WSL & Lando

An intro to WSL and an intro to Lando with extra tips and tricks.

DrupalACT Meetup 2023/07

Joshua Graham
Technical Developer
Digital and Emerging Technology

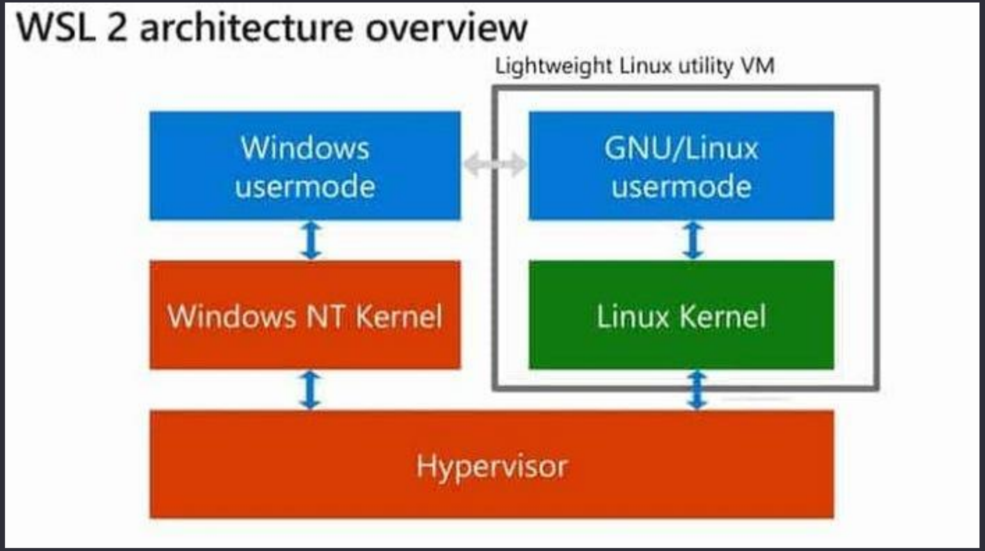
The better the question. The better the answer.
The better the world works.

EY
Building a better
working world

Windows Subsystem for Linux (WSL)

What is WSL [2]?

WSL is an optimised, lightweight, integrated, virtualised developer-oriented Linux environment that "runs" inside windows



Okay, I use windows, why should I use WSL?

For *speed !* We usually develop our Drupal websites in a Linux environment, believe or not.

When we use the same host OS filesystem as the Docker containers – we get a significant speed increase! Easy access to Linux tools is also plus.

https://www.urtech.ca/2019/09/solved-everything-you-need-to-know-about-the-new-windows-subsystem-for-linux-version-2/

Docker - WSL vs Hyper V ?

WSL is the better option in 99.99% or use cases (unless you aren't doing Drupal work?)

- Speed - Docker starts up quick + Filesystem access is faster (for Linux containers).
- More container resources – Under Hyper-V it was a set limit, with WSL more memory can be dynamically used.
- Ability to use common Linux tools - instead of installing the tools on every container.
- Runs on Windows 10 Home edition
- Con: File access to the WSL files from windows is slower - best to use software with WSL extensions or native WSL.
- Con: Limited functionally of Windowed Linux programs.
- Con: Sometimes you need to restart the WSL & Docker process due to memory bugs.

https://docs.docker.com/desktop/wsl/
https://superuser.com/questions/1561465/docker-on-hyper-v-vs-wsl-2

Docker - WSL vs Hyper V Backend?

Hyper-V:

- Moderate filesystem speed if the files must be in windows and the amount of files is small.
- Networking can be buggy for complex applications in WSL - VPNs and Peer-Peer2P software.
- Networking is piped differently - Firewall might not intrude in Hyper-V when it does in WSL.
- Mounting files works differently – Uses windows file sharing service, firewall might still block.
- Windows containers are reportedly better for Hyper-V backend and high I/O workloads where CPU workloads are more suited to WSL*.

* According to Bing Chat GPT

https://www.docker.com/blog/docker-hearts-wsl-2/

Docker - WSL vs Hyper V ? - Metrics

How long does a Drush clear cache “drush cr” take?

Project Type	Docker Type	Duration
GovCMS SaaS	Hyper-V	11 seconds
GovCMS SaaS	WSL	7 seconds
Lando	Hyper-V	63 seconds
Lando	WSL	7 seconds

For Lando projects, that a 9x increase in performance. NPM/yarn projects will also see a similar increase.

Tested with over 20 iterations, tested in Windows Virtual Machine

Docker - WSL - Installation

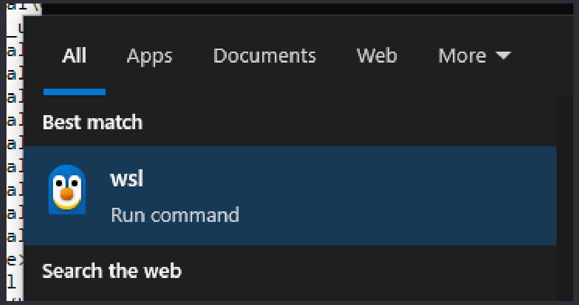
Requirements: Windows 10 version 2004 and higher (Build 19041 and higher) or Windows 11

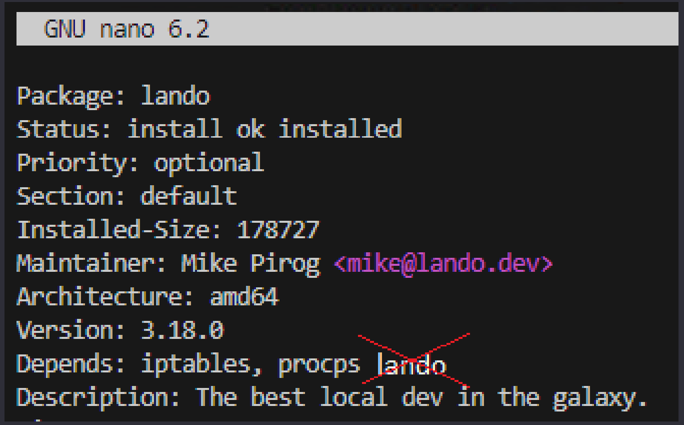
1. Run this command as Administrator on in the command prompt (cmd.exe)
   - `wsl -install` - it will download and install Ubuntu for WSL.
2. Install Docker
   - Ensure "Use WSL 2 based engine" and "Use Docker compose v2" are ticked (required for Lando/GovCMS).
   - Ensure, "Resources" => WSL integration => "Enable integration " and "Ubuntu" distribution is ticked/selected.
3. Check Windows WSL settings.
   - Command line (cmd.exe)
   - `wsl --set-default-version 2`, `wsl --set-version Ubuntu 2` and `wsl -set-default Ubuntu`.

Lando - Installation (install in WSL, not Windows!)

To install Lando (not required if only using GovCMS projects):

1. Open a WSL terminal by typing “wsl” into your start menu.
2. Run `wget https://files.lando.dev/installer/lando-x64-stable.deb`
3. Run `sudo dpkg -i --ignore-depends=docker-ce lando-x64-stable.deb`
4. (slightly technical), Open an administrative text-editor and remove remove ‘docker-ce’ from the lando packages dependency key. (fixes future package manager updates)
   - `sudo nano /var/lib/dpkg/status`
   - Crtl + W (to search), Delete text and save!
5. Run `exit` (done, use the same process for updates to Lando too!)





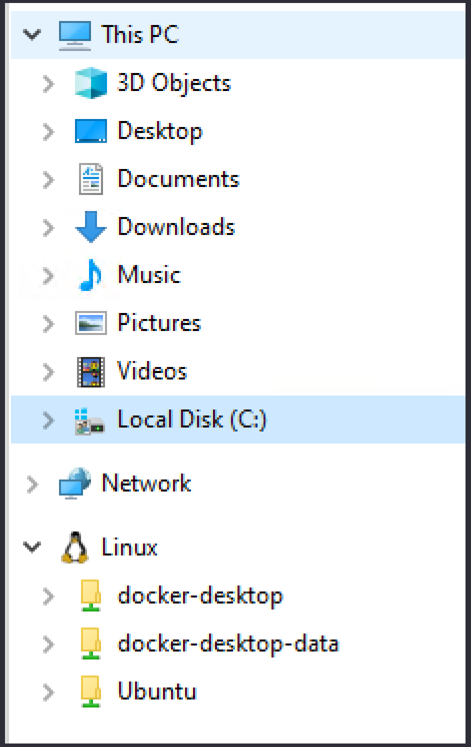
```
 GNU nano 6.2

Package: lando
Status: install ok installed
Priority: optional
Section: default
Installed-Size: 178727
Maintainer: Mike Pirog <mike@lando.dev>
Architecture: amd64
Version: 3.18.0
Depends: iptables, procps ~~lando~~
Description: The best local dev in the galaxy.
```

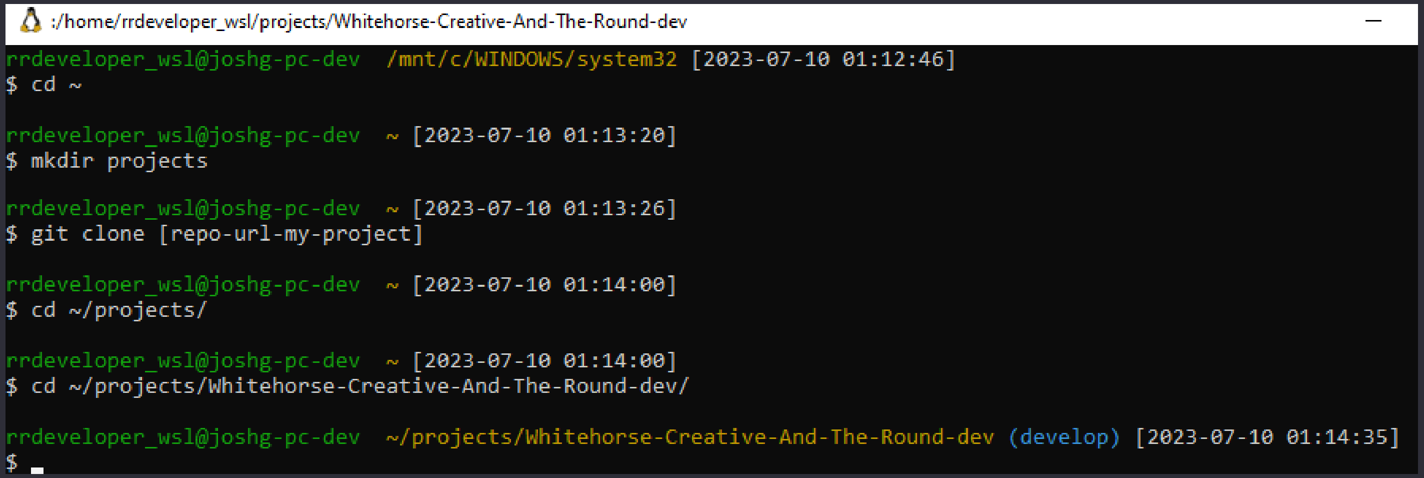
https://docs.lando.dev/guides/setup-lando-on-windows-with-wsl-2.html

