Development

TABLE OF CONTENTS

Local Drupal site setup

Local development is best done using containers and DDEV. Setting up a local site is a completely painless process.

Pick one of these options after installing Docker and Ddev:

FIRST OPTION

Using the DDEV Quickstart guides to install Drupal, Wordpress, TYPO3, Backdrop, Magento, Laravel etc. at https://ddev.readthedocs.io/en/stable/users/quickstart/#drupal

```
mkdir my-drupal10-site
cd my-drupal10-site
ddev config --project-type=drupal10 --docroot=web --create-docroot
ddev start
ddev composer create "drupal/recommended-project" --no-install
ddev composer require drush/drush --no-install
ddev composer install
ddev drush site:install -y
ddev drush uli
ddev launch
```

OR

SECOND OPTION

From https://www.drupal.org/docs/official_docs/en/_local_development_guide.html

Start by specifying your SITE_NAME using export:

```
export SITE_NAME=d9site
export SITE_NAME=clientsite
```

Here are all the steps:

```
export SITE_NAME=my-drupal-site
composer create-project drupal/recommended-project $SITE_NAME
cd $SITE_NAME
ddev config --docroot=web --project-name=$SITE_NAME --project-type=drupal9
ddev start
ddev exec drush site-install --account-name=admin --account-pass=admin
```

Checking Your Permissions

During the wizard installation, or when your welcome page first loads, you might see a warning about the permissions settings on your `/sites/web/default` directory and one file inside that directory: `settings.php`.

After the installation script runs, Drupal will try to set the web/sites/default directory permissions to read and execute for all groups: this is a 555 permissions setting. It will also attempt to set permissions for default/settings.php to read-only, or 444. If you encounter this warning, run these two chmod commands from your project's root directory. Failure to do so poses a security risk:

```
chmod 555 web/sites/default
```

```
chmod 444 web/sites/default/settings.php
```

To verify that you have the correct permissions, run this `ls` command with the a, l, h, and d switches and check that your permissions match the following output:

```
$ ls -alhd web/sites/default web/sites/default/settings.php

dr-xr-xr-x 8 sammy staff 256 Jul 21 12:56 web/sites/default
-r--r--r-- 1 sammy staff 249 Jul 21 12:12 web/sites/default/settings.php
```

You are now ready to develop a Drupal website on your local machine.

Converting existing site (non-composer based) to use composer

Composerize Drupal

Also for manual steps

Composer best practices for Drupal 8

https://www.lullabot.com/articles/drupal-8-composer-best-practices

DDEV

For local Docker container development on any platform, there is no better tool than DDEV. This is a well-documented, well-supported tool by the Amazing Randy Fay. You can get help from him or some of the other friendly folks on Discord almost instantly.

From the docs:

- Lots of built-in help: ddev help and ddev help <command>. You'll find examples and explanations.
- DDEV Documentation
- DDEV Stack Overflow for support and frequently asked questions. We respond quite quickly here and the results provide quite a library of user-curated solutions.
- DDEV issue queue for bugs and feature requests
- Interactive community support on Discord for everybody, plus sub-channels for CMS-specific questions and answers.
- ddev-contrib repo provides a number of vetted user-contributed recipes for extending and using DDEV. Your contributions are welcome.
- awesome-ddev repo has loads of external resources, blog posts, recipes, screencasts, and the like. Your contributions are welcome.
- Twitter with tag #ddev will get to us, but it's not as good for interactive support, but we'll answer anywhere.

Local config - your .ddev/config.local.yaml

From https://ddev.readthedocs.io/en/stable/users/extend/config_yaml

- You can override the config.yaml with extra files named `config.*.yaml\`. For example, many teams use `config.local.yaml` for configuration that is specific to one environment, and that is not intended to be checked into the team's default config.yaml.
- You could add a `config.selwyn.yaml` for Selwyn-specific values.
- Use ddev start (or ddev restart) after making changes to get the changes to take effect.

In the endless quest for speed in local development, try using NFS or Mutagen on MAC OS. Apparently the WSL2 setup on Windows 10/11 is the fastest performer for DDEV at th time of this writing.

