How do I use Bash on Windows from the Visual Studio Code integrated terminal?

HimanshuHimanshu 3,56911 gold badge1111 silver badges88 bronze badges

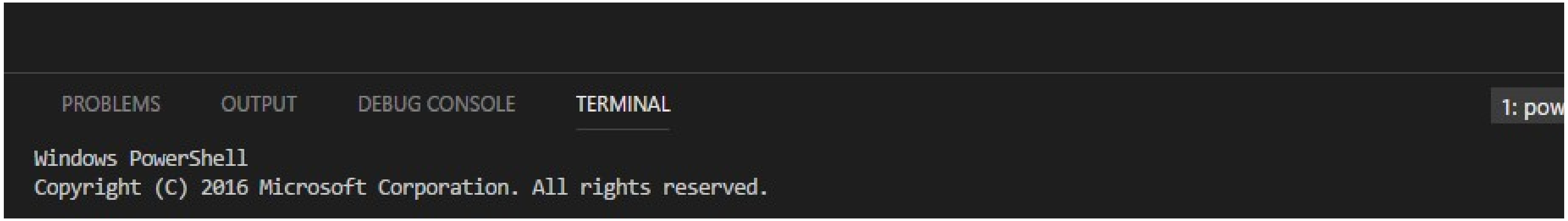
Visual Studio Code on Windows uses PowerShell by default as the integrated terminal. If you want to use Bash from Visual Studio Code, what steps should be followed?

asked Mar 5 '17 at 9:51

[Mahade Walid](#)Mahade Walid

7,3345 gold badges14 silver badges24 bronze badges

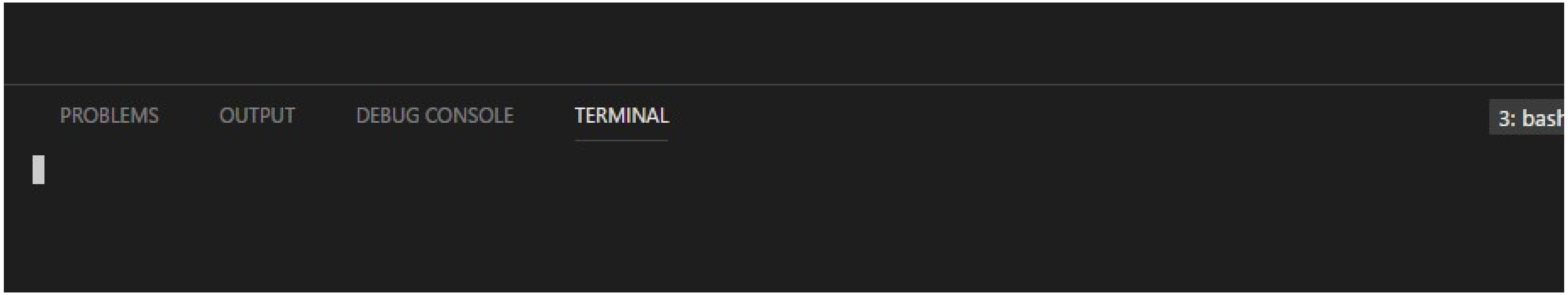
1. Install Git from https://git-scm.com/download/win
2. Open Visual Studio Code and press and hold `Ctrl` + ``` ` ``` to open the terminal.



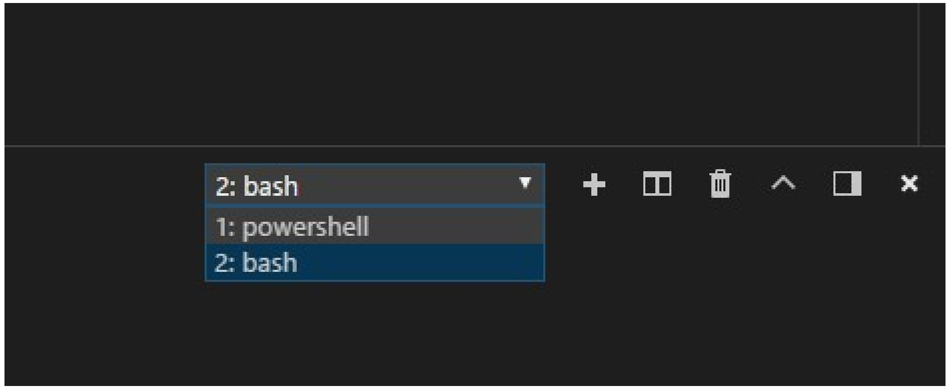
```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.
```

