From Lando To DDEV

A side by side migration

Bernardo Martinez 21/3/2023

https://www.drupal.org/u/bernardm28
https://www.linkedin.com/in/bernardm28

Goals

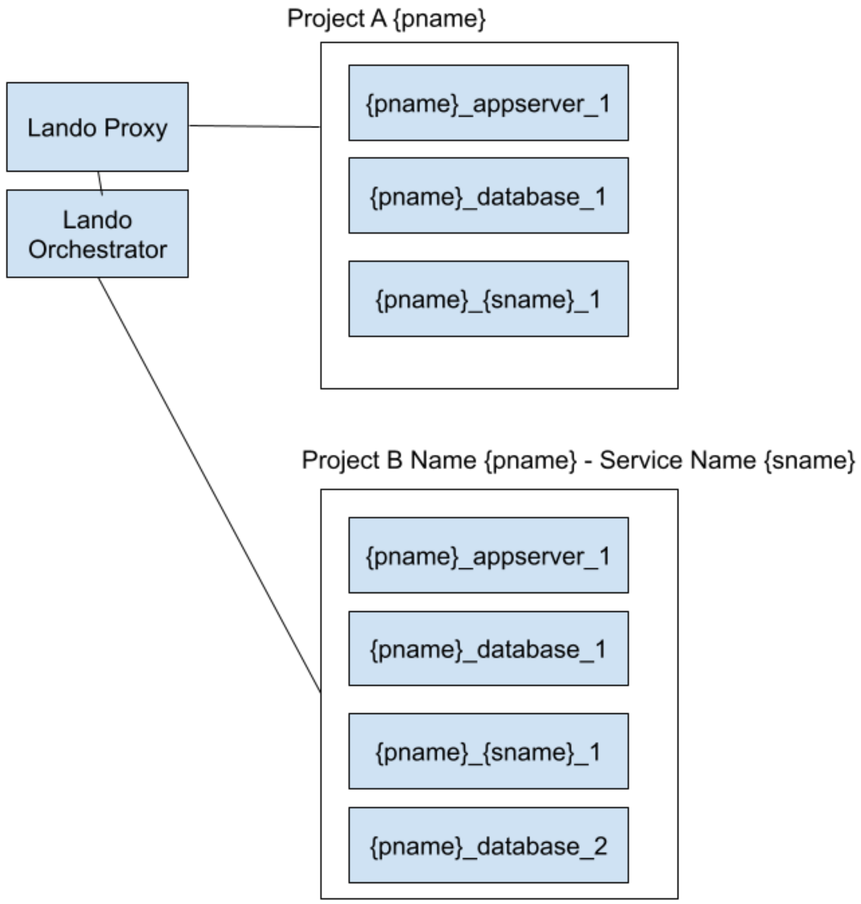
What you might get from this talk

- An overview of some tradeoffs between Lando and DDEV.
- How to migrate an astro project and two Drupal websites.
- An introduction to DDEV community and where to ask for support.

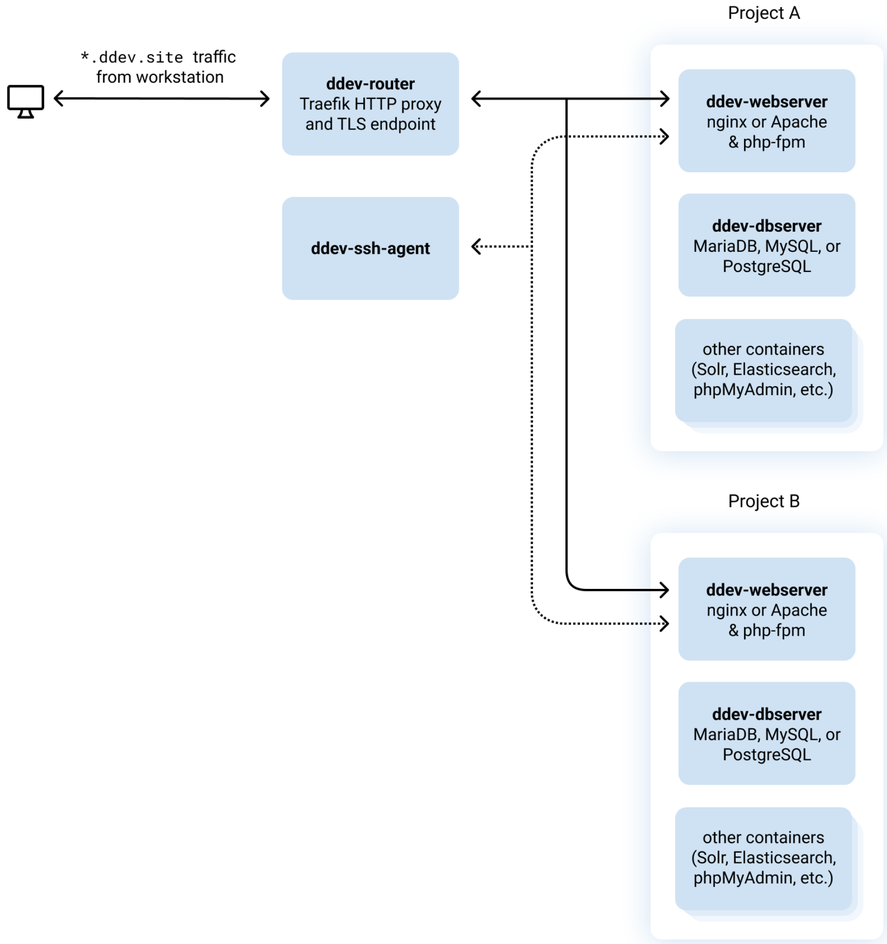
Table of content

Topics covered

LANDO vs DDEV architecture



https://docs.lando.dev/core/v3/networking.html



https://ddev.readthedocs.io/en/latest/users/usage/architecture/

Common Commands

Most used Commands

- ddev start
- ddev describe
- ddev restart

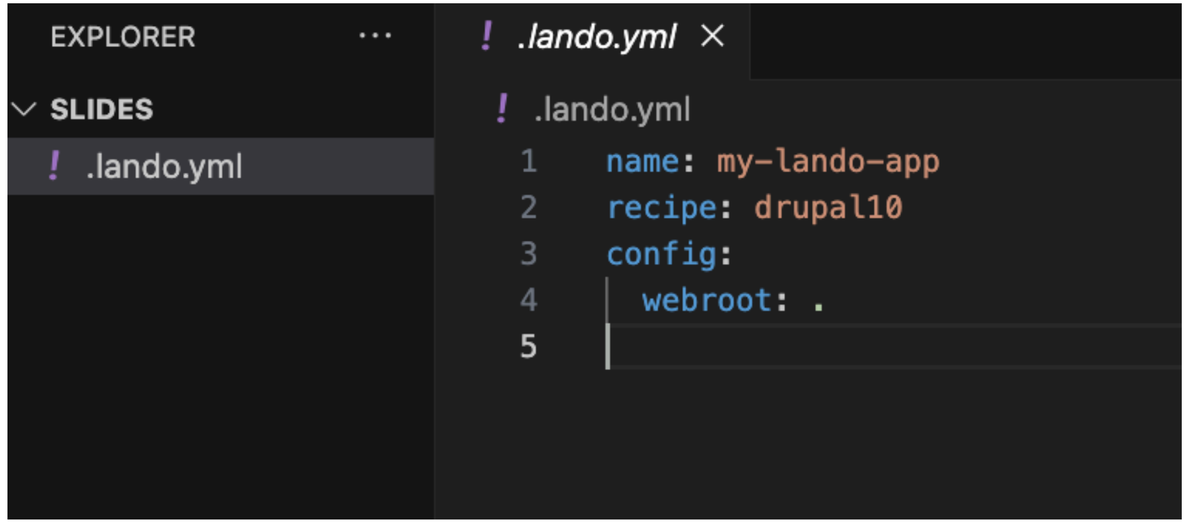
Lando	DDEV	Action
start	start	starts containers
info	describe	containers info
init	config	initial config
config		displays config
rebuild	restart	
restart	restart	
destroy	delete	
	launch	opens browser
poweroff	poweroff	
	snapshot	
share	share	
setup		loads plugins
update		updates plugins
	auth ssh	Loads ssh keys

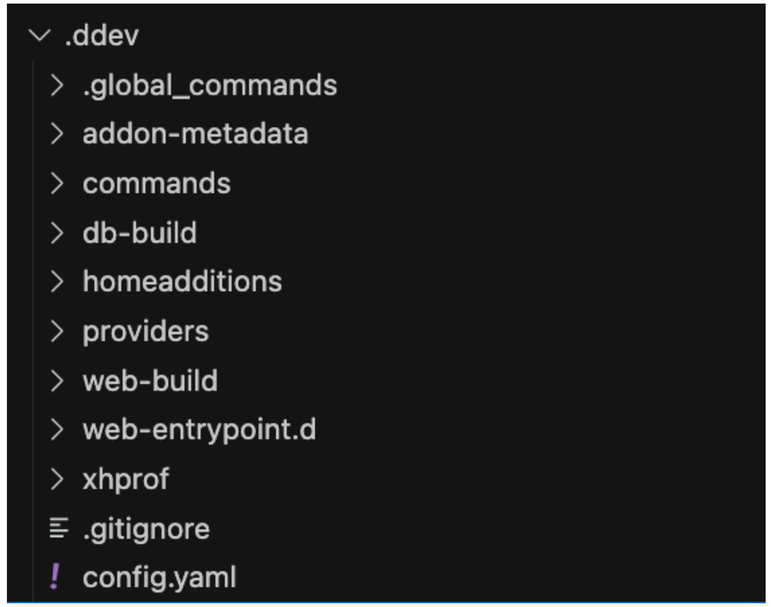
https://docs.lando.dev/cli/

https://ddev.readthedocs.io/en/latest/users/usage/commands/

Project structure

Lando init vs ddev config

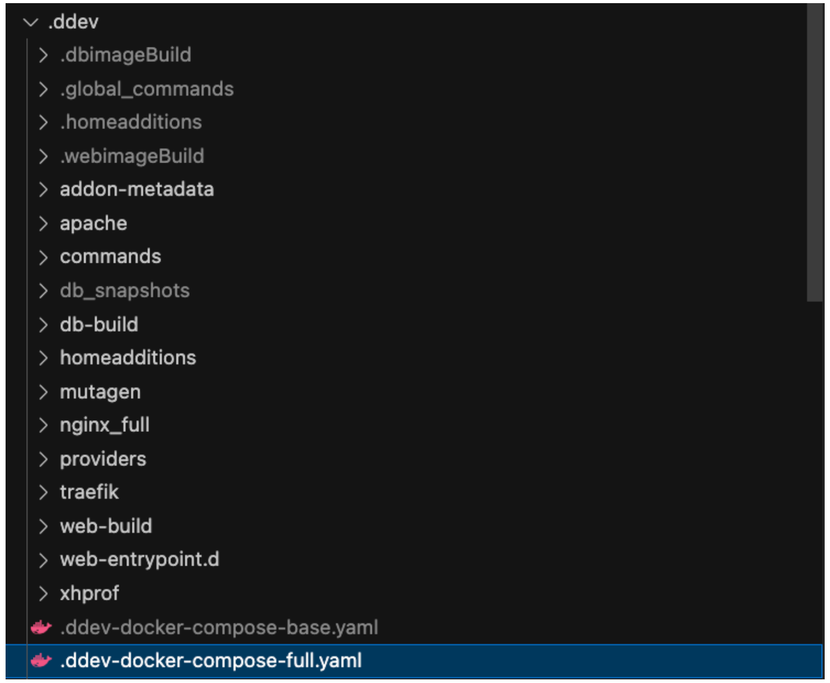




DDEV includes 263 lines of comments explaining each inside config.yml. They can be deleted but they explain the boilerplate code.

