Недавно я писал статью про wp-cli и профилирование работы известной cms. Теперь решил поразмыслить и постараться упростить жизнь разработчиков, избавив их от лишней настройки окружения при разработке под WordPress. Я придумал простой механизм быстрого развертывания и переноса готового сайта WordPress с помощью wp-cli, git и docker. Если вам интересна эта тема, читайте далее.

Содержание

Введение

Сразу важное замечание. Моя статья не будет best practice. Я вообще не нашел никаких готовых практик для разработки под wordpress, хотя это самый популярный движок для сайтов в мире. Каждый придумывает и внедряет свой велосипед и не особо им делится с остальными. Ничего готового я нагуглить не смог, поэтому все, что будет ниже, я придумал сам, прокручивая в голове несколько дней идею.

Статья будет в первую очередь интересна разработчикам, так как она упрощает разворачивание окружения для разработки. Это не для эксплуатации сайта в проде. Я постарался максимально упростить работу программиста, избавив его от настройки веб сервера, mysql сервера. Ему не придется делать дамбы базы, чтобы переносить сайт с хостинга на хостинг. Все будет храниться в репозитоии. Расскажу обо всем по порядку.

Второе важное замечание. Для того, чтобы повторить написанное ниже, вам нужно уметь работать с **docker** и **git**. Я не буду останавливаться на описании этих инструментов. Буду сразу приводить готовые команды. С помощью данной статьи будет наглядно продемонстрировано, в чем удобство docker для разработки. Многие системные администраторы до сих пор не понимают этого, так как docker усложнил процессы управления инфраструктурой, но многократно упростил разработку. Так что контейнеры надо изучать и по возможности внедрять там, где это уместно.

Описание CI/CD для WordPress

Опишу простыми словами, что я далее буду делать. Сразу поясню, что считать описанное выше полноценным конвейером нельзя. Строго говоря, заголовок статьи не соответствует полностью содержимому, но я не придумал ничего лучше. Моя основная задача, чтобы статью смогли найти те, кто будут искать что-то по этой теме. А термин ci/cd сейчас на слуху.

По шагам мы выполним следующую настройку для того, чтобы упростить и автоматизировать разработку под WordPress.

1. Подготовим конфигурацию docker-compose для быстрого автоматического разворачивания веб сервера с базой данных. Будем использовать стандартный набор контейнеров от разработчиков wordpress.
2. В рабочее окружение добавим контейнер с wp-cli, который будет выполнять начальную настройку cms под наши задачи - устанавливать тему, нужные плагины, тестовые записи и т.д.
3. Напишем несколько простых bash скриптов для бэкапа и восстановления БД.
4. Запушим все настройки в репозиторий git, который будем использовать как шаблон для разворачивания новых сайтов.
5. Проведем автоматическую преднастройку нового сайта с помощью подготовленной конфигурации.
6. Запушим сайт в отдельный репозиторий для удобной работы над ним.
7. Выполним полуавтоматический перенос сайта на другой сервер буквально за пару кликов.

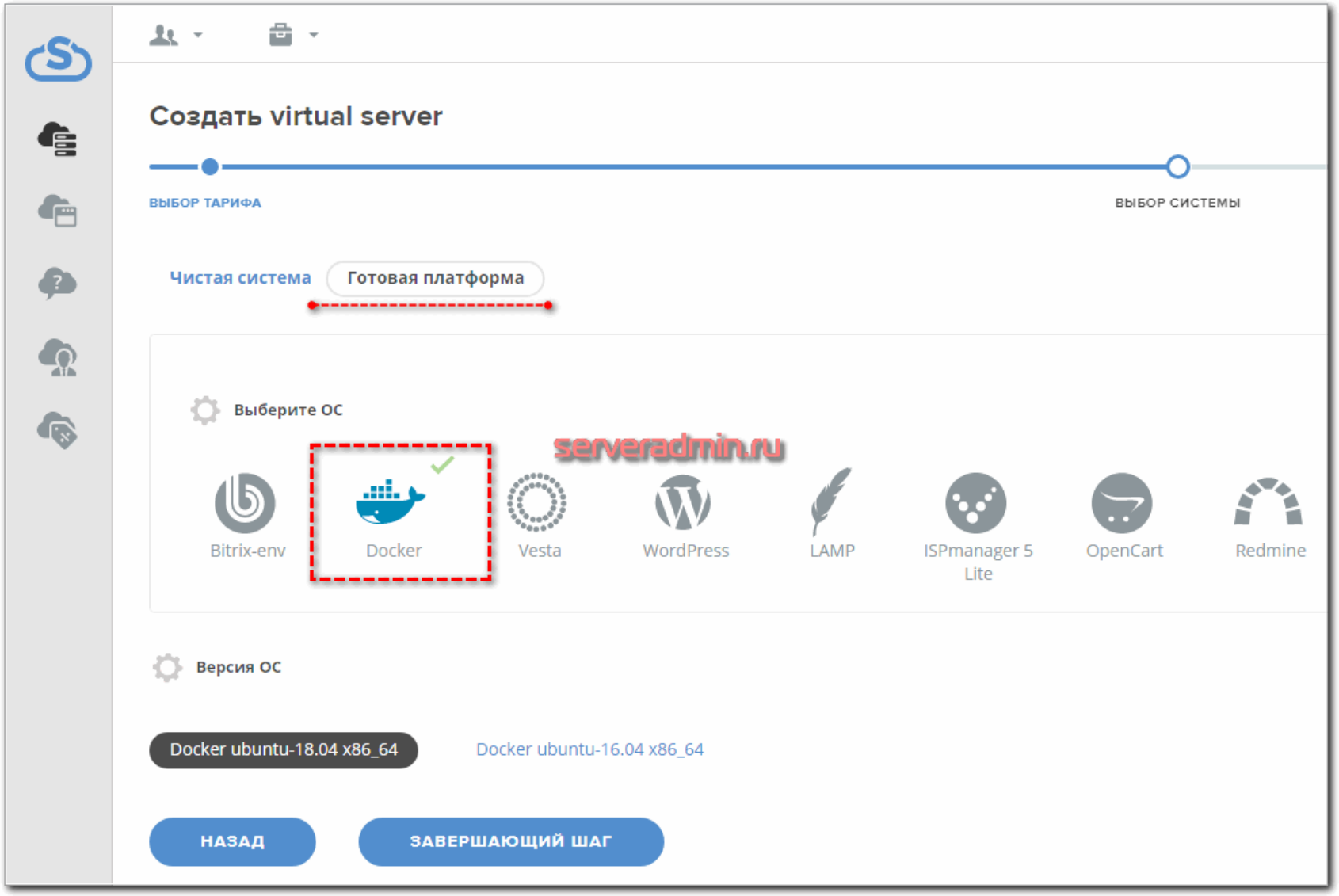
Далее в статье я буду использовать бесплатные репозитории от gitlab.com. Для быстрого старта этого достаточно. Но если у вас там будут храниться тяжелые проекты, то вас может не порадовать скорость доступа к ним. Тогда рекомендую настроить свой личный gitlab сервер. Сделать это не трудно. Я лично пользуюсь своим, как для хранения кода, так и образов docker.

