**Linux Server**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#linux-server)

We typically recommend newcomers to oTree to deploy to Heroku (see instructions [here](https://otree.readthedocs.io/en/latest/server/heroku.html#heroku)), or to use their own personal computer as a temporary server (see [here](https://otree.readthedocs.io/en/latest/server/adhoc.html#server-adhoc)).

However, you may prefer to run oTree on a proper Linux server. Reasons may include:

* Your lab doesn’t have internet
* You want full control over server configuration
* You want better performance (local servers have less latency)

If you are experienced in Django server setup, you just need to know that setting up an oTree server is the same as any Django project, except:

* You need Redis
* You start the server with otree runprodserver, rather than a WSGI server.

For those who want full details on the setup, read the below sections, which are for Ubuntu 16.04.

用Ubuntu 20.04测试可以完成下列测试，不建议用更早的版本

在开始之前可以设定根用户

设定根用户root密码：sudo passwd root

切换根用户：su root

退出根用户账户：exit

**Install apt-get packages**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#install-apt-get-packages)

Run:

sudo apt-get install python3-pip python3-dev libpq-dev postgresql postgresql-contrib redis-server git

**Create a virtualenv**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#create-a-virtualenv)

创建虚拟环境可能不成功，在这之前可能需要再装一个东西（注意提示）：

sudo apt-get install python3-venv

It’s a best practice to use a virtualenv:

python3 -m venv venv\_otree

To activate this venv every time you start your shell, edit your .bashrc file using nano:

nano ~/.bashrc

And add this line to the end:

source ~/venv\_otree/bin/activate

To save and exit, press Ctrl+O, Enter, and Ctrl+X.

Close and reopen your terminal window. You should see (venv\_otree) at the beginning of your prompt. If it’s not there, try adding the above lines to ~/.bash\_profile instead of ~/.bashrc, because some systems use a different filename.

**Database (Postgres)**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#database-postgres)

oTree’s default database is SQLite, which is fine for local development, but insufficient for production. We recommend you use PostgreSQL.

Change users to the postgres user, so that you can execute some commands:

sudo su - postgres

Then start the Postgres shell:

psql

Once you’re in the shell, create a database and user:

CREATE DATABASE django\_db;

CREATE USER otree\_user WITH PASSWORD 'mydbpassword';

GRANT ALL PRIVILEGES ON DATABASE django\_db TO otree\_user;

Exit the SQL prompt:

\q

Exit out of the postgres user and return to your regular command prompt:

exit

Now you should tell oTree to use Postgres instead of SQLite. Set the DATABASE\_URL environment variable on your server. This allows you to continue to use SQLite on your development machine, while using Postgres on your production server.

If you used the values in the example above (username otree\_user, password mydbpassword and database django\_db),

Run:

nano ~/.bashrc

(Or, nano ~/.bash\_profile, whichever file you edited in the previous step.)

Then add this line to the end of the file:

export DATABASE\_URL=postgres://otree\_user:mydbpassword**@localhost**/django\_db

To save and exit, press Ctrl+O, Enter, and Ctrl+X. Then close and reopen your terminal and confirm with echo $DATABASE\_URL that it was set properly.

Once DATABASE\_URL is defined, oTree will use it instead of the default SQLite. (This is done via [dj\_database\_url](https://pypi.python.org/pypi/dj-database-url).)

When you run otree resetdb later, if you get an error that says “password authentication failed for user”, find your hba\_auth.conf file, and on the lines for IPv4 and IPv6, change the METHOD from md5 (or whatever it currently is) to trust.

**Install Redis**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#install-redis)

If you installed redis-server through apt-get as instructed earlier, Redis should be running on port 6379. You can test with redis-cli ping, which should output PONG.

If there was an installation problem, you can try installing Redis from an alternate source, e.g. [here](https://launchpad.net/~chris-lea/+archive/ubuntu/redis-server).

