Tile map server

Введение

На этой странице показано, как можно использовать OpenStreetMap Carto для реализации тайлового сервера с использованием того же программного обеспечения, что и OpenStreetMap. Он включает в себя пошаговые инструкции по установке Tile Server на основе Ubuntu и ограничен описанием некоторых передовых методов, учитывая, что основной целью этого сайта является предоставление руководств по настройке среды разработки OpenStreetMap Carto и предложение рекомендации по редактированию стиля.
OSM Tile Server — это веб-сервер, специализирующийся на доставке растровых карт, представляющих их как статические тайлы и способный выполнять рендеринг в реальном времени или предоставлять кэшированные изображения. Веб-программное обеспечение, принятое OpenStreetMap, представляет собой HTTP-сервер Apache вместе со специальным подключаемым модулем с именем mod_tile и связанным с ним внутренним стеком, способным генерировать плитки во время выполнения; программы и библиотеки объединены в цепочку для создания сервера тайлов.
Как это часто бывает с OpenStreetMap, существует множество способов достижения цели, и почти все компоненты имеют альтернативы, которые имеют различные преимущества и недостатки. В этом руководстве описывается стандартный процесс установки OSM Tile Server, используемого на OpenStreetMap.org.
Он состоит из следующих основных компонентов:

- Mapnik
- Apache
- Mod_tile
- renderd
- osm2pgsql
- PostgreSQL/PostGIS database, to be installed locally (suggested) or remotely (might be slow, depending on the network).
- carto
- openstreetmap-carto

Все упомянутое программное обеспечение является открытым исходным кодом.
Для тайлового сервера требуется база данных PostGIS , в которой хранятся геопространственные объекты, заполненные инструментом osm2pgsql из данных OSM. Кроме того, необходим каталог файловой системы, включающий файл OSM.xml , символы карты (проверьте подкаталог openstreetmap-carto/symbols) и шейп-файлы (проверьте подкаталог openstreetmap-carto/data). OSM.xml предварительно создается инструментом carto из стиля openstreetmap-carto (project.mml и все связанные файлы CartoCSS, включенные в openstreetmap-carto).
Когда веб-сервер Apache получает запрос от браузера, он вызывает подключаемый модуль mod_tile , который, в свою очередь, проверяет, была ли плитка уже создана (из предыдущего рендеринга) и кэширована, чтобы она была готова к использованию; в этом случае mod_tile немедленно отправляет плитку обратно на веб-сервер. И наоборот, если запрос необходимо отобразить, то он ставится в очередь на серверную часть рендеринга , которая отвечает за вызов Mapnik для выполнения фактического рендеринга; renderd — это процесс- демон ,

включенный в исходники mod_tile и связанный с mod_tile через очереди UNIX. визуализированныйявляется стандартным бэкэндом, используемым в настоящее время www.openstreetmap.org, даже если некоторые реализации OSM используют Tirex ; Mapnik извлекает данные из базы данных PostGIS в соответствии с информацией о стиле openstreetmap-carto и динамически отображает тайл. renderd возвращает созданный тайл на веб-сервер и, в свою очередь, в браузер.
Демон renderd реализует механизм очередей с несколькими уровнями приоритета, чтобы обеспечить актуальность просмотра с учетом доступных ресурсов рендеринга. Наивысший приоритет — для рендеринга «на лету» тайлов, еще не находящихся в кэше тайлов, два уровня приоритета для повторного рендеринга устаревших тайлов «на лету» и две очереди фонового пакетного рендеринга. Чтобы избежать проблем с каталогами, которые становятся слишком большими, и чтобы избежать слишком большого количества крошечных файлов, Mod_tile / renderd хранит визуализированные тайлы в «метатайлах», в специальной хешированной структуре каталогов.
Даже если тайловый сервер динамически генерирует тайлы во время выполнения, они также могут быть предварительно визуализированы для просмотра в автономном режиме с помощью специального инструмента с именем render_list , который обычно используется для предварительного рендеринга тайлов с низким уровнем масштабирования и требует значительного времени для выполнения процесса (десятки часов в случае, если вся планета предварительно визуализируется); эта утилита включена в mod_tile , а также в другой инструмент с именем render_expired , который предоставляет методы для разрешения истечения срока действия тайлов карты. Более подробное описание render_list и render_expired можно найти на их справочных страницах.
Справочную информацию о методе истечения срока действия тайлов можно найти в механизме истечения срока действия тайлов.

Общая настройка для Ubuntu

Обновите Ubuntu

Убедитесь, что ваша система Ubuntu полностью обновлена:

```
lsb_release -a
```

команда возвращает версию Ubuntu.

```
vladpolskiy@linux:~$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 22.04 LTS
Release:        22.04
Codename:       jammy
```

Чтобы обновить систему:

```
sudo apt-get update
```

команда загружает списки пакетов из репозиториев и «обновляет» их, чтобы получить информацию о новейших версиях пакетов и их зависимостях. Это будет сделано для всех

репозиториев и PPA. Из http://linux.die.net/man/8/apt-get

```
vladpolskiy@linux:~$ sudo apt-get update
[sudo] password for vladpolskiy:
Hit:1 http://ru.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://ru.archive.ubuntu.com/ubuntu jammy-updates InRelease [114 kB]
Get:3 http://ru.archive.ubuntu.com/ubuntu jammy-backports InRelease [99.8 kB]
Get:4 http://ru.archive.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:5 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [479
 kB]
Get:6 http://ru.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages
[203 kB]
Fetched 1,007 kB in 1s (730 kB/s)
Reading package lists... Done
vladpolskiy@linux:~$
```

```
sudo apt list --upgradable
```

команда выведит списки пакетов готовых к обновлению

```
vladpolskiy@linux:~$ apt list --upgradable
Listing... Done
apparmor/jammy-updates 3.0.4-2ubuntu2.1 amd64 [upgradable from: 3.0.4-2ubuntu2]
apt-utils/jammy-updates 2.4.6 amd64 [upgradable from: 2.4.5]
apt/jammy-updates 2.4.6 amd64 [upgradable from: 2.4.5]
base-files/jammy-updates 12ubuntu4.2 amd64 [upgradable from: 12ubuntu4]
firmware-sof-signed/jammy-updates 2.0-1ubuntu3 all [upgradable from: 2.0-1ubuntu
2]
isc-dhcp-client/jammy-updates 4.4.1-2.3ubuntu2.1 amd64 [upgradable from: 4.4.1-2
.3ubuntu2]
isc-dhcp-common/jammy-updates 4.4.1-2.3ubuntu2.1 amd64 [upgradable from: 4.4.1-2
.3ubuntu2]
libapparmor1/jammy-updates 3.0.4-2ubuntu2.1 amd64 [upgradable from: 3.0.4-2ubunt
u2]
libapt-pkg6.0/jammy-updates 2.4.6 amd64 [upgradable from: 2.4.5]
```

```
sudo apt-get -y upgrade
```

команда обновляет все устаревшие пакеты и применить исправления безопасности с выводом подсказок « Да» или «Нет », спрашивая, нужно ли устанавливать зависимые пакеты или нет. Если вы устанавливаете большой пакет или пакет с большим количеством зависимостей, или устанавливаете несколько пакетов, ответы на эти запросы не позволят вам переключиться на выполнение какой-либо работы.

```
vladpolskiy@linux:~$ sudo apt-get -y upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages will be upgraded:
  apparmor apt apt-utils base-files firmware-sof-signed isc-dhcp-client
  isc-dhcp-common libapparmor1 libapt-pkg6.0 libc-bin libc6 libgstreamer1.0-0
  libjson-c5 libldap-2.5-0 libldap-common libnetplan0 libnss-systemd
  libpam-systemd libsystemd0 libudev1 libusb-1.0-0 linux-firmware locales
  motd-news-config netplan.io python-apt-common python3-apt
  python3-distupgrade python3-gi python3-software-properties snapd
  software-properties-common systemd systemd-sysv systemd-timesyncd
  ubuntu-advantage-tools ubuntu-release-upgrader-core udev
38 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 282 MB of archives.
After this operation, 1,059 kB of additional disk space will be used.
Get:1 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 motd-news-con
fig all 12ubuntu4.2 [4,612 B]
Get:2 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libc6 amd64 2
.35-0ubuntu3.1 [3,235 kB]
Get:3 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 base-files am
d64 12ubuntu4.2 [62.7 kB]
Get:4 http://ru.archive.ubuntu.com/ubuntu jammy-updates/main amd64 libc-bin amd6
```

```
sudo apt upgrade
```

команда обновляет все устаревшие пакеты и применить исправления безопасности

```
vladpolskiy@linux:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
vladpolskiy@linux:~$
```

Установите необходимые инструменты

Основные элементы:

```
sudo apt-get -y install ca-certificates gnupg curl unzip gdal-bin \
tar wget bzip2 build-essential clang python3-psycopg2 python3-yaml \
python3-requests postgresql-client
```

```
vladpolskiy@linux:~$ sudo apt-get -y install ca-certificates gnupg curl unzip gc
al-bin \
tar wget bzip2 build-essential clang python3-psycopg2 python3-yaml \
python3-requests postgresql-client
[sudo] password for vladpolskiy:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20211016).
ca-certificates set to manually installed.
python3-requests is already the newest version (2.25.1+dfsg-2).
python3-requests set to manually installed.
python3-yaml is already the newest version (5.4.1-1ubuntu1).
python3-yaml set to manually installed.
tar is already the newest version (1.34+dfsg-1build3).
tar set to manually installed.
wget is already the newest version (1.21.2-2ubuntu1).
wget set to manually installed.
curl is already the newest version (7.81.0-1ubuntu1.3).
curl set to manually installed.
gnupg is already the newest version (2.2.27-3ubuntu2.1).
gnupg set to manually installed.
The following additional packages will be installed:
  binfmt-support clang-14 dpkg-dev fakeroot g++ g++-11 gcc gcc-11 gdal-data
```

Дополнительные элементы:

```
sudo apt-get -y install munin-node munin protobuf-c-compiler libtiff5-dev
libcairomm-1.0-dev libagg-dev lua5.1 liblua5.1-0-dev
```

```
vladpolskiy@linux:~$ sudo apt-get -y install munin-node munin protobuf-c-compile
r libtiff5-dev
libcairomm-1.0-dev libagg-dev lua5.1 liblua5.1-0-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  jo jq libalgorithm-c3-perl libauthen-sasl-perl libb-hooks-endofscope-perl
  libb-hooks-op-check-perl libcgi-fast-perl libcgi-pm-perl libclass-c3-perl
  libclass-c3-xs-perl libclass-data-inheritable-perl
  libclass-method-modifiers-perl libclass-xsaccessor-perl libclone-perl
  libcommon-sense-perl libdata-optlist-perl libdate-manip-perl libdbil
  libdeflate-dev libdevel-callchecker-perl libdevel-caller-perl
  libdevel-globaldestruction-perl libdevel-lexalias-perl
  libdevel-stacktrace-perl libdist-checkconflicts-perl
  libdynaloader-functions-perl libemail-date-format-perl libencode-locale-perl
  libeval-closure-perl libexception-class-perl libexporter-tiny-perl
  libfcgi-bin libfcgi-perl libfcgi0ldbl libfile-copy-recursive-perl
```