Вроде все рассказал для вступления. Начинаем настраивать.

Установка WordPress с помощью docker и docker-compose

Нам понадобится виртуальная машина с установленным **docker** и **docker-compose**. Я не буду останавливаться на установке этих продуктов. Тема хорошо

освещена в поиске и решается за 5-10 минут. Лично я для тестов использую хостера simplecloud и сразу заказываю виртуальную машину с докером. Есть возможность почасовой оплаты. За это его и люблю. Все автоматизировано, виртуалка создается за 2-3 минуты.


Создать virtual server
ВЫБОР ТАРИФА
ВЫБОР СИСТЕМЫ
Чистая система
Готовая платформа
Выберите ОС
serveradmin.ru
Bitrix-env
Docker
Vesta
WordPress
LAMP
ISPmanager 5 Lite
OpenCart
Redmine
Версия ОС
Docker ubuntu-18.04 x86_64
Docker ubuntu-16.04 x86_64
НАЗАД
ЗАВЕРШАЮЩИЙ ШАГ

Удобство docker в том, что нам не нужно фокусироваться на операционной системе, где все будет работать. Поэтому я вообще не делаю на этом акцента. Используйте ту систему, к которой больше привыкли. Создаем на ней директорию wordpress.

```
# cd ~ && mkdir wordpress && cd wordpress
```

Создаем в директории конфигурационный файл *docker-compose.yaml* следующего содержания.

```
version: '3'

services:
  mysql:
    image: mysql:8
    command: --default-authentication-plugin=mysql_native_password
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: wordpress
    volumes:
      - "./db:/var/lib/mysql"

  wordpress:
    image: wordpress:php7.4-apache
    ports:
      - "80:80"
    environment:
      WORDPRESS_DB_HOST: mysql
      WORDPRESS_DB_USER: root
      WORDPRESS_DB_PASSWORD: root
      WORDPRESS_DB_NAME: wordpress
    volumes:
```

```
      - "./wp:/var/www/html/"

  wp-cli:
    image: wordpress:cli
    user: "33:33"
    environment:
      WORDPRESS_DB_HOST: mysql
      WORDPRESS_DB_USER: root
      WORDPRESS_DB_PASSWORD: root
      WORDPRESS_DB_NAME: wordpress
    volumes:
      - "./wp:/var/www/html/"
      - "./configure-wp.sh:/opt/configure-wp.sh"
#      - "./sample-content.xml:/var/www/html/sample-content.xml"
command: "bash /opt/configure-wp.sh"
```

Я использую 3 контейнера:

1. **mysql:8** - контейнер с сервером баз данных mysql 8-й версии. Вы можете использовать любой другой на ваше усмотрение. Это не принципиально, так как wordpress нормально работает со всеми известными форками mysql - mariadb, percona и т.д. Ключ *default-authentication-plugin=mysql_native_password* указан для того, чтобы работал механизм авторизации по логину с паролем. В 8-й версии по умолчанию он отключен и работа ведется через сокет. Для удобства я вынес директорию с базами данных в отдельный volume, в директорию *db*. В рамках данной задачи это не принципиально, просто я так привык.
2. **wordpress:php7.4-apache** - официальный образ wordpress с docker.hub. Выбрать можете любой из предложенного списка. Я взял наиболее простой и распространенный вариант с apache. Маплю контейнер на 80-й порт хоста. Если у вас там уже что-то запущено, выберите другой порт. В данном случае это не принципиально, так как мы настраиваем окружение для разработки. Исходники сайта мапятся в отдельный volume, в директорию *wp*.
3. **wordpress:cli** - тоже официальный образ wordpress с установленным wp-cli внутри. Он нам нужен будет для преднастройки новой установки cms. К нему мапится директория с исходниками сайта, конфиг *configure-wp.sh*, содержимое которого я покажу далее. Я указал id юзера контейнера 33, так как это дефолтный id пользователя www-data в debian, который используется в контейнере wordpress. Если этого не сделать, будут проблемы с доступом к файлам со стороны wp-cli, если хостовая машина не debian или ubuntu. Я закомментировал подключение файла sample-content.xml с образцами наполнения сайта. Далее расскажу о нем подробнее.

Остальное содержимое yaml думаю нет смысла пояснять. Там просто описывается окружение. Не забудьте поменять пароли на свои более сложные. Переходим к скрипту *configure-wp.sh*, который занимается преднастройкой свежеустановленного wordpress. Без него у вас запустится стандартный установщик при первом входе на сайт. Если вас это устраивает, то можете закомментировать в контейнере запуск этого скрипта. Вот его содержимое.

```
retries=0
while :
do
  if wp core install --url="site01.ru" --title="test blog" --admin_user="admin" --admin_password="pass" --
admin_email="admin@sample.com"
  then
    break
  else
    retries=$((retries+1))
    echo "Couldn't connect to DB. Try - ${retries}. Sleeping 5 seconds and will retry ..."
    sleep 5
  fi
  if [ "${retries}" -eq "30" ]
  then
    echo "Couldn't connect to DB 30 times. Exiting."
    exit 1
  fi
done
wp theme install donovan --activate
wp theme uninstall twentynineteen
wp theme uninstall twentyseventeen
wp theme uninstall twentytwenty
wp plugin install wordpress-importer --activate
wp plugin install classic-editor --activate
wp plugin install wp-mail-smtp --activate
wp plugin install cyr3lat --activate
wp plugin install wordpress-seo --activate
wp plugin uninstall akismet
```

```
wp plugin uninstall hello
wp language core install ru_RU --activate
# curl -O https://raw.githubusercontent.com/manovotny/wptest/master/wptest.xml
curl -O https://serveradmin.ru/files/sample-content.xml
wp import sample-content.xml --authors=create
rm sample-content.xml
```

Итак, что здесь происходит. Сначала мы пытаемся подключиться к сайту и выполнить установку wordpress. За это отвечает команда **wp core install**. Обращаю внимание на ее параметры. Обязательно указать url для сайта. Причем он может быть указан в виде ip адреса. После установки WordPress будет постоянно редиректить на этот url, так что работать придется только по нему. Я предлагаю использовать не ip адреса, а сразу настоящие. Потом просто редактировать файл hosts на своей рабочей машине, с которой осуществляется подключение. Это не очень удобно, но я не смог придумать более подходящего решения, чтобы потом можно было без лишних телодвижений перенести сайт с тестовой среды в продуктовую. Лучше всего, когда в них урлы сайта полностью совпадают.