How to edit files in WSL

First, we must add the projects files to the WSL file system (Ubuntu):

1. Open “wsl” (from start menu)
2. Add your projects files you preferred location in WSL (I use ‘~/projects’) or copy/paste!
    1. `cd ~` (~ is a shorthand for /home/[username])
    2. `mkdir projects`
    3. `git clone [git-repo-url-my-project]`



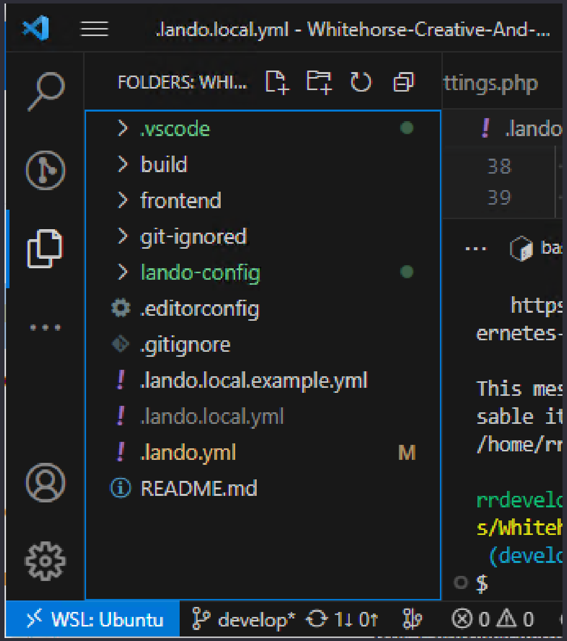

:/home/rrdeveloper_wsl/projects/Whitehorse-Creative-And-The-Round-dev

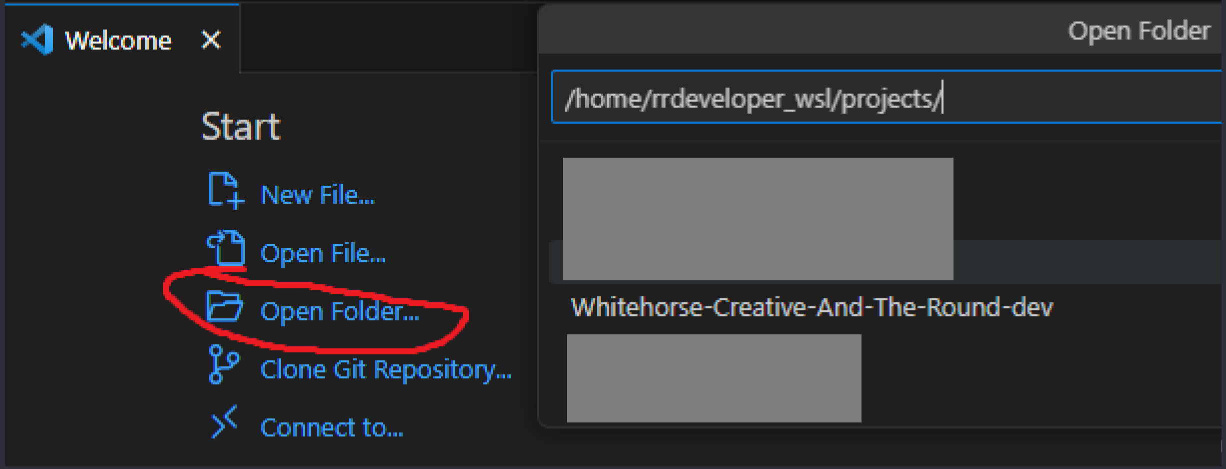
```
rrdeveloper_wsl@joshg-pc-dev  /mnt/c/WINDOWS/system32 [2023-07-10 01:12:46]
$ cd ~

rrdeveloper_wsl@joshg-pc-dev  ~ [2023-07-10 01:13:20]
$ mkdir projects

rrdeveloper_wsl@joshg-pc-dev  ~ [2023-07-10 01:13:26]
$ git clone [repo-url-my-project]

rrdeveloper_wsl@joshg-pc-dev  ~ [2023-07-10 01:14:00]
$ cd ~/projects/

rrdeveloper_wsl@joshg-pc-dev  ~ [2023-07-10 01:14:00]
$ cd ~/projects/Whitehorse-Creative-And-The-Round-dev/

rrdeveloper_wsl@joshg-pc-dev  ~/projects/Whitehorse-Creative-And-The-Round-dev (develop) [2023-07-10 01:14:35]
$ 
```

How to edit files in WSL

Now to edit, the easiest way via free software is via Microsoft Visual Studio Code, since it can inside edit in WSL (so is fast):

1. Install VS Code
2. Install the WSL extension
3. Open VS code
    1. Click WSL icon (bottom left)
    2. Click “Connect to WSL”
    3. Click “Open Folder”
    4. Browse to your created project earlier.
4. Done - Easily browse and edit files!





/home/rrdeveloper_wsl/projects/

[redacted]

Whitehorse-Creative-And-The-Round-dev

[redacted]

How to edit files in WSL - other options

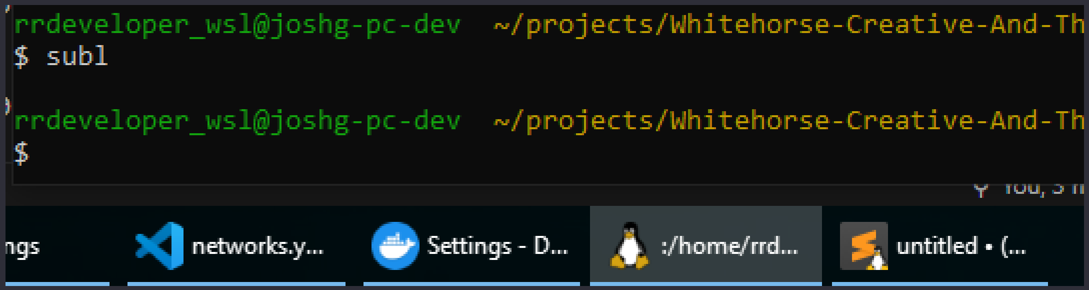
Other options besides native terminal command line tools inside (e.g. vim / nano etc.), include a windowed WSL programs nicknamed “WSLg”.

Some programs work Out of the Box after install e.g. Gnome text editor.

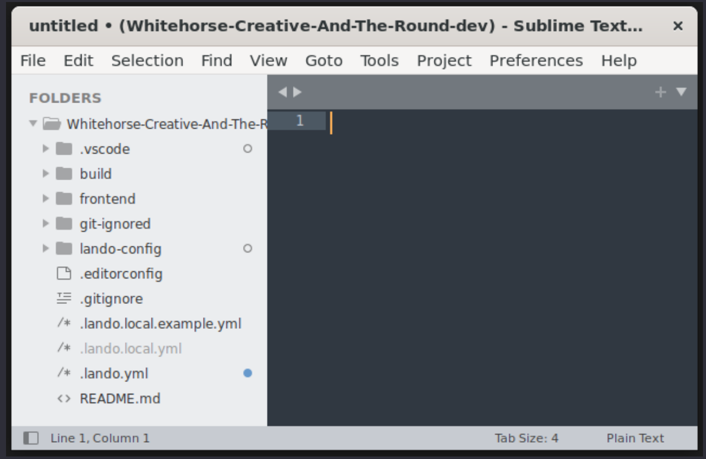
But others need a workaround

For Example for Sublime Text editor:
(YYMV though!)

# Add extra required libraries.
`sudo apt install ligl1`



```
rrdeveloper_wsl@joshg-pc-dev  ~/projects/Whitehorse-Creative-And-Th
$ subl

rrdeveloper_wsl@joshg-pc-dev  ~/projects/Whitehorse-Creative-And-Th
$
```