Проверьте предварительные условия, предложенные openstreetmap-carto.

```
FROM ubuntu:bionic

# # Зависимости стиля
RUN apt-get update && apt-get install --no-install-recommends -y \
    ca-certificates curl gnupg postgresql-client python3 python3-distutils \
    fonts-hanazono fonts-noto-cjk fonts-noto-hinted fonts-noto-unhinted \
    mapnik-utils nodejs npm ttf-unifont unzip && rm -rf /var/lib/apt/lists/*

# Kosmtik with plugins, forcing prefix to /usr because bionic sets
```

```
# npm prefix to /usr/local, which breaks the install
RUN npm set prefix /usr && npm install -g kosmtik

РАБОЧИЙ КАТАЛОГ /usr/lib/node_modules/kosmtik/
RUN kosmtik plugins --install kosmtik-overpass-layer \
                    --install kosmtik-fetch-remote \
                    --install kosmtik-overlay \
                    --install kosmtik-open-in-josm \
                    --install kosmtik-map-compare \
                    --install kosmtik-osm-data-overlay \
                    --install kosmtik-mapnik-reference \
                    --install kosmtik-geojson-overlay \
    && cp /root/.config/kosmtik.yml /tmp/.kosmtik-config.yml

# Closing section
RUN mkdir -p /openstreetmap-carto
WORKDIR /openstreetmap-carto

USER 1000
CMD sh scripts/docker-startup.sh kosmtik =====
```

установим npm, диспетчер пакетов Node.js. Для этого установите пакет npmc помощью apt:

```
sudo apt install npm
```

```
vladpolskiy@linux:~$ sudo apt install npm
[sudo] password for vladpolskiy:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  gyp javascript-common libc-ares2 libdata-dump-perl libfile-basedir-perl
  libfile-desktopentry-perl libfile-listing-perl libfile-mimeinfo-perl
  libfont-afm-perl libhtml-form-perl libhtml-format-perl libhtml-tree-perl
  libhttp-cookies-perl libhttp-daemon-perl libhttp-negotiate-perl
  libio-stringy-perl libipc-system-simple-perl libjs-events libjs-highlight.js
  libjs-inherits libjs-is-typedarray libjs-psl libjs-source-map
  libjs-sprintf-js libjs-typedarray-to-buffer liblwp-protocol-https-perl
```

перейдем в режим суперпользователя

```
sudo -i
```

```
vladpolskiy@linux:~$ sudo -i
root@linux:~#
```

проверим зависимости стиля

```
apt-get update && apt-get install --no-install-recommends -y \
    ca-certificates curl gnupg postgresql-client python3 python3-distutils \
    fonts-hanazono fonts-noto-cjk fonts-noto-hinted fonts-noto-unhinted \
    mapnik-utils nodejs npm ttf-unifont unzip && rm -rf /var/lib/apt/lists/*
```

```
vladpolskiy@linux:~$ sudo -i
root@linux:~# apt-get update && apt-get install --no-install-recommends -y \
    ca-certificates curl gnupg postgresql-client python3 python3-distutils \
    fonts-hanazono fonts-noto-cjk fonts-noto-hinted fonts-noto-unhinted \
    mapnik-utils nodejs npm ttf-unifont unzip && rm -rf /var/lib/apt/lists/*
Hit:1 http://ru.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://ru.archive.ubuntu.com/ubuntu jammy-updates InRelease [114 kB]
Get:3 http://ru.archive.ubuntu.com/ubuntu jammy-backports InRelease [99.8 kB]
Get:4 http://ru.archive.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Fetched 324 kB in 1s (350 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package ttf-unifont is not available, but is referred to by another package.
This may mean that the package is missing, has been obsoleted, or
is only available from another source
However the following packages replace it:
  fonts-unifont

E: Package 'ttf-unifont' has no installation candidate
root@linux:~#
```

<color #ed1c24>E: Package 'ttf-unifont' has no installation candidate</color>
пакет 'ttf-unifont' установим позже (сейчас не критично)
к проверке установки kosmtik вернемся после установки Node.js

Для последующих шагов установки мы предполагаем, что:
cd - по умолчанию используется ваш домашний каталог.

Настроить swap

SWAP (своп) — это механизм виртуальной памяти, при котором часть данных из оперативной памяти (ОЗУ) перемещается на хранение на HDD (жёсткий диск), SSD (твёрдотельный накопитель).
Импорт картографических данных и управление ими занимают много оперативной памяти, и обычно требуется swap.
Чтобы проверить, настроен ли раздел подкачки в вашей системе, используйте одну из следующих двух команд:

```
swapon -s
```

```
root@linux:~# swapon -s
Filename                                Type            Size            Used    P
riority
/swap.img                               file            4194300         0       -
2
root@linux:~#
```

в данном случае установлено:

- • 4gb файла подкачки

```
free -m
```

Отображение количества свободной и используемой памяти в системе (отметьте строку,

указывающую Swap):

```
root@linux:~# free -m
               total        used        free      shared  buff/cache   available
Mem:           15988         554        9557           2        5876       15093
Swap:           4095           0        4095
root@linux:~#
```

в данном случае используется:

- • ОЗУ: 544mb из доступных 16gb
- • файл подкачки: 0mb из доступных 4gb.

Если у вас нет активного раздела подкачки, особенно если ваша физическая память мала, вам следует добавить файл подкачки. Создайте файл подкачки размером 2G в корневой файловой системе:

```
sudo fallocate -l 2G /swapfile
```

```
root@linux:~# sudo fallocate -l 2G /swapfile
root@linux:~#
```

Установим правильный тип разрешений.

```
sudo chmod 600 /swapfile
```

```
root@linux:~# sudo chmod 600 /swapfile
```

Используйте mkswap утилиту, чтобы настроить файл как область подкачки Linux:

```
sudo mkswap /swapfile
```

```
root@linux:~# sudo mkswap /swapfile
Setting up swapspace version 1, size = 2 GiB (2147479552 bytes)
no label, UUID=e869d3f9-7178-4137-b24a-8ca6aa9861cd
```

Включить файл подкачки

```
sudo mkswap /swapfile
```

```
root@linux:~# sudo swapon /swapfile
```

проверим результат

```
free -m
```

```
root@linux:~# free -m
               total        used        free      shared  buff/cache   available
Mem:           15988         563        9544           2        5880       15083
Swap:           6143           0        6143
root@linux:~#
```

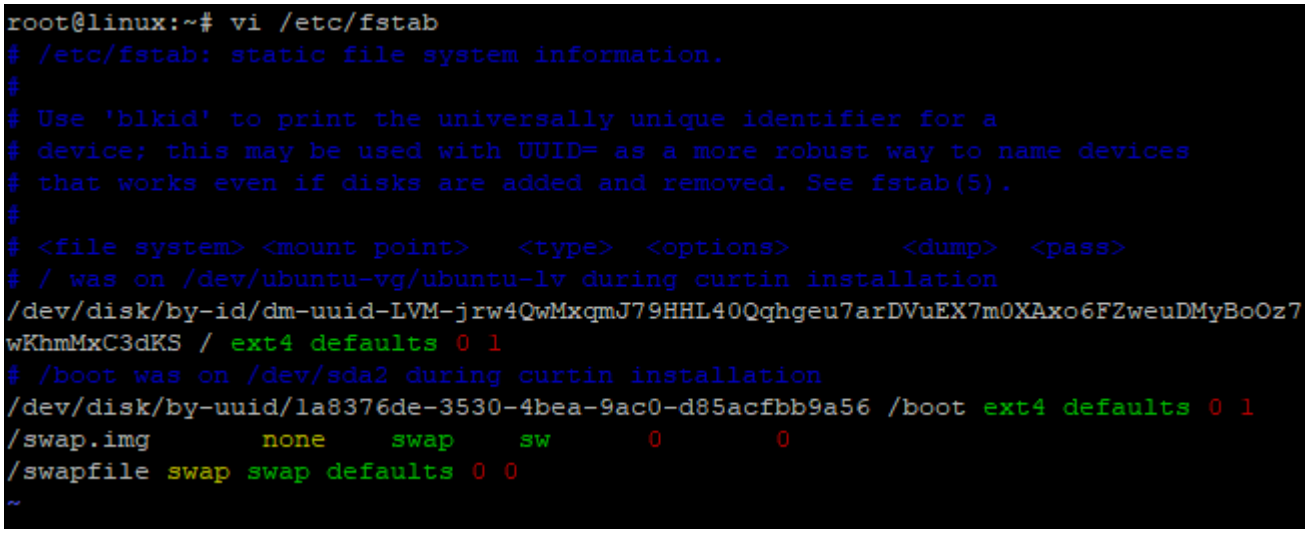
Чтобы сделать изменение постоянным, откройте /etc/fstab файл и добавьте следующую строку:

/etc/fstab

```
/swapfile swap swap defaults 0 0
```

Откроем файл fstab, находящийся в папке ets в редакторе:

```
vi /etc/fstab
```



- i -начало редактирования
- esc -конец редактирования
- :wq! -сохранение и выход из редактора

Swap в linux
Рекомендованный размер swap в linux

Настройка переменных локализации

- Locale — это набор переменных среды, которые определяют язык, страну и настройки кодировки символов (или любые другие особые предпочтения варианта) для ваших приложений и сеанса оболочки в системе Linux. Эти переменные среды используются системными библиотеками и локальными приложениями в системе.
- Конфигурационный файл находится /etc/default/locale.
- Просмотрим, какие локали в настоящее время определены для текущей учетной записи пользователя:

```
locale
```

```
root@linux:~# locale
LANG=en_US.UTF-8
LANGUAGE=
LC_CTYPE="en_US.UTF-8"
LC_NUMERIC="en_US.UTF-8"
LC_TIME="en_US.UTF-8"
LC_COLLATE="en_US.UTF-8"
LC_MONETARY="en_US.UTF-8"
LC_MESSAGES="en_US.UTF-8"
LC_PAPER="en_US.UTF-8"
LC_NAME="en_US.UTF-8"
LC_ADDRESS="en_US.UTF-8"
LC_TELEPHONE="en_US.UTF-8"
LC_MEASUREMENT="en_US.UTF-8"
LC_IDENTIFICATION="en_US.UTF-8"
LC_ALL=
```

Чтобы установить локализацию ru_RU:

```
export LANGUAGE=ru_RU.UTF-8
export LANG=ru_RU.UTF-8
export LC_ALL=ru_RU.UTF-8
```

- Экспортированные переменные можно поместить в файл /etc/environment.
- Новые локали также можно создать, выполнив:

```
sudo locale-gen ru_RU ru_RU.UTF-8
sudo dpkg-reconfigure locales
```

- если ошибка изменения локализации:
- -bash: warning: setlocale: LC_ALL: cannot change locale

Создание пользователя UNIX

- Мы предполагаем, что вы уже создали пользователя для входа во время установки Ubuntu Server, который будет использоваться для запуска сервера тайлов. Предположим, что выбранное вами имя пользователя — alisa. В этом документе каждый раз, когда будет упоминается alisa, замените его своим фактическим именем пользователя.
- Если вам нужно создать нового пользователя:

```
sudo useradd -m alisa
sudo passwd alisa
```

```
root@linux:~# sudo useradd -m alisa
sudo passwd alisa
New password:
Retype new password:
```

- Установите пароль при появлении запроса.
- Повторите пароль для правильности ввода.