DDEV Describe

Compared to lando info and lando config




```
Project: ddev.com ~/Projects/ddev/ddev-website/ddev.com https://ddev.com.ddev.site
Docker platform: docker-desktop
Router: traefik
```

SERVICE	STAT	URL/PORT	INFO
web	OK	https://ddev.com.ddev.site InDocker: web:4321,443,80,8025 Host: 127.0.0.1:55956,55957	php PHP8.1 nginx-fpm docroot:'dist' Perf mode: mutagen NodeJS:20
Mailpit		Mailpit: https://ddev.com.ddev.site:8026 `ddev mailpit`	
astro-dev		InDocker: localhost:4321 https://ddev.com.ddev.site:4321 http://ddev.com.ddev.site:4322	
All URLs		https://ddev.com.ddev.site, https://127.0.0.1:55956, http://ddev.com.ddev.site, http://127.0.0.1:55957	

DDEV perks

Items that don’t need to be migrated

- NPM built-in
- XDebug built-in
- Drush alias built-in
- DB Snapshots
- Bash conditional logic
- Offline support

Enable Xdebug by adding the `xdebug: true` line to your `.lando.yml`.

When using a recipe, add it under the `config` key:

```yml
name: mywebsite
recipe: drupal10
config:
  xdebug: true
```

Otherwise, override your php service, usually named `appserver`:

```yml
name: mywebsite
services:
  appserver:
    webroot: web
    xdebug: true
```

Rebuild your environment.

```sh
lando rebuild -y
```

Finally, create a custom `launch.json` file in your workspace in order to map paths so that XDebug

Add Node Service

Assuming you're starting with a "Lando-ized" app, open the `.lando` directory. In this example we'll assume you're using a very basic LAM

To install our frontend tooling we need to be able to run Node. Fortur a basic Node service to our app:

```yml
name: myapp
recipe: lamp

services:
  node:
    type: node
    build:
      - npm install
```

Note the `build` section. This bash command will automatically run when we any Node packages specified in our package.json file.

```
services:
  appserver:
    environment:
      DRUSH_OPTIONS_URI: "https:/mysite.lndo.site/"
```

Making Tooling Available on the CLI

Almost there! All our services are installed, but how do we run a command on the fly, say starting a watch task or running `lando npm install hot-new-thing` to start experimenting with a new package? We could SSH into our node container, but that's SO 2016. Instead, we'll expose our new tooling via the CLI by adding this `tooling` section to our `.lando.yml` file:

```yml
tooling:
  npm:
    service: node
  node:
    service: node
  gulp:
    service: node
  yarn:
    service: node
```

After restarting your app, you should be able to run `lando node`, `lando gulp` or `lando npm` and have the corresponding commands run. This is particularly useful if you want to kickoff a watch task you might have configured, say `lando gulp watch`.

https://docs.lando.dev/node/frontend-tooling.html

First Migration - events, tooling, build, proxy

- Events -> Hooks
- Tooling -> DDEV Custom commands
- Build config -> DDEV custom images or ddev hooks.
- Proxy -> DDEV router
- Extra:
  - Conditional bash logic.
  - Color coding output

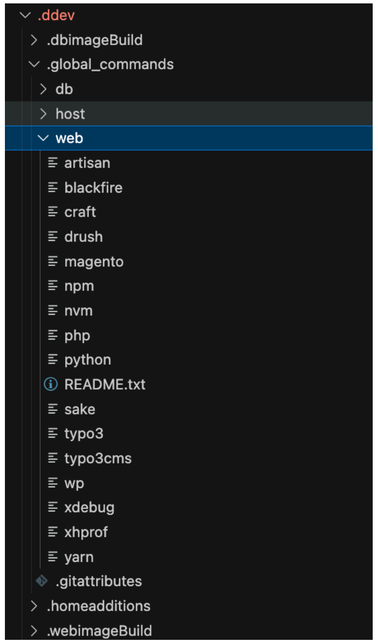
Lando tooling vs DDEV custom commands

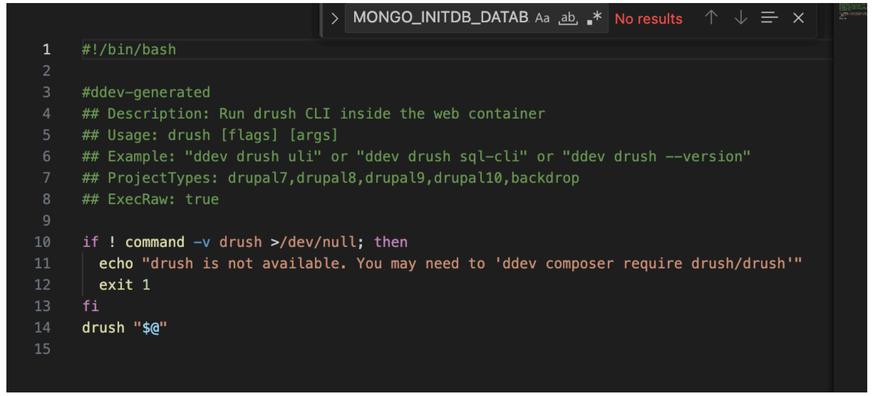
Placement and steps required.

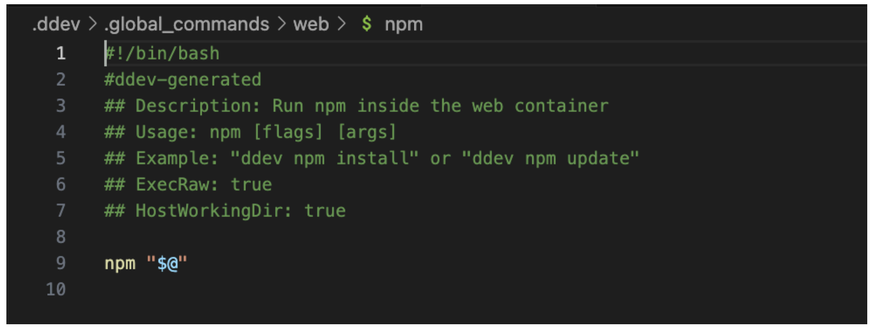
```yaml
tooling:
  build:
    description: Manually invokes all our build steps
    cmd:
      - appserver: composer install
      - node: yarn install
      - node: yarn sass
  test:
    description: Run ALL THE TESTS
    cmd:
      - appserver: composer test
      - node: yarn test
```

```bash
lando test && lando build
```

https://docs.lando.dev/core/v3/tooling.html







https://docs.lando.dev/lando-101/lando-tooling.html
https://ddev.readthedocs.io/en/latest/users/extend/custom-commands/#notes-for-all-command-types

Lando Events vs DDEV hooks

LANDO	APP
pre-bootstrap-config	pre-destroy
pre-bootstrap-tasks	post-destroy
pre-bootstrap-engine	pre-init
pre-bootstrap-app	post-init
post-bootstrap-config	pre-rebuild
post-bootstrap-tasks	post-rebuild
post-bootstrap-engine	pre-start
post-bootstrap-app	post-start
pre-engine-build	pre-stop
post-engine-build	post-stop
pre-engine-destroy	pre-uninstall
post-engine-destroy	post-uninstall
pre-engine-run	ready
post-engine-run	
pre-engine-start	
post-engine-start	
pre-engine-stop	
post-engine-stop	

You can also hook into `pre` and `post` events for all tooling commands. For example, the command `lando db-import` should expose `pre-db-import` and `post-db-import`.

https://ddev.readthedocs.io/en/latest/users/configuration/hooks/
https://docs.lando.dev/core/v3/events.html

Supported Command Hooks

- `pre-start`: Hooks into `ddev start`. Execute tasks before the project environment starts.

> **Tip**
>
> Only `exec-host` tasks can run during `pre-start` because the containers are not yet running. See Supported Tasks below.

- `post-start`: Execute tasks after the project environment has started.
- `pre-import-db` and `post-import-db`: Execute tasks before or after database import.
- `pre-import-files` and `post-import-files`: Execute tasks before or after files are imported.
- `pre-composer` and `post-composer`: Execute tasks before or after the `composer` command.
- `pre-stop`, `pre-config`, `post-config`, `pre-exec`, `post-exec`, `pre-pull`, `post-pull`, `pre-push`, `post-push`, `pre-snapshot`, `post-snapshot`, `pre-restore-snapshot`, `post-restore-snapshot`: Execute as the name suggests.
- `post-stop`: Hooks into `ddev stop`. Execute tasks after the project environment stopped.

> **Tip**
>
> Only `exec-host` tasks can run during `post-stop`. See Supported Tasks below.

Lando Proxy vs DDEV router

Both use traefik

Routing to a different port

You can suffix the domain with `:PORT` to change the default `port` from `80` to `PORT`. Note that this is the port that your service exposes from within Lando and not an external port. In the below example, this means that `appserver` exposes port `8888` and we want `myapp.lndo.site` to route our request into Lando at `appserver:8888`.

```
proxy:
  appserver:
    - myapp.lndo.site:8888
```

```
services:
  node:
    type: node:20
    port: 4321
    ssl: 4321
    build:
      - npm install
    command: npm run dev -- --host

proxy:
  node:
    - astro.lndo.site:4321
```

Exposing Extra Ports via `ddev-router`

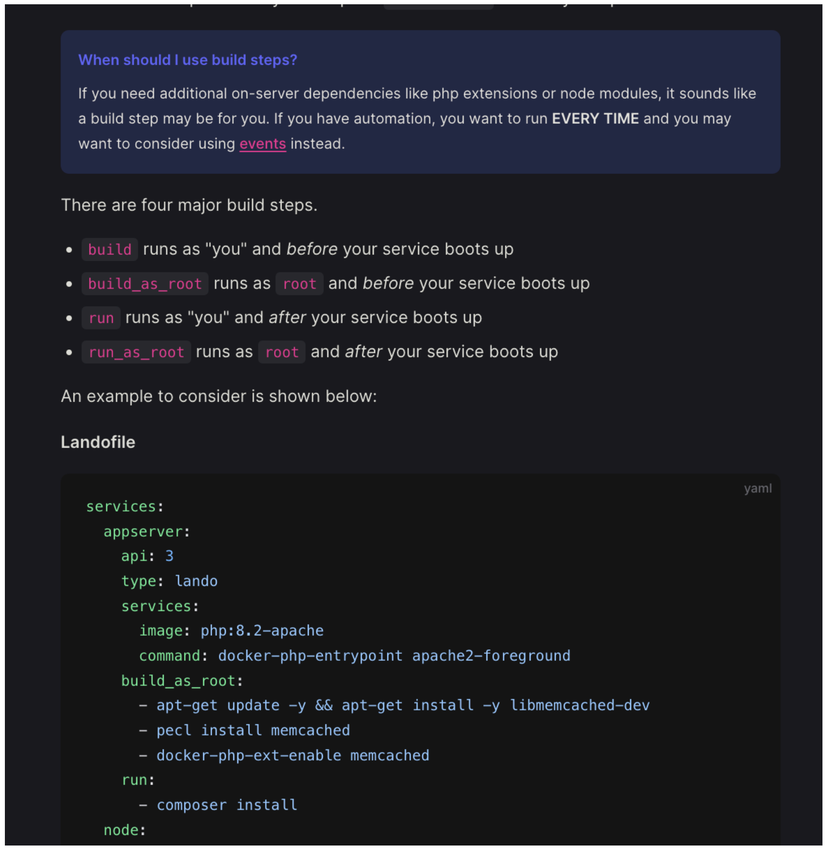
If your `web` container has additional HTTP servers running inside it on different ports, those can be exposed using `web_extra_exposed_ports` in `.ddev/config.yaml`. For example, this configuration would expose a `node-vite` HTTP server running on port 3000 inside the `web` container, via `ddev-router`, to ports 9998 (HTTP) and 9999 (HTTPS), so it could be accessed via `https://<project>.ddev.site:9999`:

```
web_extra_exposed_ports:
  - name: node-vite
    container_port: 3000
    http_port: 9998
    https_port: 9999
```

https://ddev.readthedocs.io/en/stable/users/extend/traefik-router/
https://docs.lando.dev/core/v3/proxy.html

Lando build vs DDEV (custom images or hooks with bash conditional)

Adding extra settings to a given container



When should I use build steps?