3. Open the command palette using `Ctrl` + `Shift` + `P`.
4. Type - Select Default Shell

5. Select Git Bash from the options
6. Click on the + icon in the terminal window
7. The new terminal now will be a Git Bash terminal. Give it a few seconds to load Git Bash



8. You can now toggle between the different terminals as well from the dropdown in terminal.



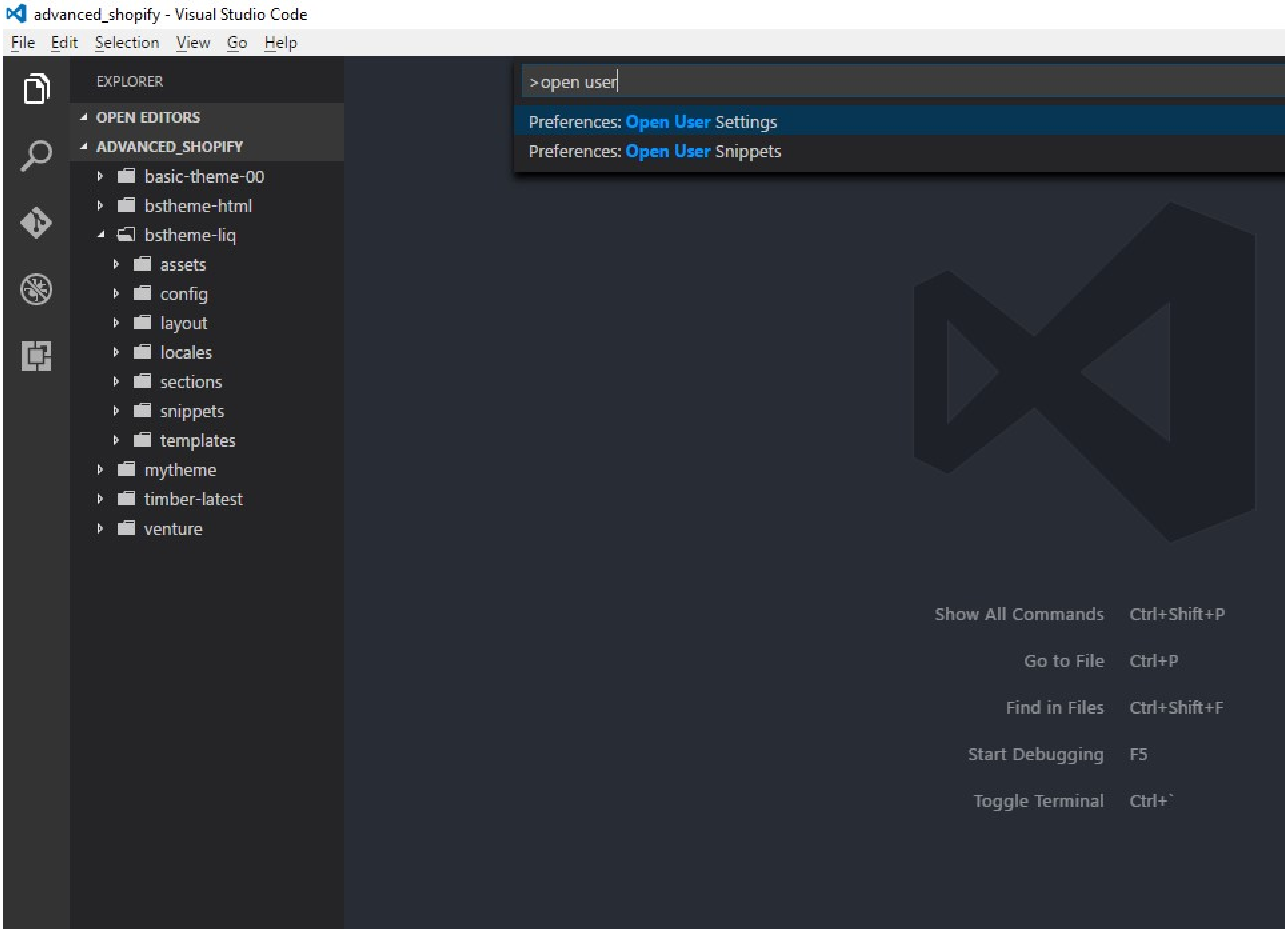
answered May 25 '18 at 11:17

You no longer need to type in bash.exe path manually. This answer is deprecated. Now you can switch to bash

> directly, if you have git installed in the default path. If you installed git to a different path you need to use the below solution.

