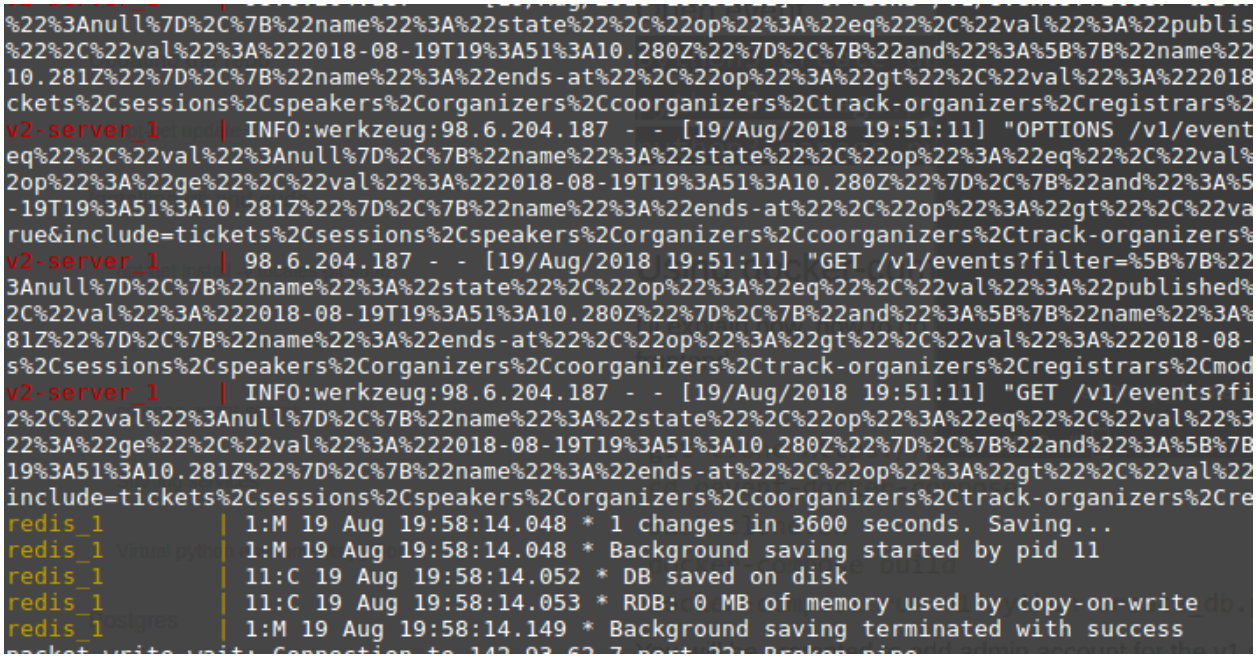

Installing Open Event Server

I'm going to walk you through the process of installing Open Event Server and possible issues you might encounter, this will probably be helpful if you're stuck with it, but most importantly in building some sort of installer as I've stated numerous times before.

Requirements for its installation

Debian based distro or unix-like with aptitude package manager

Enough memory and storage, (depends on whether you put the database on your system or not)

Knowledge of UNIX-like systems

First, here are the commands, I tried to make them as easy as a copy-paste but it probably won't work depending on it, so I'll talk to you about what each thing does

```
apt-get update
apt-get install -y locales git sudo
locale-gen en_US.UTF-8
if [ -z "$LANG" ];then export LANG=en_US.UTF-8;fi

export DEBIAN_FRONTEND=noninteractive DEBCONF_NONINTERACTIVE_SEEN=true
git clone https://github.com/kreijstal-contributions/open-event-api.git --depth=1
```

```
cd open-event-server
#libffi6 libffi-dev
apt-get install -y redis-server tmux  libssl-dev vim postgresql postgresql-contrib
build-essential python3-dev libpq-dev libevent-dev libmagic-dev python3-pip wget
ca-certificates python3-venv curl && update-ca-certificates && apt-get clean -y

service redis-server start
service  postgresql start
cp .env.example .env
#sed -i -e 's/SERVER_NAME/#SERVER_NAME/g' .env
#The above was used because before the repository came with an annoying SERVER_NAME
which only listened to localhost, so if you accessed with 127.0.0.1, it didn't work
#pip install virtualenv
#python 2.7.14
#virtualenv .

python3 -m venv .
#pip install pip==9.0.1
source bin/activate
#pip3 install -r requirements/tests.txt
pip3 install --no-cache-dir -r requirements.txt
#pip install eventlet
cat << EOF | su - postgres -c psql
-- Create the database user:
CREATE USER john WITH PASSWORD 'start';

-- Create the database:
CREATE DATABASE oevent OWNER=john
                                  LC_COLLATE='en_US.utf8'
                                  LC_CTYPE='en_US.utf8'
                                  ENCODING='UTF8'

TEMPLATE=template0;
EOF
python3 create_db.py admin@admin.com admin
python3 manage.py db stamp head
python3 manage.py runserver
```

Ok, let's start with the beginning, we assume here you know how to open a linux terminal, and that you're on linux.