If you need additional on-server dependencies like php extensions or node modules, it sounds like a build step may be for you. If you have automation, you want to run **EVERY TIME** and you may want to consider using events instead.

There are four major build steps.

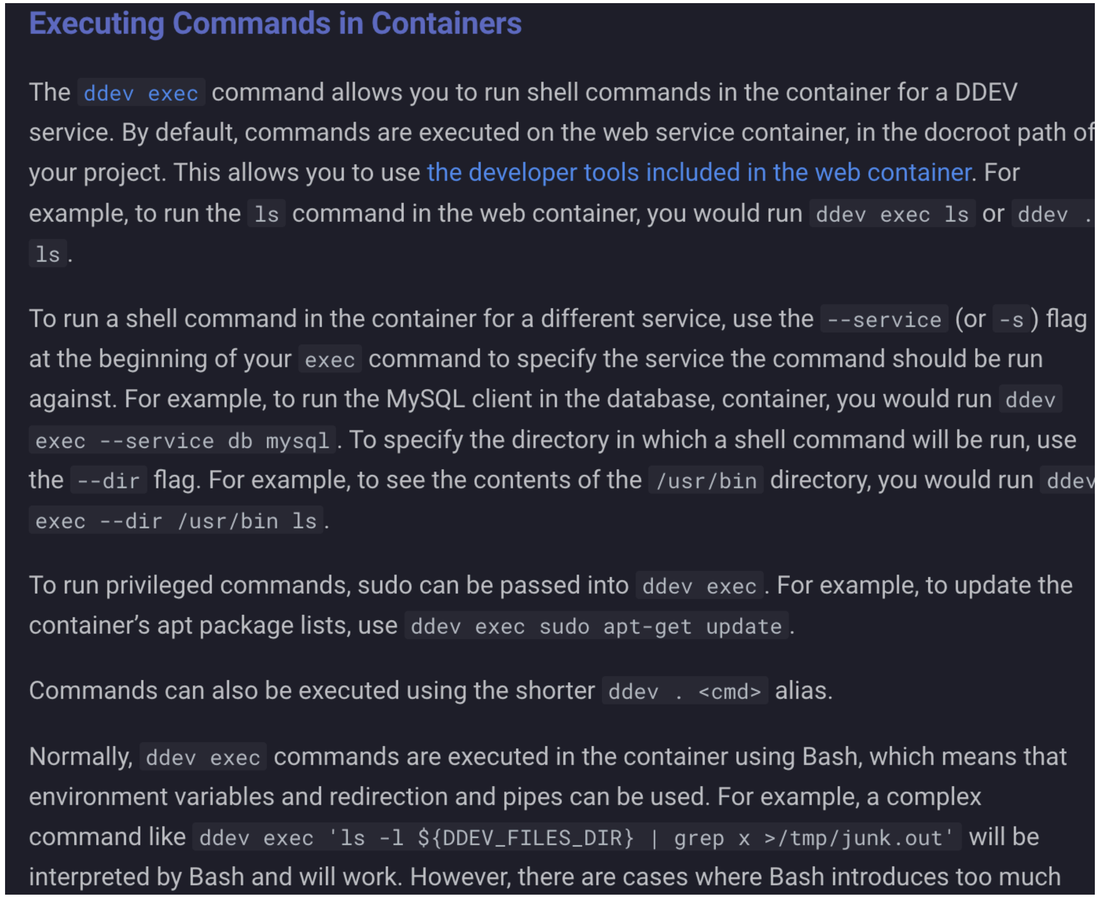
- `build` runs as "you" and *before* your service boots up
- `build_as_root` runs as `root` and *before* your service boots up
- `run` runs as "you" and *after* your service boots up
- `run_as_root` runs as `root` and *after* your service boots up

An example to consider is shown below:

Landofile

```yaml
services:
  appserver:
    api: 3
    type: lando
    services:
      image: php:8.2-apache
      command: docker-php-entrypoint apache2-foreground
    build_as_root:
      - apt-get update -y && apt-get install -y libmemcached-dev
      - pecl install memcached
      - docker-php-ext-enable memcached
    run:
      - composer install
  node:
```

https://docs.lando.dev/core/v3/lando-service.html#build-steps



Executing Commands in Containers

The `ddev exec` command allows you to run shell commands in the container for a DDEV service. By default, commands are executed on the web service container, in the docroot path of your project. This allows you to use the developer tools included in the web container. For example, to run the `ls` command in the web container, you would run `ddev exec ls` or `ddev . ls`.

To run a shell command in the container for a different service, use the `--service` (or `-s`) flag at the beginning of your `exec` command to specify the service the command should be run against. For example, to run the MySQL client in the database, container, you would run `ddev exec --service db mysql`. To specify the directory in which a shell command will be run, use the `--dir` flag. For example, to see the contents of the `/usr/bin` directory, you would run `ddev exec --dir /usr/bin ls`.

To run privileged commands, sudo can be passed into `ddev exec`. For example, to update the container's apt package lists, use `ddev exec sudo apt-get update`.

Commands can also be executed using the shorter `ddev . <cmd>` alias.

Normally, `ddev exec` commands are executed in the container using Bash, which means that environment variables and redirection and pipes can be used. For example, a complex command like `ddev exec 'ls -l ${DDEV_FILES_DIR} | grep x >/tmp/junk.out'` will be interpreted by Bash and will work. However, there are cases where Bash introduces too much

https://ddev.readthedocs.io/en/latest/users/usage/cli/#snapshotting-and-restoring-a-database

Environment variables (provided)

Comparison of environment variables

For reference, an example of the default container envvars inside of the LAMP recipe/example is shown below:

```
LANDO_WEBROOT_USER=www-data
LANDO_WEBROOT_GROUP=www-data
LANDO_WEBROOT_UID=33
LANDO_WEBROOT_GID=33
LANDO_HOST_UID=501
LANDO_HOST_GID=20
LANDO_HOST_USER=me
LANDO_CA_CERT=/lando/certs/lndo.site.pem
LANDO_CA_KEY=/lando/certs/lndo.site.key
LANDO_CONFIG_DIR=/Users/pirog/.lando
LANDO_DOMAIN=lndo.site
LANDO_HOST_HOME=/Users/pirog
LANDO_HOST_OS=darwin
LANDO_HOST_IP=host.docker.internal
LANDO_MOUNT=/app
LANDO_APP_NAME=lamp
LANDO_APP_ROOT=/Users/pirog/work/lando/examples/lamp
LANDO_APP_ROOT_BIND=/Users/pirog/work/lando/examples/lamp
LANDO_INFO=[{"service":"appserver","urls":["http://lamp.lndo.site","https://lam
LANDO_WEBROOT=/app/.
LANDO_SERVICE_TYPE=php
LANDO_SERVICE_NAME=appserver
```

Useful variables for container scripts are:

- `DDEV_DOCROOT`: Relative path from approot to docroot
- `DDEV_FILES_DIR`: *Deprecated*, first directory of user-uploaded files
- `DDEV_FILES_DIRS`: Comma-separated list of directories of user-uploaded files
- `DDEV_HOSTNAME`: Comma-separated list of FQDN hostnames
- `DDEV_MUTAGEN_ENABLED`: `true` if Mutagen is enabled
- `DDEV_PHP_VERSION`: Current PHP version
- `DDEV_PRIMARY_URL`: Primary URL for the project
- `DDEV_PROJECT`: Project name, like `d8composer`
- `DDEV_PROJECT_TYPE`: `drupal8`, `typo3`, `backdrop`, `wordpress`, etc.
- `DDEV_ROUTER_HTTP_PORT`: Router port for HTTP
- `DDEV_ROUTER_HTTPS_PORT`: Router port for HTTPS
- `DDEV_SITENAME`: Project name, like `d8composer`
- `DDEV_TLD`: Top-level project domain, like `ddev.site`
- `DDEV_WEBSERVER_TYPE`: `nginx-fpm`, `apache-fpm`, or `nginx-gunicorn`
- `IS_DDEV_PROJECT`: If `true`, PHP is running under DDEV

https://docs.lando.dev/core/v3/env.html#default-environment-variables

https://ddev.readthedocs.io/en/latest/users/extend/custom-commands/#environment-variables-provided

Bash conditional logic

```
web_extra_daemons:
  - name: astro-dev-daemon
    command: bash -c 'npm install && touch /var/tmp/npminstalldone && npm run dev -- --host'
    directory: /var/www/html

hooks:
  post-start:
    - exec: bash -c 'while [ ! -f /var/tmp/npminstalldone ]; do sleep 1; done'
    - exec: bash -c 'if [ ! -d /var/www/html/dist ]; then npm run build; fi'
    - exec: echo -e "                                        NOTICE\n
To troubleshoot any isues run \e[35mddev describe\e[0m or \e[35mddev logs --follow --time\e[0m \n"
```

What is \e[35mddev?

```
Starting ddev-router if necessary...
 Container ddev-router  Running
Waiting for additional project containers to become ready...
All project containers are now ready.
                                NOTICE
 ================================================================================
 ================================================================================
 The Astro dev container is ready
 Hot Module Reloadin (HMR) is avaliable at https://ddev.com.ddev.site:4321
 To troubleshoot any isues run ddev describe or ddev logs --follow --time

Successfully started ddev.com
Project can be reached at https://ddev.com.ddev.site https://127.0.0.1:62053
```

Example

```
! .lando.yml
1   name: demo
2   recipe: lamp
3   config:
4     webroot: dist
5   services:
6     node:
7       type: node:20
8       port: 4321
9       ssl: 4321
10      build:
11        - npm install
12      command: npm run dev -- --host
13
14  proxy:
15    node:
16      - astro.lndo.site:4321
17
18  tooling:
19    npm:
20      service: node
```

What does it do?