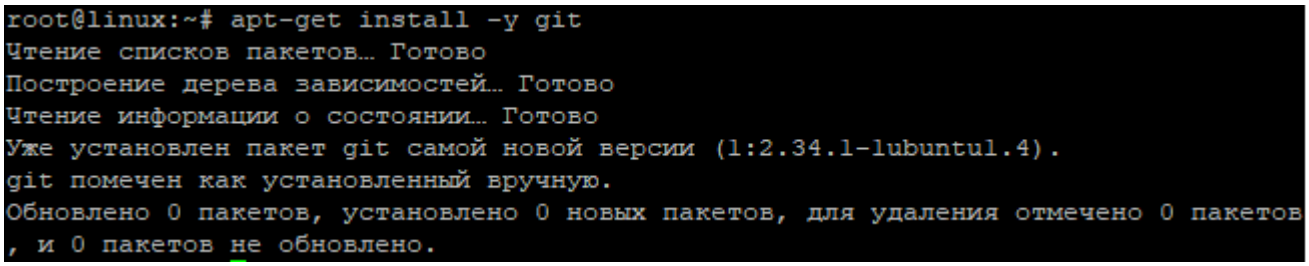
Установить Git

Иногда Git может быть уже предустановлен.

```
git --version
```

в случае отсутствия установим:

```
apt-get install -y git
```



- Справочник Pro Git

Установка библиотеки Mapnik

Mapnik используется для рендеринга данных OpenStreetMap в тайлы, управляемые веб-сервером Apache через renderd и mod_tile. Эта установка предусматривает, что все команды выполняются в окружении обычной учетной записи (не root) с использованием «sudo». Не пытайтесь и не делайте все в окружении root; система не будет работать. (выход: exit)

```
exit
```

Создайте в корне каталог src для загрузки исходников

```
mkdir -p ~/src
```

и перейдите в него:

```
cd ~/src
```


```
sudo apt install python-is-python3
```

df

Зависимость FreeType в Ubuntu 16.04 LTS

Установить библиотеку Mapnik из пакета

Или установите Mapnik из исходников

Установить Boost из пакета

В качестве альтернативы установите последнюю версию Boost из исходного кода

Установите HarfBuzz из пакета

HarfBuzz — это движок формирования текста OpenType .

Его можно установить из пакета, но лучше загрузить более обновленную версию исходного кода и скомпилировать ее. Для установки из пакета:

```
sudo apt-get install -y libharfbuzz-dev
```

Установите HarfBuzz из исходного кода

HarfBuzz — это движок формирования текста OpenType .

Его можно установить из пакета, но лучше загрузить более обновленную версию исходного кода и скомпилировать ее. Для установки из пакета:

```
sudo apt-get install -y libharfbuzz-dev
```

Собрать библиотеку Mapnik из исходников

На момент написания статьи Mapnik 3.0 является текущей стабильной версией и должна использоваться. Ветка для последней версии Mapnik из серии 3.0.x — v3.0.x.

Загрузите последние исходники Mapnik:

```
cd ~/src
git clone -b v3.0.x https://github.com/mapnik/mapnik
```

```
cd mapnik
git submodule update --init
source bootstrap.sh
./configure CUSTOM_CXXFLAGS="-D_GLIBCXX_USE_CXX11_ABI=0" CXX=${CXX} CC=${CC}
make
```

После успешной компиляции Mapnik используйте следующую команду для его установки в вашу систему:

```
make test
sudo make install
cd ~/
```

Связывания Python не включены по умолчанию. Вам нужно будет добавить их отдельно.

- Установка предварительных условий:

```
sudo apt-get install -y python-setuptools python3-setuptools
```

Только в случае, если вы установили *boost* из пакета, вам также необходимо:

```
sudo apt-get install -y libboost-python-dev
```

Не выполняйте указанную выше установку *libboost-python-dev* с **boost** , скомпилированным из исходного кода.

Установите переменные BOOST , если вы установили boost из исходников:

```
export BOOST_PYTHON_LIB=boost_python
export BOOST_THREAD_LIB=boost_thread
export BOOST_SYSTEM_LIB=boost_system
```

- Загрузите и скомпилируйте *python-mapnik* . Мы по-прежнему используем ветку v3.0.x:

```
cd ~/src
git clone -b v3.0.x https://github.com/mapnik/python-mapnik.git
cd python-mapnik
sudo -E python3 setup.py develop
sudo -E python3 setup.py install
```

Примечание: Mapnik и *mapnik-config* (часть Mapnik) должны быть установлены перед этой настройкой.

Затем вы можете убедиться, что Mapnik установлен правильно .

Убедитесь, что Mapnik был правильно установлен

Сообщить номер версии Mapnik и указать путь к каталогу входных плагинов

```
mapnik-config -v
```

```
mapnik-config --input-plugins
```

Убедитесь, что Python установлен . Также убедитесь, что pip установлен .

Затем проверьте с помощью Python 3:

```
python3 -c "import mapnik;print(mapnik.__file__)"
```

Если используется Python 2.7 (не Ubuntu 20.04 LTS), используйте эту команду для проверки:

```
python -c "import mapnik;print mapnik.__file__"
```

Он должен вернуть путь к привязкам python *(например, /usr/lib/python2.7/dist-packages/mapnik/*init*.pyc)*. Если python отвечает без ошибок, то библиотека Mapnik была найдена Python.

Настроить брандмауэр

Если вы готовите удаленную виртуальную машину, настройте брандмауэр так, чтобы разрешить удаленный доступ к локальному порту 80 и локальному порту 443.

Если вы используете облачную виртуальную машину, сама виртуальная машина также должна быть настроена на открытие этого порта.

Установить HTTP-сервер Apache

Apache бесплатный HTTP-сервер с открытым исходным кодом — один из самых популярных веб-серверов в мире. Он хорошо документирован и широко использовался на протяжении большей части истории Интернета, что делает его отличным выбором по умолчанию для хостинга веб-сайта .

Чтобы установить Apache:

```
sudo apt-get install -y apache2 apache2-dev
```

Службу Apache можно запустить с помощью

```
sudo service apache2 start
```

Ошибка «Не удалось включить APR_TCP_DEFER_ACCEPT» в Ubuntu на Windows возникает из-за этой опции сокета, которая изначально не поддерживается Windows. Чтобы ее обойти, отредактируйте */etc/apache2/apache2.con*f с помощью

```
sudo nano /etc/apache2/apache2.conf
```

и добавьте следующую строку в конец файла:

```
AcceptFilter http none
```

Чтобы проверить, установлен ли Apache, направьте ваш браузер на IP-адрес вашего сервера (например, http://localhost). Страница должна отображать домашнюю страницу Apache по умолчанию. Также эта команда позволяет проверить корректность работы:

```
curl localhost| grep 'It works!'
```

Настройки Apache, принятые серверами тайлов OpenStreetMap, можно найти в соответствующей конфигурации Chef .

Как узнать IP-адрес вашего сервера

Вы можете выполнить следующую команду, чтобы узнать публичный IP-адрес вашего сервера:

```
wget https://ipinfo.io/ip -qO -
```

Вы можете протестировать Apache, открыв его через браузер по адресу **http://your-server-ip**

Установить Mod_tile из пакета

Mod_tile — это модуль Apache для эффективного рендеринга и обслуживания фрагментов карты для www.openstreetmap.org с использованием Mapnik.

Mod_tile/render для Ubuntu 18.04 и Ubuntu 20.04

В Ubuntu 18.04 (bionic) и Ubuntu 20.04 (focal) *mod_tile/renderd* можно установить, добавив OpenStreetMap PPA , поддерживаемый командой «Администраторы OpenStreetMap»:

```
sudo add-apt-repository -y ppa:osmadmins/ppa
sudo apt-get update
```

Также подходит вышеупомянутый таладж PPA .

После добавления PPA *mod_tile/renderd* можно установить из пакета с помощью следующей команды:

```
sudo apt-get install -y libapache2-mod-tile # this includes both mod-tile
and renderd
```

Mod_tile/renderd для Ubuntu 21.04

В Ubuntu 21.04 (hirsute) пакет доступен и может быть установлен с помощью

```
sudo apt-get install -y libapache2-mod-tile renderd
```

Установить Mod_tile из исходников

В качестве альтернативы установке Mod_tile через PPA, мы можем скомпилировать его из репозитория GitHub.

Чтобы удалить ранее установленный PPA и связанные с ним пакеты:

```
sudo apt-get purge -y libapache2-mod-tile
sudo apt -y autoremove
sudo add-apt-repository -y --remove ppa:osmadmins/ppa
sudo add-apt-repository -y --remove ppa:talaj/osm-mapnik
```

Для компиляции Mod_tile:

```
sudo apt-get install -y autoconf autogen
mkdir -p ~/src ; cd ~/src
git clone https://github.com/openstreetmap/mod_tile.git

# Alternative repository:
# git clone -b switch2osm git://github.com/SomeoneElseOSM/mod_tile.git

cd mod_tile
./autogen.sh && ./configure && make && sudo make install && sudo make
install-mod_tile && sudo ldconfig
cd ~/
```

Проверьте также
https://github.com/openstreetmap/mod_tile/blob/master/docs/build/building_on_ubuntu_20_04.md

Процесс рендеринга, реализуемый mod_tile и renderd, подробно описан в соответствующем файле readme на GitHub .

Установка Python

Проверьте, установлен ли Python :

```
sudo apt-get install -y python3 python3-distutils

# Verify Python installation:
python -V
python3 -V
```

Установите Yaml и диспетчер пакетов для Python

Это необходимо для запуска скриптов/индексов *OpenStreetMap-Carto*.

```
sudo apt-get install -y python-yaml

pip -V # to verify whether pip is already installed
sudo apt-get install -y python3-pip
python3 -m pip install --upgrade pip
```

Установите утилиты Mapnik

Пакет Mapnik Utilities включает shapeindex.

```
sudo apt-get install -y mapnik-utils
```

Установить openstreetmap-carto

```
mkdir -p ~/src
cd ~/src
git clone https://github.com/gravitystorm/openstreetmap-carto.git
cd openstreetmap-carto
```

Для получения дополнительной информации прочтите примечания по установке .

Установите шрифты, необходимые для openstreetmap-carto

В настоящее время используются шрифты Noto.