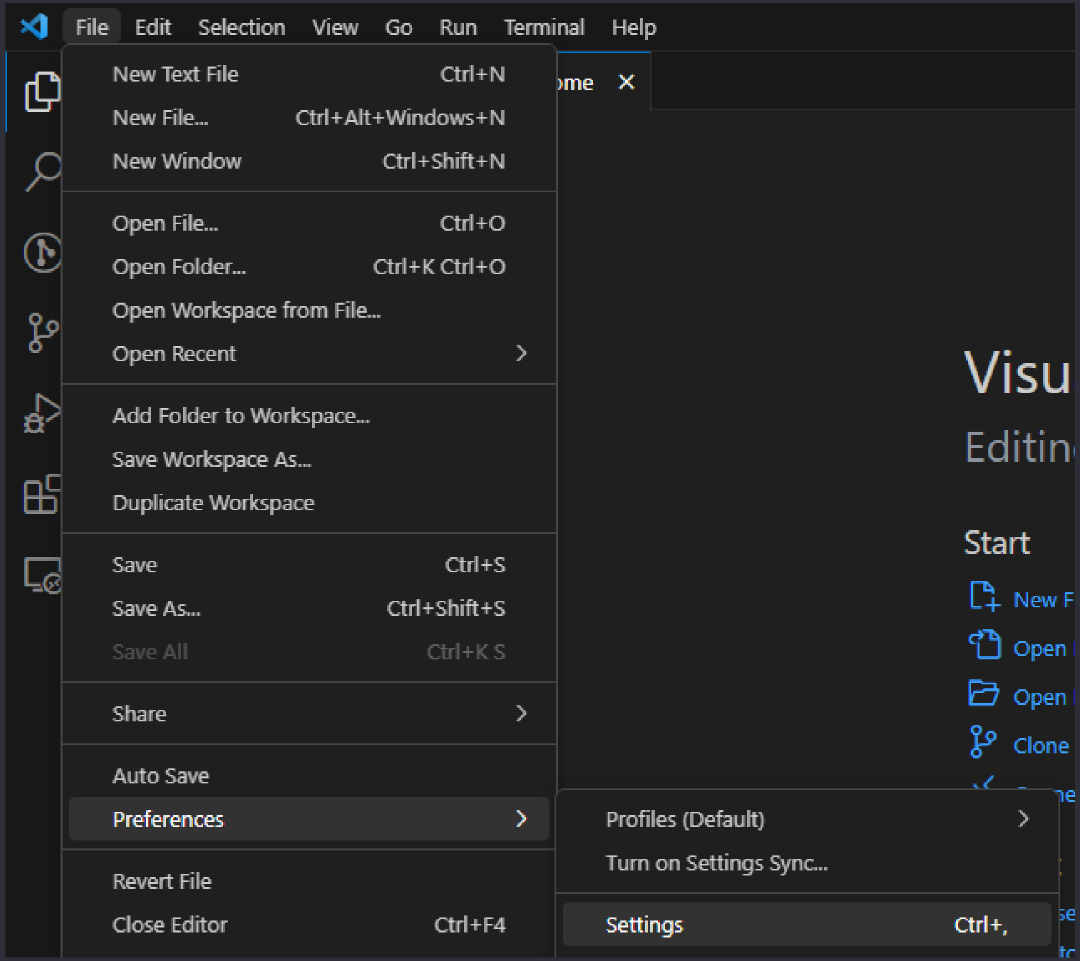


https://learn.microsoft.com/en-us/windows/wsl/tutorials/gui-apps
https://www.sublimetext.com/docs/linux_repositories.html

WSL & VS code – extra tip: file syntax highlight and login script

To add Highlighting for PHP files, perform the following steps:

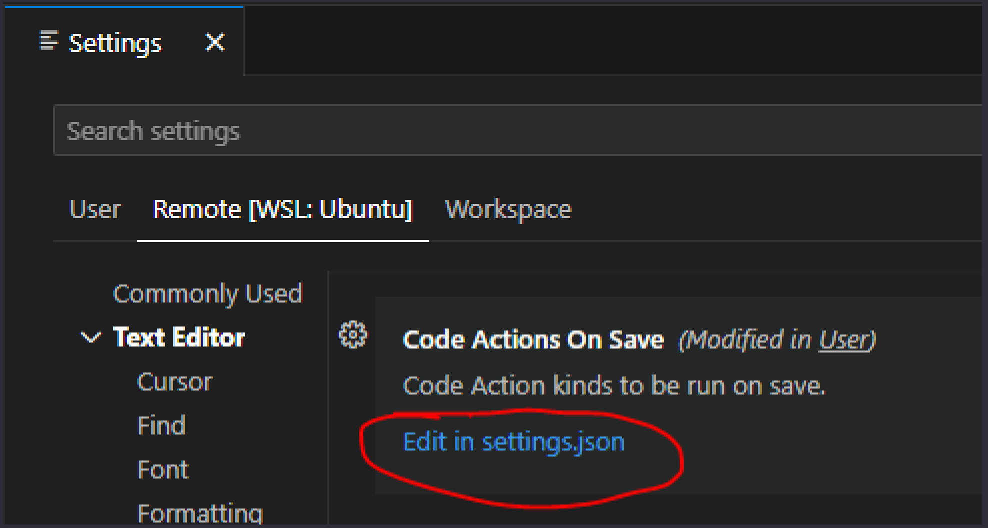
Open VS Code settings:



WSL & VS code - extra tip: file syntax highlight and login script

1. Select the “Remote [WSL: Ubuntu]” Tab.

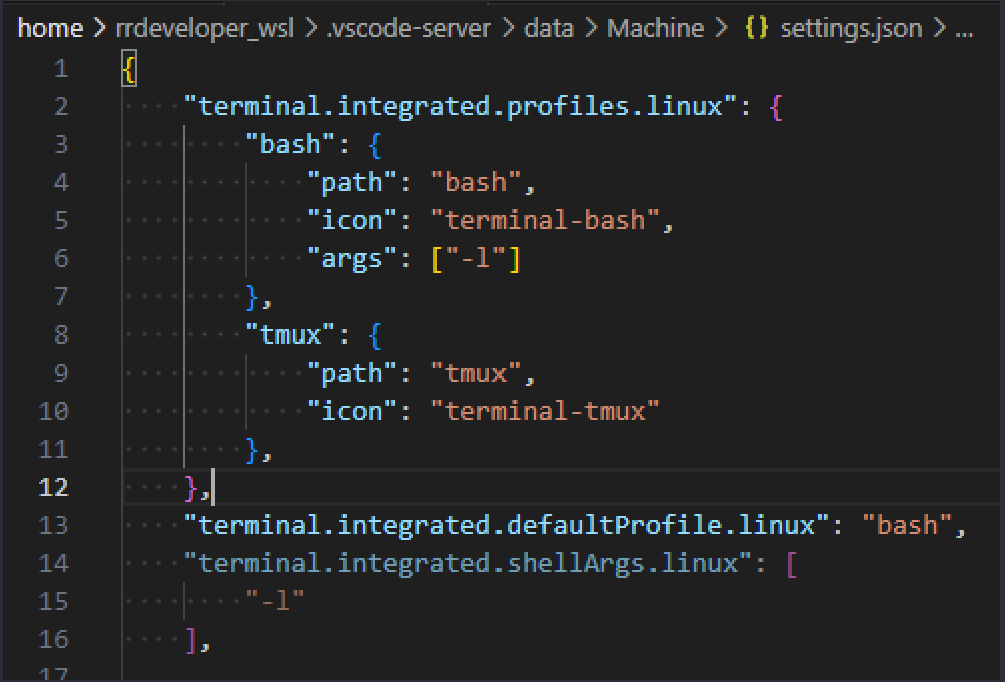
2. Scroll down until to see a setting that needs to be set in settings.json, then click the link below it to open the json file.



Code Actions On Save *(Modified in User)*

Code Action kinds to be run on save.

Edit in settings.json



home > rrdeveloper_wsl > .vscode-server > data > Machine > {} settings.json > ...

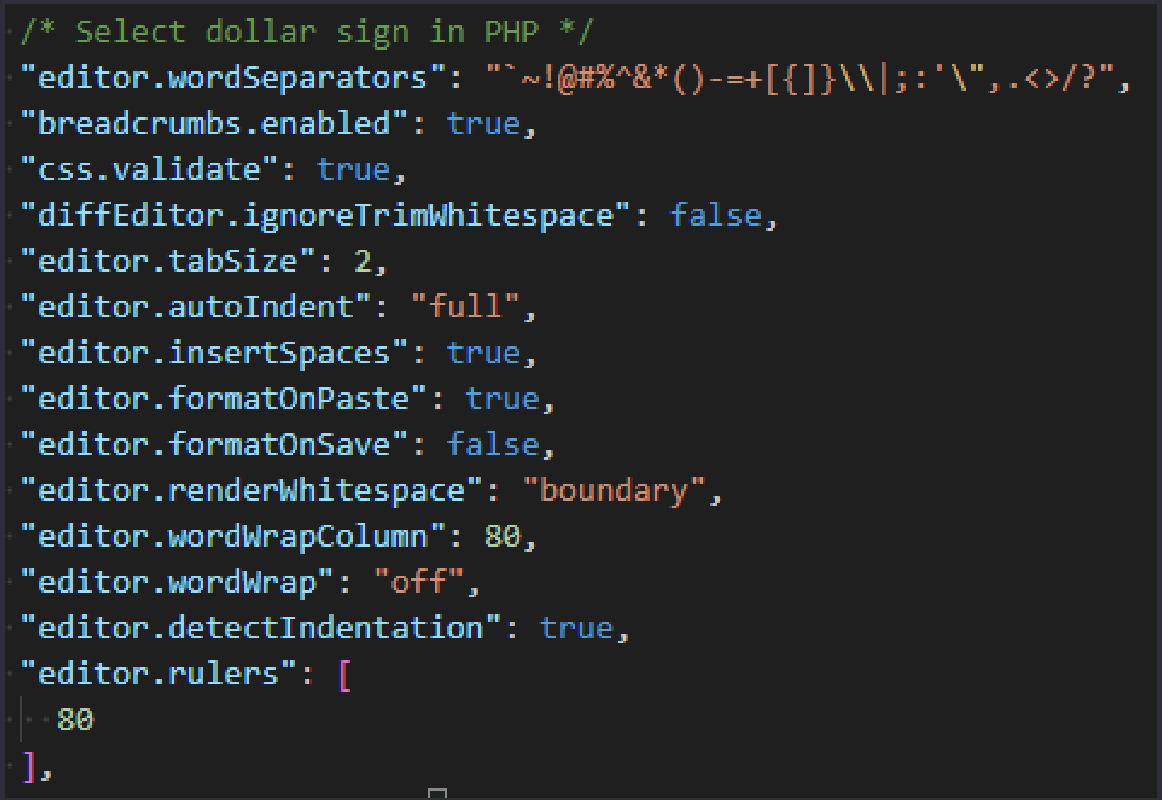
```
{
    "terminal.integrated.profiles.linux": {
        "bash": {
            "path": "bash",
            "icon": "terminal-bash",
            "args": ["-l"]
        },
        "tmux": {
            "path": "tmux",
            "icon": "terminal-tmux"
        },
    },
    "terminal.integrated.defaultProfile.linux": "bash",
    "terminal.integrated.shellArgs.linux": [
        "-l"
    ],
```