NFS

```
router_http_port: \"80\"
router_https_port: \"443\"
timezone: America/Chicago
# and for nfs
nfs_mount_enabled: true
```

MUTAGEN

```
# instead of nfs, use mutagen
nfs_mount_enabled: false
mutagen_enabled: true
```

setup aliases in ddev

I love short linux aliases like ll (or just l) for listing files. If you spend time poking around the file system in your containers this makes life so much better. A cool new feature since Ddev v15.1 lets you add aliases using this technique

Use ddev ssh to “ssh” into the container and then type ll to list the files in a directory.

Either copy `.ddev/homeadditions/bash_aliases.example` to `.ddev/homeadditions/bash_aliases` and add them there!

OR

Create a file `.ddev/homeadditions/.bash_aliases` with these contents: note. those are the letter L lower case (as in lima).

```
alias ll=\"ls -lhAp"
alias l=\"ls -lhAp"
```

Note. don’t use `.homeadditions` - use the `homeadditions`.

Upgrading ddev

After you install a new version of ddev, run `ddev stop` and then `ddev config` to reconfigure things for your project. Just press enter for all the questions. It keeps things rolling smoothly. Run `ddev start` to start it all back up again

Show others your ddev local site

Sharing your DDEV-Local site via a public URL using `ddev share` and ngrok. by Mike Anello

https://www.drupaleasy.com/blogs/ultimike/2019/06/sharing-your-ddev-local-site-public-url-using-ddev-share-and-ngrok

Email Capture and Review

MailHog is a mail catcher which is configured to capture and display emails sent by PHP in the development environment.

After your project is started, access the MailHog web interface at its default port:

http://mysite.ddev.site:8025

Please note this will not intercept emails if your application is configured to use SMTP or a 3rd-party ESP integration. If you are using SMTP for outgoing mail handling (Swiftmailer or SMTP modules for example), update your application configuration to use localhost on port 1025 as the SMTP server locally in order to use MailHog.

`ddev launch -m` will launch the MailHog UI.

DDEV and Xdebug

This is a magical match made in heaven. To enable or disable Xdebug use

`$ ddev xdebug on`

and

`$ ddev xdebug off`

Note. This will slow everything down because xdebug has a significant performance impact so be sure to disable it when you are finished with your debugging session.

In phpstorm, you can uncheck the following settings:

- force break at first line when no path mapping is specified
- force break at first line when a script is outside the project

Note. we usually use port 9000 for xdebug look in `.ddev/php/xdebug_report_port.ini` for the real port settings. Recently for a project I found it set to 11011

The contents of the file are:

```
[PHP]

xdebug.remote_port=11011
```

For phpstorm, if you start listening for a debug connection, it should automatically try to create a debug server config for you. If it doesn't manually create one

e.g name: tea.ddev.site

host tea.ddev.site

port: 80

debugger: xdebug

check use path mappings

for docroot specify: /var/www/html/docroot (i.e. wherever index.php is)

Command line or drush debugging

For command line or drush debugging (xdebug, phpstorm)

```
ddev ssh
```

```
export PHP_IDE_CONFIG=\"serverName=d8git.ddev.site\"
```

or

```
export PHP_IDE_CONFIG=\"serverName=inside-mathematics.ddev.site\"
```

confirm debug is turned on

```
php -i | grep debug
```

You should see:

```
xdebug support => enabled
```

Also you can confirm the port

set a server in phpstorm that matches the name `d8git.ddev.site` or `inside-mathematics.ddev.site`.

Configure the server to use path mappings

/Users/selwyn/Sites/ddev 82 ---> /var/www/html

click listen for debug connections button

set breakpoint and run

replace `d8git.ddev.site` with the name of your project

NOTE!!!!. You must execute drush from the vendor dir or you will always be ignored:

```
../vendor/drush/drush/drush fixmat
```

If it doesn't seem to work, try enable Break at first line in PHP scripts - something will always stop then.

more at https://stackoverflow.com/questions/50283253/how-can-i-step-debug-a-drush-command-with-ddev-and-phpstorm

Use drush commands in your shell with DDEV

If you do local development, you can use syntax like `ddev drush cst` to execute drush commands in the container. This is slower than running on your native system because they are executed in the container. I prefer using drush directly on the host computer.