Чтобы установить их (кроме Noto Emoji Regular и Noto Sans Arabic UI Regular/Bold):

```
sudo apt-get install -y fonts-noto-cjk fonts-noto-hinted fonts-noto-unhinted
fonts-hanazono ttf-unifont
```

Установка шрифтов Noto (при наличии следует использовать шрифты с подсказками):

```
cd ~/src
git clone https://github.com/googlefonts/noto-emoji.git
git clone https://github.com/googlefonts/noto-fonts.git

sudo cp noto-emoji/fonts/NotoColorEmoji.ttf /usr/share/fonts/truetype/noto
sudo cp noto-emoji/fonts/NotoEmoji-Regular.ttf
```

```
/usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoSansArabicUI/NotoSansArabicUI-Regular.ttf
/usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoNaskhArabicUI/NotoNaskhArabicUI-
Regular.ttf /usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoSansArabicUI/NotoSansArabicUI-Bold.ttf
/usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoNaskhArabicUI/NotoNaskhArabicUI-Bold.ttf
/usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoSansAdlam/NotoSansAdlam-Regular.ttf
/usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoSansAdlamUnjoined/NotoSansAdlamUnjoined-
Regular.ttf /usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoSansChakma/NotoSansChakma-Regular.ttf
/usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoSansOsage/NotoSansOsage-Regular.ttf
/usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoSansSinhalaUI/NotoSansSinhalaUI-
Regular.ttf /usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoSansArabicUI/NotoSansArabicUI-Regular.ttf
/usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoSansCherokee/NotoSansCherokee-Bold.ttf
/usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoSansSinhalaUI/NotoSansSinhalaUI-Bold.ttf
/usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoSansSymbols/NotoSansSymbols-Bold.ttf
/usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoSansArabicUI/NotoSansArabicUI-Bold.ttf
/usr/share/fonts/truetype/noto
sudo cp noto-fonts/unhinted/ttf/NotoSansSymbols2/NotoSansSymbols2-
Regular.ttf /usr/share/fonts/truetype/noto
sudo cp noto-fonts/hinted/ttf/NotoSansBalinese/NotoSansBalinese-Regular.ttf
/usr/share/fonts/truetype/noto
sudo cp noto-fonts/archive/hinted/NotoSansSyriac/NotoSansSyriac-Regular.ttf
/usr/share/fonts/truetype/noto

mkdir NotoSansSyriacEastern-unhinted
cd NotoSansSyriacEastern-unhinted
wget
https://noto-website-2.storage.googleapis.com/pkgs/NotoSansSyriacEastern-unh
inted.zip
unzip NotoSansSyriacEastern-unhinted.zip
sudo cp NotoSansSyriacEastern-Regular.ttf /usr/share/fonts/truetype/noto
cd ..

sudo apt install fontconfig
```

В конце:

```
sudo fc-cache -fv
fc-list
fc-list | grep Emoji
```

DejaVu Sans используется как необязательный резервный шрифт для систем без Noto Sans. Если установлены все шрифты Noto, его никогда не следует использовать.

```
sudo apt-get install -y fonts-dejavu-core
```

Дополнительную информацию смотрите в примечаниях к шрифтам .

Старый унифонт Средний шрифт

Шрифт *unifont Medium* (нижняя метка), который был включен в предыдущие версии ОС, теперь больше не доступен и заменен на Unifont Medium (верхняя). Предупреждения, связанные с недоступностью *unifont Medium*, не актуальны и являются следствием старого решения разработчиков OpenStreetMap поддерживать как старый шрифт Ubuntu 12.04, так и более новую версию (в верхнем регистре).

Один из способов избежать предупреждения — удалить ссылку на «unifont Medium» в *openstreetmap-carto/style.xml* .

Другой альтернативный способ удаления предупреждения о строчных буквах *unifont Medium* — установка старого шрифта «unifont Medium» (используется в Ubuntu 12.10):

```
mkdir -p ~/src ; cd ~/src
mkdir OldUnifont
cd OldUnifont
wget
http://http.debian.net/debian/pool/main/u/unifont/unifont_5.1.20080914.orig.
tar.gz
tar xvfz unifont_5.1.20080914.orig.tar.gz
unifont-5.1.20080914/font/precompiled/unifont.ttf
sudo cp unifont-5.1.20080914/font/precompiled/unifont.ttf
/usr/share/fonts/truetype/unifont/OldUnifont.ttf
sudo fc-cache -fv
fc-list | grep -i unifont # both uppercase and lowercase fonts will be
listed
```

Обратите внимание, что описанная выше операция установки бесполезна, она просто удаляет предупреждение.

Установите Node.js

Установите Node.js с Ubuntu 20.04 LTS:

```
# Check installations:
```

```
node -v
nodejs -v
npm -v

# If components needs to be installed, run the following:
sudo apt install -y nodejs npm
```

Перейдите на страницу проверки версий Node.js.

Дополнительные примечания по Node.js: другие способы его установки:

- NodeSource Node.js Двоичные дистрибутивы . Проверьте соответствующие инструкции по установке .
- Стандартный режим с помощью Ubuntu Advanced Packaging Tool (APT)
- Инструменты управления версиями, такие как:
  - n (Интерактивное управление версиями Node.js)
  - nvm (менеджер версий узлов)

Список [https://ircama.github.io/osm-carto-tutorials/nodejs-commands|полезных команд]] для управления *Node.js* доступен на специальной странице.

Указанная выше версия Node.js также поддерживает установку TileMill и Carto.

Версия дистрибутива из менеджера пакетов APT

Последние версии Ubuntu поставляются с Node.js (пакет *nodejs*) и npm (пакет *npm*) в репозиториях по умолчанию. В зависимости от того, какую версию Ubuntu вы используете, эти пакеты могут содержать устаревшие релизы; тот, что идет с Ubuntu 16.04, не будет последним, но он должен быть стабильным и достаточным для запуска Kosmtik и Carto. Вместо этого TileMill требуется *nodejs-legacy* (или старая версия node, установленная через инструмент управления версиями Node.js).

Для *carto* мы установим *nodejs* :

```
sudo apt-get install -y nodejs npm
node -v 2>/dev/null || sudo ln -fs /usr/bin/nodejs /usr/local/bin/node
nodejs -v
node -v
npm -v
```

Установите Node.js с помощью инструмента управления версиями

В качестве альтернативы предлагается использовать инструмент управления версиями *Node.js*, который упрощает интерактивное управление различными версиями *Node.js* и позволяет выполнять обновление до последней версии. Мы будем использовать n .

Установить *n* :

```
mkdir -p ~/src ; cd ~/src
git clone https://github.com/tj/n.git
cd n
sudo make install # To uninstall: sudo make uninstall
cd ..
```

Некоторые программы (например, *Kosmtik и carto*) принимают последнюю версию узла LTS (*sudo n lts*), другие (например, *Tilemill*) работают с v6.14.1 (*sudo n 6.14.1*).

Для *carto* мы установим последнюю версию LTS:

```
sudo n lts
```

Проверьте версии Node.js

Чтобы получить номера установленных версий:

```
node -v
npm -v
```

Установите carto и создайте XML-таблицу стилей Mapnik

Carto — это компилятор таблиц стилей, преобразующий проекты CartoCSS в таблицы стилей Mapnik XML.

Согласно текущей документации openstreetmap-carto , минимальная версия carto (CartoCSS), которую можно установить, — 0.18 . Поскольку carto компилирует таблицы стилей openstreetmap-carto, рекомендуется сохранить ту же версию, что и в документации openstreetmap-carto (вместо простой установки последней версии carto).

Последнюю версию carto 1.2.0 можно установить с помощью

```
sudo npm install -g carto
```

Это работает с Ubuntu 20.04 LTS.

До Ubuntu 18.04 LTS эта версия выдает предупреждения типа «Стили не соответствуют селектору слоев .text-low-zoom» .

Чтобы избежать этих предупреждений, установите версию 0 carto :

```
sudo npm install -g carto@0
```

На момент написания статьи это должна быть версия 0.18.2 .

Если установка не удалась, возможно, это связано с несовместимостью с npm/Node.js. Чтобы исправить это, попробуйте понизить версию Node.js.

Чтобы проверить установленную версию:

```
carto -v
```

При запуске carto вам необходимо указать версию API Mapnik через **-a** опцию. Для принятия версии документация openstreetmap-carto предлагает некоторые рекомендации.

Чтобы вывести список всех известных версий API в установленном программном обеспечении узла , выполните следующую команду:

```
npm install mapnik-reference
node -e "console.log(require('mapnik-reference'))"
```

Спецификации для каждой версии API также задокументированы в репозитории carto .

Вам следует использовать версию API, наиболее близкую к установленной у вас версии Mapnik (уточните у *mapnik-config -v*).

Протестируйте *carto* и создайте *style.xml* из стиля openstreetmap-carto :

```
cd ~/src
cd openstreetmap-carto
carto -a "3.0.22" project.mml > style.xml
ls -l style.xml
```

При выборе соответствующей версии API вы не должны получить никаких соответствующих предупреждающих сообщений.

Команда *sudo apt-get install -y node-carto* может установить старую версию *Carto* , несовместимую с Openstreetmap Carto, и ее следует избегать.

Установите PostgreSQL и PostGIS

PostgreSQL — это реляционная база данных, а PostGIS — ее пространственный расширитель, позволяющий хранить в ней географические объекты, такие как картографические данные; он выполняет функцию, схожую с SDE от ESRI или пространственным расширением Oracle. PostgreSQL + PostGIS используются для широкого спектра функций, таких как рендеринг карт, геокодирование и анализ.

В настоящее время протестированными версиями OpenstreetMap Carto являются PostgreSQL 10 и PostGIS 2.4:

Также должна подойти более старая или более новая версия PostgreSQL .

В Ubuntu имеются предварительно упакованные версии postgis и postgresql, поэтому их можно просто установить через менеджер пакетов Ubuntu.

```
sudo apt-get update
```

```
sudo apt-get install -y postgresql postgis
```

Дополнительные компоненты:

```
sudo apt-get install -y postgresql-contrib postgresql-12-postgis-3
postgresql-12-postgis-3-scripts
```

Вам необходимо запустить базу данных:

```
sudo service postgresql start
```

Примечание: используемый порт PostgreSQL — 5432 (по умолчанию).

Создайте экземпляр PostGIS

Теперь вам нужно создать базу данных PostGIS. По умолчанию различные программы, включая openstreetmap-carto *(ref. project.mml)*, предполагают, что база данных называется gis . Вам нужно создать базу данных PostgreSQL и настроить на ней расширение PostGIS.

Схема кодировки символов, которая будет использоваться в базе данных, — *UTF8* , а принятый параметр сортировки — *en_GB.utf8* . (U&«...«Экранированный синтаксис Unicode, используемый в *project.mml*, должен работать только в том случае, если кодировкой сервера является UTF8 . Это также соответствует тому, что указано в коде конфигурации PostgreSQL Chef .)

```
sudo -u postgres createuser -s $USER
createdb gis --encoding="UTF8" --lc-collate="en_GB.UTF-8" --lc-
ctype="en_GB.UTF-8" --template=template0
psql -d gis -c 'CREATE EXTENSION postgis; CREATE EXTENSION hstore;'
```

Примечание: **ERROR: invalid locale name: «en_GB.UTF-8»** означает, что локаль *en_GB.UTF-8* не установлена. После установки локали необходимо перезапустить базу данных, чтобы загрузить локаль.
Перейти к следующему шагу.
Если на другом хосте:

Установите переменные среды

```
export PGHOST=localhost
export PGPORT=5432
export PGUSER=postgres
export PGPASSWORD=postgres_007%
```

```
HOSTNAME=localhost # set it to the actual ip address or host name
createdb gis --host="$HOSTNAME" --encoding="UTF8" --lc-collate="en_GB.UTF-8"
--lc-ctype="en_GB.UTF-8" --template=template0
```

Если вы получили следующую ошибку:
ERROR: invalid locale name: «en_GB.utf8»

то вам нужно добавить локаль «en_GB.utf8» с помощью следующей команды:

```
sudo dpkg-reconfigure locales
```

И выберите «en_GB.UTF-8 UTF-8» на первом экране («Locales to be generated»). После этого будет предложено перезапустить базу данных:

```
sudo service postgresql restart
```

Если вы получили следующую ошибку:
ERROR: new collation (en_GB.utf8) is incompatible with the collation of the template database (en_US.UTF-8)
HINT: Use the same collation as in the template database, or use template0 as template.
вам нужно использовать template0 для gis:

```
psql -U postgres -h $HOSTNAME -c "CREATE DATABASE gis ENCODING 'UTF-8'
LC_COLLATE 'en_GB.utf8' LC_CTYPE 'en_GB.utf8' TEMPLATE template0"
# alternative command: createdb -E UTF8 -l en_GB.UTF8 -O postgres  -T
template0 gis
```