3. Add the setting on the right for to force a “login” flag for bash, so it runs “~/bash_profile”.

WSL & VS code – extra tip: file syntax highlight and login script

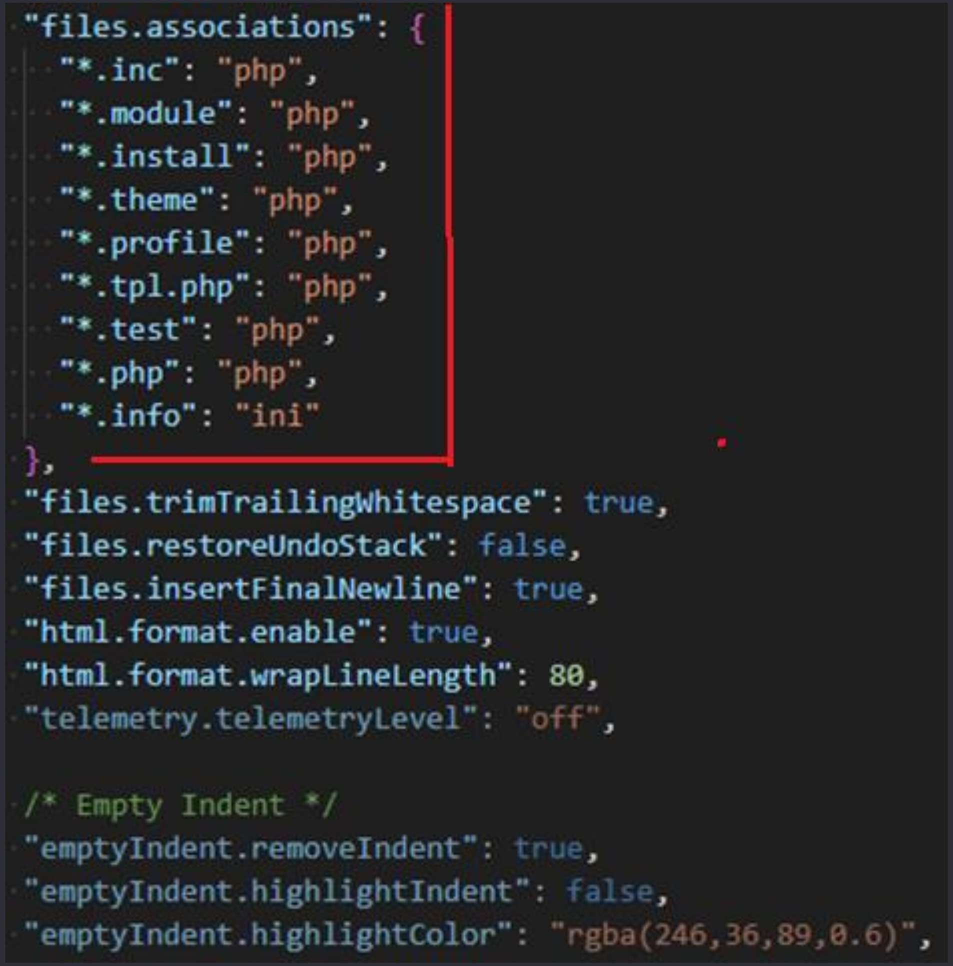
File highlighting can be set via more settings in the same file.

Twig VS code extension needs to be installed separately.



```
/* Select dollar sign in PHP */
"editor.wordSeparators": "`~!@#%^&*()-=+[{]}\\|;:'\",.<>/?",
"breadcrumbs.enabled": true,
"css.validate": true,
"diffEditor.ignoreTrimWhitespace": false,
"editor.tabSize": 2,
"editor.autoIndent": "full",
"editor.insertSpaces": true,
"editor.formatOnPaste": true,
"editor.formatOnSave": false,
"editor.renderWhitespace": "boundary",
"editor.wordWrapColumn": 80,
"editor.wordWrap": "off",
"editor.detectIndentation": true,
"editor.rulers": [
  80
],
```

https://www.drupal.org/docs/develop/development-tools/editors-and-ides/configuring-visual-studio-code
https://github.com/Microsoft/vscode/issues/2036

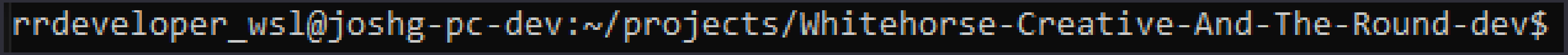


```
"files.associations": {
  "*.inc": "php",
  "*.module": "php",
  "*.install": "php",
  "*.theme": "php",
  "*.profile": "php",
  "*.tpl.php": "php",
  "*.test": "php",
  "*.php": "php",
  "*.info": "ini"
},
"files.trimTrailingWhitespace": true,
"files.restoreUndoStack": false,
"files.insertFinalNewline": true,
"html.format.enable": true,
"html.format.wrapLineLength": 80,
"telemetry.telemetryLevel": "off",

/* Empty Indent */
"emptyIndent.removeIndent": true,
"emptyIndent.highlightIndent": false,
"emptyIndent.highlightColor": "rgba(246,36,89,0.6)",
```

WSL & VS code - extra tip: file syntax highlight and login script

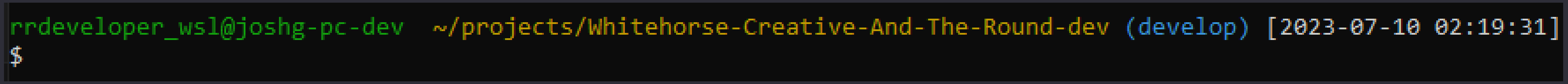
A login script is useful to show git branch or start a SSH agent.
(An SSH agent is required for SSH git repos to perform git actions in VS code’s integrated git plugin).

OLD prompt:

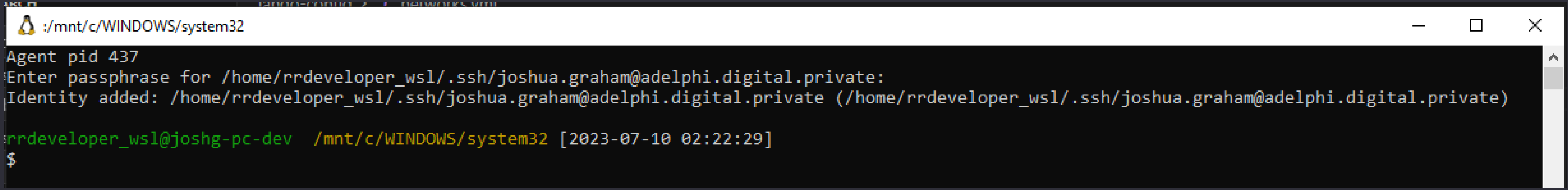


```
rrdeveloper_wsl@joshg-pc-dev:~/projects/Whitehorse-Creative-And-The-Round-dev$
```

New prompt:



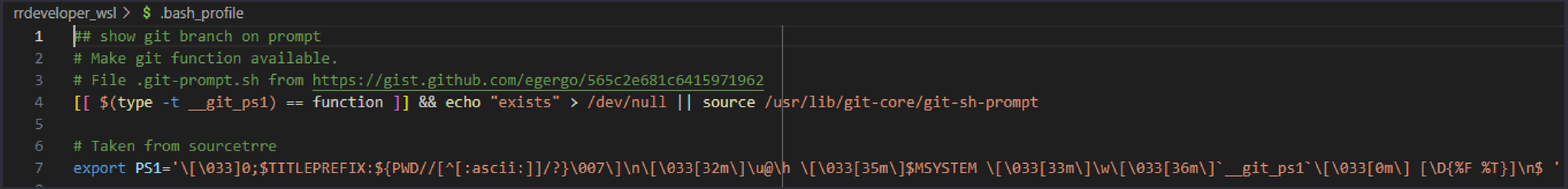
```
rrdeveloper_wsl@joshg-pc-dev  ~/projects/Whitehorse-Creative-And-The-Round-dev (develop) [2023-07-10 02:19:31]
$
```



```
Agent pid 437
Enter passphrase for /home/rrdeveloper_wsl/.ssh/joshua.graham@adelphi.digital.private:
Identity added: /home/rrdeveloper_wsl/.ssh/joshua.graham@adelphi.digital.private (/home/rrdeveloper_wsl/.ssh/joshua.graham@adelphi.digital.private)

rrdeveloper_wsl@joshg-pc-dev  /mnt/c/WINDOWS/system32 [2023-07-10 02:22:29]
$
```

WSL & VS code – extra tip: file syntax highlight and login script

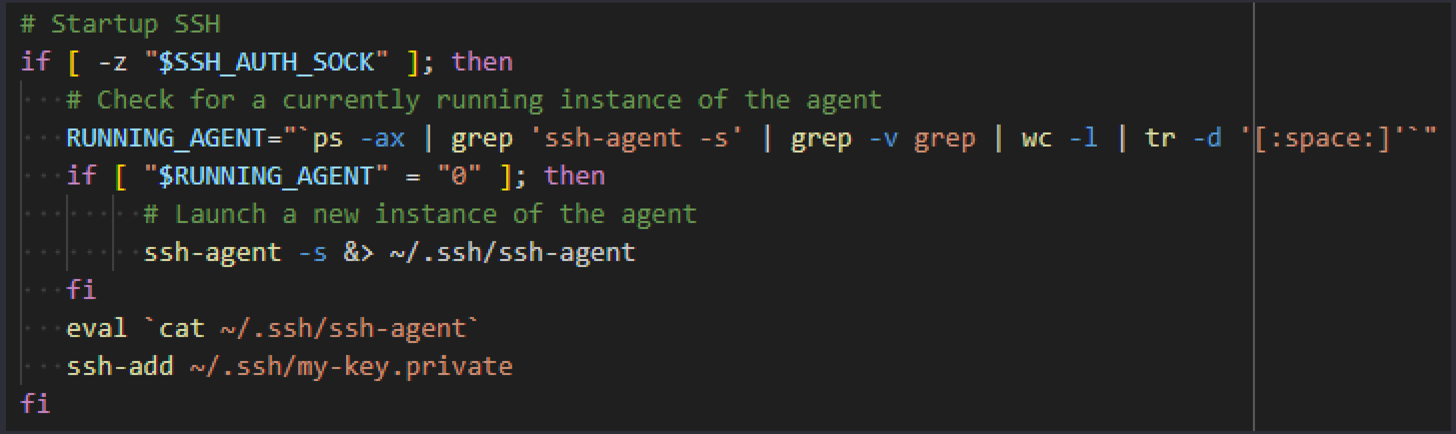
Setup the new prompt:



rrdeveloper_wsl > $ .bash_profile

```
## show git branch on prompt
# Make git function available.
# File .git-prompt.sh from https://gist.github.com/egergo/565c2e681c6415971962
[[ $(type -t __git_ps1) == function ]] && echo "exists" > /dev/null || source /usr/lib/git-core/git-sh-prompt

# Taken from sourcetrre
export PS1='\[\033]0;$TITLEPREFIX:${PWD//[^[:ascii:]]/?}\007\]\n\[\033[32m\]\u@\h \[\033[35m\]$MSYSTEM \[\033[33m\]\w\[\033[36m\]`__git_ps1`\[\033[0m\] [\D{%F %T}]\n$ '
```

SSH Agent (also won’t re-prompt for multiple VS code windows!