Description of commands

Updating the repo

```
apt-get update
```

Ok, so let's start with some assumptions, you're either on ubuntu, or debian, apt-get is on every debian-based distro, so you can do this on linux-mint as well, but if you're using other distro you might have to consult with your package manager guide where to get the packets that are needed.

Unfortunately the open-event-server has dependencies and in order to get them we've used the package manager of ubuntu, that means some packages might not be possible to be found on every distro, and there is currently not support for that.

Installing git and sudo

```
apt-get install -y locales git sudo
```

Ok, let's also assume you're on a barebones ubuntu installation, as many people are so let's work on some very basic requirements that you probably already fulfil, git is not installed by default, so here we install it, we also install sudo, and the -y flag is to avoid being asked for confirmation.

Setting the locales

```
locale-gen en_US.UTF-8
```

```
if [ -z "$LANG" ];then export LANG=en_US.UTF-8;fi
```

Okay so basically some very bare bones linux distributions don't have the UTF-8 locale enabled, here we generate it in case it hasn't, and if the $LANG variable doesn't exist, then we add the en_US locale.

Really ensuring it works everywhere, especially because once you start installing packages you might get prompted or errors might come up

Cloning the repo

Explanation

```
git clone https://github.com/fossasia/open-event-server.git
--depth=1
```

Here we clone the repo with git clone from github.

However we use the --depth=1 flag, that means we only and just only get the latest version, git was designed as a version control tool, not as a source host, so when you clone a project with the purpose of installing it, you're downloading the whole revision history, and some revision histories can get pretty, pretty large, slowing the download time.

If you're a contributor and you want to revert some git commits, or you want to debug and see what causes something, then feel free to download everything, but if you're downloading the git with the purpose to use the software, you really don't need to download the revision history.

```
cd open-event-server
```

A bit of trivia for those who didn't know, cd means "Change Directory", here we're changing our current directory to open-event-server

```
#libffi6 libffi-dev
apt-get install -y redis-server tmux  libssl-dev vim
postgresql postgresql-contrib build-essential python3-dev
libpq-dev libevent-dev libmagic-dev python3-pip wget
ca-certificates python3-venv curl && update-ca-certificates
&& apt-get clean -y
```

Okay, here we use apt-get to install redi-server, which is a dependency for open-event, and postgresql, we also download python3 plus some important packets like virtualenv for python3.

Starting services

In this specific case we're installing these servers on our server but it doesn't necessarily have to be that way, if you have an external postgres or redis server you can use that as well.

```
service redis-server start
```

We start the redis-server

```
service  postgresql start
```

We start the postgres service

```
cp .env.example .env
```

Virtual python enviroments

`python3 -m venv .`

We initialize the virtualenv on the current directory, that is .

`source bin/activate`

We activate the virtualenv generated by .venv

`pip3 install --no-cache-dir -r requirements.txt`

Postgres

```
cat << EOF | su - postgres -c psql
-- Create the database user:
CREATE USER john WITH PASSWORD 'start';

-- Create the database:
CREATE DATABASE oevent OWNER=john
                                    LC_COLLATE='en_US.utf8'
                                    LC_CTYPE='en_US.utf8'
                                    ENCODING='UTF8'
TEMPLATE=template0;
EOF
```

A very clever use of command line pipes, basically psql is the command you use to interact with the database, but it is usually interactive, however on the command line you can very easily pipe typed commands using cat, this is equivalent as if we executed `su - postgres -c psql` and then typed everything until EOF, so next time you need to automate something that prompts you from input, you can use this!

Also psql won't generally accept connections from anyone buy postgres, even if you're root that's why we're using `su - postgres -c psql` which means, run psql as user postgres, you don't need to use su, if you know how to change users with sudo

Open event

`python3 create_db.py admin@admin.com admin`

`python3 manage.py db stamp head`

`python3 manage.py runserver`

Using docker-compose install both versions

I'll explain now, how to do it using docker-compose, which concurrently boots both server and frontend.

```
git clone https://github.com/Kreijstal/oevent-docker-compose.git
cd oevent-docker-compose
bash clone.sh
docker-compose build
docker-compose run v1 python create_db.py
```

You will be prompted to add admin account for the v1 server. After you do that just do

`docker-compose up`

(v2 log in settings you can change it in docker-compose.yml)
And that's it! Way faster than the other way, requires even less code, and almost always succeed), the catch is that you have to have docker installed in your machine
Sources

Tags: installer, open-event, explanation, guide, tutorial, information, docker-compose, docker