# DigitalOcean droplet install guide

This install guide describes how I have setup a Nginx powered Ubuntu 14.04 droplet on DigitalOcean for future reference.

The information recorded is a collection of information found around the web, supplemented with my own lines and insights.

Please ping me or send PR for incorrect or obsolete information.

## Index


1. Add users
2. Setup SSH Keys
3. Nginx, PHP5-FPM, GD/ImageMagick, Sendmail
4. Configure Nginx
5. Configure Git
    - *Flavour 1*: Bare repository
    - *Flavour 2*: Repository with submodules
6. Setup Dropbox sync


## Add users

1. Change root password after login:
    - `ssh root@<ip address droplet>`
    - `passwd`
2. Update system:
    1. `apt-get update`
    1a. `apt-get dist-upgrade` (watch out: upgrades the Ubuntu distribution version!)
    2. `apt-get upgrade`
    3. `apt-get autoremove && apt-get autoclean`
3. Add users:
    - `adduser <username>`
4. Add root privileges for added users:
    - `visudo`
    - now add for every user: `<username> ALL=(ALL:ALL) ALL`
5. Add a `.bash_profile` file to the user's home directory: `/root/` or `/home/<username>`:
    - `cd ~`
    - `touch .bash_profile` (if not already present)
6. Edit `.bash_profile` file:
    - `nano .bash_profile` (or edit via Transmit's SFTP))

7. And add the following 4 lines (including the empty second line) to the `.bash_profile` file:
    `# ~/.bash_profile`

`export LC_ALL=en_US.UTF-8`
`export LANG=en_US.UTF-8`

8. Reload `.bash_profile`:
   - `source .bash_profile` (or open a new Terminal window)
9. Stop forwarding locale from the client:
   - Open the local `ssh_config` file in Sublime Text: `sublime /etc/ssh_config`
   - Now out-comment the following line: `SendEnv LANG LC_*`

10. Upload from the `/git` folder the `gitconfig.sample` config file to: `/home/<username>` and after uploading rename to `.gitconfig` extension

11. Repeat steps 5 to 9 for every user added in step 3 (SSH login one user at a time)!

12. Now login (again) as root user and generate and reconfigure the (missing) locales:
    - `locale-gen en_US en_US.UTF-8`
    - `dpkg-reconfigure locales`

# Setup SSH keys

1. If there is no `~/.ssh` directory in your user's home directory, create one:
   - `mkdir ~/.ssh` (login with the user first!)
2. If there is no `id_rsa` and `id_rsa.pub` key combo, then generate one:
   - `ssh-keygen -t rsa`
3. Now from your local machine (assuming your public key can be found at `~/.ssh/id_rsa.pub`) enter the following command:
   - `ssh user@hostname/ipadress 'cat >> ~/.ssh/authorized_keys' < ~/.ssh/id_rsa.pub`

4. If you now close the connection, and then attempt to establish SSH access, you should no longer be prompted for a password

# Install Nginx, PHP5-FPM, GD/ImageMagick and Sendmail

To check if `packagename` was installed, type:

- `dpkg -s <packagename>` or

- `dpkg -l <packagename>`

- Install Nginx:
  - `apt-get install nginx`
  - test run and then (re)start: `nginx -tt && service nginx restart`
- Install PHP:
  - `apt-get install php5-fpm`
  - then (re)start: `service php5-fpm restart`
  - check `dpkg -l php5-fpm`
- Install GD:
  - `apt-get install php5-gd`
  - check if succes: `dpkg --get-selections | grep php`
  - or `dpkg -l php5-gd`

- Install ImageMagick:
  - `apt-get install php5-imagick`
  - check: `dpkg --get-selections | grep php`
  - or `dpkg -l php5-imagick`
- Install Sendmail:
  - `apt-get install sendmail`
  - check `dpkg -l sendmail`
  - now configure sendmail: `sendmailconfig` ()basically say 'Y' to all questions)
  - edit the `/etc/hosts` file:
    - replace the `127.0.0.1 …` / `127.0.1.1 …` lines with:

      127.0.0.1 localhost.localdomain localhost
      127.0.1.1 hostname.example.com hostname/do-droplet-name
      ip-address/do-droplet-ip hostname.example.com hostname/do-droplet-name
    - to get hostname/do-droplet-name, type: `hostname` , and…
    - to get ip-address/do-droplet-ip, type: `hostname -I`
    - now restart hostname: `service hostname restart`
    - when the domain is `test.example.com` , the hostname/do-droplet-name `carrot` , and the ip-address/do-droplet-ip `177.55.162.226` , the configuration looks like this:

      127.0.0.1 localhost.localdomain localhost
      127.0.1.1 test.example.com carrot
      177.55.162.226 test.example.com carrot
    - **Important note**:
      When multiple domains (e.g. `example.com` and `my.example.com` ) on the same DigitalOcean droplet, then please let me now *how to configure to send from both domains*?
    - **More infromation**:
      Setting the Hostname & Fully Qualified Domain Name (FQDN) on Ubuntu 12.04 or CentOS 6.4
  - to apply the new host name without rebooting the system type:
    - `hostname example.com` , then then check you have the FQDN:
    - `hostname -f`
- Do a bit of cleanup:
  - `apt-get autoremove && apt-get autoclean`

# Configure Nginx

1. Login (SFTP) as `root` user with Transmit f.i.
2. Upload from the `/nginx` folder the config files to: `/etc/nginx` , frist the `hbp5` and (when using Kirby CMS) the `kirby` folders

3. Backup the default `nginx.conf` and `mime.types` files by renaming them like this: `nginx.conf~` and `mime.types~`

4. Upload also from the `/nginx` folder the `nginx.conf` and `mime.types` files

5. Now delete the `default` file in `sites-available` folder and upload (again from the `/nginx` folder) the following files:

   `no-default`
   `example.com`
   `ssl.example.com`

6. Rename and edit the files accordingly

7. Delete the `default` symlink in `sites-enabled`

8. Login with `root` user and active the desired virtual hosts:
   - `ln -s /etc/nginx/sites-available/no-default /etc/nginx/sites-enabled/no-default`
   - `ln -s /etc/nginx/sites-available/example.com /etc/nginx/sites-enabled/example.com`
9. Test and restart Nginx:
   - `nginx -tt && service nginx restart`

# Configure Git

## *Flavour 1*: Bare repository (Use git submodules? See 'flavour 2' below!)

1. Install Git:
   - `apt-get install git-core`
   - `apt-get autoremove && apt-get autoclean`
2. Create the following folders with `root` user ( `-p` makes sure all the directories that don't exists already are created, not just the last one):
   - `mkdir -p /usr/share/nginx/www/example.com/public`
   - `mkdir -p /usr/share/nginx/www/stage.example.com/public`
   - `mkdir -p /usr/share/nginx/repo/example.git`
   - `mkdir -p /usr/share/nginx/repo/stage.example.git`
3. Now change the group and ownership of the `/public` and `/example.git` folders:
   - `sudo chown -R example:example /usr/share/nginx/www/example.com/public`
   - `sudo chown -R example:example /usr/share/nginx/repo/example.git`
   - etc.

4. Move the `/usr/share/nginx/html/50x.html` file to the newly created `/www` directory: `/usr/share/nginx/www`, and then delete the `/html` directory

5. Login (SSH) with `example` user and initialize the bare Git repositories:
   - `cd /usr/share/nginx/repo/example.git`
   - `git init --bare`
   - Repeat for staging domain
6. Upload to the `/usr/share/nginx/repo/example.git/hooks` folder of the bare git repository the `post-receive.bare.sample` file, located in the `/git/hooks` folder (make sure to enter the correct virtual host, etc.) and after uploading rename to `post-receive`.

- **Important note:** make sure to login (either SSH or SFTP) with the correct `example` user when uploading the file, otherwise the file group/owner will be incorrect!

7. Set permissions of the `post-receive` file to `775`

8. Repeat for staging (and other possible) (sub)domain(s)

9. Now add to the `/usr/share/nginx/repo/example.git/info` folder of the bare git repositories the `sparse-checkout.sample` file (set permissions to `664`), located in the `/git/info` folder (make sure to enter the correct path-to-files) and after uploading rename to `sparse-checkout`.

10. Now add the remote stage and production repositories to your local repository:
    - `git remote add stage ssh://example@hostname-or-ip/usr/share/nginx/repo/stage.exammple.git`
    - `git remote add production ssh://user@hostname-or-ip/usr/share/nginx/repo/example.git`

## *Flavour 2*: Repository with submodules (Don't use git submodules? See 'flavour 1' above!)

Based on this!

1. Install Git:
    - `apt-get install git-core`
    - `apt-get autoremove && apt-get autoclean`
2. Create the following folders with `root` user (`-p` makes sure all the directories that don't exists already are created, not just the last one):
    - `mkdir -p /usr/share/nginx/www/example.com/public`
    - `mkdir -p /usr/share/nginx/www/stage.example.com/public`
3. Now change the group and ownership of the `/public` folders:
    - `sudo chown -R example:example /usr/share/nginx/www/example.com/public`
    - etc.