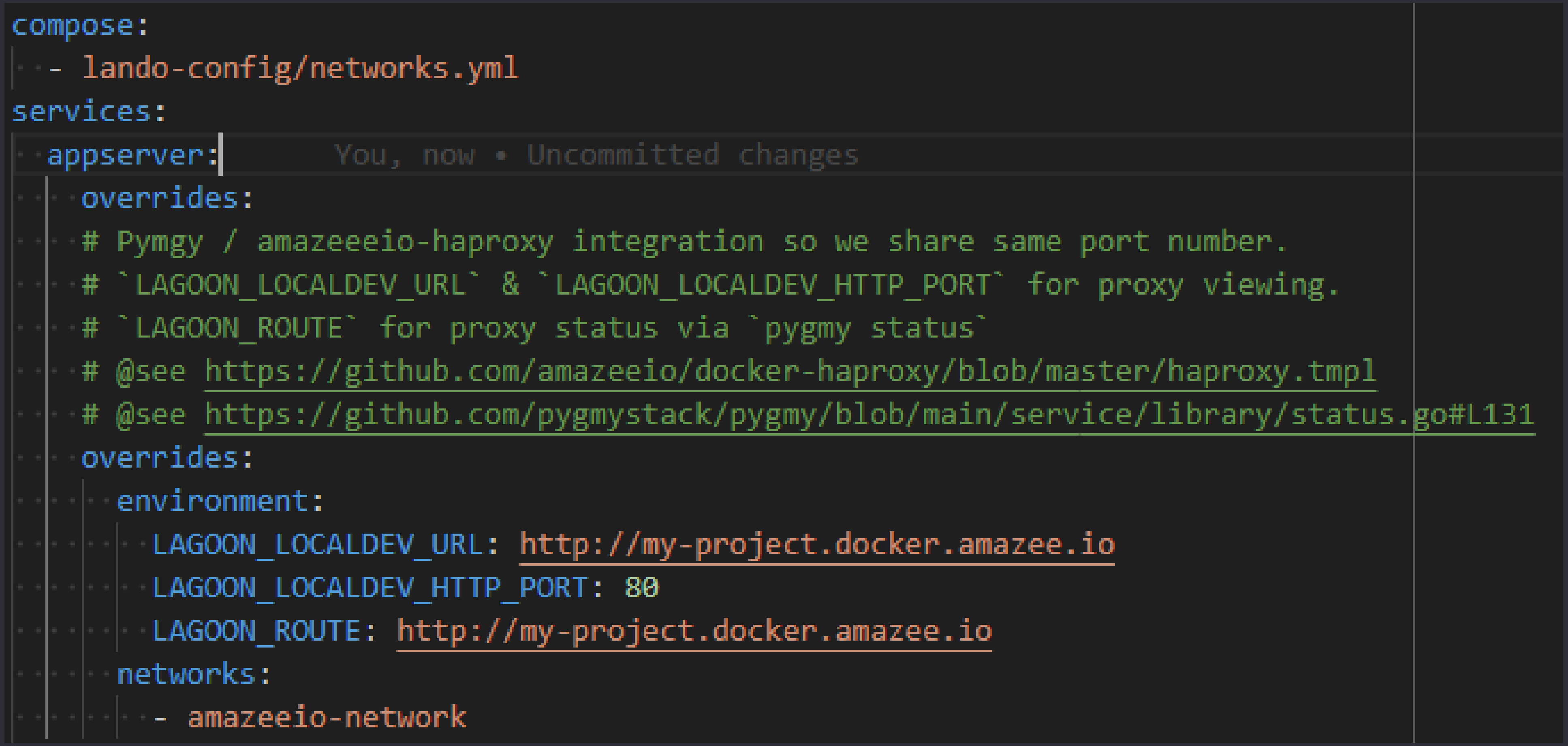


```
# Startup SSH
if [ -z "$SSH_AUTH_SOCK" ]; then
   # Check for a currently running instance of the agent
   RUNNING_AGENT="`ps -ax | grep 'ssh-agent -s' | grep -v grep | wc -l | tr -d '[:space:]'`"
   if [ "$RUNNING_AGENT" = "0" ]; then
        # Launch a new instance of the agent
        ssh-agent -s &> ~/.ssh/ssh-agent
   fi
   eval `cat ~/.ssh/ssh-agent`
   ssh-add ~/.ssh/my-key.private
fi
```

https://code.visualstudio.com/docs/remote/troubleshooting#_setting-up-the-ssh-agent

Using GovCMS reverse proxy - pygmy and Lando at the same time?

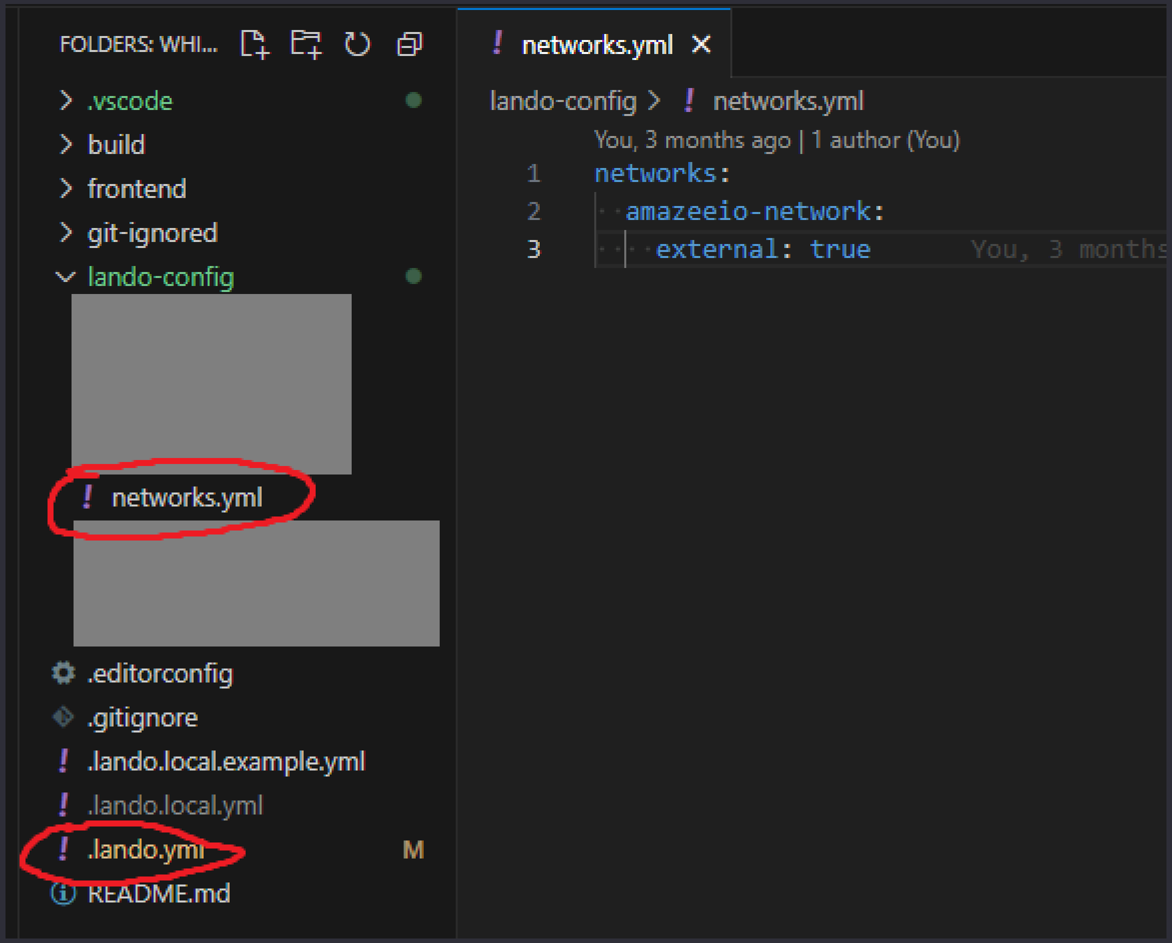
If want to reuse the pygmy reverse proxy for Lando so you don’t have to use different port numbers for different types of projects, you can make Lando containers use the proxy.



```yaml
compose:
  - lando-config/networks.yml
services:
  appserver:        You, now • Uncommitted changes
    overrides:
    # Pymgy / amazeeeio-haproxy integration so we share same port number.
    # `LAGOON_LOCALDEV_URL` & `LAGOON_LOCALDEV_HTTP_PORT` for proxy viewing.
    # `LAGOON_ROUTE` for proxy status via `pygmy status`
    # @see https://github.com/amazeeio/docker-haproxy/blob/master/haproxy.tmpl
    # @see https://github.com/pygmystack/pygmy/blob/main/service/library/status.go#L131
    overrides:
      environment:
        LAGOON_LOCALDEV_URL: http://my-project.docker.amazee.io
        LAGOON_LOCALDEV_HTTP_PORT: 80
        LAGOON_ROUTE: http://my-project.docker.amazee.io
      networks:
        - amazeeio-network
```

Using GovCMS reverse proxy - pygmy and Lando at the same time?

networks.yml contents.