To do this install PHP as well drush launcher. Once these are working, you can `cd` into the project directory and issue commands like `drush cr`, `drush cst` or `drush cim -y` etc. It is *so* very quick and smooth. (Note. this is the case with MacOS and Linux but I don't really know how it works on Windows.)

Details for Drush Launcher

From Installation of Drush Launcher

- To be able to call drush from anywhere, install the Drush Launcher. Launcher is a small program which listens on your $PATH and hands control to a site-local Drush that is in the /vendor directory of your Composer project.

Luckily, DDEV enables this functionality by default (Thanks Randy!)

Load your data from an Acquia site

Using the drush aliases assuming the site is called `abc` and you want the `prod` (production) database:

```
$ drush @abc.prod sql-dump >dbprod.sql
$ gzip dbprod.sql
$ ddev import-db --src=dbprod.sql.gz
```

Of course this works with any site where you've set up your drush aliases.

Cleanup some disk space

Free up disk space used by previous docker image versions. This does no harm.

```
ddev delete images
```

also

```
docker system prune
```

and

```
docker image prune -a
```

List all docker volumes

```
docker volume ls
```

DDEV General cleanup topic

Accessing specific containers

To ssh into a specific service e.g. from a docker-composer.chromedriver.yml the service is listed under "services:" like:

```
services:
  chromedriver
```

Use

```
ddev ssh -s chromedriver
```

or for selenium, use:

```
ddev ssh -s selenium
```

DDEV Troubleshooting

Running out of docker disk space

if ddev won't start and shows:

```
Creating ddev-router ... done
Failed to start ddev82: db container failed: log=, err=container exited, please use 'ddev logs -s db` to find out why it failed
```

Looking in the log, you might see:

```
preallocating 12582912 bytes for file ./ibtmp1 failed with error 28
2020-03-16 14:27:54 140144158233920 [ERROR] InnoDB: Could not set the file size of './ibtmp1'. Probably out of disk space
```

That is the clue.

You can kill off images using

```
ddev delete images
```

or the more drastic

```
docker rmi -f $(docker images -q)
```

Q. Deleting the images: Does that mean it will delete the db snapshots? A. No, docker images are the versioned images that come from dockerhub, they're are always replaceable.

Absolutely nothing you do with ddev will delete your snapshots - you have to remove them manually

They're stored in .ddev/db_snapshots on the host (under each project)

also

```
docker system prune
```

and

```
docker system prune --volumes
```

prunes every single thing, destroys all ddev databases and your composer cache.

DDEV won't start

ddev pull or ddev start failed with error something like:

```
Pull failed: db container failed: log=, err=health check timed out: labels map[com.ddev.site-name:inside-mathematics com.docker.compose.service:db] timed out without becomin
```

Or like this:

```
$ ddev start
Starting inside-mathematics...
Pushing mkcert rootca certs to ddev-global-cache
Pushed mkcert rootca certs to ddev-global-cache
Creating ddev-inside-mathematics-db ... done
Creating ddev-inside-mathematics-dba ... done
Creating ddev-inside-mathematics-web ... done

Creating ddev-router ... done

Failed to start inside-mathematics: db container failed: log=, err=health check timed out: labels map[com.ddev.site-name:inside-mathematics com.docker.compose.service:db] tir
```

This is almost always caused by a corrupted database, most often in a larger database. Since v0.17.0, this is generally only caused by docker being shut down in an ungraceful way. Unfortunately, both Docker for Windows and Docker for Mac shut down without notifying the container during upgrade, with a manual Docker exit, or at system shutdown. It can be avoided by stopping or removing your projects before letting Docker exit.