Скрипт пытается 30 раз подключиться к БД с интервалом в 5 секунд. Если не получается, выходит. Значит с запуском сайта какие-то проблемы, так как такого интервала обычно достаточно, чтобы все запустилось. Дальше по смыслу все и так понятно. Ставится и активируется тема donovan, дефолтные удаляются. Устанавливаются некоторые плагины. Привел их для примера. Вы можете выбрать свои. Лишние плагины удаляем. Потом устанавливаем русский язык. В самом конце я заливаю тестовые данные - посты, юзеры, картинки и т.д. Я закомментировал заливку из предложенного файла wptest.xml, так как там много всего. Возможно, вам он пригодится, поэтому не стал удалять. Лично мне больше нравится набор sample данных, которые предлагает тема sahifa, поэтому заливаю их. В интернете не знаю, где их взять, залил на свой сайт. С него же и загружаю их в скрипте. Если у вас есть свой файл с подобными данными, то вы можете просто положить его в эту же директорию и подключать в файле yaml. Там у меня указан конфиг для этого, просто строка закомментирована.

По сути у нас все готово для автоматической установки и запуска WordPress с помощью docker. Переходим в папку *~/wordpress* и запускаем контейнеры в интерактивном режиме, чтобы сразу же увидеть результат работы. Потом можно будет запустить в режиме демона. Не забудьте положить файл *sample-content.xml* в директорию с yaml файлом, иначе получите ошибку в конце работы скрипта *configure-wp.sh*.

```
# docker-compose up
```

```
[root@centos8 wordpress]# docker-compose up
```

```
Pulling mysql (mysql:8)...
8: Pulling from library/mysql
d121f8d1c412: Pull complete
f3cebc0b4691: Pull complete
1862755a0b37: Pull complete
489b44f3dbb4: Pull complete
690874f836db: Pull complete
baa8be383ffb: Pull complete
55356608b4ac: Pull complete
dd35ceccb6eb: Pull complete
429b35712b19: Pull complete
162d8291095c: Pull complete
5e500ef7181b: Pull complete
af7528e958b6: Pull complete
Digest: sha256:e1bfe11693ed2052cb3b4e5fa356c65381129e87e38551c6cd6ec532ebe0e808
Status: Downloaded newer image for mysql:8
Pulling wordpress (wordpress:php7.4-apache)...
php7.4-apache: Pulling from library/wordpress
d121f8d1c412: Already exists
58b3577b786a: Pull complete
60538287851f: Pull complete
c53ff72fe225: Pull complete
79b018c8773f: Pull complete
fbe3e00ac4b0: Pull complete
ff35226e1df8: Pull complete
ab3b1d46dd82: Pull complete
b29cdd230d9a: Pull complete
d466b05cf627: Pull complete
771f930f6d23: Pull complete
b89a2786f2a3: Pull complete
c35594c34f69: Pull complete
8e3c480bd8bf: Pull complete
2d3e26ca1157: Pull complete
```

```
cc0d53b93bc3: Pull complete
dbcd12305020: Pull complete
90356c70a472: Pull complete
ceb2ac363e49: Pull complete
202d7e2f6c6c: Pull complete
Digest: sha256:6d86f59c5ad70e3f0eadff229b96f0ae16fe7d4fd2dde67fe4c2db5dcaddd650
Status: Downloaded newer image for wordpress:php7.4-apache
Pulling wp-cli (wordpress:cli)...
cli: Pulling from library/wordpress
df20fa9351a1: Pull complete
b358d6dbbdff: Pull complete
0232d962484c: Pull complete
0c1d3ac04d2a: Pull complete
16de917e7920: Pull complete
4688fcc9c773: Pull complete
9790b0ac94c0: Pull complete
4d1da0bfefaa: Pull complete
510744afa01d: Pull complete
5a10b044b8e5: Pull complete
8aa71dcbb813: Pull complete
af5c22b5f081: Pull complete
17f8433e8782: Pull complete
b26f62f60054: Pull complete
8e61c6973544: Pull complete
Digest: sha256:71428bdb5c9c72bbf9001bf68ec2dac50a97238284d5fbf20dcc85f2f09ab4a1
Status: Downloaded newer image for wordpress:cli
Creating wordpress_wordpress_1 ... done
Creating wordpress_mysql_1     ... done
Creating wordpress_wp-cli_1    ... done
Attaching to wordpress_mysql_1, wordpress_wordpress_1, wordpress_wp-cli_1
wordpress_1  | WordPress not found in /var/www/html - copying now...
mysql_1      | 2020-09-14 11:26:32+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.21-1debian10
started.
```

```
mysql_1      | 2020-09-14 11:26:32+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
mysql_1      | 2020-09-14 11:26:32+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.21-1debian10
started.
mysql_1      | 2020-09-14 11:26:32+00:00 [Note] [Entrypoint]: Initializing database files
mysql_1      | 2020-09-14T11:26:32.780545Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.21) initializing
of server in progress as process 46
mysql_1      | 2020-09-14T11:26:32.786377Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
wordpress_1  | WARNING: /var/www/html is not empty! (copying anyhow)
wordpress_1  | Complete! WordPress has been successfully copied to /var/www/html
wp-cli_1     | Error: This does not seem to be a WordPress installation.
wp-cli_1     | Pass --path=`path/to/wordpress` or run `wp core download`.
wp-cli_1     | Couldn't connect to DB. Try - 1. Sleeping 5 seconds and will retry ...
wordpress_1  | [14-Sep-2020 11:26:33 UTC] PHP Warning:  mysqli::__construct(): (HY000/2002): Connection refused in
Standard input code on line 22
wordpress_1  |
wordpress_1  | MySQL Connection Error: (2002) Connection refused
mysql_1      | 2020-09-14T11:26:33.377702Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
mysql_1      | 2020-09-14T11:26:34.776366Z 6 [Warning] [MY-010453] [Server] root@localhost is created with an empty
password ! Please consider switching off the --initialize-insecure option.
wordpress_1  |
wordpress_1  | MySQL Connection Error: (2002) Connection refused
mysql_1      | 2020-09-14 11:26:37+00:00 [Note] [Entrypoint]: Database files initialized
mysql_1      | 2020-09-14 11:26:37+00:00 [Note] [Entrypoint]: Starting temporary server
mysql_1      | 2020-09-14T11:26:37.739434Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.21) starting as
process 93
mysql_1      | 2020-09-14T11:26:37.751839Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
mysql_1      | 2020-09-14T11:26:37.952788Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
wp-cli_1     | Error: Error establishing a database connection. This either means that the username and password
information in your `wp-config.php` file is incorrect or we can’t contact the database server at `mysql`. This could mean
your host’s database server is down.
wp-cli_1     | Couldn't connect to DB. Try - 2. Sleeping 5 seconds and will retry ...
mysql_1      | 2020-09-14T11:26:38.039547Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Socket:
/var/run/mysqld/mysqlx.sock
```

```
mysql_1      | 2020-09-14T11:26:38.145276Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
mysql_1      | 2020-09-14T11:26:38.145406Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support TLS.
Encrypted connections are now supported for this channel.
mysql_1      | 2020-09-14T11:26:38.146875Z 0 [Warning] [MY-011810] [Server] Insecure configuration for --pid-file:
Location '/var/run/mysqld' in the path is accessible to all OS users. Consider choosing a different directory.
mysql_1      | 2020-09-14T11:26:38.163716Z 0 [System] [MY-010931] [Server] /usr/sbin/mysqld: ready for connections.
Version: '8.0.21'  socket: '/var/run/mysqld/mysqld.sock'  port: 0  MySQL Community Server - GPL.
mysql_1      | 2020-09-14 11:26:38+00:00 [Note] [Entrypoint]: Temporary server started.
mysql_1      | Warning: Unable to load '/usr/share/zoneinfo/iso3166.tab' as time zone. Skipping it.
mysql_1      | Warning: Unable to load '/usr/share/zoneinfo/leap-seconds.list' as time zone. Skipping it.
wordpress_1  |
wordpress_1  | MySQL Connection Error: (2002) Connection refused
mysql_1      | Warning: Unable to load '/usr/share/zoneinfo/zone.tab' as time zone. Skipping it.
mysql_1      | Warning: Unable to load '/usr/share/zoneinfo/zone1970.tab' as time zone. Skipping it.
mysql_1      | 2020-09-14 11:26:40+00:00 [Note] [Entrypoint]: Creating database wordpress
mysql_1      |
mysql_1      | 2020-09-14 11:26:40+00:00 [Note] [Entrypoint]: Stopping temporary server
mysql_1      | 2020-09-14T11:26:40.271979Z 11 [System] [MY-013172] [Server] Received SHUTDOWN from user root. Shutting
down mysqld (Version: 8.0.21).
wordpress_1  |
wordpress_1  | MySQL Connection Error: (2002) Connection refused
mysql_1      | 2020-09-14T11:26:42.791419Z 0 [System] [MY-010910] [Server] /usr/sbin/mysqld: Shutdown complete (mysqld
8.0.21)  MySQL Community Server - GPL.
wp-cli_1     | Error: Error establishing a database connection. This either means that the username and password
information in your `wp-config.php` file is incorrect or we can’t contact the database server at `mysql`. This could mean
your host’s database server is down.
wp-cli_1     | Couldn't connect to DB. Try - 3. Sleeping 5 seconds and will retry ...