Adds a node service
Exposes and maps port 4321
Installs the npm packages
Runs the command on a background thread

Creates a new url for node service

Adds the npm command

.ddev > ! config.yaml

Find Aa ab .* No result

You, 37 seconds ago | 3 authors (Bernardo Martinez and others)

```yaml
name: ddev.com
type: php
docroot: dist
php_version: "8.1"
webserver_type: nginx-fpm
xdebug_enabled: false
additional_hostnames: []
additional_fqdns: []
use_dns_when_possible: true
composer_version: "2"
web_environment: []      Bernardo Martinez, 4 months ago • Add instructions to run locally with ddev (#111…
nodejs_version: "20"
omit_containers: ["db"]
disable_upload_dirs_warning: true
web_extra_exposed_ports:
    - name: astro-dev
      container_port: 4321
      http_port: 4322
      https_port: 4321
# The extra -- in `npm run dev -- --host` is a Vite requirement
# https://github.com/vitejs/vite/discussions/3396
web_extra_daemons:
  - name: astro-dev-daemon
    command: bash -c 'npm install && touch /var/tmp/npminstalldone && npm run dev -- --host'
    directory: /var/www/html
hooks:
  post-start:
    - exec: bash -c 'while [ ! -f /var/tmp/npminstalldone ]; do sleep 1; done'
    - exec: bash -c 'if [ ! -d /var/www/html/dist ]; then npm run build; fi'
    - exec: echo -e "                                NOTICE                                          \n
                    ================================================================================\n
                    ================================================================================\n
                     The Astro dev container is ready \n
                     Hot Module Reloadin (HMR) is avaliable at \e[32m${DDEV_PRIMARY_URL}:4321\e[0m \n
                     To troubleshoot any isues run \e[35mddev describe\e[0m or \e[35mddev logs --follow --time\e[0m \n"
```

Maps the container port to the host

Installs the packages and it adds the background thread

Waits for npm install to be done and runs the build

ddev.com config

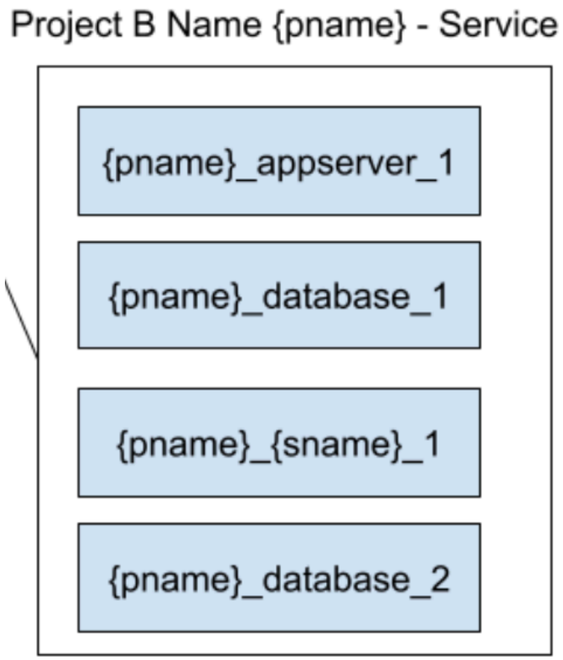
https://github.com/ddev/ddev.com/blob/main/.ddev/config.yaml

Second Migration - services, build steps, database

- Services -> custom commands
- Build steps -> custom docker images or hooks
- disable_settings_management - items to take into consideration.
- Database
- Lets create:
  - Drupal 10 project
  - Add npm
  - Add gulp or vite

Database Management

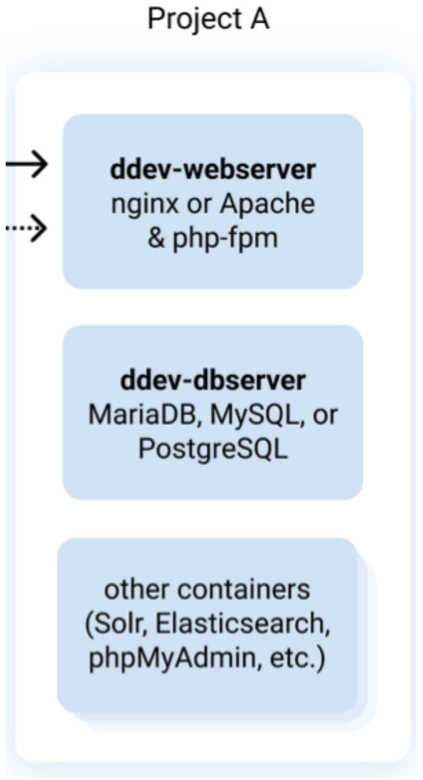
Lando vs DDEV



Lando adds a new docker image per database

Any kind of database is allowed as a new docker image gets created and added.

Creates performance bottle necks on multisite installs.



DDEV has one database image

One can add multiple databases of the same type on the one image.

Adding mysql and Postgres on one project requires a DDEV add-on or custom docker file.

https://docs.lando.dev/guides/db-import.html

https://ddev.readthedocs.io/en/stable/users/usage/database-management/

Handy database commands

One less item to worry about

- ddev tableplus
- ddev querious
- ddev dbeaver
- ddev sequelpro
- ddev sequence

https://ddev.readthedocs.io/en/stable/users/usage/database-management/

Environment variables (user generated)

Comparison of environment variables

You can accomplish this using the `env_file` top level config in your Landofile.

```yml
env_file:
  - defaults.env
  - extras/special.env
```

These files are relative to your projects root directory.

These files will need to take the form below. Note that this is **not a yaml file**.

```yml
DB_HOST=localhost
DB_USER=root
DB_PASS=s1mpl3
```

Providing Custom Environment Variables to a Container

You can set custom environment variables in several places:

- The project's `web_environment` setting in `.ddev/config.yaml` or `.ddev/config.*.yaml`:

```
web_environment:
- MY_ENV_VAR=someval
- MY_OTHER_ENV_VAR=someotherval
```

- The global `web_environment` setting in `.ddev/global_config.yaml`.
- An optional, project-level `.ddev/.env` file, which could look something like this:

```
MY_ENV_VAR='someval'
MY_OTHER_ENV_VAR='someotherval'
```

https://docs.lando.dev/core/v3/env.html

https://ddev.readthedocs.io/en/latest/users/extend/customization-extendibility/#providing-custom-environment-variables-to-a-container

Environment variables

DDEV vs Lando

```yaml
name: slides
type: drupal10
docroot: web
php_version: "8.1"
webserver_type: nginx-fpm
xdebug_enabled: false
additional_hostnames: []
additional_fqdns: []
database:
    type: mariadb
    version: "10.4"
use_dns_when_possible: true
web_environment:
    - DRUPAL_PRIVATE=../super/secret/path
    - DRUPAL_TEMP=../secret/location
    - DRUPAL_CONFIG_SYNC=../secret/path/config/sync
    - DRUSH_OPTIONS_URI=$DDEV_PRIMARY_URL #only needed if one disables settings management.
composer_version: "2"
web_environment: []
```

- An optional, project-level `.ddev/.env` file, which could look something like this:

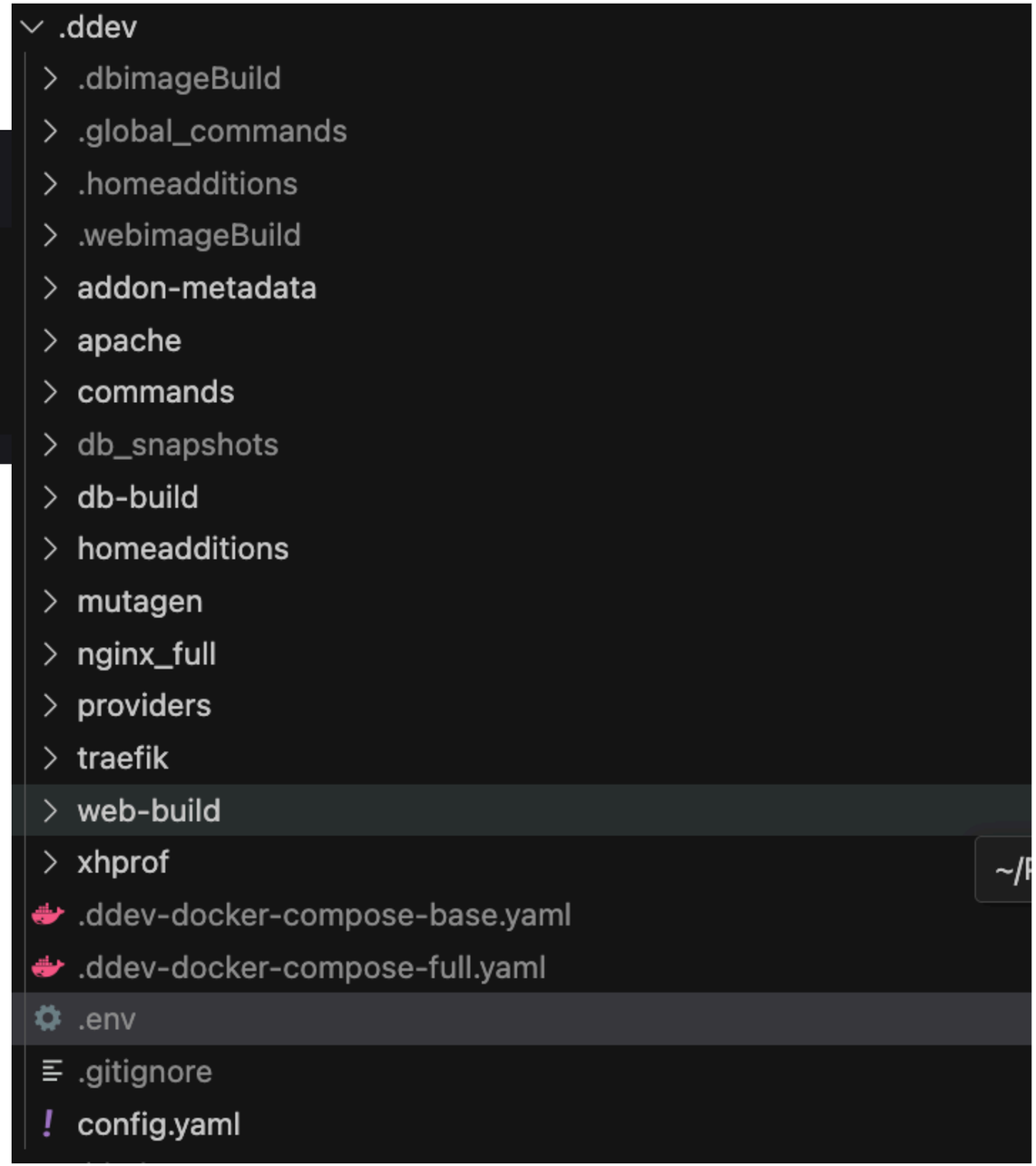
```
MY_ENV_VAR='someval'
MY_OTHER_ENV_VAR='someotherval'
```

https://ddev.readthedocs.io/en/latest/users/extend/customization-extendibility/#providing-custom-environment-variables-to-a-container

DDEV .env files

You can accomplish this using the `env_file` top level config in your Landofile.

```
env_file:
  - defaults.env
  - extras/special.env
```



Example

! .lando.yml

You, 1 second ago | 2 authors (Bernardo Martinez and others)

```yaml
name: demo
recipe: drupal10
config:
  webroot: docroot
  xdebug: false
  php: '8.2'
services:
  appserver:
    xdebug: false
    build_as_root:
      - apt-get update
      - apt-get install libxss1
      - echo "Run additional build commands here. Run lando rebuild after updating this file."
      - curl -sL https://deb.nodesource.com/setup_14.x | bash -
      - apt install -y nodejs
    overrides:
      # Pass SSH keys.
      volumes:
        - type: bind
          # Linux user: add 'export LANDO_SSH_AUTH_SOCK="${SSH_AUTH_SOCK}"' at the end of your ~/.bashrc:
          # Mac user: MacOS specific path is here as the variable default value, nothing to do.
          source: "${LANDO_SSH_AUTH_SOCK:-/run/host-services/ssh-auth.sock}"
          target: /run/host-services/ssh-auth.sock
      environment:
        DRUSH_OPTIONS_URI: "https://mywebsite.lndo.site/"      You, now • Uncommitted changes
        SSH_AUTH_SOCK: /run/host-services/ssh-auth.sock
      cap_add:
        - SYS_ADMIN
tooling:
  blt:
    service: appserver
    cmd: /app/vendor/bin/blt
  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
  ssh-fix:
    service: appserver
    description: Fix ssh auth sock permission for MacOS users. Lando rebuild fixes the problem as well.
    cmd: "/bin/chgrp www-data /run/host-services/ssh-auth.sock && /bin/chmod g+w /run/host-services/ssh-auth.sock"
    user: root
events:
  post-start:
    - appserver: test -e ~/.ssh/config || printf 'Host *\n  AddKeysToAgent yes\n' > ~/.ssh/config