.vscode
build
frontend
git-ignored
lando-config
[redacted]
networks.yml
[redacted]
.editorconfig
.gitignore
.lando.local.example.yml
.lando.local.yml
.lando.yml M
README.md

networks.yml

lando-config > networks.yml

You, 3 months ago | 1 author (You)

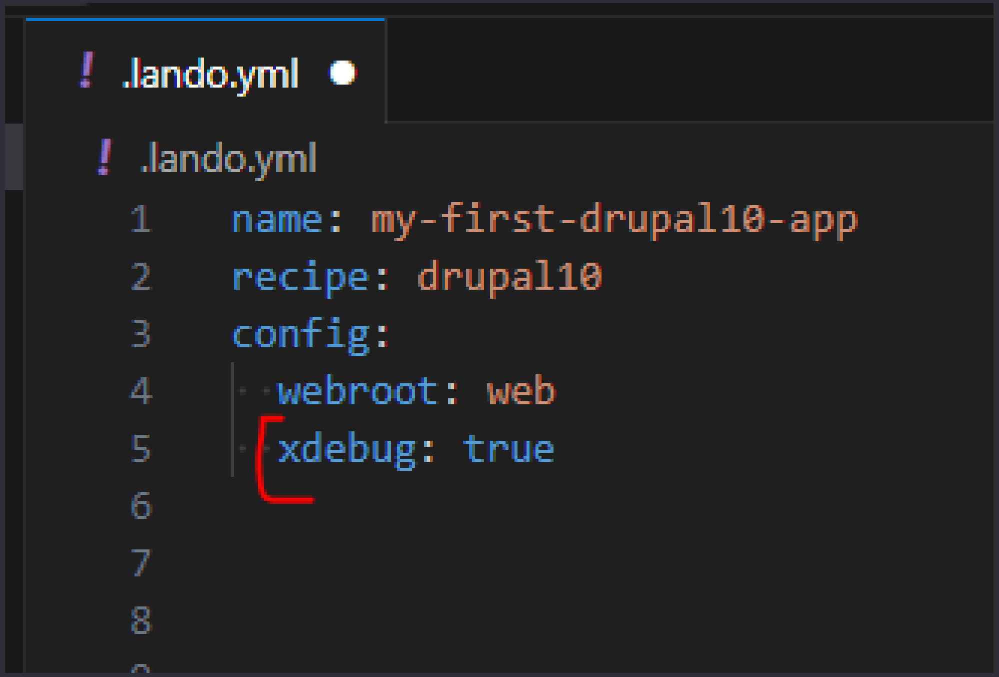
```
networks:
  amazeeio-network:
    external: true        You, 3 months
```

- PS: you can always use a ".lando.local.yml" file for a local override as to not annoy other developers!

Lando and Xdebug (with VS code)

For Xdebug, I'm talking about Xdebug 3 which is the default in Lando PHP 8.1. Some commands in differ for Xdebug 2.

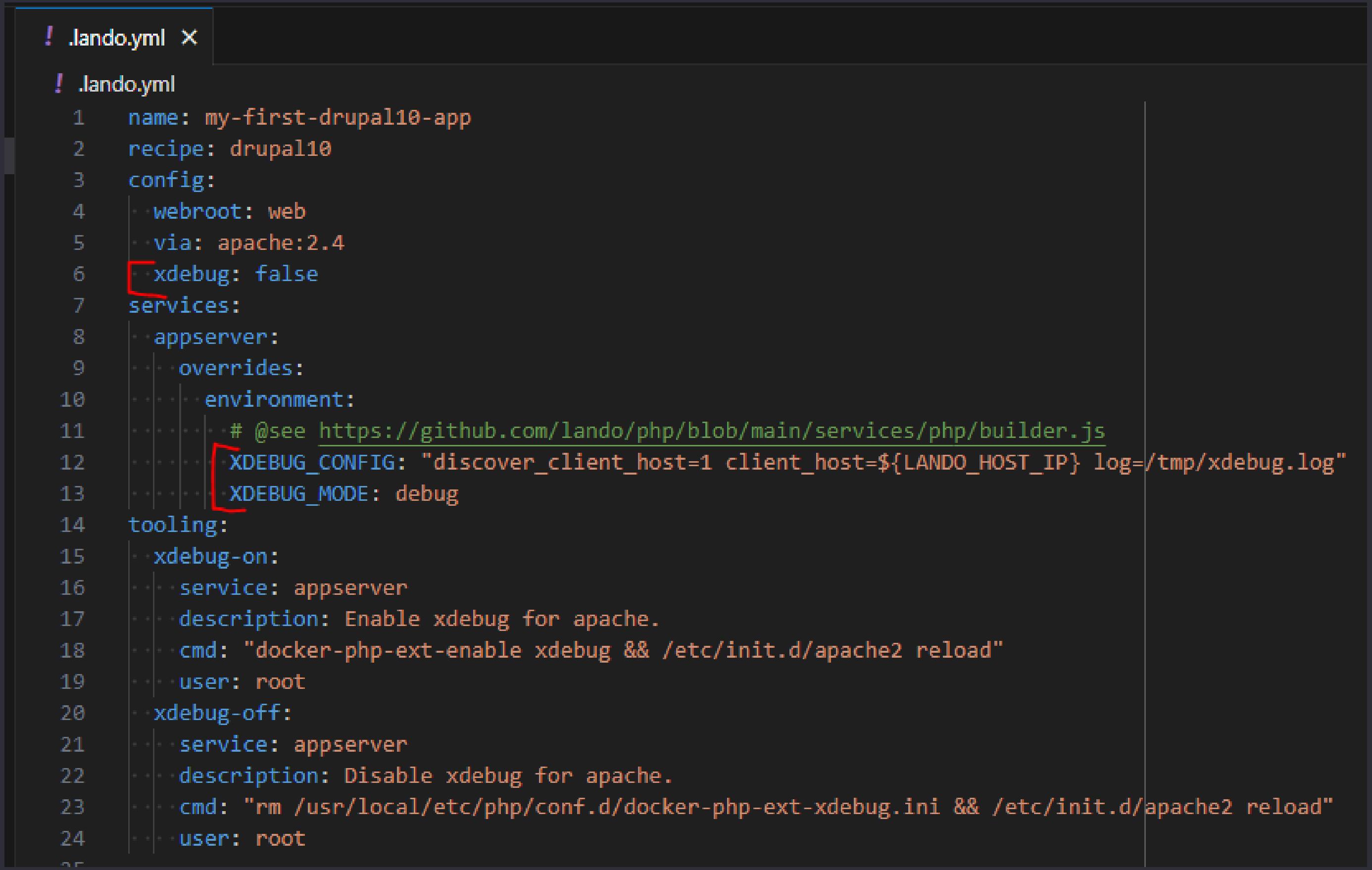
To use Xdebug with lando, the easiest way is just set the 'xdebug' flag to true and run `lando rebuild`. However this will slow up your application since it is always running.



```
name: my-first-drupal10-app
recipe: drupal10
config:
  webroot: web
  xdebug: true
```

Lando and Xdebug (with VS code)

Another way is using lando's tooling to add commands to enable and disable xdebug.



```yaml
name: my-first-drupal10-app
recipe: drupal10
config:
  webroot: web
  via: apache:2.4
  xdebug: false
services:
  appserver:
    overrides:
      environment:
        # @see https://github.com/lando/php/blob/main/services/php/builder.js
        XDEBUG_CONFIG: "discover_client_host=1 client_host=${LANDO_HOST_IP} log=/tmp/xdebug.log"
        XDEBUG_MODE: debug
tooling:
  xdebug-on:
    service: appserver
    description: Enable xdebug for apache.
    cmd: "docker-php-ext-enable xdebug && /etc/init.d/apache2 reload"
    user: root
  xdebug-off:
    service: appserver
    description: Disable xdebug for apache.
    cmd: "rm /usr/local/etc/php/conf.d/docker-php-ext-xdebug.ini && /etc/init.d/apache2 reload"
    user: root
```

https://github.com/lando/lando/issues/2718#issuecomment-740842589

Run:

- `` `lando xdebug-on` ``

And

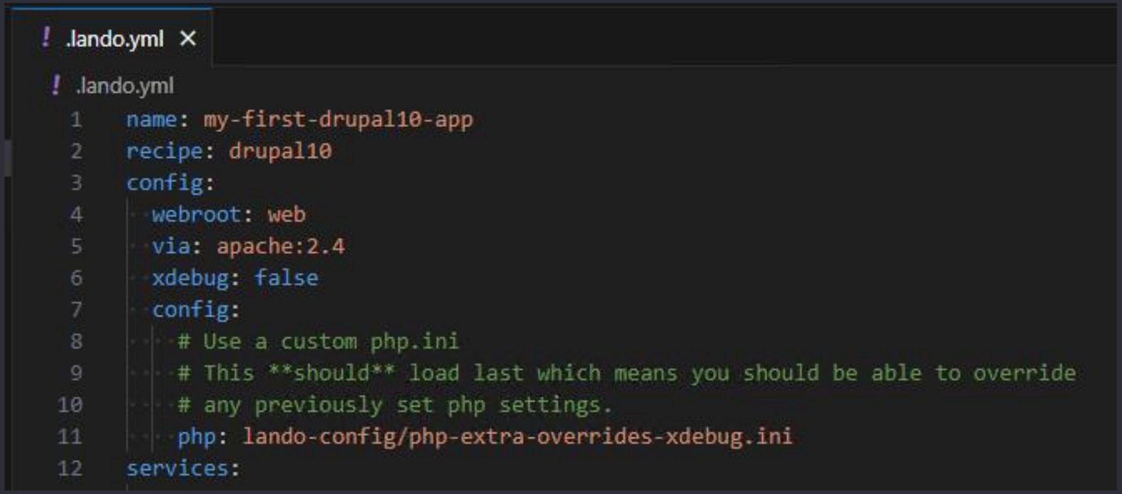
- `` `lando xdebug-off` ``

PS: Example assumes Apache as the web server, adjust tooling for nginx web server.