mysql_1      | 2020-09-14 11:26:43+00:00 [Note] [Entrypoint]: Temporary server stopped
mysql_1      |
mysql_1      | 2020-09-14 11:26:43+00:00 [Note] [Entrypoint]: MySQL init process done. Ready for start up.
mysql_1      |
mysql_1      | 2020-09-14T11:26:43.481313Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.21) starting as
process 1
```

```
mysql_1      | 2020-09-14T11:26:43.489285Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
mysql_1      | 2020-09-14T11:26:43.687866Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
mysql_1      | 2020-09-14T11:26:43.770276Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Bind-address:
'::' port: 33060, socket: /var/run/mysqld/mysqlx.sock
mysql_1      | 2020-09-14T11:26:43.819136Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
mysql_1      | 2020-09-14T11:26:43.819278Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support TLS.
Encrypted connections are now supported for this channel.
mysql_1      | 2020-09-14T11:26:43.821171Z 0 [Warning] [MY-011810] [Server] Insecure configuration for --pid-file:
Location '/var/run/mysqld' in the path is accessible to all OS users. Consider choosing a different directory.
mysql_1      | 2020-09-14T11:26:43.838111Z 0 [System] [MY-010931] [Server] /usr/sbin/mysqld: ready for connections.
Version: '8.0.21'  socket: '/var/run/mysqld/mysqld.sock'  port: 3306  MySQL Community Server - GPL.
wordpress_1  | AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.19.0.3.
Set the 'ServerName' directive globally to suppress this message
wordpress_1  | AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.19.0.3.
Set the 'ServerName' directive globally to suppress this message
wordpress_1  | [Mon Sep 14 11:26:45.423530 2020] [mpm_prefork:notice] [pid 1] AH00163: Apache/2.4.38 (Debian) PHP/7.4.10
configured -- resuming normal operations
wordpress_1  | [Mon Sep 14 11:26:45.423590 2020] [core:notice] [pid 1] AH00094: Command line: 'apache2 -D FOREGROUND'
mysql_1      | mbind: Operation not permitted
wp-cli_1     | sendmail: can't connect to remote host (127.0.0.1): Connection refused
wp-cli_1     | Success: WordPress installed successfully.
mysql_1      | mbind: Operation not permitted
wp-cli_1     | Installing Donovan (1.6.1)
wp-cli_1     | Warning: Failed to create directory '/etc/X11/fs/.wp-cli/cache/': mkdir(): Permission denied.
wp-cli_1     | Downloading installation package from https://downloads.wordpress.org/theme/donovan.1.6.1.zip...
wp-cli_1     | Unpacking the package...
wp-cli_1     | Installing the theme...
wp-cli_1     | Theme installed successfully.
wp-cli_1     | Activating 'donovan'...
wp-cli_1     | Success: Switched to 'Donovan' theme.
wp-cli_1     | Success: Installed 1 of 1 themes.
mysql_1      | mbind: Operation not permitted
wp-cli_1     | Deleted 'twentynineteen' theme.
```

```
wp-cli_1     | Success: Deleted 1 of 1 themes.
mysql_1      | mbind: Operation not permitted
wp-cli_1     | Deleted 'twentyseventeen' theme.
wp-cli_1     | Success: Deleted 1 of 1 themes.
mysql_1      | mbind: Operation not permitted
wp-cli_1     | Deleted 'twentytwenty' theme.
wp-cli_1     | Success: Deleted 1 of 1 themes.
wp-cli_1     | Installing WordPress Importer (0.7)
wp-cli_1     | Warning: Failed to create directory '/etc/X11/fs/.wp-cli/cache/': mkdir(): Permission denied.
mysql_1      | mbind: Operation not permitted
wp-cli_1     | Downloading installation package from https://downloads.wordpress.org/plugin/wordpress-importer.0.7.zip...
wp-cli_1     | Unpacking the package...
wp-cli_1     | Installing the plugin...
wp-cli_1     | Plugin installed successfully.
wp-cli_1     | Activating 'wordpress-importer'...
wp-cli_1     | Plugin 'wordpress-importer' activated.
wp-cli_1     | Success: Installed 1 of 1 plugins.
wp-cli_1     | Installing Classic Editor (1.6)
wp-cli_1     | Warning: Failed to create directory '/etc/X11/fs/.wp-cli/cache/': mkdir(): Permission denied.
mysql_1      | mbind: Operation not permitted
wp-cli_1     | Downloading installation package from https://downloads.wordpress.org/plugin/classic-editor.1.6.zip...
wp-cli_1     | Unpacking the package...
wp-cli_1     | Installing the plugin...
wp-cli_1     | Plugin installed successfully.
wp-cli_1     | Activating 'classic-editor'...
wp-cli_1     | Plugin 'classic-editor' activated.
wp-cli_1     | Success: Installed 1 of 1 plugins.
wp-cli_1     | Installing WP Mail SMTP by WPForms (2.3.1)
wp-cli_1     | Warning: Failed to create directory '/etc/X11/fs/.wp-cli/cache/': mkdir(): Permission denied.
mysql_1      | mbind: Operation not permitted
wp-cli_1     | Downloading installation package from https://downloads.wordpress.org/plugin/wp-mail-smtp.2.3.1.zip...
wp-cli_1     | Unpacking the package...
wp-cli_1     | Installing the plugin...
```

```
wp-cli_1     | Plugin installed successfully.
wp-cli_1     | Activating 'wp-mail-smtp'...
wp-cli_1     | Plugin 'wp-mail-smtp' activated.
wp-cli_1     | Success: Installed 1 of 1 plugins.
mysql_1      | mbind: Operation not permitted
wp-cli_1     | Installing Cyr to Lat enhanced (3.5)
wp-cli_1     | Warning: Failed to create directory '/etc/X11/fs/.wp-cli/cache/': mkdir(): Permission denied.
wp-cli_1     | Downloading installation package from https://downloads.wordpress.org/plugin/cyr3lat.3.5.zip...
wp-cli_1     | Unpacking the package...
wp-cli_1     | Installing the plugin...
wp-cli_1     | Plugin installed successfully.
wp-cli_1     | Activating 'cyr3lat'...
wp-cli_1     | Plugin 'cyr3lat' activated.
wp-cli_1     | Success: Installed 1 of 1 plugins.
wp-cli_1     | Installing Yoast SEO (14.9)
wp-cli_1     | Warning: Failed to create directory '/etc/X11/fs/.wp-cli/cache/': mkdir(): Permission denied.
mysql_1      | mbind: Operation not permitted
wp-cli_1     | Downloading installation package from https://downloads.wordpress.org/plugin/wordpress-seo.14.9.zip...
wp-cli_1     | Unpacking the package...
wp-cli_1     | Installing the plugin...
wp-cli_1     | Plugin installed successfully.
wp-cli_1     | Activating 'wordpress-seo'...
wp-cli_1     | Plugin 'wordpress-seo' activated.
wp-cli_1     | Success: Installed 1 of 1 plugins.
wp-cli_1     | Uninstalled and deleted 'akismet' plugin.
wp-cli_1     | Success: Uninstalled 1 of 1 plugins.
wp-cli_1     | Uninstalled and deleted 'hello' plugin.
wp-cli_1     | Success: Uninstalled 1 of 1 plugins.
wp-cli_1     | Downloading translation from https://downloads.wordpress.org/translation/core/5.5.1/ru_RU.zip...
wp-cli_1     | [14-Sep-2020 11:27:16 UTC] PHP Warning:  Declaration of
WP_CLI\LanguagePackUpgrader::download_package($package, $check_signatures = false) should be compatible with
WP_Upgrader::download_package($package, $check_signatures = false, $hook_extra = Array) in
phar:///usr/local/bin/wp/vendor/wp-cli/language-command/src/WP_CLI/LanguagePackUpgrader.php on line 51
```

```
wp-cli_1     | Warning: Declaration of WP_CLI\LanguagePackUpgrader::download_package($package, $check_signatures = false)
should be compatible with WP_Upgrader::download_package($package, $check_signatures = false, $hook_extra = Array) in
phar:///usr/local/bin/wp/vendor/wp-cli/language-command/src/WP_CLI/LanguagePackUpgrader.php on line 51
wp-cli_1     | Warning: Failed to create directory '/etc/X11/fs/.wp-cli/cache/': mkdir(): Permission denied.
wp-cli_1     | Unpacking the update...
wp-cli_1     | Installing the latest version...
wp-cli_1     | Removing the old version of the translation...
wp-cli_1     | Translation updated successfully.
wp-cli_1     | Language 'ru_RU' installed.
wp-cli_1     | Success: Language activated.
wp-cli_1     | Success: Installed 1 of 1 languages.
wp-cli_1     | Starting the import process...
mysql_1      | mbind: Operation not permitted
wp-cli_1     | Failed to import product_tag Accessories
Failed to import product_tag album
Failed to import product_tag albums
Failed to import product_tag cap
Failed to import product_tag color
Failed to import product_type external
Failed to import product_type grouped
Failed to import product_tag Jackets
Failed to import product_tag men
Failed to import product_tag mug
Failed to import product_tag music
Failed to import product_tag pants
Failed to import product_tag Pullover
Failed to import product_tag shorts
Failed to import product_type simple
Failed to import product_tag T-shirt
Failed to import product_type variable
Failed to import product_cat Accessories
Failed to import product_cat Jackets
Failed to import product_cat Music
```