Если вы получили следующую ошибку:
ERROR: new encoding (UTF8) is incompatible with the encoding of the template database (SQL_ASCII)
HINT: Use the same encoding as in the template database, or use template0 as template.
(ошибка, которая обычно возникает в Ubuntu на Windows с WSL), затем добавьте также *TEMPLATE template0;* например, используйте следующую команду:

```
psql -U postgres -h $HOSTNAME -c "CREATE DATABASE gis ENCODING 'UTF-8'
LC_COLLATE 'en_GB.utf8' LC_CTYPE 'en_GB.utf8' TEMPLATE template0"
# alternative command: createdb -E UTF8 -l en_GB.utf8 -O postgres -T
template0 gis
```

Установите флажок, чтобы создать базу данных в разделе диска, где достаточно свободного места. Если вам необходимо использовать табличное пространство, отличное от используемого по умолчанию, выполните следующие команды вместо предыдущих (пример: табличное пространство имеет местоположение */tmp/db*):

```
sudo mkdir /mnt/db # Suppose this is the tablespace location
sudo chown postgres:postgres /mnt/db
psql -U postgres -h $HOSTNAME -c "CREATE TABLESPACE gists LOCATION
'/mnt/db'"
psql -U postgres -h $HOSTNAME -c "ALTER DATABASE gis SET TABLESPACE gists"
```

Создайте расширения *postgis и hstore* :

```
psql -U postgres -h $HOSTNAME -c "\connect gis"
psql -U postgres -h $HOSTNAME -d gis -c "CREATE EXTENSION postgis"
psql -U postgres -h $HOSTNAME -d gis -c "CREATE EXTENSION hstore"
```

Если вы получили следующую ошибку
ERROR: could not open extension control file »/usr/share/postgresql/9.3/extension/postgis.control»: No such file or directory
то вы, возможно, устанавливаете PostgreSQL 9.3 (вместо 9.5), для чего вам также понадобится:

```
sudo apt-get install postgis postgresql-9.3-postgis-scripts
```

Установите его и повторите команды создания расширения. Обратите внимание, что PostgreSQL 9.3 в настоящее время не поддерживается openstreetmap-carto.

Добавьте пользователя и предоставьте доступ к базе данных gis

Для того, чтобы приложение могло получить доступ к базе данных *gis* , необходим пользователь DB с тем же именем, что и у вашего пользователя UNIX. Предположим, что ваш пользователь UNIX — *tileserver* .

```
psql -d gis -c "create user tileserver;grant all privileges on database gis
to postgres;"
psql -d gis -c 'create user "www-data";grant all privileges on database gis
to "www-data";'

psql -d gis -c 'ALTER TABLE geometry_columns OWNER TO postgres;'
psql -d gis -c 'ALTER TABLE spatial_ref_sys OWNER TO  postgres;'
```

Включение удаленного доступа к PostgreSQL

Если вы находитесь на другом хосте, для удаленного доступа к PostgreSQL вам необходимо отредактировать *pg_hba.conf* :

```
sudo nano /etc/postgresql/*/main/pg_hba.conf
```

и добавьте следующую строку:

```
host    all             all             <your IP set>/<your netmask>
md5
```

host all all 0.0.0.0/0 md5— это правило контроля доступа, которое позволяет любому человеку входить в систему с любого адреса, указав действительный пароль (ключевое слово md5).

Затем отредактируйте *postgresql.conf* :

```
sudo vi /etc/postgresql/*/main/postgresql.conf
```

и установить *listen_addresses* = '*'

Наконец, необходимо перезапустить БД:

```
sudo /etc/init.d/postgresql restart
```

Проверьте, доступна ли база данных *gis* . Чтобы вывести список всех баз данных, определенных в PostgreSQL, выполните следующую команду:

```
psql -U postgres -h $HOSTNAME -c "\l+"
```

Полученный отчет должен включать базу данных *ГИС* , как показано в следующей таблице:

Name	Owner	Encoding	Collate	Ctype	Access privileges
gis	postgres	UTF8	en_US.utf8	en_US.utf8	=Tc/postgres
					postgres=CTc/postgres
					tileserver=CTc/postgres

Настройка базы данных

Настройки PostgreSQL по умолчанию не очень хороши для очень больших баз данных, таких как OSM. Правильная настройка может просто удвоить производительность.

Минимальные требования к настройке

Установите пользователя *postgres* для доверия :

```
sudo vi /etc/postgresql/*/main/pg_hba.conf
# change: local   all             postgres
peer
# to:     local   all             postgres
trust
```

После выполнения вышеуказанных изменений перезапустите БД:

```
sudo service postgresql restart
```

Запустите *tune-postgis.sh* :

```
export POSTGRES_USER=postgres
export PG_MAINTENANCE_WORK_MEM=256MB
export PG_WORK_MEM=16MB
export psql=psql

cd ~/src
cd openstreetmap-carto
bash scripts/tune-postgis.sh
```

Если postgres не настроен на *доверие* , возникает следующая ошибка: *psql: error: FATAL: Peer authentication failed for user «postgres»* при запуске tune-postgis.sh .

Чтобы очистить каталог данных и снова повторить *tune-postgis.sh* : rm -rf data.

Необязательные дополнительные требования к настройке

В вики PostgreSQL есть страница, посвященная настройке базы данных.

В блоге Пола Нормана есть интересная заметка об оптимизации базы данных, которая используется ниже.

maintenance_work_memНастройки по умолчанию work_memслишком низкие для рендеринга.11: оба параметра следует увеличить для более быстрой загрузки данных и более быстрых запросов (сканирования индекса).

Консервативные настройки для 4 ГБ ВМ — *work_mem=32MB* и *maintenance_work_mem=256MB*. На машине с достаточным объемом памяти вы можете установить их как *work_mem=256MB* и *maintenance_work_mem=1GB*.

Кроме того, важными настройками являются *shared_buffersи write-ahead-log (wal)*. Есть также некоторые другие настройки, которые вы, возможно, захотите изменить специально для импорта.

Чтобы отредактировать файл конфигурации PostgreSQL с помощью редактора nano :

```
sudo nano /etc/postgresql/*/main/postgresql.conf
```

а если вы используете PostgreSQL 9.3 (не поддерживается):

```
sudo nano /etc/postgresql/9.3/main/postgresql.conf
```

Рекомендуемые минимальные настройки:

```
shared_buffers = 128MB
min_wal_size = 1GB
max_wal_size = 2GB
work_mem = 32MB # check comments for better tuning
maintenance_work_mem = 256MB
autovacuum = off
fsync = off
```

Последние два варианта позволяют ускорить импорт: первый отключает автоматическую очистку во время импорта и позволяет запустить очистку в конце; второй приводит к повреждению данных в случае отключения электроэнергии и опасен. Если во время импорта данных произойдет отключение электроэнергии, вам придется удалить данные из базы данных и выполнить повторный импорт, но это быстрее. Просто не забудьте вернуть эти настройки обратно после импорта. fsync не влияет на время запроса после загрузки данных.

Настройки PostgreSQL, принятые OpenStreetMap, можно найти в книге PostgreSQL Chef Cookbook : специальные настройки PostgreSQL для серверов тайлов OpenStreetMap описаны в соответствующей конфигурации Tileserver Chef .

Для установки dev&test на системе с 16 ГБ ОЗУ предлагаются следующие настройки:12:

```
shared_buffers = 2GB
```

```
work_mem = 256MB
maintenance_work_mem = 1GB
wal_level = minimal
synchronous_commit = off
min_wal_size = 1GB
max_wal_size = 2GB
checkpoint_segments = 60
checkpoint_timeout = 15min
checkpoint_completion_target = 0.9
default_statistics_target = 1000
autovacuum = off
fsync = off
```

default_statistics_target можно даже увеличить до 10000.

При выполнении обновлений базы данных периодически запускайте ANALYZE.

Чтобы остановить и запустить базу данных:

```
sudo /etc/init.d/postgresql stop
sudo /etc/init.d/postgresql start
```

Вы можете получить ошибку и вам нужно увеличить размер разделяемой памяти. Отредактируйте **/etc/sysctl.d/30-postgresql-shm.conf** и запустите *sudo sysctl -p /etc/sysctl.d/30-postgresql-shm.conf*. Параметр типа *kernel.shmmax=17179869184и kernel.shmall=4194304* может быть подходящим для размера сегмента 16 ГБ.

Для управления и поддержания конфигурации серверов, работающих под управлением OpenStreetMap, используется инструмент управления конфигурацией Chef .

Для PostgreSQL принята конфигурация postgresql/attributes/default.rb .

Установить Osm2pgsql

Osm2pgsql — это специальное программное обеспечение OpenStreetMap, используемое для загрузки данных OSM в базу данных PostGIS.

По умолчанию пакетные версии Osm2pgsql — 0.88.1-1 на Ubuntu 16.04 LTS и 0.96.0 на Ubuntu 18.04 LTS. Тем не менее, предлагаются более свежие версии, доступные на OpenStreetMap Osmadmins PPA или компиляция программного обеспечения из исходников .

Чтобы установить osm2pgsql:

```
sudo apt install -y osm2pgsql
```

Чтобы установить Osm2pgsql из Osmadmins PPA:

```
sudo add-apt-repository -y ppa:osmadmins/ppa
```

```
apt-key adv --keyserver keyserver.ubuntu.com --recv
A438A16C88C6BE41CB1616B8D57F48750AC4F2CB
sudo apt-get update
sudo apt-get install -y osm2pgsql
```

Перейдите в раздел «Получить извлечение данных OpenStreetMap» .

Сгенерировать Osm2pgsql из исходников

Эта альтернативная процедура установки генерирует самый последний исполняемый файл путем компиляции исходных кодов.

Установите необходимые зависимости:

```
sudo apt-get install -y make cmake g++ libboost-dev libboost-system-dev \
  libboost-filesystem-dev libexpat1-dev zlib1g-dev \
  libbz2-dev libpq-dev libgeos-dev libgeos++-dev libproj-dev lua5.2 \
  liblua5.2-dev
```

Загрузить osm2pgsql:

```
mkdir -p ~/src ; cd ~/src
git clone git://github.com/openstreetmap/osm2pgsql.git
```

Подготовка к компиляции, компиляция и установка:

```
cd osm2pgsql
mkdir build
cd build
cmake -DCMAKE_BUILD_TYPE=Release ..
make
sudo make install
cd
```

Получите извлечение данных OpenStreetMap

Вам необходимо загрузить соответствующий файл .osm или .pbf, который затем можно загрузить в ранее созданный экземпляр PostGIS через *osm2pgsql*.

Существует много способов загрузки данных OSM.

Ссылка — Planet OSM .

Вероятно, проще всего получить PBF-файл данных OSM с сайта geofabrikсайта geofabrik .

Кроме того, BBBike.org предоставляет выдержки из более чем 200 городов и регионов по всему миру в различных форматах.

Примеры:

- Картографические данные всей планеты (76G):

```
wget -c https://planet.openstreetmap.org/pbf/planet-latest.osm.pbf
```

- Картографические данные Великобритании (847M):

```
wget -c https://download.geofabrik.de/europe/great-britain-latest.osm.pbf
```

- Только для Лихтенштейна:

```
wget https://download.geofabrik.de/europe/liechtenstein-latest.osm.pbf
wget https://download.geofabrik.de/europe/liechtenstein-latest.osm.pbf.md5

# Optionally, the following will xheck that the download wasn't corrupted:
md5sum -c liechtenstein-latest.osm.pbf.md5
```

Другой способ загрузки данных — напрямую через браузер. Проверьте эту страницу .