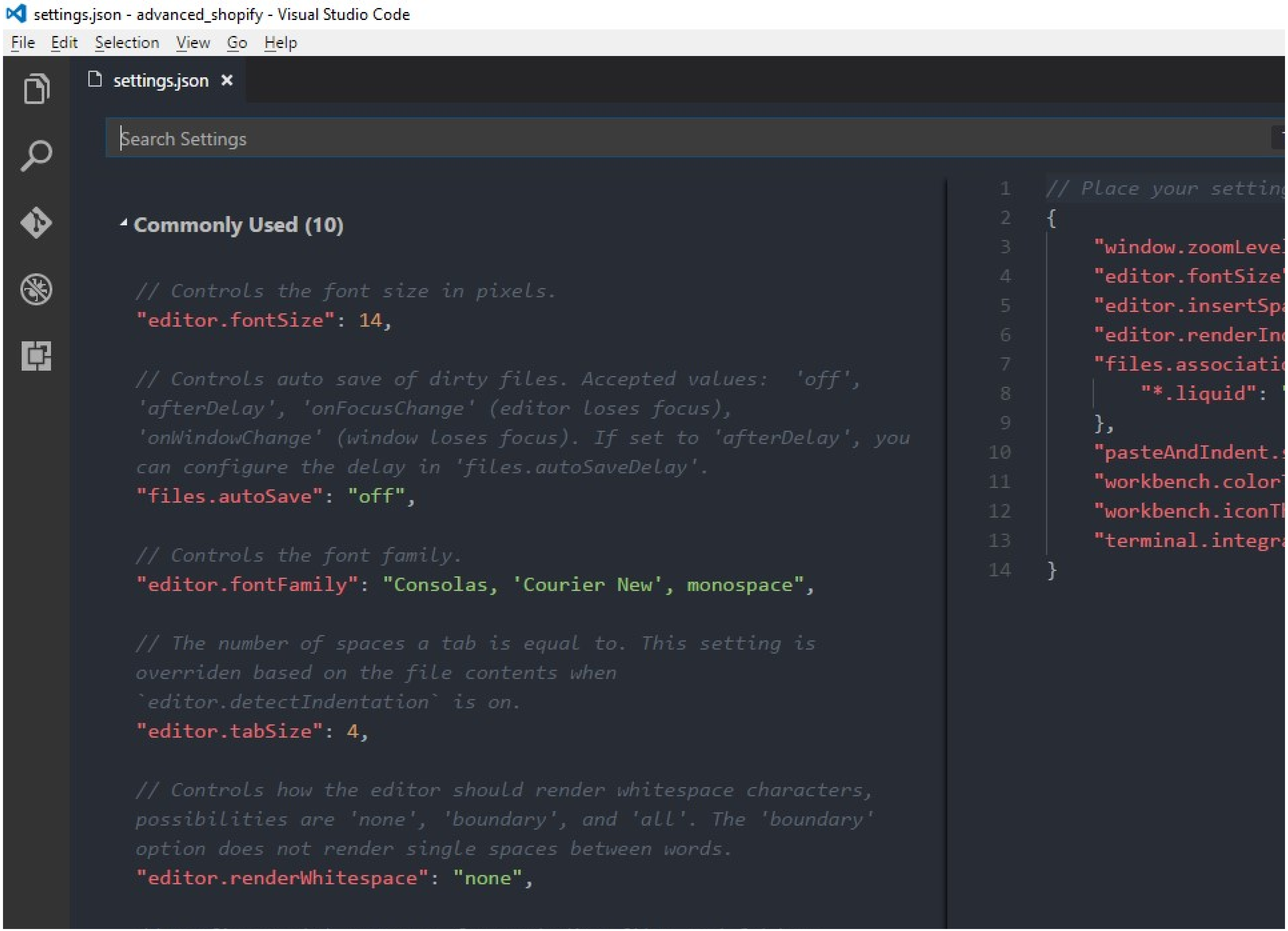
Install Git from https://git-scm.com/download/win.