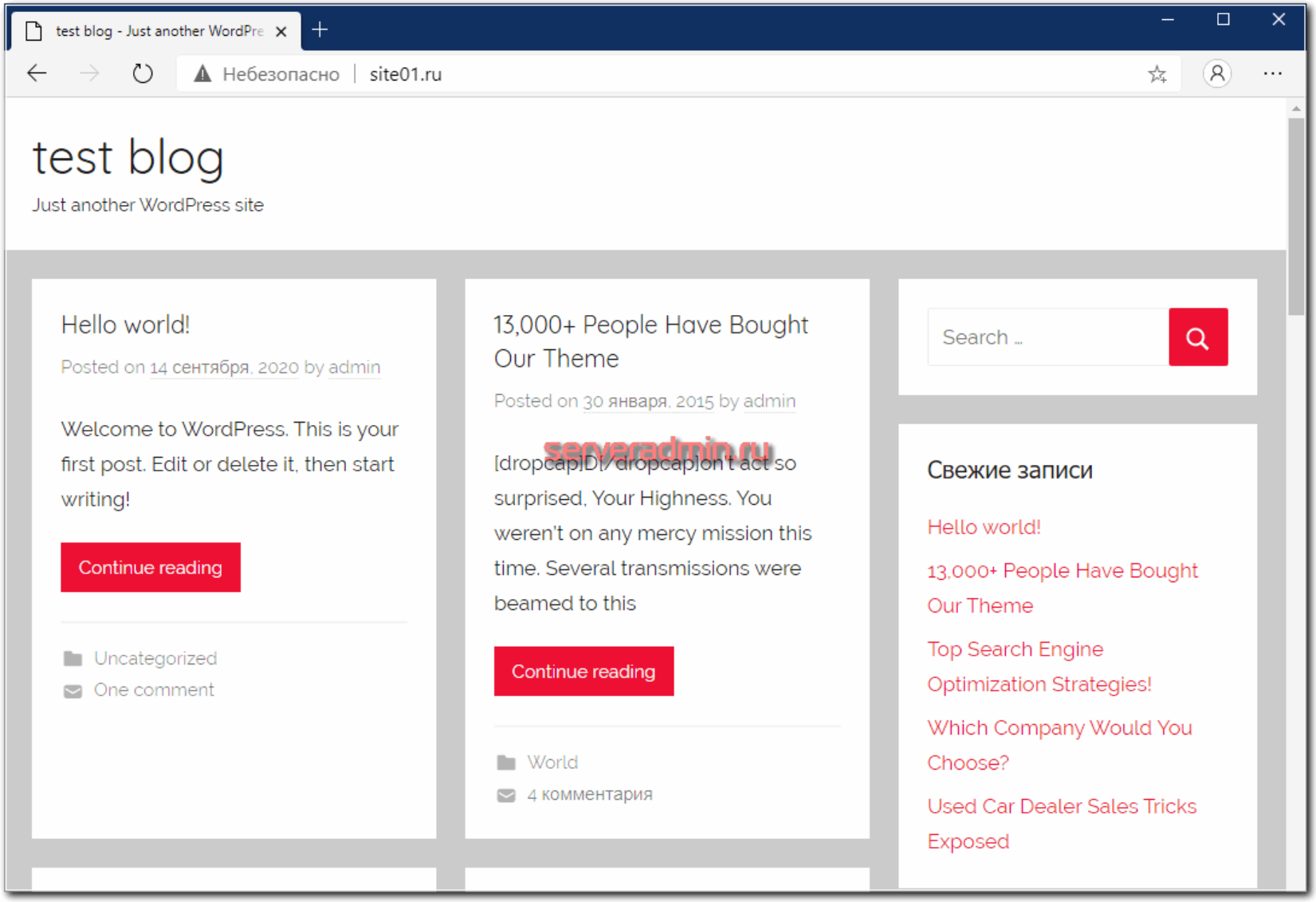
```
Failed to import product_cat pants
Failed to import product_cat Pullover
Failed to import product_cat shorts
Failed to import product_cat Sweatshirts
Failed to import product_cat T-Shirts
wp-cli_1     |
wp-cli_1     | Processing post #68 ("t-shirt-orange") (post_type: attachment)
wp-cli_1     | -- 1 of 277 (in file sample-content.xml)
wp-cli_1     | -- Mon, 14 Sep 2020 11:27:18 +0000
.....................................................................
wp-cli_1     | Failed to import “Order – February 6, 2015 @ 02:42 PM”: Invalid post type shop_order

All done. Have fun!
Remember to update the passwords and roles of imported users.
wp-cli_1 | Success: Finished importing from 'sample-content.xml' file.
```