4. Move the `/usr/share/nginx/html/50x.html` file to the newly created `/www` directory: `/usr/share/nginx/www`, and then delete the `/html` directory

5. Login (SSH) with `example` user and initialize the Git repositories:
    - `cd /usr/share/nginx/www/example.com/public`
    - `git init`
    - Repeat for staging domain
6. Upload to the `/usr/share/nginx/www/example.com/.git/hooks` folder the `post-receive.submodules.sample` file, located in the `/git/hooks` folder (make sure to enter the correct virtual host, etc.) and after uploading rename to `post-receive`.
    - **Important note:** make sure to login (either SSH or SFTP) with the correct `example` user when uploading the file, otherwise the file group/owner will be incorrect!

7. Set permissions of the `post-receive` file to `775`

8. Repeat for staging (and other (sub)domains)

9. Now add to the `/usr/share/nginx/www/example.com/.git/info` folder the `sparse-checkout.sample` file (set permissions to `664`), located in the `/git/info` folder (make sure to enter the correct path-to-files) and after uploading rename to `sparse-checkout`.

10. Now add the remote stage and production repositories to your local repository:
    - `git remote add stage ssh://example@hostname-or-ip/usr/share/nginx/www/stage.exammple.com/public`
    - `git remote add production ssh://user@hostname-or-ip/usr/share/nginx/www/example.com/public`

# Setup Dropbox sync

1. Login (SHH) with your special `dropbox` user
2. Make sure to be your home directory: `cd`
3. Download Dropbox client:
    - Stable 32-bit: `wget -O dropbox.tar.gz "http://www.dropbox.com/download/?plat=lnx.x86"`
    - Stable 64-bit: `wget -O dropbox.tar.gz "http://www.dropbox.com/download/?plat=lnx.x86_64"`
4. Sanity check to make sure we're not going to clog our home directory:
    - `tar -tzf dropbox.tar.gz`
5. Extract:
    - `tar -xvzf dropbox.tar.gz`
6. Run dropboxd:
    - `~/.dropbox-dist/dropboxd`
7. You should see output like this:
    - `This client is not linked to any account... Please visit https://www.dropbox.com/cli_link?host_id=7d44a557aa58f285f2da0x67334d02c1 to link this machine.`
8. Go to the URL given, and after succes dropboxd will create a ~/Dropbox folder and start synchronizing it after this step
9. It is recommended to download the official Dropbox CLI to start the dropbox daemon (as an unprivileged user) and get its status:
    - `mkdir -p ~/bin`
    - `wget -O ~/bin/dropbox.py "http://www.dropbox.com/download?dl=packages/dropbox.py"`
    - `chmod 755 ~/bin/dropbox.py`
    - `~/bin/dropbox.py help`
10. Change the too restrictive default Dropbox folder permissions:
    - `chmod 755 ~/Dropbox`
11. Create symlink to the `dropbox` file in the `~/.dropbox-dist/` folder (necessary for folowing steps)
    - `ln -s ~/.dropbox-dist/dropbox-lnx.x86-2.10.28/dropbox ~/.dropbox-dist/dropbox`
12. To run dropbox on system startup, login (SSH) as `root` user and then create: `touch /etc/init.d/dropbox`
13. Edit the newly created file and add the contents of the `/dropbox/dropbox.sample` file: `nano /et/init.d/dropbox`
14. Make the script is executable and add it to the system startup:
    - `chmod +x /etc/init.d/dropbox`
    - `update-rc.d dropbox defaults`
15. Now control the Dropbox client like any other Ubuntu service:

- `service dropbox start|stop|reload|force-reload|restart|status`

16. Still logged in as `root` user, then add symbolic links to the `www` content folders like this:
    - example.com `ln -s /home/dropbox/Dropbox/example.com/content/ /usr/share/nginx/www/example.com/public/content`
    - stage.example.com `ln -s /home/dropbox/Dropbox/example.com/content/ /usr/share/nginx/www/stage.example.com/public/content`

---

**Note 1**: If something went wrong during the install you can try again by deleting all the Dropbox files in the home directory: `rm -rf .dropbox* Dropbox`. And start again by downloading the files you need.

**Note 2**: If you want to change the Dropbox account it is linked to, unlink it from the first account, then kill the running dropbox process, start it up again (with `~/.dropbox-dist/dropboxd &`) and obtain the new host_id with `dbreadconfig.py`. If you don't restart the dropbox client, it will give the same host_id (which for some reason causes it to be unable to change the account it is linked to).