**Push your code to the server**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#push-your-code-to-the-server)

You can get your code on the server using SCP, SFTP, Dropbox, etc. If you are interested in using Git (which is somewhat more advanced), see the instructions [here](https://otree.readthedocs.io/en/latest/server/git.html#git-generic).

For this tutorial, we will assume you are storing your files under /home/my\_username/oTree.

**Reset the database on the server**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#reset-the-database-on-the-server)

On the server, cd to the folder containing your oTree project. Install the requirements and reset the database:

pip3 install -r requirements.txt

otree resetdb

**Running the server**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#running-the-server)

If you are just testing your app locally, you can use the usual devserver command.

However, when you want to use oTree in production, you need to run the production server, which can handle more traffic.

Note: oTree does not run with typical Django WSGI servers like gunicorn, because it is ASGI based.

**Testing the production server**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#testing-the-production-server)

From your project folder, run:

otree runprodserver 8000

Then navigate in your browser to your server’s IP/hostname followed by :8000.

If you’re not using a reverse proxy like Nginx or Apache, you probably want to run oTree directly on port 80. This requires superuser permission, so let’s use sudo, but add some extra args to preserve environment variables like PATH, DATABASE\_URL, etc:

sudo -E env "PATH=$PATH" otree runprodserver 80

Try again to open your browser; this time, you don’t need to append :80 to the URL, because that is the default HTTP port.

Notes:

* unlike devserver, runprodserver does not restart automatically when your files are changed.
* runprodserver automatically runs Django’s collectstatic to collect your files under \_static\_root/. If you have already run collectstatic, you can skip it with --no-collectstatic.
* 将上述命令写进一个sh文件里，用WinSCP或VSC的远程连接上传至程序文件夹内，用sh ./start\_server.sh启动

**Set remaining environment variables**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#set-remaining-environment-variables)

这些环境变量开启服务器登录验证

Run:

nano ~/.bashrc

(Or, nano ~/.bash\_profile, whichever file you edited previously.)

Then add these lines to the end of the file (substitute your own values):

export OTREE\_ADMIN\_PASSWORD=my\_password

*#no use#export OTREE\_PRODUCTION=1 # uncomment this line to enable production mode*

export OTREE\_AUTH\_LEVEL=DEMO

To save and exit, press Ctrl+O, Enter, and Ctrl+X.

修改环境变量要重新启动连接一次终端

下面的不用管

**(Optional) Process control system**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#optional-process-control-system)

Once the server is working as described above, it’s a good practice to use a process control system like Supervisord or Circus. This will restart your processes in case they crash, keep it running if you log out, etc.

**Circus**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#circus)

Install Circus, then create a circus.ini in your project folder, with the following content:

[watcher:webapp]

cmd = otree

args = runprodserver 80

use\_sockets = **True**

copy\_env = **True**

Then run:

sudo -E env "PATH=$PATH" circusd circus.ini

If this is working properly, you can start it as a daemon:

sudo -E env "PATH=$PATH" circusd --daemon circus.ini --log-output=circus-logs.txt

To stop circus, run:

circusctl stop

**Supervisor**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#supervisor)

As an alternative to Circus, you can install supervisor:

sudo apt-get install supervisor

If you install supervisor through apt-get, it will be installed as a service, and will therefore automatically start when your server boots. (You can also install supervisor with pip, but unlike oTree it’s only compatible with Python 2, so you should install it into your system’s Python 2 installation, rather than your Python 3 virtualenv.)

In the supervisor config dir /etc/supervisor/conf.d/, create a file otree.conf with the following content:

[program:otree]

command=/home/my\_username/venv\_otree/bin/otree runprodserver 80

directory=/home/my\_username/oTree

stdout\_logfile=/home/my\_username/otree-supervisor.log

stderr\_logfile=/home/my\_username/otree-supervisor-errors.log

autostart=true

autorestart=true

environment=

PATH="/home/my\_username/venv\_otree/bin/:*%(ENV\_PATH)s*",

DATABASE\_URL="postgres://otree\_user:otree@localhost/django\_db",

OTREE\_ADMIN\_PASSWORD="my\_password", *# password for oTree web admin*

OTREE\_PRODUCTION="", *# can set to 1*

OTREE\_AUTH\_LEVEL="", *# can set to STUDY or DEMO*

Set directory to the dir containing your project (i.e. with settings.py).