Если у вас примерно такой вывод, значит все в порядке. Теперь отредактируйте у себя на рабочем месте файл hosts, добавив туда информацию:

```
10.20.1.23 site01.ru
```

10.20.1.23 - адрес сервера, на котором мы только что запустили преднастроенный сайт на wordpress. Можете перейти по адресу site01.ru и посмотреть, что получилось.



Сайт полностью установлен и готов к работе. С помощью скрипта *configure-wp.sh* вы можете как угодно кастомизировать начальную установку wordpress. Если работаете на потоке, вам это сэкономит много времени. К примеру, если у вас много своих плагинов или тем, то просто положите их в отдельную директорию, копируйте и активируйте с помощью скрипта для wp-cli.

Бэкап базы данных через wp-cli

Прежде чем залить созданные выше конфиги в репозиторий, предлагаю добавить туда пару скриптов, с помощью которых можно будет бэкапить и заливать базу данных wordpress. Так как будет использоваться wp-cli, не придется вводить отдельно имя базы и учетку для доступа туда. Wp-cli все возьмет из настроек wordpress и выгрузит базу стандартным mysqldump.

Скрипт для экспорта базы *db-export.sh*.

```
#!/bin/bash

docker-compose run --rm wp-cli wp db export --add-drop-table
mv ./wp/wordpress-*.sql .
gzip wordpress-*.sql
```

Тут все просто. Запускаем контейнер с wp-cli и экспортируем базу. После того, как контейнер отработает, удаляем его. Базу данных забираем из директории wp и кладем в корень папки, где лежит сам скрипт. Дальше объясню, зачем мы это делаем.

Проверьте работу скрипта.

```
# chmod +x db-export.sh
# ./db-export.sh
Success: Exported to 'wordpress-2020-09-15-062339e.sql'.
```

Теперь напишем скрипт для импорта базы *db-import.sh*.

```
#!/bin/bash

gunzip wordpress-*.sql.gz
mv wordpress-*.sql ./wp

dbname=`ls -l ./wp | grep sql | awk '{print $9}'`

docker-compose run --rm wp-cli wp db import $dbname
rm ./wp/$dbname
```

Здесь выполняем все то же самое, только в обратном направлении. Распаковываем базу, кладем в директорию wp и импортируем через контейнер wp-cli. Проверяйте его работу.

```
# chmod +x db-import.sh
# ./db-import.sh
Success: Imported from 'wordpress-2020-09-15-062339e.sql'.
```

После импорта база данных удаляется. Напоминаю, что мы это делаем не для бэкапов, а для переноса сайта. Дальше я подробнее раскрою эту тему. На текущем моменте нам надо убедиться в том, что скрипты нормально работают - экспортируют и импортируют базу.

Создание репозитория git для шаблона сайта

Прежде чем мы загрузим все наше хозяйство в git, добавим еще пару файлов. Нам надо добавить сюда еще один скрипт - *chmod.sh*.

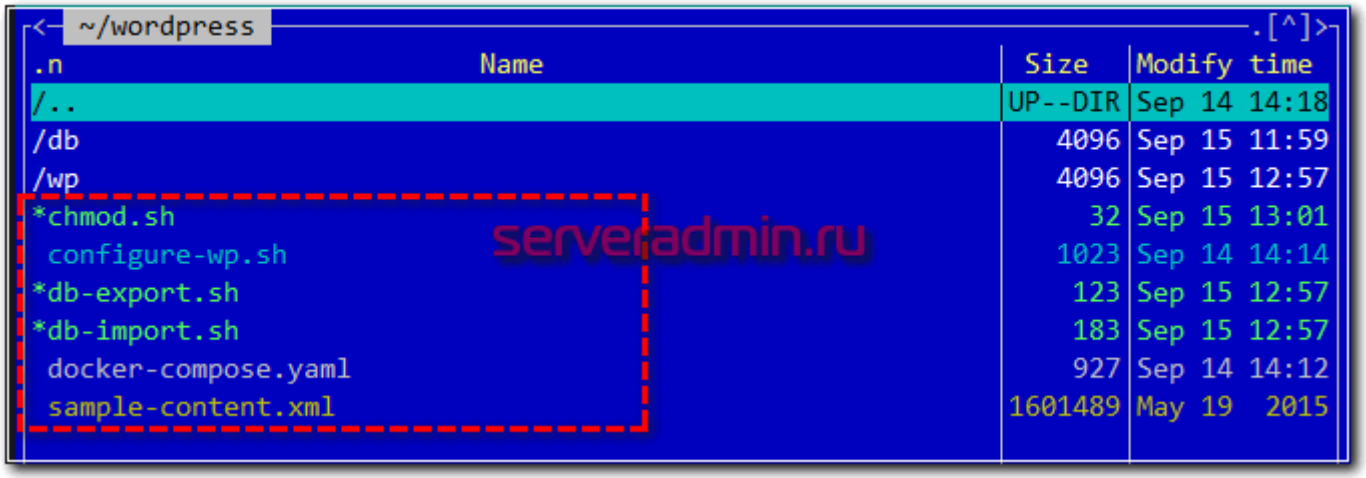
```
#!/bin/bash

chown -R 33:33 wp/
```

Он нам понадобится для выставления прав на исходники сайта wordpress после переноса. И сюда же, в директорию добавляем еще один файл - *.gitignore*.

```
/wp/*
/db/*
```

В настоящий момент нам не надо ничего от самого сайта загружать в репозиторий, так как мы собираем шаблон. У нас все готово. В директории wordpress должны лежать следующие файлы, содержимое которых я привел выше.