Then open Visual Studio Code and open the command palette using `Ctrl` + `Shift` + `P`. Then type "open user setting", and then select "Open User Settings" from the drop down menu.



advanced_shopify - Visual Studio Code
File Edit Selection View Go Help
EXPLORER
OPEN EDITORS
ADVANCED_SHOPIFY
basic-theme-00
bstheme-html
bstheme-liq
assets
config
layout
locales
sections
snippets
templates
mytheme
timber-latest
venture
>open user
Preferences: Open User Settings
Preferences: Open User Snippets
Show All Commands Ctrl+Shift+P
Go to File Ctrl+P
Find in Files Ctrl+Shift+F
Start Debugging F5
Toggle Terminal Ctrl+`

Then this tab will open up with default settings on left and your settings on the right:


settings.json - advanced_shopify - Visual Studio Code
File Edit Selection View Go Help
settings.json ×
Search Settings
Commonly Used (10)

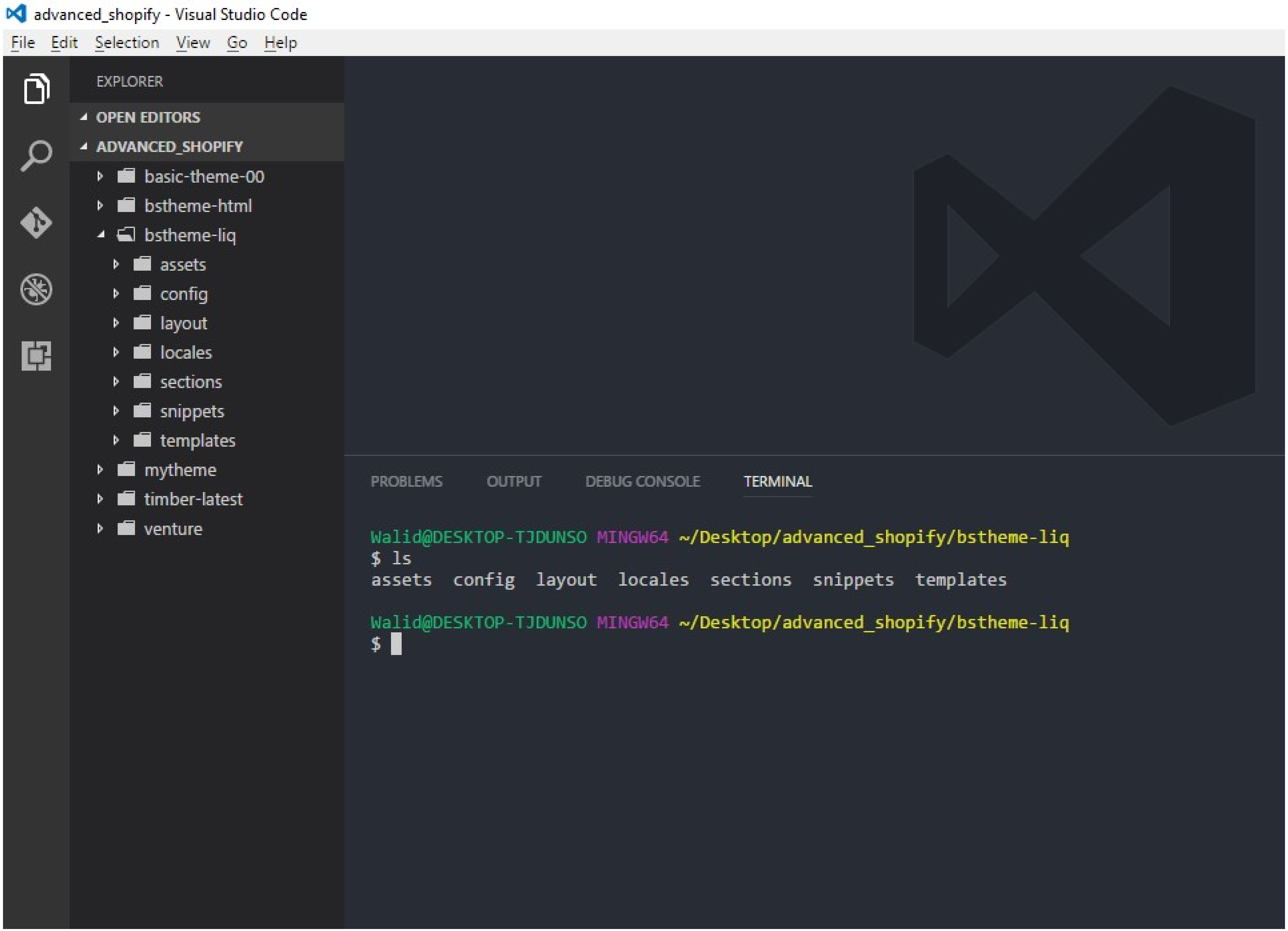
```
// Controls the font size in pixels.
"editor.fontSize": 14,

// Controls auto save of dirty files. Accepted values:  'off',
'afterDelay', 'onFocusChange' (editor loses focus),
'onWindowChange' (window loses focus). If set to 'afterDelay', you
can configure the delay in 'files.autoSaveDelay'.
"files.autoSave": "off",

// Controls the font family.
"editor.fontFamily": "Consolas, 'Courier New', monospace",

// The number of spaces a tab is equal to. This setting is
overriden based on the file contents when
`editor.detectIndentation` is on.
"editor.tabSize": 4,

// Controls how the editor should render whitespace characters,
possibilities are 'none', 'boundary', and 'all'. The 'boundary'
option does not render single spaces between words.
"editor.renderWhitespace": "none",
```

```
// Place your settin
{
    "window.zoomLeve
    "editor.fontSize
    "editor.insertSp
    "editor.renderIn
    "files.associati
        "*.liquid": 
    },
    "pasteAndIndent.
    "workbench.color
    "workbench.iconT
    "terminal.integr
}
```