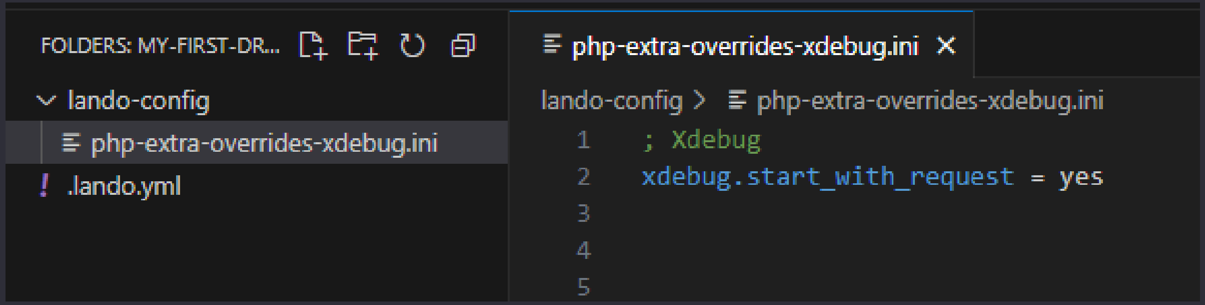
Lando and Xdebug (with VS code)

If you need remote connect always on, a php.ini injection is required.
Usually you need marker to trigger debugging, so then a marker is not required (useful for ajax urls):

- http://my-project.docker.amazee.io/my-page?XDEBUG_SESSION_START=idekey
- XDEBUG_SESSION_START=idekey php path/to/myfile.php



```yaml
name: my-first-drupal10-app
recipe: drupal10
config:
  webroot: web
  via: apache:2.4
  xdebug: false
  config:
    # Use a custom php.ini
    # This **should** load last which means you should be able to override
    # any previously set php settings.
    php: lando-config/php-extra-overrides-xdebug.ini
services:
```

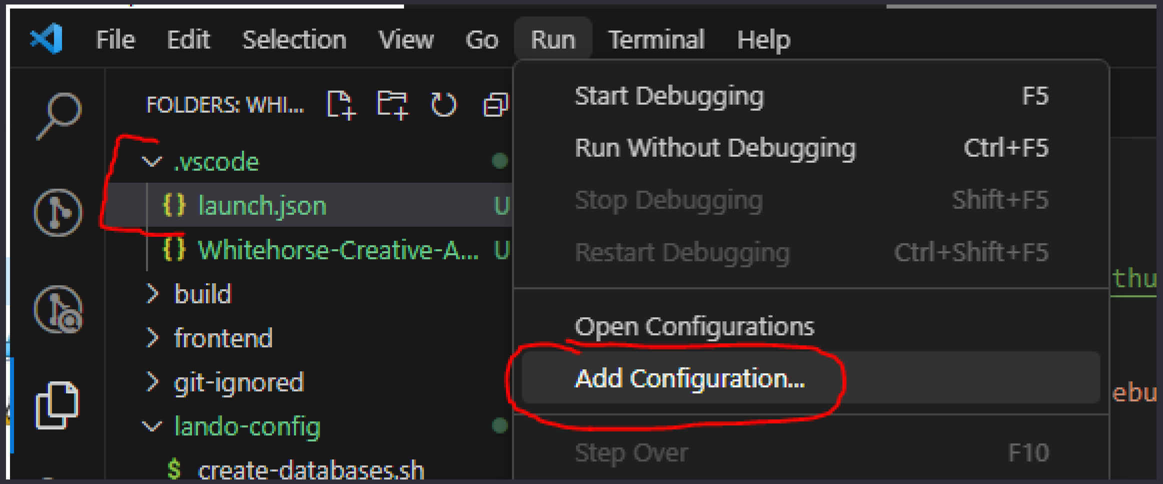

FOLDERS: MY-FIRST-DR...
lando-config
php-extra-overrides-xdebug.ini
! .lando.yml
php-extra-overrides-xdebug.ini ✕
lando-config > php-extra-overrides-xdebug.ini

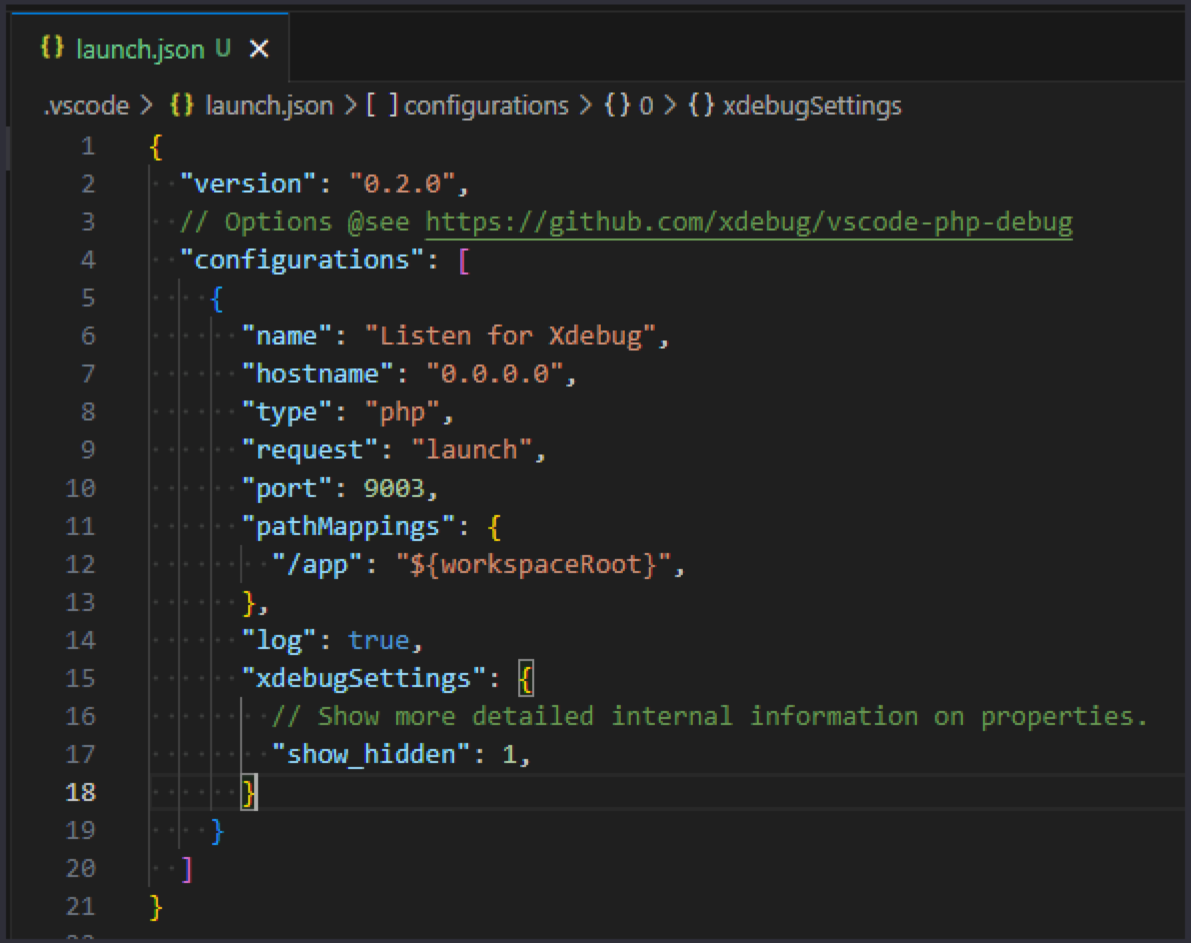
```ini
; Xdebug
xdebug.start_with_request = yes
```

https://www.jetbrains.com/help/phpstorm/debugging-a-php-cli-script.html#starting-cli-debugging-session

Lando and Xdebug (with VS code)

In VS Code, add lunch configuration by going to “Run” then “Add Configuration”. You will see a launch.json file created on the left.



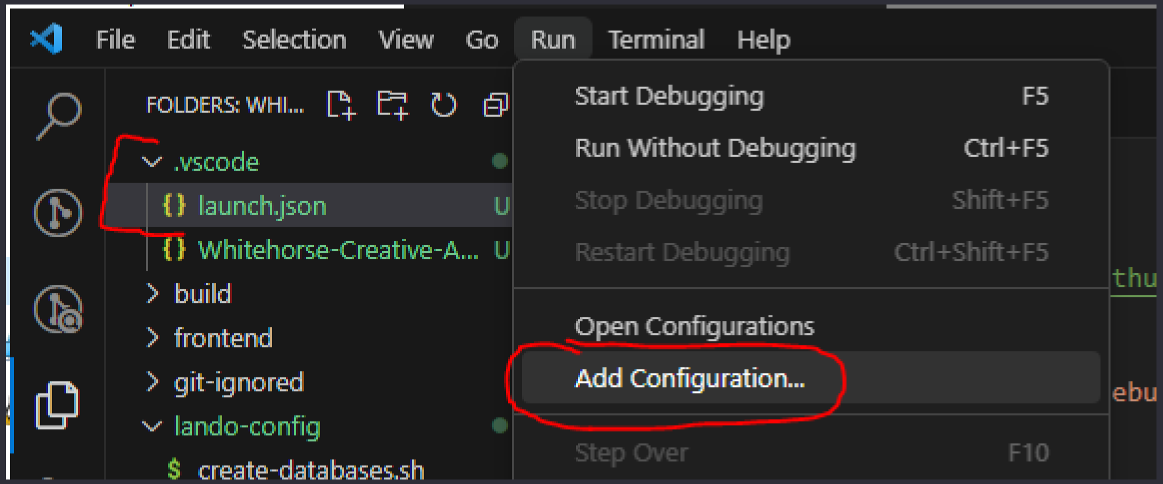

launch.json U ×
.vscode > launch.json > [ ] configurations > { } 0 > { } xdebugSettings