```

4 things everyone does

- Install node some way through a service, bash file or build command.
- Add drush URI option
- Add xdebug
- Patch bugs (ssh-fix or otherwise)

https://github.com/Drupal4Gov/drupal4gov.us/blob/develop/.lando.yml

```yaml
name: drupal4gov
recipe: acquia
config:
  webroot: docroot
  xdebug: false
  php: '8.1'
services:
  appserver:
    xdebug: false
    build_as_root:
      - apt-get update
      - apt-get install libxss1
      - echo "Run additional build commands here. Run lando rebuild after updating this file."
      - curl -sL https://deb.nodesource.com/setup_14.x | bash -
      - apt install -y nodejs
    overrides:
      # Pass SSH keys.
      volumes:
        - type: bind
          # Linux user: add 'export LANDO_SSH_AUTH_SOCK="${SSH_AUTH_SOCK}"' at the end of your ~/.bashrc:
          # Mac user: MacOS specific path is here as the variable default value, nothing to do.
          source: "${LANDO_SSH_AUTH_SOCK:-/run/host-services/ssh-auth.sock}"
          target: /run/host-services/ssh-auth.sock
      environment:
        DRUSH_OPTIONS_URI: "https://drupal4gov.lndo.site/"
        SSH_AUTH_SOCK: /run/host-services/ssh-auth.sock
      cap_add:
        - SYS_ADMIN
tooling:
  blt:
    service: appserver
    cmd: /app/vendor/bin/blt
  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
  ssh-fix:
    service: appserver
    description: Fix ssh auth sock permission for MacOS users. Lando rebuild fixes the problem as well.
    cmd: "/bin/chgrp www-data /run/host-services/ssh-auth.sock && /bin/chmod g+w /run/host-services/ssh-auth.sock"
    user: root
events:
  post-start:
    - appserver: test -e ~/.ssh/config || printf 'Host *\n  AddKeysToAgent yes\n' > ~/.ssh/config
```

.lando.yml

You, 1 second ago | 2 authors (Bernardo Martinez and others)

```yaml
name: demo
recipe: drupal10
config:
  webroot: docroot
  xdebug: false
  php: '8.2'
services:
  appserver:
    xdebug: false
    build_as_root:
      - apt-get update
      - apt-get install libxss1
      - echo "Run additional build commands here. Run lando rebuild after updating this file."
      - curl -sL https://deb.nodesource.com/setup_14.x | bash -
      - apt install -y nodejs
    overrides:
      # Pass SSH keys.
      volumes:
        - type: bind
          # Linux user: add 'export LANDO_SSH_AUTH_SOCK="${SSH_AUTH_SOCK}"' at the end of your ~/.bashrc:
          # Mac user: MacOS specific path is here as the variable default value, nothing to do.
          source: "${LANDO_SSH_AUTH_SOCK:-/run/host-services/ssh-auth.sock}"
          target: /run/host-services/ssh-auth.sock
      environment:
        DRUSH_OPTIONS_URI: "https://mywebsite.lndo.site/"      You, now • Uncommitted changes
        SSH_AUTH_SOCK: /run/host-services/ssh-auth.sock
      cap_add:
        - SYS_ADMIN
tooling:
  blt:
    service: appserver
    cmd: /app/vendor/bin/blt
  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
  ssh-fix:
    service: appserver
    description: Fix ssh auth sock permission for MacOS users. Lando rebuild fixes the problem as well.
    cmd: "/bin/chgrp www-data /run/host-services/ssh-auth.sock && /bin/chmod g+w /run/host-services/ssh-auth.sock"
    user: root
events:
  post-start:
    - appserver: test -e ~/.ssh/config || printf 'Host *\n  AddKeysToAgent yes\n' > ~/.ssh/config

```

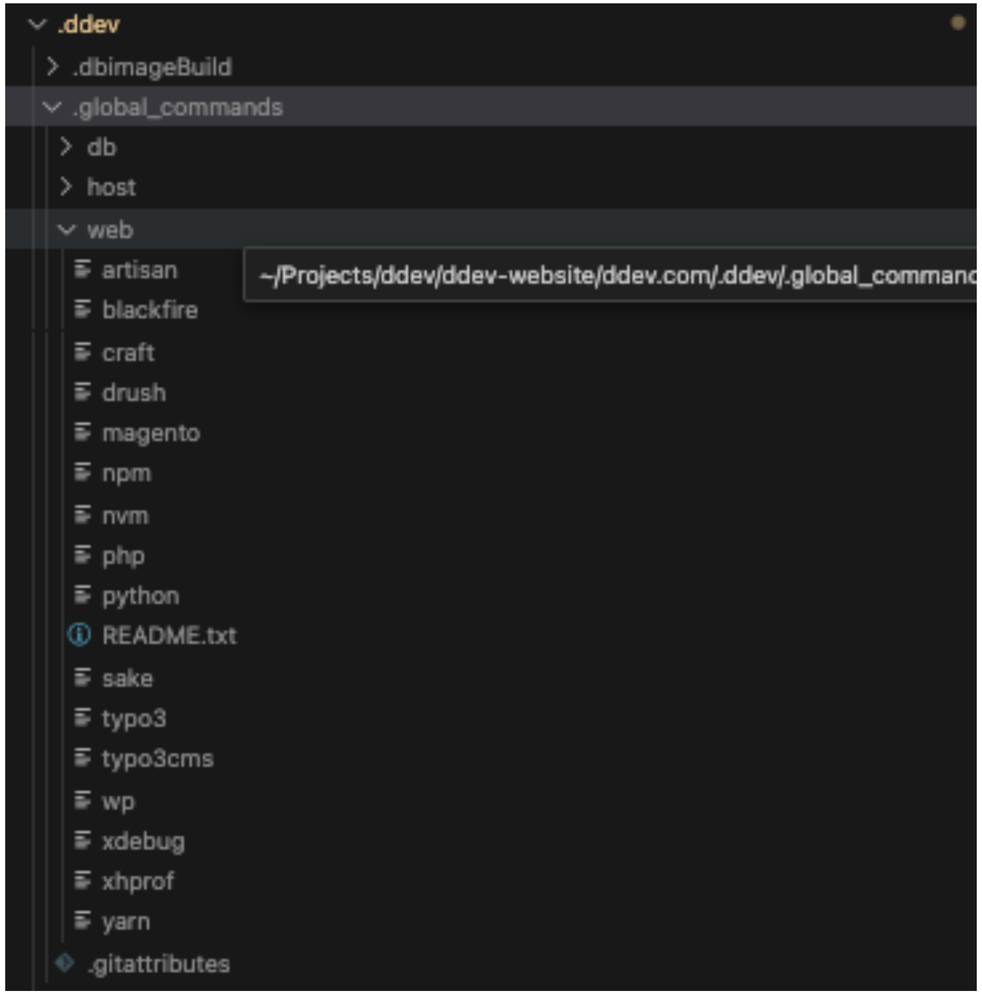
```
name: yourwebsite
type: drupal10
docroot: web
php_version: "8.2"
webserver_type: nginx-fpm
xdebug_enabled: false
additional_hostnames: []
additional_fqdns: []
database:
    type: mariadb
    version: "10.11"
performance_mode: mutagen
use_dns_when_possible: true
composer_version: “2"
nodejs_version: "20"
web_environment:
    - DRUPAL_PRIVATE=../web/sites/sbagov/files/private
    - DRUPAL_CONFIG_SYNC=../config/sync
    - DRUSH_OPTIONS_URI=$DDEV_PRIMARY_URL
disable_settings_management: true
```

What might not be needed

```
disable_settings_management: true
DRUSH_OPTIONS_URI=$DDEV_PRIMARY_URL
```

But where are NPM and Xdebug?

They are built-in commands.



disable_settings_management

What does it do?

Tells DDEV to use a specific project type without creating setting files and creates creates the .ddev/.gitignore

web/sites/default/settings.ddev.php

web/sites/default/settings.php

```
// Automatically generated include for settings managed by ddev.
if (getenv('IS_DDEV_PROJECT') == 'true' && file_exists(__DIR__ . '/settings.ddev.php')) {
  include __DIR__ . '/settings.ddev.php';
}
```

https://ddev.readthedocs.io/en/latest/users/usage/cms-settings/#controlling-or-removing-cms-settings

IS_LANDO_PROJECT?

Might be a good workaround

web/sites/default/settings.php
web/sites/default/settings.local.php
web/sites/default/settings.ddev.php
web/sites/default/settings.lando.php

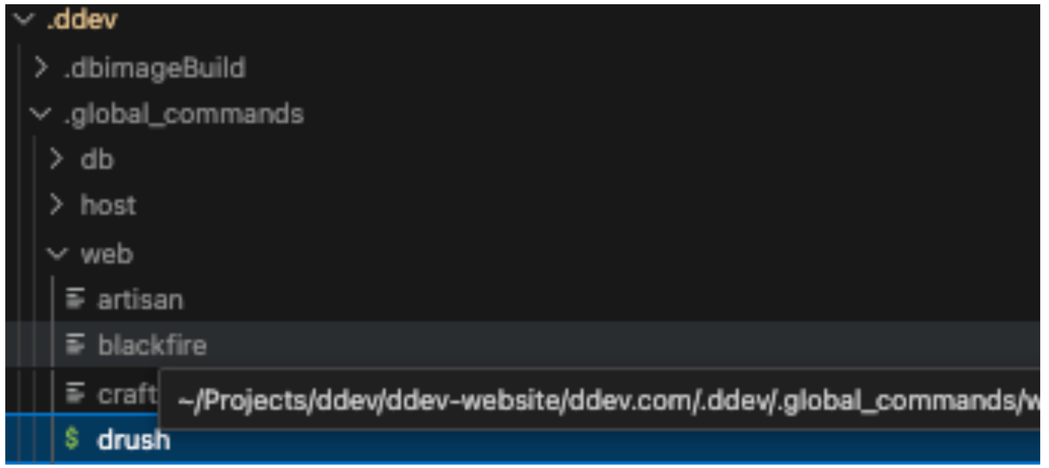
```php
// Automatically generated include for settings managed by ddev.
if (getenv('IS_DDEV_PROJECT') == 'true' && file_exists(__DIR__ . '/settings.ddev.php')) {
  include __DIR__ . '/settings.ddev.php';
}

/**
 * Load local development override configuration, if available.
 *
 * Create a settings.local.php file to override variables on secondary (staging,
 * development, etc.) installations of this site.
 *
 * Typical uses of settings.local.php include:
 * - Disabling caching.
 * - Disabling JavaScript/CSS compression.
 * - Rerouting outgoing emails.
 *
 * Keep this code block at the end of this file to take full effect.
 */
#

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

```php
if (getenv('LANDO_INFO')) {
   $lando_info = json_decode(getenv('LANDO_INFO'), TRUE);

   $settings['trusted_host_patterns'] = ['.*'];
   $settings['hash_salt'] = md5(getenv('LANDO_HOST_IP'));

  $databases['default']['default'] = [
     'database' => $lando_info['database']['creds']['database'],
     'username' => $lando_info['database']['creds']['user'],
     'password' => $lando_info['database']['creds']['password'],
     'host' => $lando_info['database']['internal_connection']['host'],
     'port' => $lando_info['database']['internal_connection']['port'],
     'driver' => 'pgsql',
     'namespace' => 'Drupal\\Core\\Database\\Driver\\pgsql',
  ];
}
```

https://www.drupal.org/u/mr_scumbag

But what about the BLT command