Now copy this line of code to your own settings page (the pane on the right hand side) and save - `"terminal.integrated.shell.windows": "C:\\Program Files\\Git\\bin\\bash.exe"`

Note: `"C:\\Program Files\Git\bin\bash.exe"` is the path where the `bash.exe` file is located from the Git installation. If you are using the Windows Subsystem for Linux (WSL) Bash shell, the path would be `"C:\Windows\System32\bash.exe"`

Now press `Ctrl` + `` ` `` to open up the terminal from Visual Studio Code. And you will have Bash -



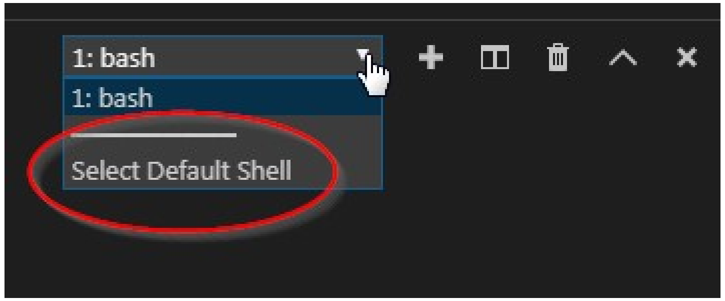
```
Walid@DESKTOP-TJDUNSO MINGW64 ~/Desktop/advanced_shopify/bstheme-liq
$ ls
assets  config  layout  locales  sections  snippets  templates

Walid@DESKTOP-TJDUNSO MINGW64 ~/Desktop/advanced_shopify/bstheme-liq
$ 
```

answered Mar 5 '17 at 9:51

[Mahade Walid](#)Mahade Walid

7,3345 gold badges14 silver badges24 bronze badges

Updated: Newer versions of Visual Studio Code have the *Select Default Shell* command in the terminal pull down menu:



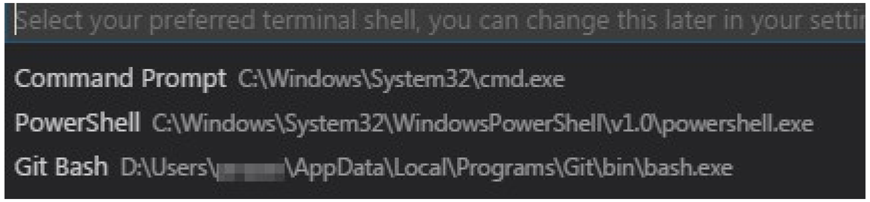
Remember that it just lists the shells that are in your %PATH% environment variable. For shells that aren't in your path, see other answers.

Extra tip: when you start bash it will just execute `.bashrc`, if you have initialization commands in `.bash_profile` you must copy it to `.bashrc`. It's essential for using [Conda enviroments](#) in Git Bash.

The easiest way now (at least from Visual Studio Code 1.22 on) is to type `Shift` + `Ctrl` + `P` to open the *Command Palette* and type:

```
Select Default Shell
```

Now you can easily select your preferred shell between the ones **found in your path**:



Select your preferred terminal shell, you can change this later in your setti

Command Prompt C:\Windows\System32\cmd.exe

PowerShell C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe

Git Bash D:\Users\[redacted]\AppData\Local\Programs\Git\bin\bash.exe

For shells that aren't in your %PATH%, see the other answers.