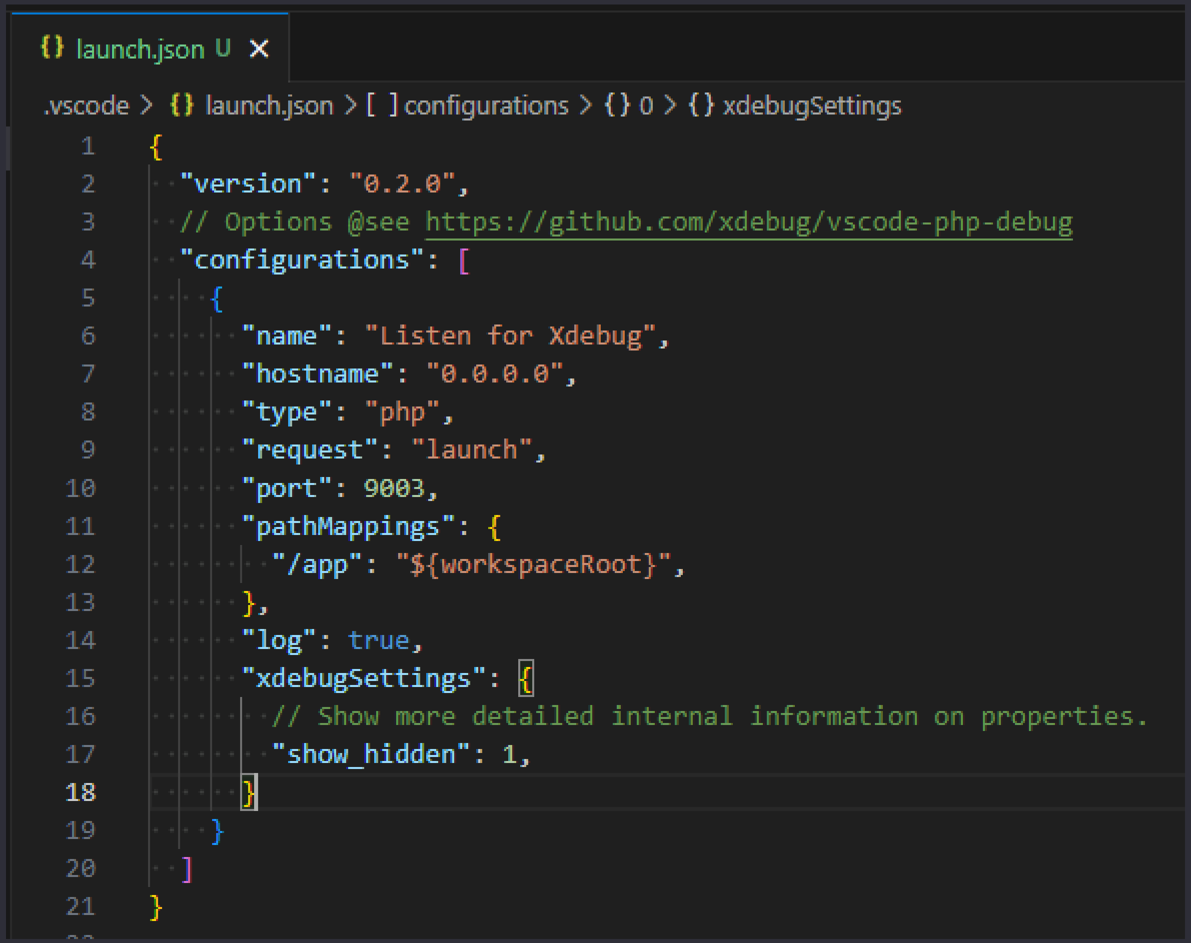
```json
{
  "version": "0.2.0",
  // Options @see https://github.com/xdebug/vscode-php-debug
  "configurations": [
    {
      "name": "Listen for Xdebug",
      "hostname": "0.0.0.0",
      "type": "php",
      "request": "launch",
      "port": 9003,
      "pathMappings": {
        "/app": "${workspaceRoot}",
      },
      "log": true,
      "xdebugSettings": {
        // Show more detailed internal information on properties.
        "show_hidden": 1,
      }
    }
  ]
}
```

https://github.com/xdebug/vscode-php-debug
https://xdebug.org/docs/dbgp

Lando and Xdebug (with VS code)

In VS Code, add lunch configuration by going to “Run” then “Add Configuration”. You will see a launch.json file created on the left.




launch.json U
.vscode > launch.json > [ ] configurations > { } 0 > { } xdebugSettings

```json
{
  "version": "0.2.0",
  // Options @see https://github.com/xdebug/vscode-php-debug
  "configurations": [
    {
      "name": "Listen for Xdebug",
      "hostname": "0.0.0.0",
      "type": "php",
      "request": "launch",
      "port": 9003,
      "pathMappings": {
        "/app": "${workspaceRoot}",
      },
      "log": true,
      "xdebugSettings": {
        // Show more detailed internal information on properties.
        "show_hidden": 1,
      }
    }
  ]
}
```

https://github.com/xdebug/vscode-php-debug
https://xdebug.org/docs/dbgp

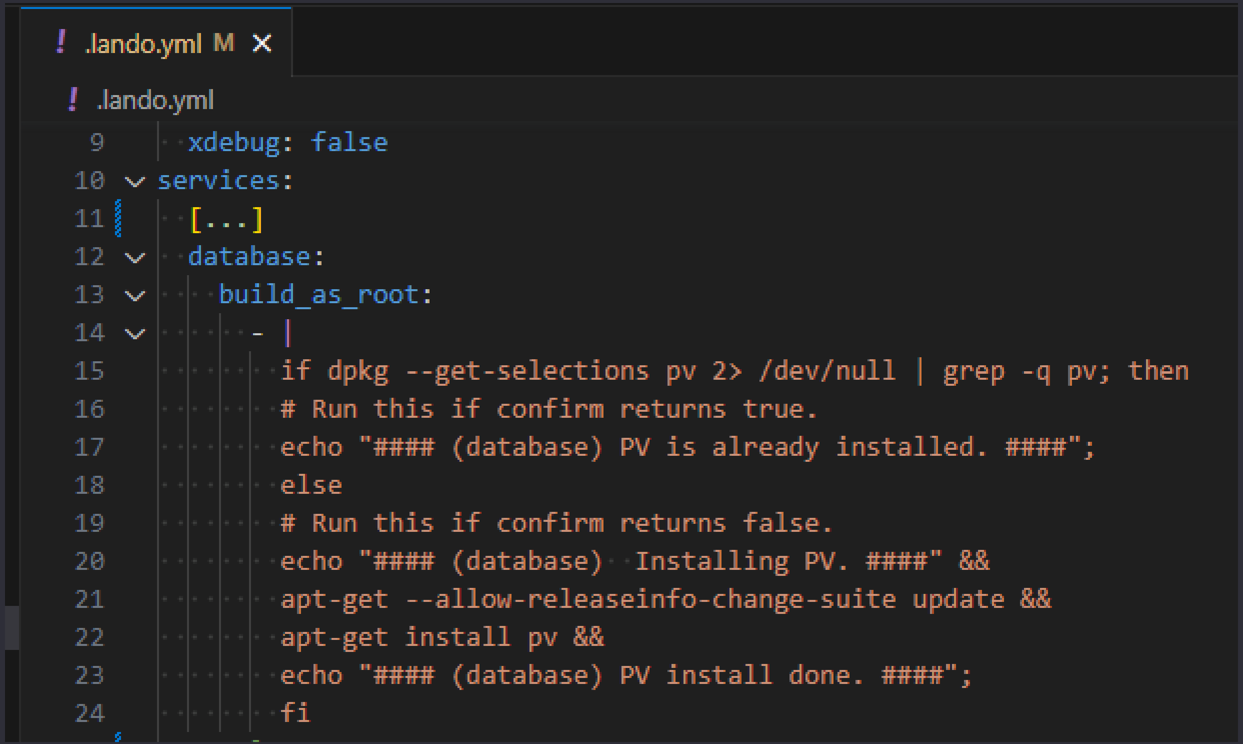
Lando always show SQL file import progress

Conventionally, we put the database sql file dump in to lando's "/app" folder (usually the same place as where the .lando.yml file is) and run `lando db-import myfile.sql.gz`.

However Lando does not always show progress. Probably is because it guesses based on file size and but it cannot take computer performance into consideration.

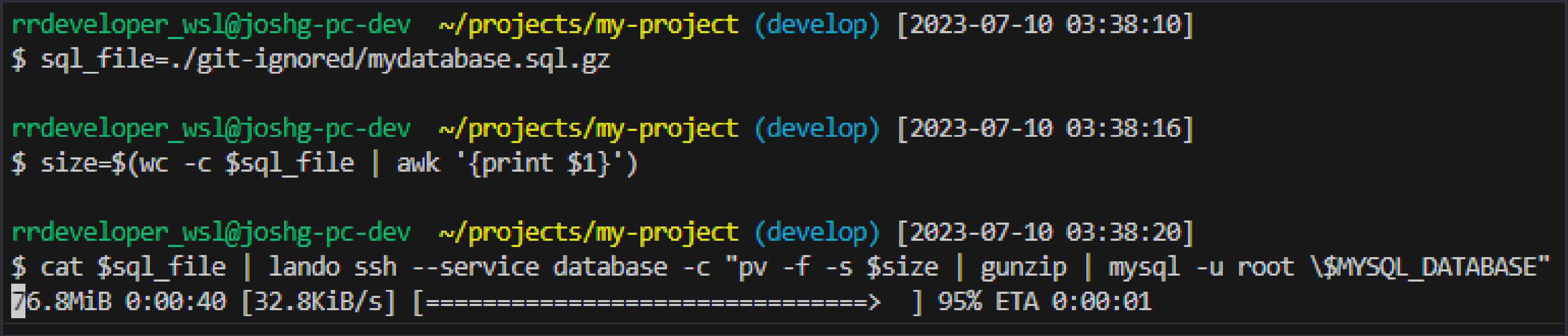
Here, we can manually pipe the database file in via "pv", but before we do that, we should make sure it's installed first.

Running a "sudo apt-get install pv" will do but a lando rebuild will remove it, so we can add the following code to ensure it's ways there.



```
  xdebug: false
services:
  [...]
  database:
    build_as_root:
      - |
        if dpkg --get-selections pv 2> /dev/null | grep -q pv; then
        # Run this if confirm returns true.
        echo "#### (database) PV is already installed. ####";
        else
        # Run this if confirm returns false.
        echo "#### (database)  Installing PV. ####" &&
        apt-get --allow-releaseinfo-change-suite update &&
        apt-get install pv &&
        echo "#### (database) PV install done. ####";
        fi
```

Lando always show SQL file import progress



```
rrdeveloper_wsl@joshg-pc-dev  ~/projects/my-project (develop) [2023-07-10 03:38:10]
$ sql_file=./git-ignored/mydatabase.sql.gz

rrdeveloper_wsl@joshg-pc-dev  ~/projects/my-project (develop) [2023-07-10 03:38:16]
$ size=$(wc -c $sql_file | awk '{print $1}')

rrdeveloper_wsl@joshg-pc-dev  ~/projects/my-project (develop) [2023-07-10 03:38:20]
$ cat $sql_file | lando ssh --service database -c "pv -f -s $size | gunzip | mysql -u root \$MYSQL_DATABASE"
76.8MiB 0:00:40 [32.8KiB/s] [===============================>  ] 95% ETA 0:00:01
```

Download - Slides and example code

Download link for slides and same codes (codes coming soon!):

https://github.com/silverham/DrupalACT-2023-07-WSL-Lando

Questions?