В качестве альтернативы можно использовать JOSM (выберите область для загрузки данных OSM: меню JOSM, Файл, Загрузить из OSM; вкладка Slippy map; перетащите карту правой кнопкой мыши, масштабируйте колесиком мыши или Ctrl + клавиши со стрелками; перетащите рамку левой кнопкой мыши, чтобы выбрать область для загрузки. Также предлагается плагин Continuous Download . Когда нужный регион будет локально доступен, выберите Файл, Сохранить как, <filename>.osm. Дайте ему допустимое имя файла и проверьте также соответствующий каталог, в котором этот файл сохранен.

В любом случае избегайте использования слишком маленьких участков.

OpenStreetMap — это *открытые данные* . Лицензия OSM — Open Database License.

Загрузить данные в PostGIS

В документации osm2pgsql приведена вся необходимая информация для использования этого инструмента ETL, включая соответствующие параметры командной строки .

osm2pgsql использует **overcommit**, как и многие научные приложения и приложения для работы с большими данными, что требует корректировки настроек ядра:

```
sudo sysctl -w vm.overcommit_memory=1
```

Чтобы загрузить данные из файла *.osm* или *.pbf* в PostGIS, выполните следующую команду:

```
cd ~/src
cd openstreetmap-carto

export OSM2PGSQL_CACHE=${OSM2PGSQL_CACHE:-512}
```

```
export OSM2PGSQL_NUMPROC=${OSM2PGSQL_NUMPROC:-1}
export OSM2PGSQL_DATAFILE=${OSM2PGSQL_DATAFILE:-data.osm.pbf}

osm2pgsql \
--cache $OSM2PGSQL_CACHE \
--number-processes $OSM2PGSQL_NUMPROC \
--hstore \
--multi-geometry \
--database gis \
--slim \
--drop \
--style openstreetmap-carto.style \
--tag-transform-script openstreetmap-carto.lua \
[.osm or .pbf file]
```

[.osm or .pbf file]: замените это на уже загруженный вами файл *.osm* или *.pbf* , например, *liechtenstein-latest.osm.pbf* .

При наличии доступной памяти установите **export OSM2PGSQL_CACHE=2500**; для процесса импорта будет выделено 2,5 ГБ памяти.

Опция **–create** загружает данные в пустую базу данных, а не пытается добавить их в существующую.

Ретранслируем на *OSM2PGSQL_NUMPROC*, если у вас доступно больше ядер, вы можете настроить его соответствующим образом.

В руководстве osm2pgsql подробно описывается использование и все параметры.

Перейти к следующему шагу .

При использовании другого сервера:

```
cd ~/src
cd openstreetmap-carto
HOSTNAME=localhost # set it to the actual ip address or host name
osm2pgsql -s -C 300 -c -G --hstore --style openstreetmap-carto.style --tag-
transform-script openstreetmap-carto.lua -d gis -H $HOSTNAME -U postgres
[.osm or .pbf file]
```

Обратите внимание, что предлагаемый процесс принимает опцию -s(–slim), которая использует временные таблицы, поэтому его выполнение занимает больше места на диске (и очень медленно), при этом используется меньше оперативной памяти. Вы можете добавить –drop опцию с -s(–slim), чтобы также удалить временные таблицы после импорта, в противном случае вы также найдете временные таблицы *nodes , ways и rels* (эти таблицы изначально были чистыми «вспомогательными» таблицами для систем с малым объемом памяти, но сегодня они широко используются, поскольку они также являются предпосылкой для обновлений).

Если все в порядке, можно переходить к следующему шагу .

Обратите внимание, что используются следующие элементы:

- hstore
- openstreetmap- carto.стиль
- скрипт LUA openstreetmap -carto.lua
- Имя базы данных ГИС

В зависимости от размера входного файла команда *osm2pgsql* может выполняться очень долго. Интересная страница, связанная с бенчмарками Osm2pgsql, связывает размеры систем hw/sw с соответствующими цифрами для импорта данных OpenStreetMap.

Примечание: если вы получили следующую ошибку:

```
node_changed_mark failed: ERROR:  prepared statement "node_changed_mark"
does not exist
```

выполните следующую команду для вашего *оригинального .osm* :

```
sed "s/action='modify' //" < original.osm | > fixedfile.osm
```

Затем обработайте *fixedfile.osm* .

Если вы получаете такие ошибки:

```
Error reading style file line 79 (fields=4)
flag 'phstore' is invalid in non-hstore mode
Error occurred, cleaning up
```

или этот:

```
Postgis Plugin: ERROR:  column "tags" does not exist
LINE 8: ...ASE WHEN "natural" IN ('mud') THEN "natural" ELSE tags->'wet...
```

то вам нужно включить расширение *hstore* в базу данных с помощью **CREATE EXTENSION hstore**;и также добавить флаг –hstore в osm2pgsql . Включение расширения *hstore* и использование его с *osm2pgsql* исправит эти ошибки.

Создайте папку данных

Для этого шага требуется не менее 18 ГБ HD и соответствующая *RAM/swap* (лучше 24 ГБ HD). 8 ГБ HD будет недостаточно. При 1 ГБ RAM настройка swap обязательна.

```
python3 -m pip install psycopg2-binary
cd ~/src
cd openstreetmap-carto
scripts/get-external-data.py
```

Чтобы очистить процедуру *get-external-data.py* и перезапустить ее с нуля, удалите каталог *данных* (rm -r data).

Настройте подкачку , чтобы предотвратить появление следующего сообщения:

```
INFO:root:Checking table water_polygons
Killed
```

Способ загрузки шейп-файлов серверами тайлов OpenStreetMap описан в соответствующей конфигурации Chef .

Для получения дополнительной информации прочтите скрипт загрузки .

Создание индексов и предоставление пользователям

Создайте частичные индексы для ускорения запросов, включенных в *project.mml* , и предоставьте доступ ко всем таблицам *ГИС* , чтобы избежать ошибок рендеринга при доступе к таблицам с пользователем *tileserver* .

- Добавьте частичные геометрические индексы, указанные *openstreetmap-carto* для обеспечения эффективного улучшения запросов:

```
cd ~/src
cd openstreetmap-carto
HOSTNAME=localhost # set it to the actual ip address or host name
psql -d gis -f indexes.sql
```

Альтернативный режим:

```
cd ~/src
cd openstreetmap-carto
scripts/indexes.py | psql -d gis
```

При использовании другого хоста:

```
HOSTNAME=localhost # set it to the actual ip address or host name
cd ~/src
cd openstreetmap-carto
scripts/indexes.py | psql -U postgres -h $HOSTNAME -d gis
```

Альтернативный режим с другим хостом:

```
HOSTNAME=localhost # set it to the actual ip address or host name
psql -U postgres -h $HOSTNAME -d gis -f indexes.sql
```

- Создайте пользователя PostgreSQL **« tilerserver »** (если он еще не существует) и предоставьте права на все таблицы базы данных gis для пользователя **« tilerserver »** и для всех зарегистрированных пользователей:

```
psql -d gis <<\eof
REVOKE CONNECT ON DATABASE gis FROM PUBLIC;
GRANT CONNECT ON DATABASE gis TO "www-data";
```

```
GRANT CONNECT ON DATABASE gis TO "tileserver";
eof
```

Чтобы вывести список всех таблиц, доступных в базе данных *ГИС* , выполните следующую команду:

```
psql -d gis -c "\dt+"
```

или:

```
psql -U postgres -h $HOSTNAME -d gis -c "\dt+"
```

База данных должна включать таблицы *rels , ways и nodes* (созданные в *–slim* режиме *osm2pgsql*) для обеспечения возможности обновлений.

В следующем примере вывода использовался *–slim* режим *osm2pgsql* :

Schema	Name	Type	Owner
public	planet_osm_line	table	postgres
public	planet_osm_nodes	table	postgres
public	planet_osm_point	table	postgres
public	planet_osm_polygon	table	postgres
public	planet_osm_rels	table	postgres
public	planet_osm_roads	table	postgres
public	planet_osm_ways	table	postgres
public	spatial_ref_sys	table	postgres

Фактически, таблицы *planet_osm_rels , planet_osm_ways , planet_osm_nodes* доступны, как описано в разделе Структура базы данных Pgsql .

Более подробную информацию можно найти в модели данных OpenStreetMap на сайте Mapbox.

Для получения дополнительной информации ознакомьтесь с пользовательскими индексами .

Настроить рендер

Далее нам нужно подключить *renderd* и *mod_tile* к веб-серверу Apache, чтобы он был готов принимать запросы на тайлы.

Получите каталог плагинов Mapnik:

```
mapnik-config --input-plugins
```

Это должен быть */usr/local/lib/mapnik/input* , или */usr/lib/mapnik/3.0/input* , или другой.

Отредактируйте файл конфигурации renderd с помощью предпочитаемого вами редактора:

```
sudo vi /usr/local/etc/renderd.conf
```

Примечание: при установке *mod_tile* из пакета путь будет */etc/renderd.conf* .

```
sudo vi /etc/renderd.conf
```

В *[mapnik]* разделе измените значение параметра plugins_dir так, чтобы оно соответствовало значению, возвращаемому функцией mapnik-config –input-plugins:

Пример (при установке Mapnik 3.0 из пакета):

```
plugins_dir=/usr/lib/mapnik/3.0/input/
```

С Mapnik 2.2 из пакета:

```
plugins_dir=/usr/lib/mapnik/2.2/input/
```

С Mapnik 3.0 из источников:

```
plugins_dir=/usr/local/lib/mapnik/input/
```

В *[mapnik]* разделе также измените значение следующих параметров:

```
font_dir=/usr/share/fonts
font_dir_recurse=true
```

В *[default]* разделе измените значение XML и HOST на следующее.

```
XML=/home/tileserver/src/openstreetmap-carto/style.xml
HOST=localhost
```

Обратите внимание, что URI должен быть установлен на **/osm_tiles/**.

Также замените все ;** на ;xxx=** (например, на vi :1,$s/^;** /;xxx=** /g).

В приведенном выше примере мы предполагаем, что ваш домашний каталог — **/home/tileserver** . Измените его на ваш фактический домашний каталог.

Пример файла:

```
[renderd]
stats_file=/var/run/renderd/renderd.stats
;socketname=/var/run/renderd/renderd.sock
num_threads=4
tile_dir=/var/lib/mod_tile

[mapnik]
plugins_dir=/usr/lib/mapnik/3.0/input/
font_dir=/usr/share/fonts
font_dir_recurse=true
```

```
[default]
URI=/osm_tiles/
TILEDIR=/var/lib/mod_tile
XML=/home/tileserver/src/openstreetmap-carto/style.xml
HOST=localhost
TILESIZE=256
```

Сохраните файл.

Проверьте наличие каталога **/var/run/renderd**, в противном случае создайте его с помощью *sudo mkdir /var/run/renderd/renderd.sock.*

Проверьте это, чтобы убедиться:

```
ls -l /home/tileserver/src/openstreetmap-carto/style.xml
grep '^;xxx=\*\*' /usr/local/etc/renderd.conf
```

В случае ошибки проверьте имя пользователя и повторите попытку **/usr/local/etc/renderd.conf**.