See the complete Visual Studio Code shell reference. There's lot of meaty stuff.

answered May 3 '18 at 16:17

nevesneves

18.5k15 gold badges102 silver badges130 bronze badges

Press and hold `Ctrl` + ` to open the terminal. Inside the terminal, type `bash` to use Git Bash in the terminal. Note: Make sure you have Git Bash installed on your machine.

If you wish to use PowerShell again, just type `powershell` in the terminal. To use the Windows command line, type `cmd` in the terminal.

The setting you choose will be used as your default.

answered Sep 6 '17 at 14:03

Isaac SekamatteIsaac Sekamatte

3,2951 gold badge25 silver badges31 bronze badges

For me this is the only combination worked!

```
"terminal.integrated.shell.windows": "C:\\Program Files\\Git\\git-cmd.exe",
"terminal.integrated.shellArgs.windows": [
  "--command=usr/bin/bash.exe",
  "-l",
  "-i"
]
```

With git-bash.exe as the `...shell.windows`, every time the bash was opening outside VS!!

Thank God it worked finally!! Else, I was planning to wipe out VS completely and reinstall it (making me to reinstall all my extensions and redo my customizations!)

answered Jun 3 '18 at 12:26

Things has been a little bit changed due to the latest updates on Visual Studio Code. The following steps work for me.

1. Press `Ctrl` + `Shift` + `P` to open the Visual Studio Code command palate.

2. Type `>preferences: Open Settings (JSON)` in the text area.

3. Add the following lines at the end of the JSON file which is displayed in your right hand pane.

   ```
   "terminal.integrated.shell.windows": "C:\\Program Files\\Git\\bin\\bash.exe"
   ```

4. Close and reopen your Visual Studio Code instance.

answered Nov 26 '18 at 16:08

I followed this tutorial from Paul DeCarlo to use the Bash from the Windows Subsystem for Linux (WSL) instead of what comes with Git Bash for Windows. They are the same steps as above in the answer, but use the below in your User Settings instead.

```
"terminal.integrated.shell.windows": "C:\\Windows\\sysnative\\bash.exe",
```

This worked for me the first time... which is rare for this stuff.

answered Feb 7 '18 at 3:06

Jay KilleenJay Killeen

2,3045 gold badges29 silver badges54 bronze badges

This, at least for me, will make Visual Studio Code open a new Bash window as an external terminal.

If you want the integrated environment you need to point to the `sh.exe` file inside the `bin` folder of your Git installation.

So the configuration should say `C:\\<my-git-install>\\bin\\sh.exe`.

answered Dec 8 '17 at 12:07

ChristerChrister

1,60316 silver badges33 bronze badges

It depends on whether you have installed Git Bash in the current user only or all users:

If it is installed on all users then put `"terminal.integrated.shell.windows": "C:\\Program Files\\Git\\bin\\bash.exe"` in your User Settings (`Ctrl` + `Comma`).

If it is installed on only the current user then put `"terminal.integrated.shell.windows": "C:\\Users\\<name of your user>\\AppData\\Local\\Programs\\Git\\bin\\bash.exe"` in your User Settings (`Ctrl` + `Comma`).

If the methods listed above do not work then you should try Christer's solution which says -

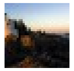
> If you want the integrated environment you need to point to the `sh.exe` file inside the `bin` folder of your Git

installation.

So the configuration should say `C:\\<my-git-install>\\bin\\sh.exe`.

Note: The sh.exe and bash.exe appear completely same to me. There should be no difference between them.