To fix, `ddev remove --remove-data`, then `ddev start`. This may fail and suggest this bazooka version:

```
ddev stop --remove-data --omit-snapshot
```

PHPStorm

All the PHPStorm Drupal magic is at https://www.jetbrains.com/help/phpstorm/drupal-support.html#view_drupal_api_documentation

Setting up PHPStorm and Drupal

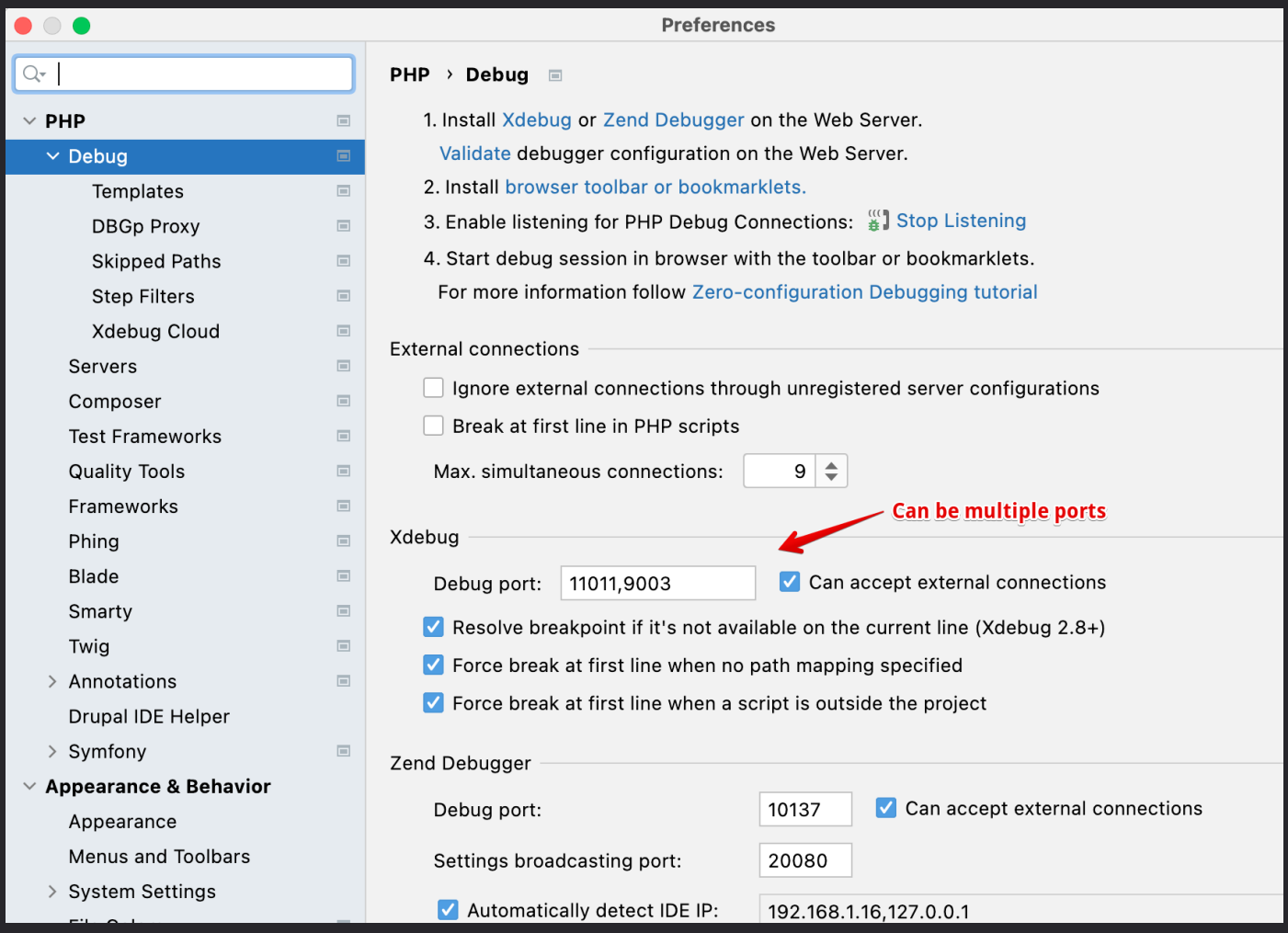
https://www.drupal.org/docs/develop/development-tools/configuring-phpstorm