Установите скрипт инициализации renderd , скопировав пример скрипта инициализации, включенный в его пакет.

```
sudo cp ~/src/mod_tile/debian/renderd.init /etc/init.d/renderd
```

Примечание: при установке mod_tile из пакета указанная выше команда не нужна.

Предоставьте разрешение на выполнение.

```
sudo chmod a+x /etc/init.d/renderd
```

Примечание: при установке *mod_tile* из пакета указанная выше команда не нужна.

Отредактируйте файл сценария инициализации

```
sudo vi /etc/init.d/renderd
```

Измените следующие переменные:

```
DAEMON=/usr/local/bin/$NAME
DAEMON_ARGS="-c /usr/local/etc/renderd.conf"
RUNASUSER=tileserver
```

Важное примечание: при установке *mod_tile* из пакета сохраните *DAEMON=/usr/bin/$NAMEи DAEMON_ARGS=«-c /etc/renderd.conf»*.

В *RUNASUSER=tileserver* мы предполагаем, что ваш пользователь — *tileserver* . Измените его на ваше фактическое имя пользователя.

Сохраните файл.

Создайте следующий файл и установите его владельцем (вашего фактического пользователя).

```
sudo mkdir -p /var/lib/mod_tile
sudo chown tileserver:tileserver /var/lib/mod_tile
```

Примечание: при установке *mod_tile* из пакета указанные выше команды не нужны.

Снова измените его на свое фактическое имя пользователя.

Затем запустите службу renderd.

```
sudo systemctl daemon-reload
sudo systemctl start renderd
sudo systemctl enable renderd
```

При использовании WSL renderd *необходимо* запустить с помощью следующей команды:

```
sudo service renderd start
```

Следующий вывод является обычным:

```
renderd.service is not a native service, redirecting to systemd-sysv-install
Executing /lib/systemd/systemd-sysv-install enable renderd
```

Если *systemctl* не установлен (например, Ubuntu 14.4), используйте следующие команды:

```
sudo update-rc.d renderd defaults
sudo service renderd start
```

Настроить Apache

Создайте файл загрузки модуля.

```
sudo vi /etc/apache2/mods-available/mod_tile.load
```

Вставьте следующую строку в файл.

```
LoadModule tile_module /usr/lib/apache2/modules/mod_tile.so
```

Сохраните его. Создайте символическую ссылку.

```
sudo ln -s /etc/apache2/mods-available/mod_tile.load /etc/apache2/mods-
enabled/
```

Затем отредактируйте файл виртуального хоста по умолчанию.

```
test -f /etc/apache2/sites-enabled/000-default.conf || sudo ln -s
/etc/apache2/sites-available/000-default.conf /etc/apache2/sites-enabled
sudo vi /etc/apache2/sites-enabled/000-default.conf
```

После строки ниже пропишите следующие строки *<VirtualHost *:80>*

```
# Load all the tilesets defined in the configuration file into this virtual
host
LoadTileConfigFile /usr/local/etc/renderd.conf
# Socket where we connect to the rendering daemon
ModTileRenderdSocketName /var/run/renderd/renderd.sock
# Timeout before giving up for a tile to be rendered
ModTileRequestTimeout 3
# Timeout before giving up for a tile to be rendered that is otherwise
missing
ModTileMissingRequestTimeout 60
```

Примечание: при установке mod_tile из пакета установите *LoadTileConfigFile /etc/renderd.conf*.

```
LoadTileConfigFile /etc/renderd.conf
```

Сохраните и закройте файл.

Пример:

```
<VirtualHost *:80>
    LoadTileConfigFile /etc/renderd.conf
    ModTileRenderdSocketName /var/run/renderd/renderd.sock
    ModTileRequestTimeout 3
    ModTileMissingRequestTimeout 60
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Перезапустите Apache.

```
sudo systemctl restart apache2
```

Если systemctl не установлен (например, Ubuntu 14.4):

```
sudo service apache2 restart
```

С помощью WSL перезапустите службу Apache с помощью следующих команд:

```
sudo service apache2 stop; sudo service apache2 start
<code>
```

Проверьте локальный доступ к плиткам:
<code>
```
wget --spider http://localhost/osm_tiles/0/0/0.png
```

У вас должно получиться *Remote file exists.*, если все настроено правильно.

Затем в адресной строке веб-браузера введите

```
your-server-ip/osm_tiles/0/0/0.png
```

где вам необходимо изменить *your-server-ip* на фактический IP-адрес установленного картографического сервера.

Чтобы расширить его, указав публичный IP-адрес вашего сервера, проверьте, например, эту команду (вставьте ее вывод в браузер):

```
echo "http://`wget http://ipinfo.io/ip -qO -`/osm_tiles/0/0/0.png"
```

Вы должны увидеть фрагмент карты мира.

Поздравляем! Вы только что успешно создали свой собственный сервер плиток OSM.

Вы можете перейти в OpenLayers , чтобы отобразить скользящую карту.

Предварительный рендеринг тайлов

Предварительная визуализация плиток обычно не нужна (или нежелательна); ее основное применение — обеспечить офлайн-просмотр вместо визуализации плиток на лету. В зависимости от размера БД процедура может занять очень много времени и соответствующих дисковых данных.

Для предварительной визуализации плиток используйте команду *render_list* . Предварительно визуализированные плитки будут кэшированы в **/var/lib/mod_tile** каталоге.

Чтобы отобразить все параметры командной строки render_list :

```
render_list --help
```

Пример использования:

```
render_list -a
```

В зависимости от размера базы данных выполнение этой команды может занять очень много времени.

Следующая команда предварительно визуализирует все плитки от уровня масштабирования 0 до уровня масштабирования 10, используя 1 поток:

```
render_list -n 1 -z 0 -Z 10 -a
```

Скрипт командной строки Perl с именем render_list_geo.pl , разработанный alx77, позволяет автоматически предварительно визуализировать плитки в определенной области с использованием географических координат. Соответствующий файл README Github описывает использование и примеры.

Чтобы установить его:

```
cd ~/src
git clone https://github.com/alx77/render_list_geo.pl
cd render_list_geo.pl
```

Пример команды для генерации плиток z11 для Великобритании:

```
./render_list_geo.pl -n 1 -z 11 -Z 11 -x -9.5 -X 2.72 -y 49.39 -Y 61.26
```

Для *render_list и render_list_geo.pl* опция **-m** позволяет выбирать определенные профили, относящиеся к именованным разделам в *renderd.conf* . Если эта опция не используется, выбирается **[default]** раздел *renderd.conf* .

Устранение неполадок Apache, mod_tile и renderd

Для мониторинга сервера плиток, показывающего линию каждый раз при запросе плитки и одну линию каждый раз после завершения соответствующего рендеринга:

```
tail -f /var/log/syslog | grep " TILE "
```

Чтобы очистить кэш всех плиток osm, удалите /var/lib/mod_tile/default (используйте rm -rf, если осмелитесь) и перезапустите демон renderd:

```
sudo rm -rf /var/lib/mod_tile/default
sudo systemctl restart renderd
```

Не забудьте также очистить кэш браузера.

Если systemctl не установлен (например, Ubuntu 14.4):

```
sudo service renderd restart
```

Показать загруженные модули Apache:

```
apache2ctl -M
```

Вы должны найтитile_module (shared)

Показать конфигурацию Apache:

```
apache2ctl -S
```

В журнале должны появиться следующие сообщения:

```
Loading tile config default at /osm_tiles/ for zooms 0 - 20 from tile
directory /var/lib/mod_tile with extension .png and mime type image/png
```

Хвостовой журнал:

```
tail -f /var/log/apache2/error.log
```

Большинство проблем с конфигурацией можно обнаружить, проанализировав журнал отладки renderd ; нам нужно остановить демон и запустить renderd в фоновом режиме:

```
sudo systemctl stop renderd
```

Если systemctl не установлен (например, Ubuntu 14.4):

```
sudo service renderd stop
```

Затем управляйте выводом renderd :

```
sudo -u tileserver /usr/local/bin/renderd -fc /usr/local/etc/renderd.conf
<code>
Игнорируйте пять ошибок, связанных со //iniparser: syntax error in
/usr/local/etc/renderd.conf// с сылками на закомментированные переменные
(например, начинающиеся с ;).

Нажмите **Control-C**, чтобы завершить программу. После исправления ошибки демон
можно перезапустить с помощью:
<code>
sudo systemctl start renderd
```

Если *systemctl* не установлен (например, Ubuntu 14.4):

sudo service renderd start Проверить наличие */var/run/renderd*:

```
ls -ld /var/run/renderd
```

Проверьте, что разрешение на доступ есть *-rw-r–r– 1 tileserver tileserver*. Вы можете временно сделать

```
sudo chmod 777 /var/run/renderd
```

Проверьте наличие файла *style.xml* :

```
ls -l /home/tileserver/src/openstreetmap-carto/style.xml
```

Если он отсутствует, смотрите выше, как его создать.

Проверить наличие */var/run/renderd/renderd.sock:*

ls -ld /var/run/renderd/renderd.sock Убедитесь, что разрешения на доступ есть **srwxrwxrwx 1 tileserver tileserver.**

В случае неправильного владельца:

```
sudo chown 'tileserver' /var/run/renderd
sudo chown 'tileserver' /var/run/renderd/renderd.sock
sudo service renderd restart
```

Если каталог отсутствует:

```
sudo mkdir /var/run/renderd
sudo chown 'tileserver' /var/run/renderd
sudo service renderd restart
```

В случае, если renderd завершается с ошибкой сегментации (например, **Loading parameterization function forand then Segmentation fault**), это может быть связано с несоответствием конфигурации между плагинами Mapnik и конфигурацией renderd; проверьте параметр *plugins_dir* в */usr/local/etc/renderd.conf* .

Ошибка разрешения PostGIS означает, что таблицы БД не были предоставлены пользователю tileserver:

…An error occurred while loading the map layer 'default': Postgis Plugin: ERROR: permission denied for relation…

Чтобы исправить ошибку разрешения, выполните:

```
sudo ./install-postgis-osm-user.sh gis tileserver
```

Ошибка, связанная с отсутствием tagsстолбца в журналах *renderd* , означает, что *osm2pgsql* не был запущен с этой **–hstore** опцией.

Если все в конфигурации выглядит нормально, но карта по-прежнему не отображается без какого-либо конкретного сообщения от renderd , попробуйте выполнить перезапуск системы:

```
sudo shutdown -r now
```

Если проблема сохраняется, возможно, проблема с пользователем UNIX. Попробуйте отладку снова, установив эти переменные:

```
export PGHOST=localhost
export PGPORT=5432
export PGUSER=postgres
export PGPASSWORD=postgres_007%
```

В исключительном случае следующие команды позволяют полностью удалить Apache, mod_tile и renderd и переустановить службу:

```
sudo rm -r ~/src/mod_tile/
sudo apt-get purge apache2 apache2-dev
sudo rm -r /etc/apache2/mods-available
sudo rm /usr/local/etc/renderd.conf
sudo rm  /etc/init.d/renderd
sudo rm -rf /var/lib/mod_tile
sudo rm -rf /usr/lib/apache2
sudo rm -rf /etc/apache2/
sudo rm -rf /var/run/renderd
sudo apt-get --reinstall install apache2-bin
sudo apt-get install apache2 apache2-dev
```

Формат имен тайлов сервера тайлов OpenStreetMap

Наименование файлов и формат изображений, используемые mod_tile, описаны в Slippy map tilenames . Похожий формат также используется Google Maps и многими другими поставщиками карт.

TMS и WMS — это другие протоколы для обслуживания карт в виде фрагментов, управляемые различными бэкэндами рендеринга.

Развертывание собственной карты Slippy Map

В терминологии OpenStreetMap мозаичная веб-карта также известна как скользящая карта.

OpenStreetMap не предоставляет «официальную» библиотеку JavaScript, которую вы обязаны использовать. Вместо этого вы можете использовать любую библиотеку, которая соответствует вашим потребностям. Две самые популярные — OpenLayers и Leaflet. Обе с открытым исходным кодом.

Страница Развертывание собственной карты Slippy Map иллюстрирует, как встроить ранее установленный сервер карт в веб-сайт. Упоминается ряд возможных библиотек карт, включая некоторые релевантные (Leaflet , OpenLayers , Google Maps APIGoogle Maps API), а также множество альтернатив.

OpenLayers

Чтобы отобразить скользящую карту с помощью OpenLayers, создайте файл с именем *ol.html* в каталоге */var/www/html*.