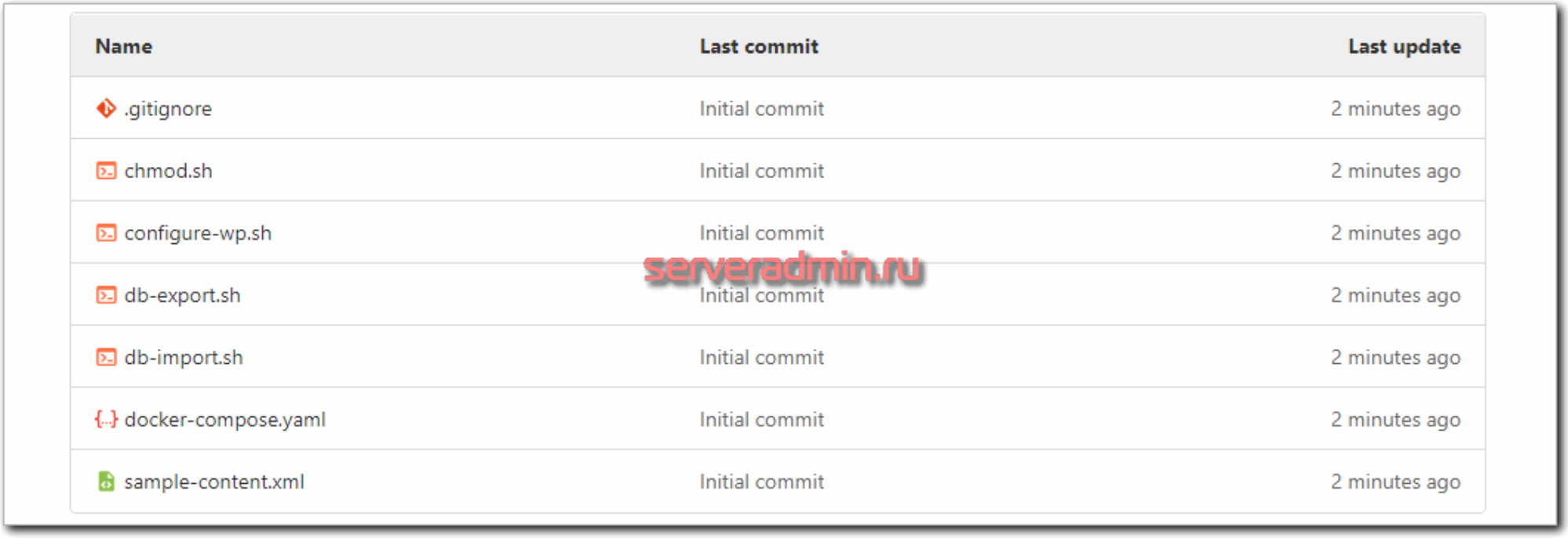


Теперь создаем репозиторий wp в gitlab и заливаем все туда. Сначала сделайте это в веб интерфейсе, а потом переходите в консоль сервера. Не забудьте добавить ssh ключ в gitlab, чтобы подключиться к репозиторию.

```
# git config --global user.name "Vladimir Zp"
# git config --global user.email "zeroxzed@gmail.com"
```

```
# git init
Initialized empty Git repository in /root/wordpress/.git/
# git remote add wp git@gitlab.com:zeroxzed/wp.git
# git add .
# git commit -m "Initial commit"
[master (root-commit) 1dfefb2] Initial commit
7 files changed, 33718 insertions(+)
create mode 100644 .gitignore
create mode 100755 chmod.sh
create mode 100644 configure-wp.sh
create mode 100755 db-export.sh
create mode 100755 db-import.sh
create mode 100644 docker-compose.yaml
create mode 100644 sample-content.xml
# git push -u wp master
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 4 threads.
Compressing objects: 100% (7/7), done.
Writing objects: 100% (9/9), 69.87 KiB | 4.11 MiB/s, done.
Total 9 (delta 0), reused 0 (delta 0)
To gitlab.com:zeroxzed/wp.git
* [new branch] master -> master
Branch 'master' set up to track remote branch 'master' from 'wp'.
```

```
[root@centos8 wordpress]# git config --global user.name "Vladimir Zp"
[root@centos8 wordpress]# git config --global user.email "zeroxzed@gmail.com"
[root@centos8 wordpress]# git init
Initialized empty Git repository in /root/wordpress/.git/
[root@centos8 wordpress]# git remote add wp git@gitlab.com:zeroxzed/wp.git
[root@centos8 wordpress]# git add .
[root@centos8 wordpress]# git commit -m "Initial commit"
[master (root-commit) 1dfefb2] Initial commit
 7 files changed, 33718 insertions(+)
 create mode 100644 .gitignore
 create mode 100755 chmod.sh
 create mode 100644 configure-wp.sh
 create mode 100755 db-export.sh
 create mode 100755 db-import.sh
 create mode 100644 docker-compose.yaml
 create mode 100644 sample-content.xml
[root@centos8 wordpress]# git push -u wp master
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 4 threads.
Compressing objects: 100% (7/7), done.
Writing objects: 100% (9/9), 69.87 KiB | 4.11 MiB/s, done.
Total 9 (delta 0), reused 0 (delta 0)
To gitlab.com:zeroxzed/wp.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'wp'.
[root@centos8 wordpress]#
```



Все, мы создали репозиторий, который будем брать за основу при разработке сайтов на wordpress. Показываю дальше, как это будет выглядеть на практике.

CI/CD wordpress в тестовой среде

Допустим, нам надо начать разрабатывать сайт на wordprtess с адресом site02.ru. Для этого мы идем на тестовый сервер и клонируем к себе репозиторий wp, который создали ранее.

```
# git clone git@gitlab.com:zeroxzed/wp.git
```

Переименовываем директорию *wp* в *site02.ru* и удаляем там папку *.git*. Для каждого сайта будем создавать отдельный именной репозиторий и с ним работать. Переходим в папку *site02.ru* и редактируем **configure-wp.sh** под нужды этого проекта. Обязательно укажите правильный url сайта. После этого запускаем контейнеры.

```
# docker-compose up
```

Дожидаемся запуска контейнеров и проверяем работу сайта. Не забудьте отредактировать файл hosts.