PHPStorm and Xdebug

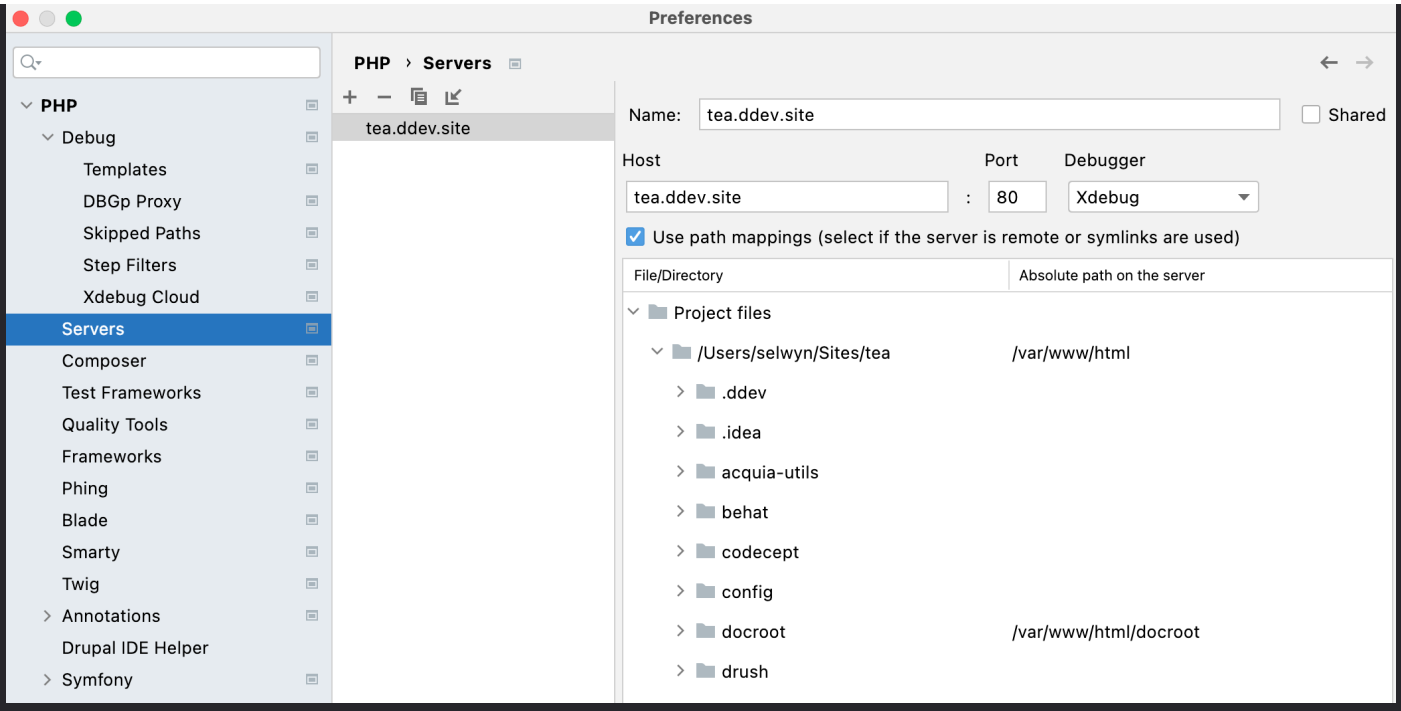
Debugging drush commands at https://www.jetbrains.com/help/phpstorm/drupal-support.html#debugging-drush-commands

PHPStorm has a series of instructions for configuring PHPStorm with Xdebug but unfortunately, nothing specifically on using it with DDEV. Fortunately it doesn't require any special setup for it to work.