```
sudo vi /var/www/html/ol.html
```

Вставьте следующий HTML-код в файл.

Вы можете настроить долготу, широту и уровень масштабирования в соответствии с вашими потребностями. Проверьте **var zoom = 2, center = [0, 0];**.

Обратите внимание, что мы используем https для openstreetmap.org .

```
<!DOCTYPE html>
<html>
<head>
<title>OpenStreetMap with OpenLayers</title>
<link rel="stylesheet" href="https://openlayers.org/en/v4.6.5/css/ol.css"
type="text/css">
<script src="https://openlayers.org/en/v4.6.5/build/ol.js"></script>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.4/jquery.min.js"></scr
ipt>
  <style>
  html,
  body,
  #map {
    height: 100%;
    margin: 0;
    padding: 0;
  }
  .ol-custom-overviewmap,
  .ol-custom-overviewmap.ol-uncollapsible {
    bottom: auto;
    left: auto;
    right: 0;
    top: 85px;
  }
  .ol-zoom {
    top: 3em;
  }
  .ol-zoom-extent {
      top: 20.6em!important;
  }
  .ol-zoomslider {
      top: 7.7em!important;
  }
  .ol-custom-fullscreen {
    bottom: auto;
    left: auto;
    right: 0;
    top: 50px;
  }
  .ol-custom-mouse-positionXY {
    top: auto;
    bottom: 4em;
    font-family: "Arial";
    font-size: 12px;
    text-shadow: 0 0 0.5em #FFE, 0 0 0.5em #FFE, 0 0 0.5em #FFE;
  }
```

```
  .ol-custom-mouse-positionHDMS {
    top: auto;
    bottom: 5em;
    font-family: "Arial";
    font-size: 12px;
    text-shadow: 0 0 0.5em #FFE, 0 0 0.5em #FFE, 0 0 0.5em #FFE;
  }
  .ol-custom-mouse-position3857 {
    top: auto;
    bottom: 6em;
    font-family: "Arial";
    font-size: 12px;
    text-shadow: 0 0 0.5em #FFE, 0 0 0.5em #FFE, 0 0 0.5em #FFE;
  }
  #ZoomElement {
    position: absolute;
    top: auto;
    left: 10px;
    bottom: 2.5em;
    text-decoration: none;
    font-family: "Arial";
    font-size: 10pt;
    text-shadow: 0 0 0.5em #FFE, 0 0 0.5em #FFE, 0 0 0.5em #FFE;
    z-index: 30;
  }
  #TSLabel {
    position: absolute;
    top: 21px;
    right: 0;
    font-family: "Arial";
    font-size: 12px;
    z-index: 30;
  }
  #osmLabel {
    position: absolute;
    top: 21px;
    left: 0;
    font-family: "Arial";
    font-size: 12px;
    z-index: 30;
  }
  #swipe {
    position: absolute;
    top: 0;
    left: -4px;
    z-index: 20;
  }
</style>
</head>
<body>
  <div class="ol-viewport">
```

```
  <input class="ol-unselectable ol-control" id="swipe" type="range"
style="width: 100%">
  <div class="ol-unselectable ol-control" id="TSLabel"> Tile Server
&#9658;</div>
  <div class="ol-unselectable ol-control" id="osmLabel">&#9668;
OpenStreetMap </div>
  <a class="ol-unselectable ol-control" id="ZoomElement"></a>
  </div>
  <div tabindex="0" id="map" class="map"></div>
  <script>
    var zoom = 2, center = [0, 0];

    // Set up the Tile Server layer
    var myTileServer = new ol.layer.Tile({
      preload: Infinity,
      source: new ol.source.OSM({
        crossOrigin: null,
        url: 'osm_tiles/{z}/{x}/{y}.png'
      })
    });

    // Set up the OSM layer
    var openStreetMap = new ol.layer.Tile({
      preload: Infinity,
      source: new ol.source.OSM({
        crossOrigin: null,
        url: 'https://{a-c}.tile.openstreetmap.org/{z}/{x}/{y}.png'
      })
    });

    if (window.location.hash !== '') {
      var hash = window.location.hash.replace('#', '');
      var parts = hash.split(';');
      if (parts.length === 3) {
        zoom = parseInt(parts[0], 10);
        center = [
          parseFloat(parts[2]),
          parseFloat(parts[1])
          ];
      }
    }

    // Set up the default view
    var myTileView = new ol.View({
      center: ol.proj.transform(center, 'EPSG:4326', 'EPSG:3857'),
      zoom: zoom
    });

    // Create the map
```

```
    var map = new ol.Map({
      layers: [myTileServer, openStreetMap],
      loadTilesWhileInteracting: true,
      target: 'map',
      controls: ol.control.defaults().extend([
        new ol.control.ScaleLine(),
        new ol.control.Zoom(),
        new ol.control.ZoomSlider(),
        new ol.control.ZoomToExtent(),
        new ol.control.FullScreen({
          className: 'ol-fullscreen ol-custom-fullscreen'
        }),
        new ol.control.OverviewMap({
          className: 'ol-overviewmap ol-custom-overviewmap'
        }),
        new ol.control.MousePosition({
          className: 'ol-mouse-position ol-custom-mouse-position3857',
          coordinateFormat: ol.coordinate.createStringXY(4),
          projection: 'EPSG:3857',
          undefinedHTML: ' '
        }),
        new ol.control.MousePosition({
          coordinateFormat: function(coord) {
            return ol.coordinate.toStringHDMS(coord);
          },
          projection: 'EPSG:4326',
          className: 'ol-mouse-position ol-custom-mouse-positionHDMS',
          target: document.getElementById('mouse-position'),
          undefinedHTML: ' '
        }),
        new ol.control.MousePosition({
          className: 'ol-mouse-position ol-custom-mouse-positionXY',
          coordinateFormat: ol.coordinate.createStringXY(4),
          projection: 'EPSG:4326',
          undefinedHTML: ' '
        }),
      ]),
      view: myTileView
    });
    map.on("moveend", function() {
      var view = map.getView();
      var center = ol.proj.transform(view.getCenter(), 'EPSG:3857',
'EPSG:4326');
      var zoom = view.getZoom();
      var zoomInfo = 'Zoom level = ' + zoom;
      document.getElementById('ZoomElement').innerHTML = zoomInfo;
      window.location.hash =
        view.getZoom() + ';' +
          Math.round(center[1]*1000000)/1000000 + ';' +
          Math.round(center[0]*1000000)/1000000;
    });
```

```
    var swipe = document.getElementById('swipe');

    openStreetMap.on('precompose', function(event) {
        var ctx = event.context;
        var width = ctx.canvas.width * (swipe.value / 100);

        ctx.save();
        ctx.beginPath();
        ctx.rect(width, 0, ctx.canvas.width - width, ctx.canvas.height);
        ctx.clip();
      });

    openStreetMap.on('postcompose', function(event) {
        var ctx = event.context;
        ctx.restore();
      });

    swipe.addEventListener('input', function() {
        map.render();
    }, false);
  </script>
</body>
</html>
```

Сохраните и закройте файл. Теперь вы можете просмотреть свою скользящую карту, введя следующий URL в браузере.

```
http://your-server-ip/ol.html
```

Чтобы расширить его, указав публичный IP-адрес вашего сервера, проверьте, например, эту команду (вставьте ее вывод в браузер):

```
echo "http://`wget http://ipinfo.io/ip -qO -`/ol.html"
```

Листовка

Leaflet — это библиотека JavaScript для встраивания карт. Она проще и меньше, чем OpenLayers.

Самый простой пример отображения вашей скользящей карты с помощью Leaflet — создание файла с именем lf.html в каталоге /var/www/html .

```
sudo vi /var/www/html/lf.html
```

Вставьте следующий HTML-код в файл. Замените your-server-ip на ваш IP-адрес и настройте долготу, широту и уровень масштабирования в соответствии с вашими потребностями.

```
<!DOCTYPE html>
<html>
<head>
<title>OpenStreetMap with Leaflet</title>
<link rel="stylesheet"
href="http://cdn.leafletjs.com/leaflet-0.6.4/leaflet.css" type="text/css">
<script src="http://cdn.leafletjs.com/leaflet-0.6.4/leaflet.js"></script>
<style>
  html,
  body,
  #map {
    height: 100%;
    margin: 0;
    padding: 0;
  }
</style>
</head>
<body>
  <div id="map" class="map"></div>
  <script>
    // Create the map
    var map = L.map('map').setView([45, 10], 3);

    // Set up the OSM layer
    L.tileLayer(
    'http://your-server-ip/osm_tiles/{z}/{x}/{y}.png'
    ).addTo(map);
  </script>
</body>
</html>
```

Сохраните и закройте файл. Теперь вы можете просмотреть свою скользящую карту, введя следующий URL в браузере.

```
http://your-server-ip/lf.html
```

Быстрый способ протестировать скользящую карту — воспользоваться онлайн-площадкой для изучения исходного кода, например, этим шаблоном JSFiddle.

В следующем примере Leaflet используется для отображения данных OpenStreetMap.

Плитки по умолчанию можно заменить на плитки вашего сервера, изменив

```
https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png
```

к http://your-server-ip/osm_tiles/{z}/{x}/{y}.png.

Чтобы отредактировать образец, нажмите Edit в JSFiddle . Затем на панели Javascript измените строку внутри кавычек, как описано выше. Нажмите Run .

Дополнения и файлы

- Установка сервера тайлов OpenStreetMap в Ubuntu
- Создание тайлового сервера вручную (Ubuntu 24.04)
- Создание тайлового сервера вручную (Ubuntu 20.04)
- PostgreSQL
- Поднимаем собственный репозиторий пакетов для Ubuntu (Debian)
- Как установить все доступные обновления для выпуска Ubuntu перед обновлением
- mod_tile

Упомянутые команды:

- sudo apt update — обновляет список пакетов последней информацией из настроенных репозиториев.
- sudo apt update — устанавливает доступные обновления для вашей текущей версии Ubuntu.
- sudo apt dist-upgrade — выполняет более тщательное обновление, которое более тщательно обрабатывает зависимости пакетов.
- sudo apt autoremove — удаляет из системы неиспользуемые или устаревшие пакеты.
- sudo restart — перезагружает систему, чтобы все изменения вступили в силу.
- sudo apt install update-manager-core — устанавливает основной пакет менеджера обновлений, если он еще не установлен.
- sudo do-release-upgrade — запускает процесс обновления выпуска Ubuntu.