```
tooling:
  blt:
    service: appserver
    cmd: /app/vendor/bin/blt
```




```
#!/bin/bash

#ddev-generated
## Description: Run drush CLI inside the web container
## Usage: drush [flags] [args]
## Example: "ddev drush uli" or "ddev drush sql-cli" or "ddev drush --version"
## ProjectTypes: drupal7,drupal8,drupal9,drupal10,backdrop
## ExecRaw: true

if ! command -v drush >/dev/null; then
  echo "drush is not available. You may need to 'ddev composer require drush/drush'"
  exit 1
fi
drush "$@"
```

Third Migration - Tools, env variables, add-ons

- Tools -> custom commands
- Build steps -> custom docker images or hooks
- Env variables
- Add-on services

Lando custom images vs DDEV custom images

Adding extra settings to a given container

2. Extending a Dockerfile

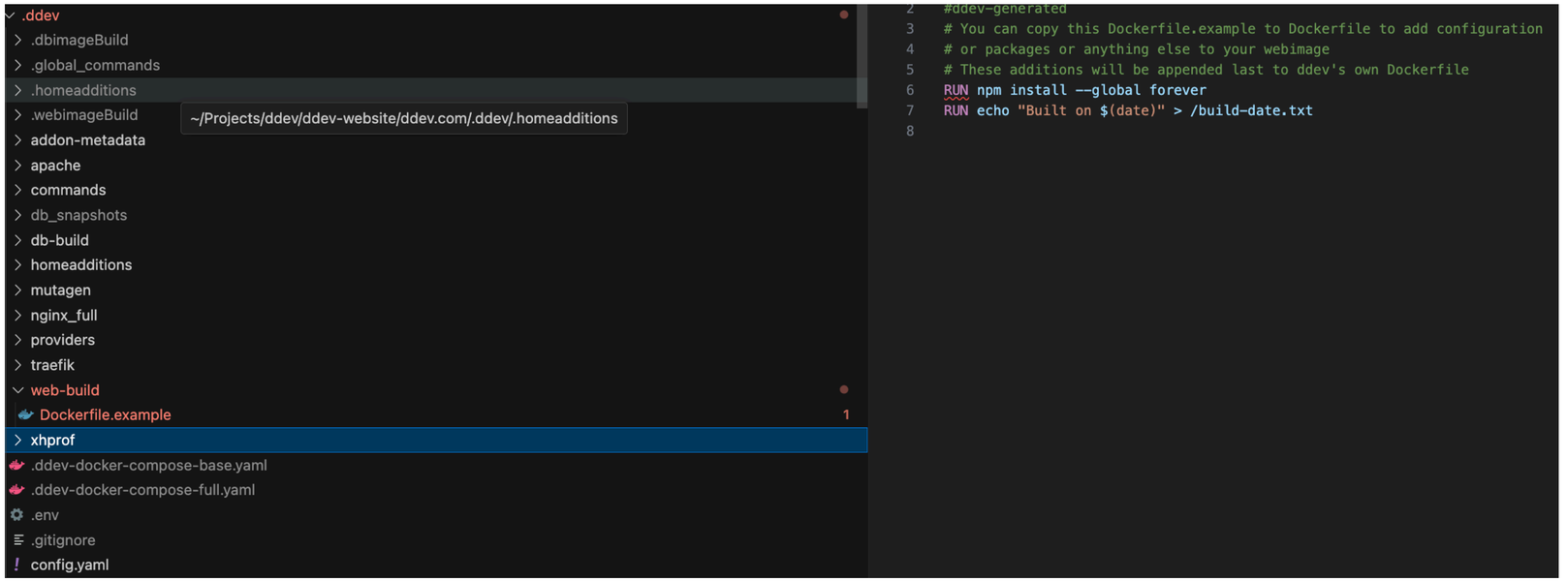
If you are planning to extend your service with *additional* build steps or would like to cache the build steps for a faster `lando rebuild` you should instead consider extending with a Dockerfile as in the example below:

.lando.yml

```
services:
  myservice:
    type: php:custom
    via: cli
    overrides:
      image: lando/php:7.4-with-node12
      build:
        context: ./
        dockerfile: Dockerfile.node
tooling:
  node:
    service: myservice
  npm:
    service: myservice
```

Dockerfile.node

```
FROM devwithlando/php:7.4-apache-2

# Choose the major node version
ENV NODE_VERSION=12

# Install node
RUN curl -sL "https://deb.nodesource.com/setup_${NODE_VERSION}.x" | bash - \
  && apt-get install -y nodejs
```



Adding Extra Dockerfiles for `webimage` and `dbimage`

For more complex requirements, you can add:

- `.ddev/web-build/Dockerfile`
- `.ddev/web-build/Dockerfile.*`
- `.ddev/db-build/Dockerfile`
- `.ddev/db-build/Dockerfile.*`

An example web image `.ddev/web-build/Dockerfile` might be:

```
RUN npm install -g gatsby-cli
```

https://docs.lando.dev/plugins/php/guides/installing-node.html#_2-extending-a-dockerfile

https://ddev.readthedocs.io/en/latest/users/extend/customizing-images/#adding-extra-dockerfiles-for-webimage-and-dbimage

LANDO plugins vs DDEVs Addons

A fairly basic example follows:

```
'use strict';

module.exports = (app, lando) => {
  // Run my custom script on all my containers after my app starts
  const buildServices = _.get(app, 'opts.services', app.services);
  app.events.on('post-start', () => lando.engine.run(_.map(buildServices, servi
    id: `${app.project}_${service}_1`,
    cmd: '/helpers/myscript.sh',
    compose: app.compose,
    project: app.project,
  })));

  // Mix in some envvars and docker labels we want to add to all our containers
  return {
    env: {
      WOODEN_SHIPS_FREE_AND_EASY: true,
      I_SAW: 'the sign',
    },
    labels: {
      'io.lando.danger-factor': 11,
    },
  };
};
```

https://docs.lando.dev/contrib/coder

https://ddev.readthedocs.io/en/latest/users/extend/additional-services/

ddev-contrib / docker-compose-services / mongodb / **docker-compose.mongo.yaml**

rfay Use expose: instead of ports: with mongodb recipe

Code | Blame | 48 lines (43 loc) · 1000 Bytes

```
version: '3.6'

services:
  mongo:
    container_name: ddev-${DDEV_SITENAME}-mongo
    image: mongo:4.0
    volumes:
    - type: "volume"
      source: mongo
      target: "/data/db"
      volume:
        nocopy: true
    restart: "no"
    expose:
    - "27017"
    labels:
      com.ddev.site-name: ${DDEV_SITENAME}
      com.ddev.approot: $DDEV_APPROOT
    environment:
    - MONGO_INITDB_ROOT_USERNAME=db
    - MONGO_INITDB_ROOT_PASSWORD=db
    - MONGO_INITDB_DATABASE=db

  mongo-express:
    container_name: ddev-${DDEV_SITENAME}-mongo-express
    image: mongo-express:0.49
    restart: "no"
    labels:
      com.ddev.site-name: ${DDEV_SITENAME}
      com.ddev.approot: ${DDEV_APPROOT}
      com.ddev.platform: ddev

    links:
    - mongo:mongo
    expose:
    - "8081"
    environment:
      VIRTUAL_HOST: $DDEV_HOSTNAME
      ME_CONFIG_MONGODB_ADMINUSERNAME: db
      ME_CONFIG_MONGODB_ADMINPASSWORD: db
      HTTP_EXPOSE: "8081:8081"

  web:
    links:
    - mongo:mongo

volumes:
  mongo:
```

Example

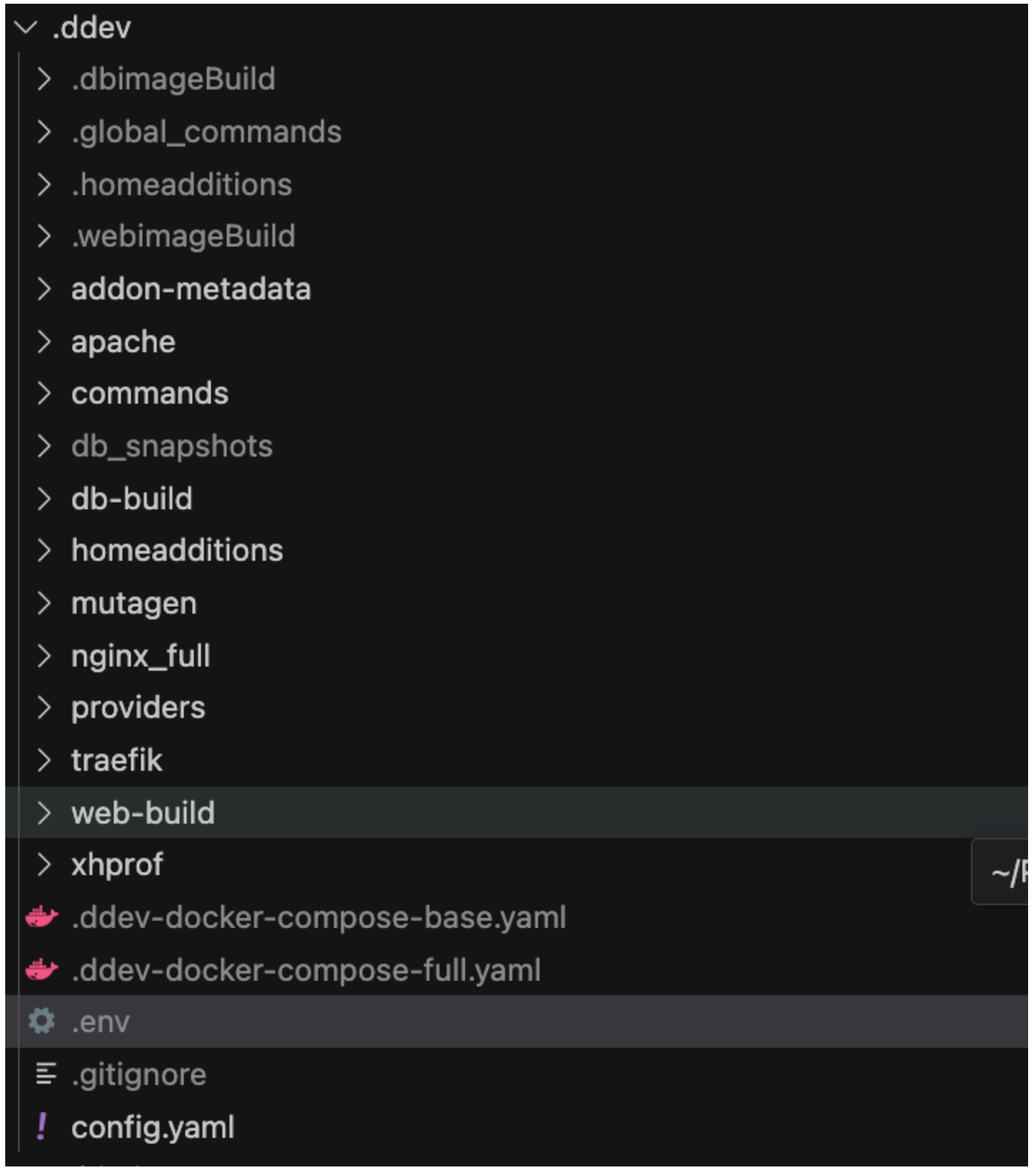
```yaml
name: midcamp
recipe: drupal10
config:
  webroot: web
env_file:
  - lando/lando.env
services:
  appserver:
    type: php:8.2
  database:
    creds:
      database: drupal10
      user: drupal10
      password: drupal10
  phpmyadmin:
    type: phpmyadmin
    host:
      - database
  mailhog:
    type: mailhog
    hogfrom:
      - appserver
  node:
    type: node:20
proxy:
  mailhog:
    - mail.midcamp.lndo.site
tooling:
  npm:
    service: node
  drush:
    service: appserver
    cmd: drush
    env:
      DRUSH_OPTIONS:URI: "https://midcamp.lndo.site"
  gulp:
    service: appserver
    description: Runs gulp
    dir: 'location where it should run'
    npm: npm run gulp
```

Items required

1. Environment file.
2. Phpmyadmin service
3. Mailhog service
4. Custom drush URI
5. Gulp custom command
6. Npm command

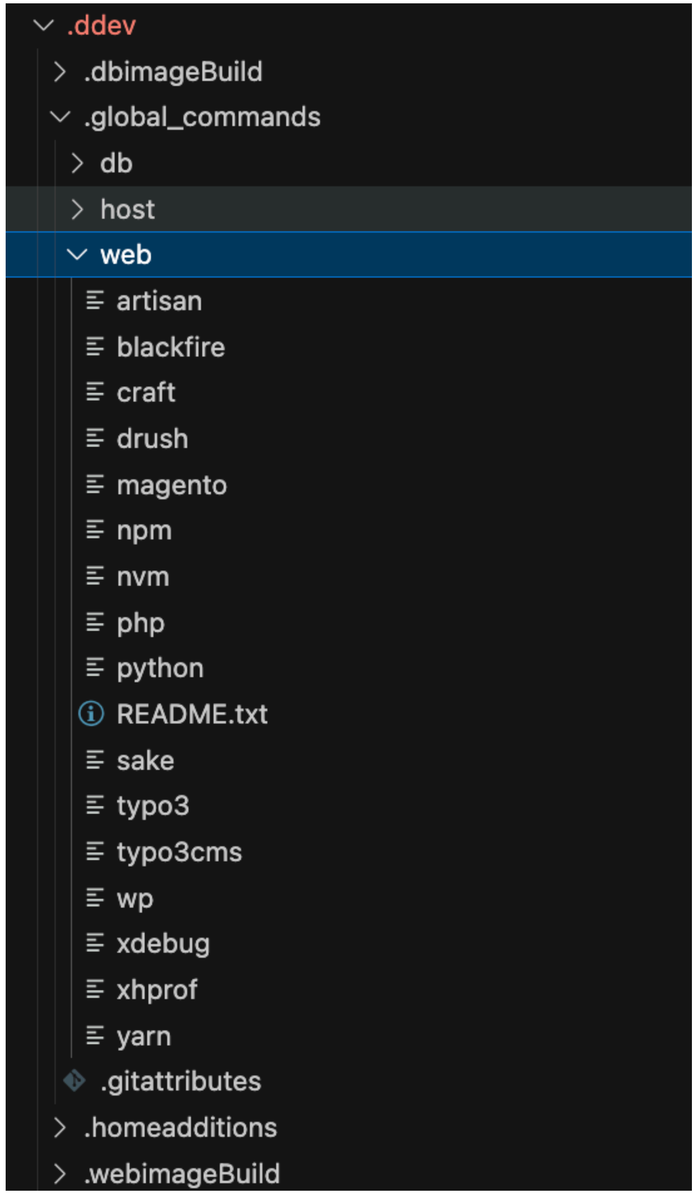
.ddev > ! config.yaml