Some settings I use

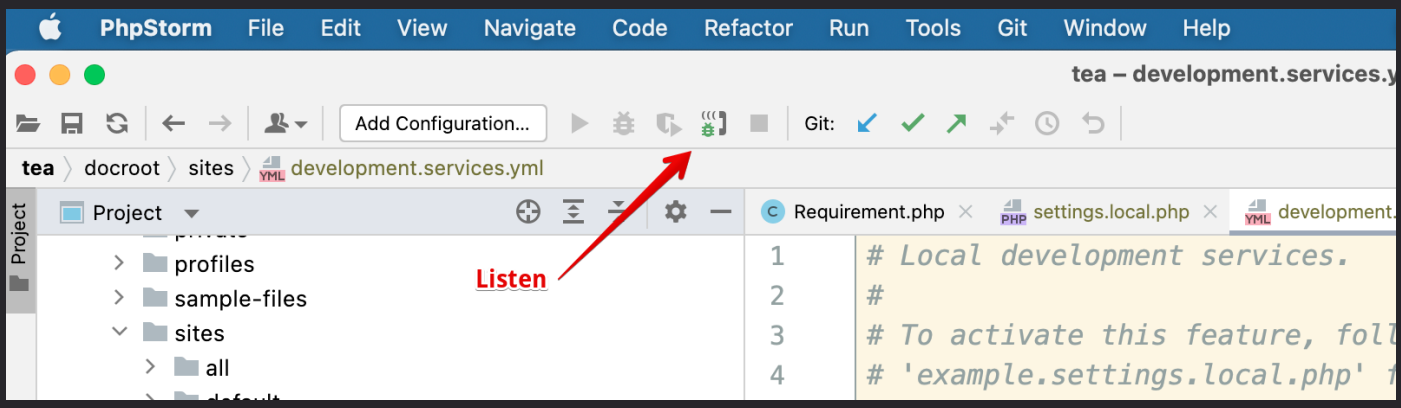


And for this project



If phpstorm doesn't stop when you set a breakpoint on some code, try deleting the server from the config debug, php, servers.

Make sure PHPStorm is listening by clicking the listen button



When you try again it will be recreated but you will probably need to specify the path (from the image above).

ADD A BREAKPOINT IN CODE

To add a breakpoint in code, use

```
xdebug_break()
```

more at https://xdebug.org/docs/all_functions

Collecting PhpStorm debugging logs

- In the Settings/Preferences dialog (⌘ ,) , go to PHP.
- From the PHP executable list, choose the relevant PHP interpreter and click next to it. In the CLI Interpreters dialog that opens, click the Open in Editor link next to the Configuration file: <path to php.ini> file. Close all the dialogs and switch to the tab where the php.ini file is opened.
- In the php.ini, enable Xdebug logging by adding the following line:
- For Xdebug 3xdebug.log="path_to_log/xdebug.log"The log file contains the raw communication between PhpStorm and Xdebug as well as any warnings or errors:
- https://www.jetbrains.com/help/phpstorm/troubleshooting-php-debugging.html#collecting-logs

Troubleshooting Xdebug with DDEV

• Use curl or a browser to create a web request. For example, curl https://d9.ddev.site

• If the IDE doesn't respond, take a look at ddev logs (`ddev logs`). If you see a message like ""PHP message: Xdebug: [Step Debug] Could not connect to debugging client. Tried: host.docker.internal:9000 (through xdebug.client_host/xdebug.client_port)" then php/xdebug (inside the container) is not able to make a connection to port 9000.

• In PhpStorm, disable the "listen for connections" button so it won't listen. Or just exit PhpStorm. With another IDE like vscode, stop the debugger from listening.

• ddev ssh: Can telnet host.docker.internal 9000 connect? If it does, you have something else running on port 9000, probably php-fpm. On the host, use sudo lsof -i :9000 -sTCP:LISTEN to find out what is there and stop it, or change the xdebug port and configure PhpStorm to use the new one . Don't continue debugging until your telnet command does not connect.

• Check to make sure that Xdebug is enabled. You can use php -i | grep Xdebug inside the container, or use any other technique you want that gives the output of phpinfo(), including Drupal's admin/reports/status/php. You should see with Xdebug v2.9.6, Copyright (c) 2002-2020 and php -i | grep "xdebug.remote_enable" should give you xdebug.remote_enable: On.

https://ddev.readthedocs.io/en/stable/users/step-debugging/

What is listening on port 9000?

To check if something is listening on port 9000 (the default port for xdebug) it's best to use

```
$ lsof -i TCP:9000
```

it will actually list the name of the process listening

i.e.

```
COMMAND    PID   USER   FD   TYPE            DEVICE SIZE/OFF NODE NAME
phpstorm 13361 selwyn   81u  IPv6 0x5d4d30caf0be07d      0t0  TCP *:cslistener (LISTEN)
```

Another option is

```
nc -z localhost 9000
```

If it says:

Connection to localhost port 9000 [tcp/cslistener] succeeded!

this means something is listening. If you get nothing, then nothing is listening.

