2Fblog%2Flaunching-wsl-programs-from-a-right-click-windows-menu&text=Launching+WSL+Programs+from+a+Right+Click+Windows+Menu+by+%40nickjanetakis&hashtags=\)](https://twitter.com/intent/tweet?url=https%3A%2F%2Fnickjanetakis.com%2Fblog%2Flaunching-wsl-programs-from-a-right-click-windows-menu&text=Launching+WSL+Programs+from+a+Right+Click+Windows+Menu+by+%40nickjanetakis&hashtags=)

 [Share \(https://www.facebook.com/sharer/sharer.php?u=https%3A%2F%2Fnickjanetakis.com%2Fblog%2Flaunching-wsl-programs-from-a-right-click-windows-menu&title=Launching+WSL+Programs+from+a+Right+Click+Windows+Menu\)](https://www.facebook.com/sharer/sharer.php?u=https%3A%2F%2Fnickjanetakis.com%2Fblog%2Flaunching-wsl-programs-from-a-right-click-windows-menu&title=Launching+WSL+Programs+from+a+Right+Click+Windows+Menu)

 [Share \(http://www.linkedin.com/shareArticle?mini=true&url=https%3A%2F%2Fnickjanetakis.com%2Fblog%2Flaunching-wsl-programs-from-a-right-click-windows-menu&title=Launching+WSL+Programs+from+a+Right+Click+Windows+Menu\)](http://www.linkedin.com/shareArticle?mini=true&url=https%3A%2F%2Fnickjanetakis.com%2Fblog%2Flaunching-wsl-programs-from-a-right-click-windows-menu&title=Launching+WSL+Programs+from+a+Right+Click+Windows+Menu)

---

## Comments

## ALSO ON NICKJANETAKIS.COM

### Configuring wsltty Which Is My ...

8 months ago • 7 comments

In this 25 minute video we'll cover both why I really enjoy using wsltty and how to ...

### 4 Use Cases for When to Use Celery in a ...

10 months ago • 8 comments

Celery helps you run code asynchronously or on a periodic schedule which ...

### Remap and Set Global Hotkeys on ...

a year ago • 13 comments

Remapping and overriding global hotkeys can be tricky, but here's an easy way to ...

### Docker Tip: Obscure E

9 months ago

If you're getting you can't expect getting differ

[2 Comments](#) [nickjanetakis.com](#) [Disqus' Privacy Policy](#)[Login](#)[Recommend](#) [Tweet](#) [Share](#)[Sort by Best](#)

Join the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS ?

**taffit** • a year ago

Thank you for this! Helped me keeping my sanity, as the Windows shell didn't work well with NeoVim.

2 ^ | v • Reply • Share ›

**Paolo** • a year ago

Nice tutorial, thanks.

You may also want to add \" around the \$(wslpath ...) for this to work with file names containing spaces

^ | v • Reply • Share ›

[Subscribe](#) [Add Disqus to your site](#) [Add Disqus](#) [Do Not Sell My Data](#)

© 2020 Nick Janetakis

Follow @nickjanetakis

2,630 followers