answered Jan 11 '18 at 14:51

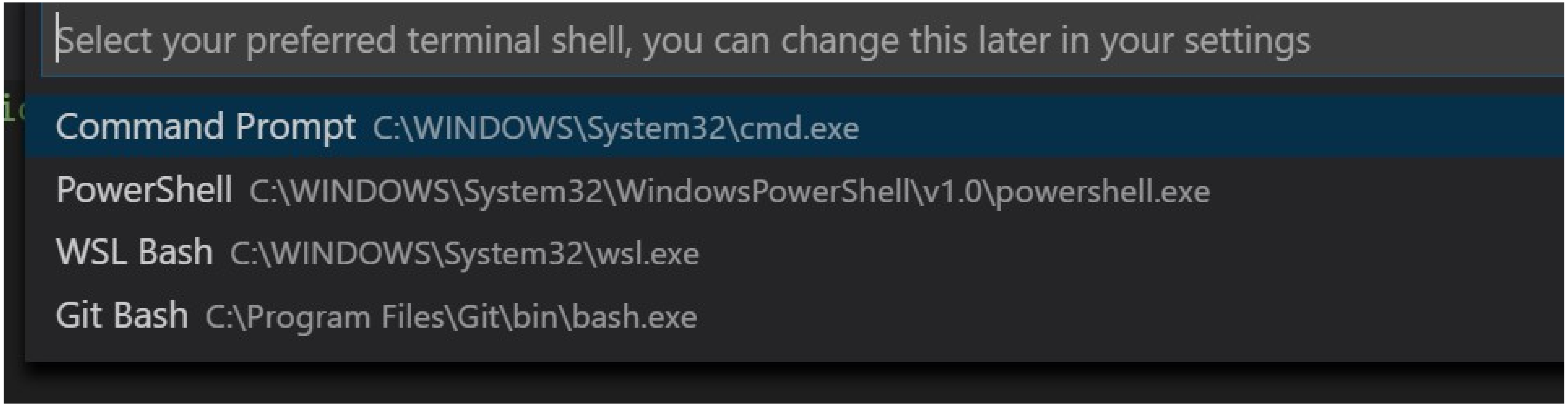


ShauryaShaurya

1013 silver badges6 bronze badges

This answer is similar to the top voted answer, but with an important distinction: a lot of the previous answers on this question focus on running **Git Bash** while my answer focuses on running **WSL Bash**.

1. Enable Windows Subsystem for Linux on your Windows 10 machine.
2. Open Visual Studio Code and press and hold `Ctrl + `` ` `` to open the terminal.
3. Open the command palette using `Ctrl + Shift + P`.
4. Type - `Select Default Shell`.
5. Select `WSL Bash` (NOT `Git Bash`) from the options.



Select your preferred terminal shell, you can change this later in your settings

Command Prompt C:\WINDOWS\System32\cmd.exe

PowerShell C:\WINDOWS\System32\WindowsPowerShell\v1.0\powershell.exe

WSL Bash C:\WINDOWS\System32\wsl.exe

Git Bash C:\Program Files\Git\bin\bash.exe

5. Click on the `+` icon in the terminal window. The new terminal now will be a WSL Bash terminal!

answered May 20 '19 at 16:59

therobinkimtherobinkim

2,22010 silver badges19 bronze badges

Add the `Git\bin` directory to the `Path` environment variable. The directory is `%ProgramFiles%\Git\bin` by default. By this way you can access Git Bash with simply *typing* `bash` in every terminal including the integrated terminal of Visual Studio Code.

How to set the path and environment variables in Windows

answered Jun 23 '18 at 9:42

I had already set up lots of **`conda`** environments on `WSL` (Bash on Ubuntu on Windows), so I wanted to use the same Bash installation on Visual Studio Code.

To do that, I just had to specify the path of this particular Bash executable (***instead** of the `Git-Bash`*) on Visual Studio Code's settings:

```
"terminal.integrated.shell.windows": "C:\\Windows\\System32\\bash.exe"
```

PS: to make sure where your Ubuntu on Bash executable is installed on your Windows machine, open the `Command prompt` (search: `cmd`) and run:

```
where bash.exe
```

answered Nov 26 '18 at 10:58

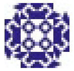
dvitsiosdvitsios

4302 silver badges7 bronze badges

Latest VS code :

- if you can't see the settings.json, go to menu File -> Preferences -> Settings (or press on `Ctrl+,`)
- Settings appear, see two tabs User (selected by default) and Workspace. Go to User -> Features -> Terminal
- Terminal section appear, see link `edit in settings.json`. Click and add `"terminal.integrated.shell.windows": "C:\\Program Files\\Git\\bin\\bash.exe",`
- Save and Restart VS code.

Bash terminal will reflect on the terminal.



answered Dec 11 '19 at 2:12

RijoRijo

2,5353 gold badges18 silver badges36 bronze badges

I happen to be consulting for a Fortune 500 company and it's sadly **Windows 7** and no administrator privileges. Thus Node.js, Npm, Visual Studio Code, etc.. were pushed to my machine - I cannot change a lot, etc...