You can also run network utility, scan port 9000 to 9003 on 127.0.0.1 (localhost)

What could be listening on port 9000?

```
$ netstat -an | grep 9000
```

```
tcp4 0 0 127.0.0.1.9000 \*.\* LISTEN
```

Other options include:

```
$ lsof -i TCP:9000
```

Which reports that `php-fpm` is listening.

```
COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME

php-fpm 732 selwyn 7u IPv4 0x4120ed57a07e871f 0t0 TCP
localhost:cslistener (LISTEN)

php-fpm 764 selwyn 8u IPv4 0x4120ed57a07e871f 0t0 TCP
localhost:cslistener (LISTEN)

php-fpm 765 selwyn 8u IPv4 0x4120ed57a07e871f 0t0 TCP
localhost:cslistener (LISTEN)
```

Setup settings.local.php and disable Cache

From Disabling cache during development

1. Copy, rename, and move the sites/example.settings.local.php to sites/default/settings.local.php:

```
$ cp sites/example.settings.local.php sites/default/settings.local.php
```

2. Open sites/default/settings.php and uncomment these lines:

```
if (file_exists($app_root . '/' . $site_path . '/settings.local.php')) {
  include $app_root . '/' . $site_path . '/settings.local.php';
}
```

This will include the local settings file as part of Drupal's settings file.

3. Open settings.local.php and make sure development.services.yml is enabled.

```
$settings['container_yamls'][] = DRUPAL_ROOT . '/sites/development.services.yml';
```

By default development.services.yml contains the settings to disable Drupal caching:

```
services:
  cache.backend.null:
    class: Drupal\Core\Cache\NullBackendFactory
```

NOTE: Do not create development.services.yml, it exists under /sites

4. In settings.local.php change the following to be TRUE if you want to work with enabled css- and js-aggregation:

```
$config['system.performance']['css']['preprocess'] = FALSE;
$config['system.performance']['js']['preprocess'] = FALSE;
```

5. Uncomment these lines in settings.local.php to disable the render cache and disable dynamic page cache:

```
$settings['cache']['bins']['render'] = 'cache.backend.null';
$settings['cache']['bins']['dynamic_page_cache'] = 'cache.backend.null';
```

Add the following lines to your sites/default/settings.local.php

```
$settings['cache']['bins']['page'] = 'cache.backend.null';
```

If you do not want to install test modules and themes, set the following to FALSE:

```
$settings['extension_discovery_scan_tests'] = FALSE;
```

6. Open sites/development.services.yml in the sites folder and add the following block to disable the twig cache and enable twig debugging:

```
parameters:
  twig.config:
    debug: true
    auto_reload: true
    cache: false
```

NOTE: If the parameters section is already present in the development.services.yml file, append the twig.config section to it.

7. Rebuild the Drupal cache (drush cr) otherwise your website will encounter an unexpected error on page reload.

Development.services.yml

I usually develop with this in sites/default/development.services.yml

```
# Local development services.
#
# To activate this feature, follow the instructions at the top of the
# 'example.settings.local.php' file, which sits next to this file.
parameters:
  http.response.debug_cacheability_headers: true
  twig.config:
    # Twig debugging:
    #
    # When debugging is enabled:
    # - The markup of each Twig template is surrounded by HTML comments that
    #   contain theming information, such as template file name suggestions.
    # - Note that this debugging markup will cause automated tests that directly
    #   check rendered HTML to fail. When running automated tests, 'debug'
    #   should be set to FALSE.
    # - The dump() function can be used in Twig templates to output information
    #   about template variables.
    # - Twig templates are automatically recompiled whenever the source code
    #   changes (see auto_reload below).
    #
    # For more information about debugging Twig templates, see
    # https://www.drupal.org/node/1906392.
    #
    # Not recommended in production environments
    # @default false
    debug: true
    # Twig auto-reload:
    #
    # Automatically recompile Twig templates whenever the source code changes.
    # If you don't provide a value for auto_reload, it will be determined
    # based on the value of debug.
    #
    # Not recommended in production environments
    # @default null
    #    auto_reload: null
    auto_reload: true
    # Twig cache:
    #
    # By default, Twig templates will be compiled and stored in the filesystem
    # to increase performance. Disabling the Twig cache will recompile the
    # templates from source each time they are used. In most cases the
    # auto_reload setting above should be enabled rather than disabling the
    # Twig cache.
    #
    # Not recommended in production environments
    # @default true
    cache: false
services:
  cache.backend.null:
    class: Drupal\Core\Cache\NullBackendFactory
```