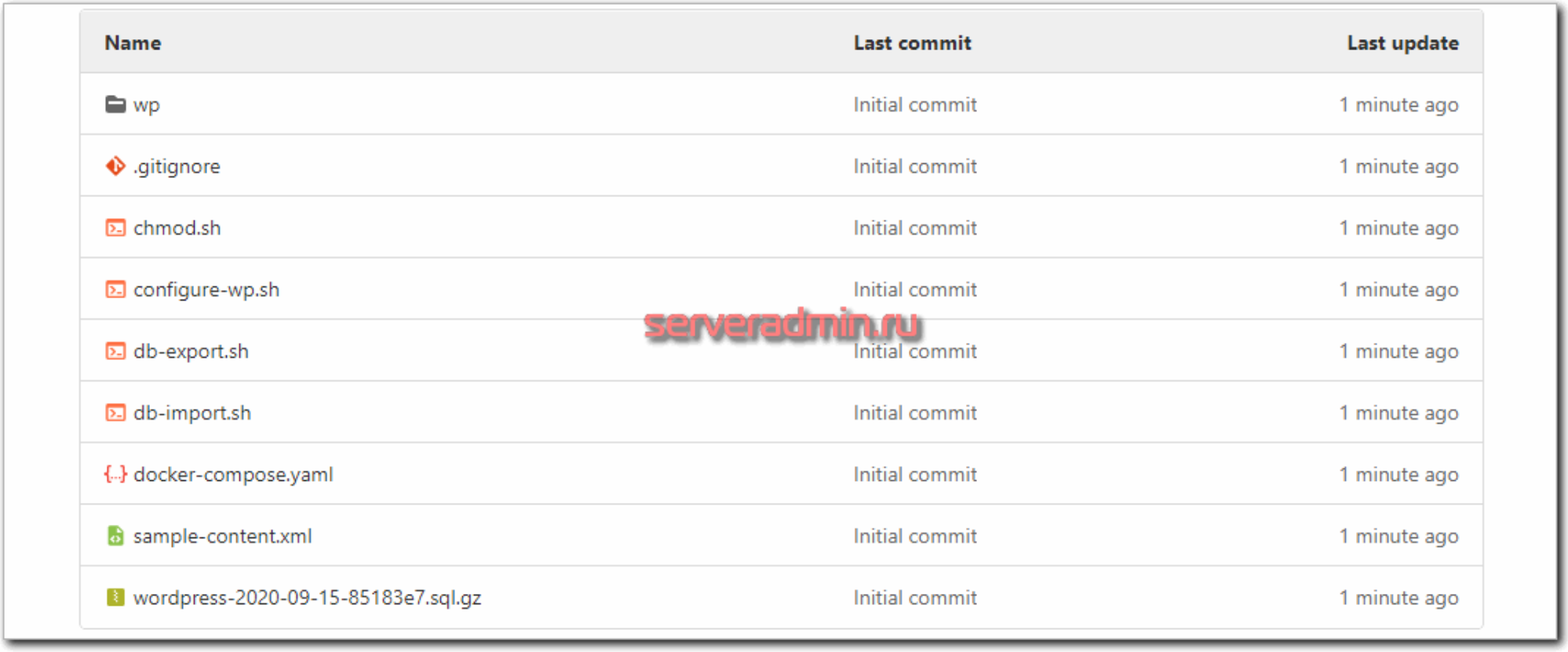
```
10.20.1.23 site02.ru
```

Если все в порядке, то выгрузим базу данных, запустив скрипт *db-export.sh*. Убедитесь, что файл с базой появился в директории со скриптом. Теперь отредактируем файл *.gitignore*, удалив оттуда строку с /wp/*, так как теперь нам нужно будет загрузить исходники сайта в репозиторий.

Далее идем в git, добавляем новый проект с названием site02.ru и загружаем туда репозиторий из директории site02.ru.

```
# git init
# git remote add site02.ru git@gitlab.com:zeroxzed/site02.ru.git
# git add .
# git commit -m "Initial commit"
# git push -u site02.ru master
```

В репозиторий улетают исходники сайта и база данных.



Name	Last commit	Last update
wp	Initial commit	1 minute ago
.gitignore	Initial commit	1 minute ago
chmod.sh	Initial commit	1 minute ago
configure-wp.sh	Initial commit	1 minute ago
db-export.sh	Initial commit	1 minute ago
db-import.sh	Initial commit	1 minute ago
docker-compose.yaml	Initial commit	1 minute ago
sample-content.xml	Initial commit	1 minute ago
wordpress-2020-09-15-85183e7.sql.gz	Initial commit	1 minute ago

Дальше вы один или с кем-то в группе работаете над сайтом, пушите изменения в репу, проверяете на тестовом сервере. Для того, чтобы показать сайт заказчику, вам достаточно будет пойти на simplecloud, создать виртуалку с docker. Склонировать себе репозиторий, не забыв перед этим запушить туда весь код и экспорт базы данных. После этого запускаете контейнеры, импортируете базу данных через скрипт *db-impost.sh* и назначаете права доступа через скрипт *chmod.sh*. В завершении меняете в файле hosts ip адрес на новый внешний ip и заходите на сайт.

Заключение

> Не понравилась статья и хочешь научить меня администрировать? Пожалуйста, я люблю учиться. Комментарии в твоем распоряжении. Расскажи, как сделать правильно!

Надеюсь, понятно описал придуманную схему. В ней очень много допущений и костылей, так как wordpress это типичный монолитный проект, который под методы современной непрерывной разработки и доставки не очень подходит. Поэтому в статье я делаю акцент, что все это для упрощения начальной разработки сайта перед запуском его в продакшн.

Когда сайт уедет в prod, нужен будет другой подход, так как:

1. Будет расти директория wp-content, где содержатся картинки и прочие файлы. Пушить все это в git не очень идея. Хотя, если размеры директории не слишком большие, то можно так делать, особенно если у вас свой git и вы нормально управляете его нагрузкой и свободным местом. Как самый простой вариант - перенести это все в s3.
2. Самая большая проблема - база данных. Хранить ее и переносить подобным образом точно неприемлемо для работающего проекта. Нужно будет что-то делать с этим, так как если вы банально обновите сайт или плагины на тестовом сервере, то вы не сможете просто так накатить изменения кода на прод, нужно будет и базу наливать, так как обновления вносят в нее изменения. Как красиво обходить этот момент лично я не прорабатывал. Буду рад, если кто-то поделится инфой на эту тему.

В общем случае я не вижу смысла prod для wordpress держать в контейнерах. Как я уже говорил ранее, это типичный монолит, к тому же весь на php, к тому же с полноценной БД. У него при разработке меняется только код, который может обновляться из репозитория. Нет смысла его в docker засовывать, если есть возможность не засовывать. Само веб окружение лучше настроить без контейнеров, так будет ниже отклик сайта. А вот для настройки рабочего окружения разработчика контейнеры в данном случае самое то.

длится 4 месяца, после чего успешные выпускники курса смогут пройти собеседования у партнеров. После обучения вы сможете:

- разрабатывать на SQL;
- проектировать БД и понимать все нюансы;
- анализировать и оптимизировать производительности запросов;
- писать сложные хранимые процедуры, функции и триггеры;
- читать план запроса.

Проверьте себя на вступительном тесте и смотрите подробнее программу по .

Помогла статья? Подписывайся на telegram канал автора

Анонсы всех статей, плюс много другой полезной и интересной информации, которая не попадает на сайт.