```yaml
type: drupal10
docroot: web
php_version: "8.1"
webserver_type: nginx-fpm
xdebug_enabled: false
additional_hostnames: []
additional_fqdns: []
database:
    type: mariadb
    version: "10.4"
use_dns_when_possible: true
composer_version: "2"
web_environment: []

# Key features of DDEV's config.yaml:

```



Items provided

1. ~~Environment file~~.
2. ~~Custom drush URI~~
3. ~~Npm command~~
4. ~~Mailhog service (Mailpit provided)~~ ddev launch -m



But What about?

1. Phpmyadmin service
2. Gulp custom command

```
+ #!/bin/bash
+ ## Description: Run gulp inside the web container
+ ## Usage: gulp [flags] [args]
+ ## Example: "ddev gulp"
+ ## ExecRaw: true
+ ## HostWorkingDir: true
+
+ npm run gulp "$@"
```

Add-ons

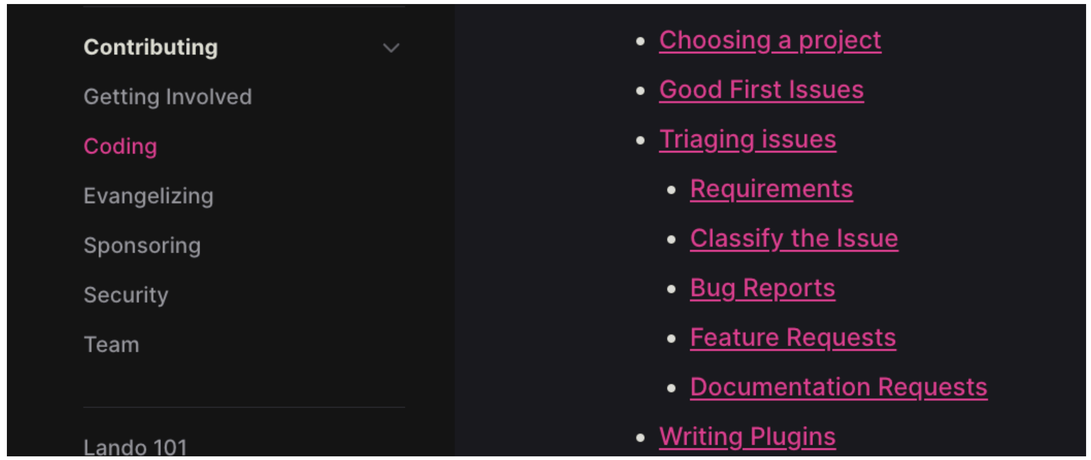
ddev get —list

ddev get ddev/ddev-phpmyadmin

https://ddev.readthedocs.io/en/latest/users/extend/additional-services/

Resources

LANDO vs DDEV code contributor workflow



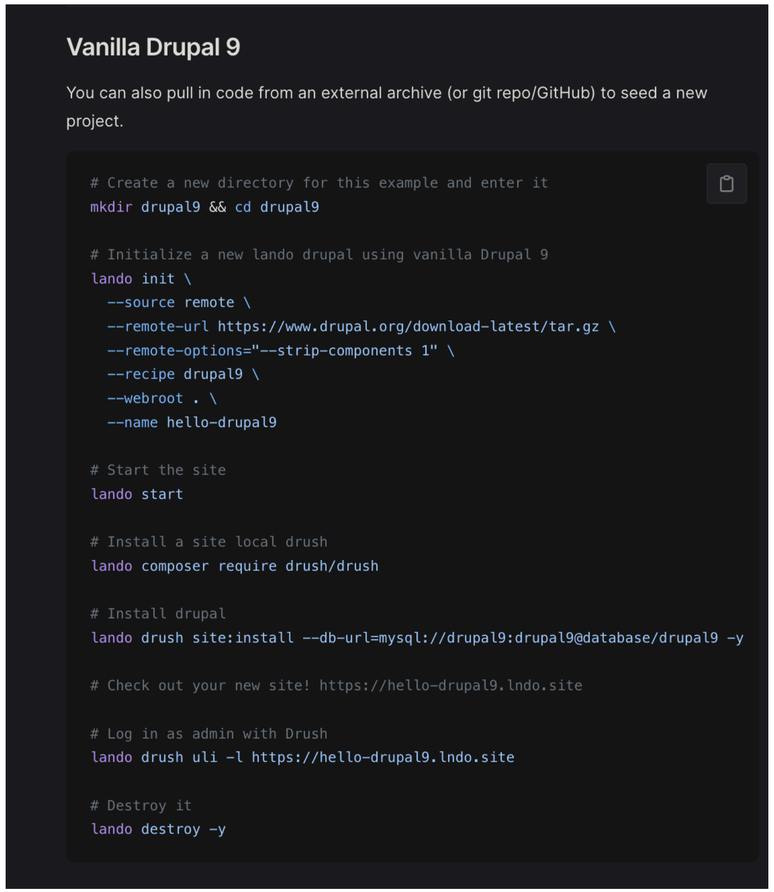
https://docs.lando.dev/contrib/

Training Date	Description and Recording
2023-07-11	Testing DDEV Features, testing DDEV v1.22.0 prereleases, how to help
2023-07-18	Maintaining and Improving the Docs
2023-07-25	Setting up a Go Development Environment
2023-08-08	Running Automated Tests Locally
2023-08-15	Maintain and improve ddev.com see outline and blog post
2023-08-22	Supporting others: How can I help? see blog
2023-08-29	Building and pushing an improved Docker image see outline
2023-09-05	Marketing: How can I help?
2023-09-12	DDEV Architecture: Go, Docker Compose, Containers, see outline
2023-09-19	DDEV Testing Infrastructure, see outline
2023-09-26	DDEV Automated Tests and How to Improve Them, Randy forgot to record! but see outline
2023-10-31	Measuring and Analyzing User Data with Amplitude see outline
2023-11-07	DDEV Add-ons: Creating, maintaining, testing see outline
2023-11-14	DDEV Hosting Providers: Creating, Maintaining, Using, see outline.

https://ddev.com/blog/contributor-training/

https://ddev.readthedocs.io/en/latest/developers/building-contributing/

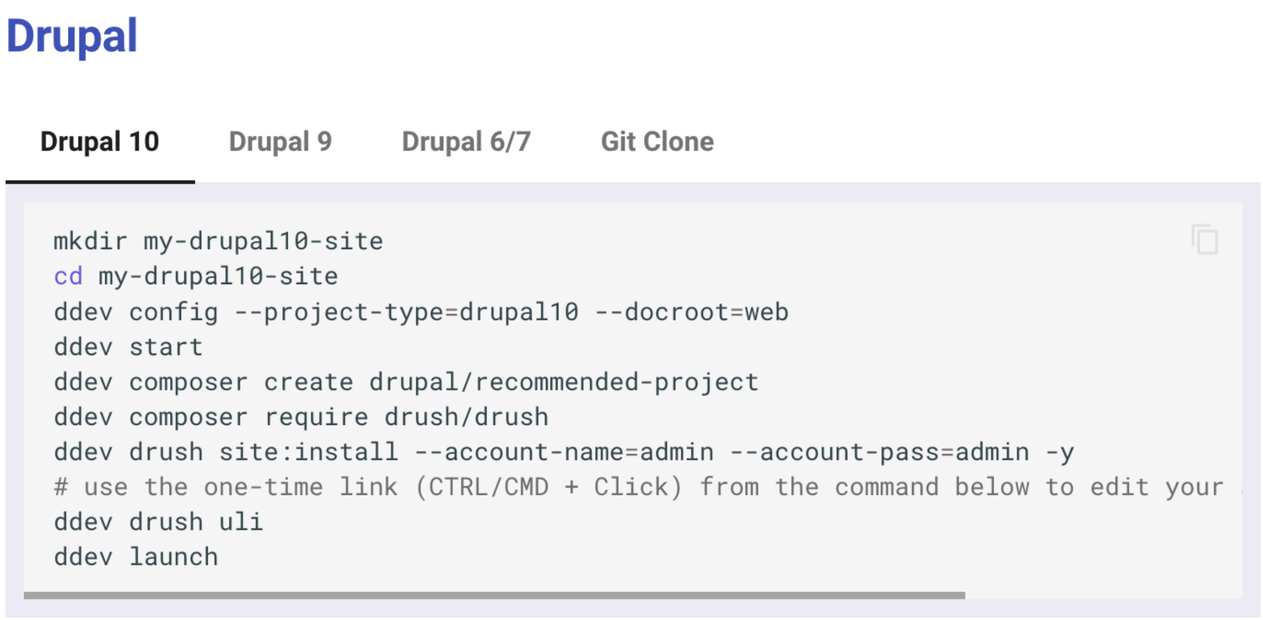
LANDO recipes VS DDEV quick starts



Vanilla Drupal 9

You can also pull in code from an external archive (or git repo/GitHub) to seed a new project.