For this computer running Windows 7:

Below are my new settings. The one not working is commented out.

```
{
    "update.channel": "none",
    "terminal.integrated.shell.windows": "C:\\Program Files\\Git\\bin\\bash.exe"

}
```

answered Apr 5 '18 at 22:42

Tom StickelTom Stickel

16.2k6 gold badges100 silver badges107 bronze badges

I tried the above answers as of vscode 1.42.1 and they worked inasmuch as to get me a git bash terminal. So, bottom line this setting works just for opening a bash shell from terminal:

```
"terminal.integrated.shell.windows": "C:\\Program Files\\Git\\bin\\bash.exe"
```

However it has the unwanted side effect of also being the shell used to build things and that breaks the MS C++ chain because the `\` character used for path separator is understood by bash as an escape character. The complete fix for me then required me to add this extra variable, setting it to powershell:

```
"terminal.integrated.automationShell.windows": "C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe"
```

Now, I can have my bash terminal and `Ctrl-Shift-B` or `F5` work without problems.

Oh, and as other posters mentioned, the source for this information is VSCode's documentation.

answered Mar 8 '20 at 4:09

Coffee_fanCoffee_fan

3984 silver badges9 bronze badges

What about detached or unrelated shells and `code [args]` support?

While other answers talk about how to configure and use the VScode integrated WSL `bash` terminal support, they don't solve the problem of "detached shells": shells which were not launched from within VScode, or which somehow get "disconnected" from the VScode server instance associated with the IDE.

Such shells can give errors like:

```
Command is only available in WSL or inside a Visual Studio Code terminal.
```

or...

```
Unable to connect to VS Code server. Error in request
```

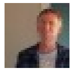
Here's a script which makes it easy to solve this problem.

I use this daily to connect shells in a `tmux` session with a specific VScode server instance, or to fix an integrated shell that's become detached from its hosting IDE.

```
#!/bin/bash
```

```
function errExit {
    echo "ERROR: $@" >&2
    exit 1
}

[[ -S $VSCODE_IPC_HOOK_CLI ]] || errExit "VSCODE_IPC_HOOK_CLI not defined or not a pipe [$VSCODE_IPC_HOOK_CLI]"
if [[ $(which code) != *vscode-server* ]]; then
    errExit "The 'code' command doesn't refer to something under .vscode-server: $(type -a code)"
fi
cat <<EOF >.codesrv-connect
# Temp file created by $(which codesrv-connect): source this into your working shell like '. .codesrv-connect'
# ( git hint: add ".codesrv-connect" to .gitignore )
#
cd "$PWD"
if ! test -S "$VSCODE_IPC_HOOK_CLI"; then
    echo "ERROR: $VSCODE_IPC_HOOK_CLI not a socket. Dead session."
else
    export VSCODE_IPC_HOOK_CLI="$VSCODE_IPC_HOOK_CLI"
    alias code=$(which code)
    echo "Done: the 'code' command will talk to socket \"$VSCODE_IPC_HOOK_CLI\" now."
    echo "You can delete .codesrv-connect when the vscode server context dies, or reuse it in other shells until then."
fi
EOF

echo "# OK: run this to connect to vscode server in a destination shell:"
echo ". $PWD/.codesrv-connect"
```

answered Mar 31 '20 at 11:40



StabledogStabledog

2,7511 gold badge27 silver badges37 bronze badges

If you already have **"bash", "powershell"** and **"cmd"** CLI's and have correct path settings then switching from one CLI to another can done by the following ways.

> `Ctrl` + `'` : Opens the terminal window with default CLI.
>
> bash + `enter` : Switch from your default/current CLI to **bash** CLI.
>
> powershell + `enter` : Switch from your default/current CLI to **powershell** CLI.
>
> cmd + `enter` : Switch from your default/current CLI to **cmd** CLI.

VS Code Version I'm using is 1.45.0

answered May 13 '20 at 17:24

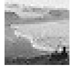
SridharKrithaSridharKritha

4,6832 gold badges33 silver badges32 bronze badges

For **scoop** users:

```
"terminal.integrated.shell.windows": "C:\\Users\\[YOUR-NAME]\\scoop\\apps\\git\\current\\usr\\bin\\bash.exe",
"terminal.integrated.shellArgs.windows": [
  "-l",
  "-i"
],
```

answered Jan 13 at 16:27



JacobJacob

1,72410 silver badges16 bronze badges

Not the answer you're looking for? Browse other questions tagged visual-studio-code git-bash vscode-settings or ask your own question.