Make sure the following is in docroot/sites/default/settings.local.php

```
/**
 * Enable local development services.
 */
$settings['container_yamls'][] = DRUPAL_ROOT . '/sites/development.services.yml';
```

Enable twig debugging output in source

In sites/default/development.services.yml set twig.config debug:true. See core.services.yml for lots of other items to change for development

```
# Local development services.
#
parameters:
  http.response.debug_cacheability_headers: true
  twig.config:
    debug: true
    auto_reload: true
    cache: false

# To disable caching, you need this and a few other items
services:
  cache.backend.null:
    class: Drupal\Core\Cache\NullBackendFactory
```

to enable put the following in settings.local.php:

```
/**
 * Enable local development services.
 */
$settings['container_yamls'][] = DRUPAL_ROOT . '/sites/development.services.yml';
```

You also need to disable the render cache in settings.local.php with:

```
$settings['cache']['bins']['render'] = 'cache.backend.null';
```

Kint

From https://www.webwash.net/how-to-print-variables-using-devel-and-kint-in-drupal/

Setup

We need both the Devel and Devel Kint Extras module. Devel Kint Extras ships with the kint-php library so installing this via Composer will take care of it automatically.

Install using Composer:

```
$ composer require drupal/devel drupal/devel_kint_extras
```

Enable both with the following Drush command:

```
$ drush en devel_kint_extras -y
```

Finally, enable Kint Extended as the Variables Dumper. To do this go to admin/config/development/devel and select Kint Extender and Save the configuration.

Add kint to a custom module

```
function custom_kint_preprocess_page(&$variables) {
  kint($variables['page']);
 }
```

Dump variables in a TWIG template

```
{{ kint(attributes) }}
```

Kint::dump

From Migrate Devel contrib module, in /docroot/modules/contrib/migrate_devel/src/EventSubscriber/MigrationEventSubscriber.php.

This is used in migrate to dump the source and destination values.

```
// We use kint directly here since we want to support variable naming.
kint_require();
\Kint::dump($Source, $Destination, $DestinationIDValues);
```

Set max levels to avoid running out of memory

Kint can run really slowly so you may have to set maxLevels this way:

Add this to settings.local.php

```
// Change kint maxLevels setting:
include_once(DRUPAL_ROOT . '/modules/contrib/devel/kint/kint/Kint.class.php');
if(class_exists('Kint')){
  // Set the maxlevels to prevent out-of-memory. Currently there doesn't seem to be a cleaner way to set this:
  Kint::$maxLevels = 4;
}
```

Replacing deprecated functions

https://drupal.stackexchange.com/questions/144147/get-taxonomy-terms

Missing module

from: https://www.drupal.org/node/2487215

If you see a PHP warning such as `The following module is missing from the file system...` (or similar) on your site, You can remove it with:

```
$ drush sql-query "DELETE FROM key_value WHERE name='module_name';"
```

You have requested a non-existent service

```
Symfony\Component\DependencyInjection\Exception\ServiceNotFoundException: You have requested a non-existent service "lingotek.content_translation". in /var/www/vendor/
```

Sometimes, when drush cr throws errors like that try `drush sqlc` and then `truncate cache_bootstrap` and `truncate cache_discovery`.

Resources

- Composer best practices for Drupal 8 from Lullabot Jan 2018
- Why DDEV by Randy Fay (Author of DDEV) from Dec 2022
- How to setup Devel and Kint on Drupal 9 by Alex Aug 2021

Back to top

Page last modified: May 14 2023.

Edit this page on GitHub

This site uses Just the Docs, a documentation theme for Jekyll.