```
# Create a new directory for this example and enter it
mkdir drupal9 && cd drupal9

# Initialize a new lando drupal using vanilla Drupal 9
lando init \
  --source remote \
  --remote-url https://www.drupal.org/download-latest/tar.gz \
  --remote-options="--strip-components 1" \
  --recipe drupal9 \
  --webroot . \
  --name hello-drupal9

# Start the site
lando start

# Install a site local drush
lando composer require drush/drush

# Install drupal
lando drush site:install --db-url=mysql://drupal9:drupal9@database/drupal9 -y

# Check out your new site! https://hello-drupal9.lndo.site

# Log in as admin with Drush
lando drush uli -l https://hello-drupal9.lndo.site

# Destroy it
lando destroy -y
```



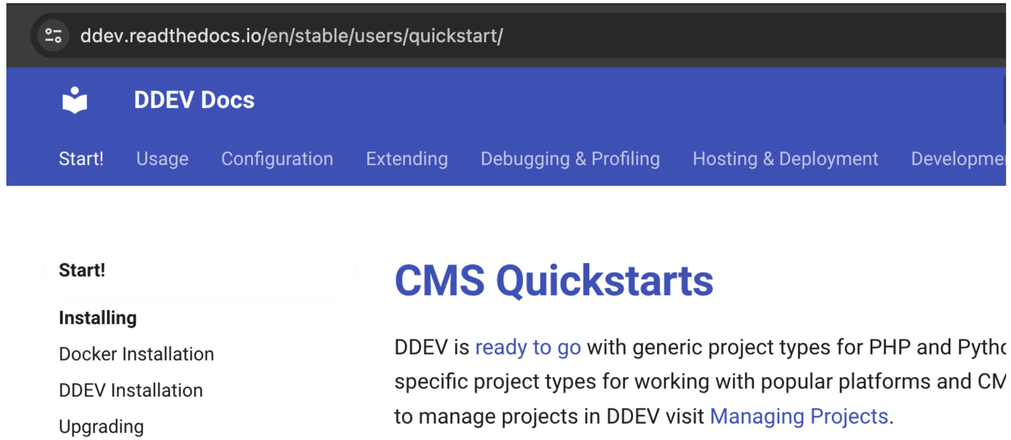
Drupal

Drupal 10 | Drupal 9 | Drupal 6/7 | Git Clone

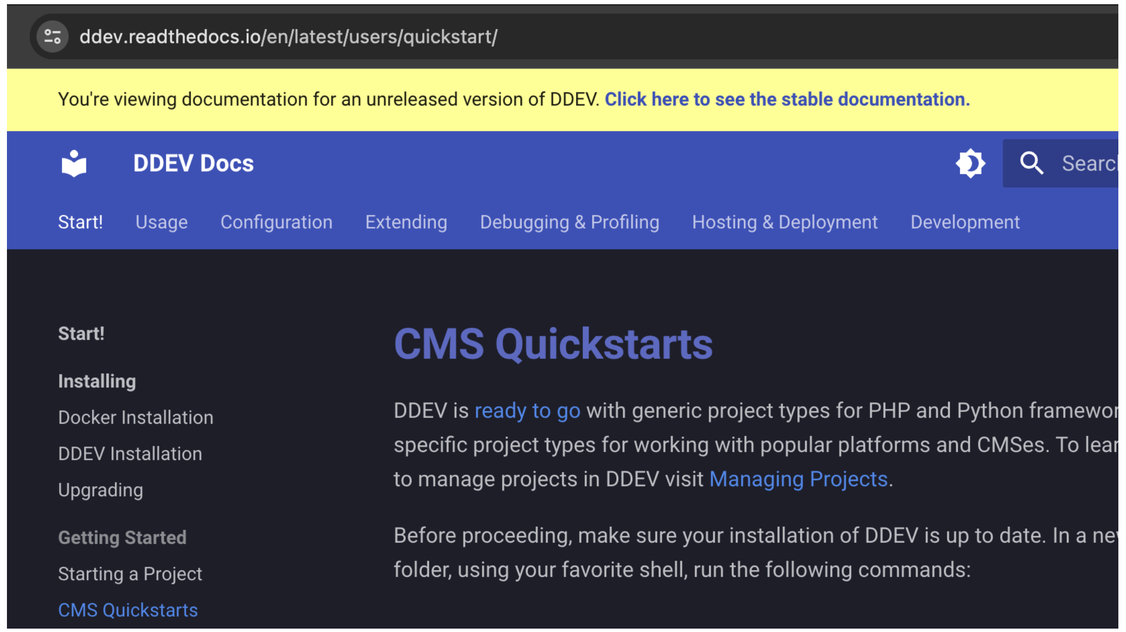
```
mkdir my-drupal10-site
cd my-drupal10-site
ddev config --project-type=drupal10 --docroot=web
ddev start
ddev composer create drupal/recommended-project
ddev composer require drush/drush
ddev drush site:install --account-name=admin --account-pass=admin -y
# use the one-time link (CTRL/CMD + Click) from the command below to edit your
ddev drush uli
ddev launch
```

https://docs.lando.dev/getting-started/first-app.html
https://ddev.readthedocs.io/en/latest/users/quickstart/#craft-cms

DDEV latest vs DDEV stable docs

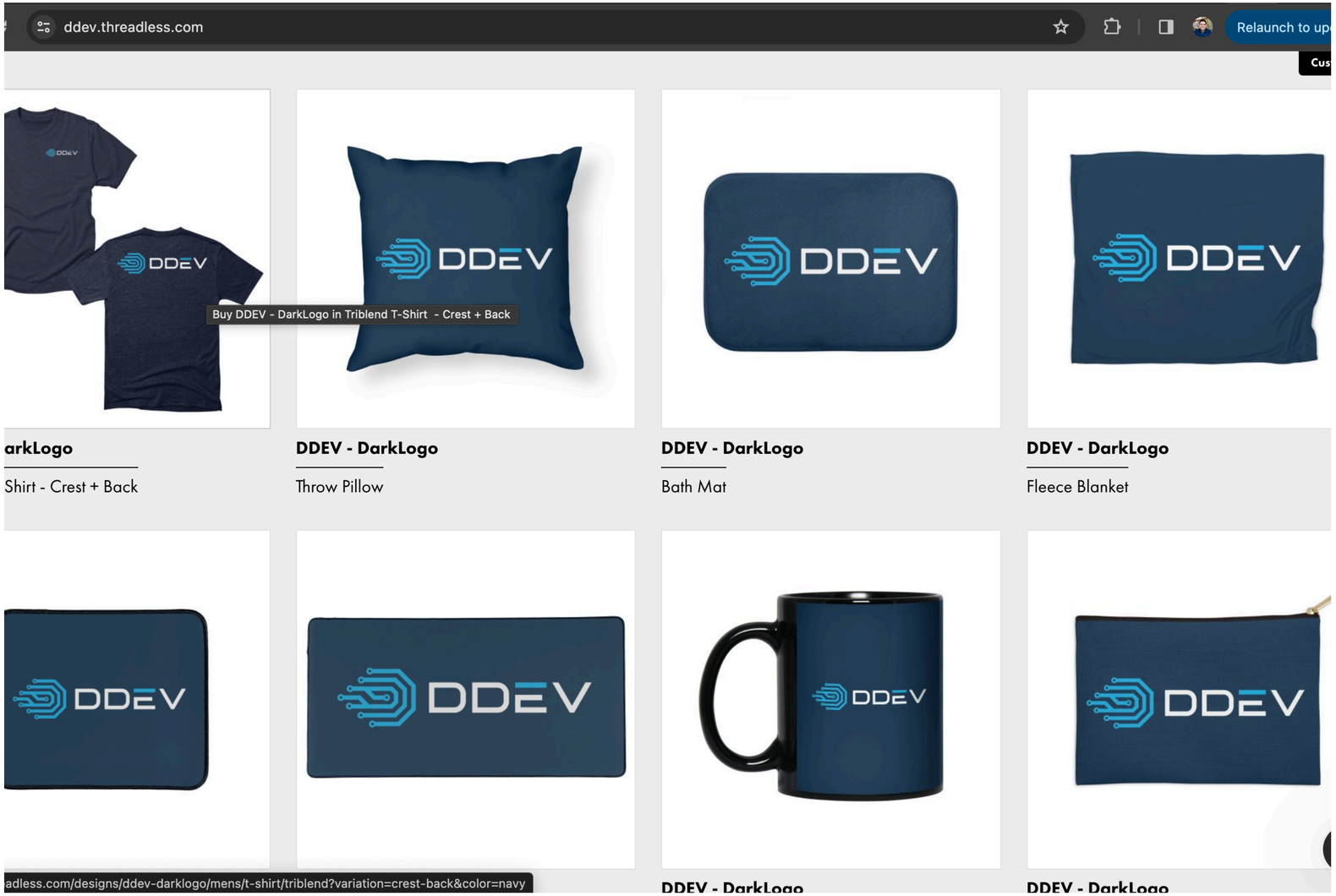


https://ddev.readthedocs.io/en/stable/users/quickstart/



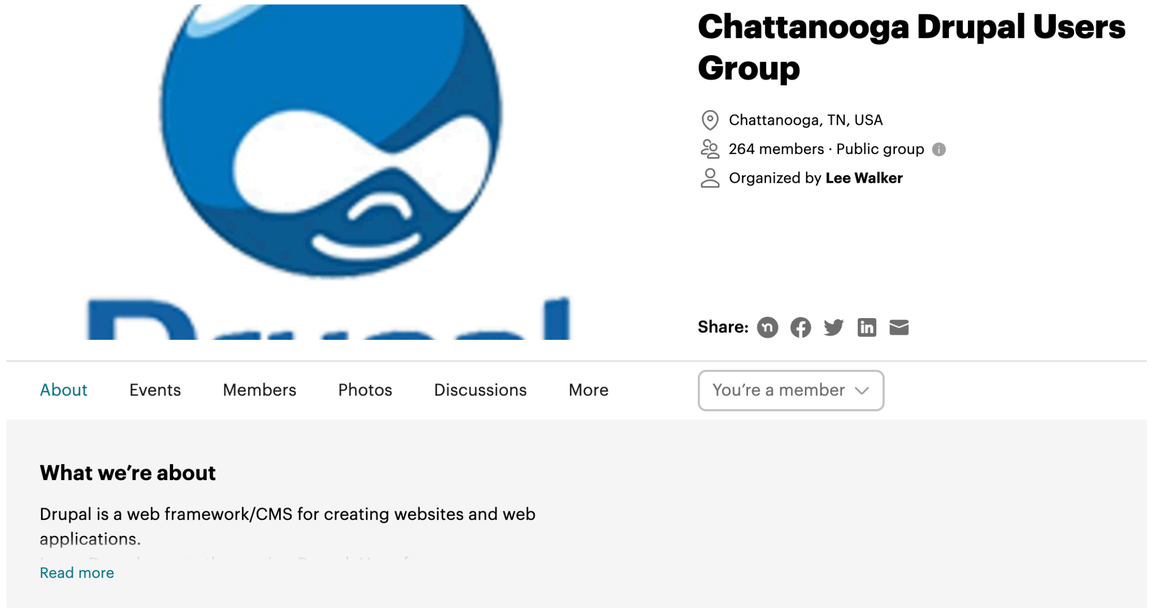
https://ddev.readthedocs.io/en/latest/users/quickstart/

Checkout the DDEV store



https://ddev.threadless.com/

Connect with us



https://www.meetup.com/Chattanooga-Drupal-Users-Group/

https://ddev.com/

https://discord.com/invite/hCZFfAMc5k

https://www.drupalcampchattanooga.com/

Feedback

https://mid.camp/8550

Questions?

THANK YOU!