DATABASE\_URL should match what you set earlier. That is, you need to set DATABASE\_URL in 2 places:

* in your .bashrc, so that otree resetdb works when you execute it as a regular user
* in your otree.conf so that otree runprodserver works when it is executed by the root user (normally supervisor runs under the root user)

To start or restart the server (e.g. after making changes), do:

sudo service supervisor restart

If this doesn’t start the server, check the stdout\_logfile you defined above, or /var/log/supervisor/supervisord.log.

**(Optional) Apache, Nginx, etc.**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#optional-apache-nginx-etc)

You cannot use Apache or Nginx as your primary web server, because oTree must be run with an ASGI server. However, you still might want to use Apache/Nginx as a reverse proxy, for the following reasons:

* You are trying to optimize serving of static files (though oTree uses Whitenoise, which is already fairly efficient)
* You need to host other websites on the same server
* You need features like SSL or proxy buffering

**Apache**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#apache)

If you want to run oTree on a subdomain of your host so that you can share port 80 with other sites hosted on the same machine, you can try the below configuration. The below example assumes oTree server is running on port 8000. For HTTPS, change 80 to 443 ws prefix to wss:

<VirtualHost \*:80>

ServerName otree.domain.com

ProxyRequests Off

ProxyPreserveHost On

ProxyPass / http://localhost:8080/

ProxyPassReverse / http://localhost:8080/

RewriteEngine On

RewriteCond %{HTTP:Connection} Upgrade [NC]

RewriteCond %{HTTP:Upgrade} websocket [NC]

RewriteRule /(.\*) ws://localhost:8000/$1 [P,L]

</VirtualHost>

**Troubleshooting**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#troubleshooting)

If you get strange behavior, such as random changes each time the page reloads, it might be caused by another oTree instance that didn’t shut down. Try stopping oTree and reload again. Also make sure that you are not sharing the same Postgres or Redis databases between two oTree instances.

**Database backups**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#database-backups)

If you just want to download the data from your study, open the admin interface in your browser and click “Data”. But if you want to back up the raw data in your Postgres database, you can use the below command:

pg\_dump -U otree\_user -h localhost django\_db > otree-$(date +"%Y-%m-%d-%H-%M").sql

(This assumes your database is set up as described above (with username otree\_user and database name django\_db, and that you are on Unix.)

If you need to restore your database to a particular backup, do like this:

psql django\_db < otree-2017-03-22-01-01.sql

**Sharing a server with other oTree users**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#sharing-a-server-with-other-otree-users)

You can share a server with other oTree users; you just have to make sure that the code and databases are kept separate, so they don’t conflict with each other.

On the server you should create a different Unix user for each person using oTree. Then each person should follow the same steps described above, but in some cases name things differently to avoid clashes:

* Create a virtualenv in their home directory (can also be named venv\_otree)
* Create a different Postgres database (e.g. postgres://otree\_user2:mydbpassword@localhost/django\_db), as described earlier, and set this in the DATABASE\_URL env var.
* Each user needs their own Redis database. By default, oTree uses redis://localhost:6379; but if another person uses the same server, they need to set the REDIS\_URL env var explicitly, to avoid clashes. You can set it to redis://localhost:6379/1, redis://localhost:6379/2, etc. (which will use databases 1, 2, etc…instead of the default database 0). Another option is to run multiple instances of Redis on different ports.

Once these steps are done, the second user can push code to the server, then run otree resetdb.

If you don’t need multiple people to run experiments simultaneously, then each user can take turns running the server on port 80 with otree runprodserver 80. However, if multiple people need to run experiments at the same time, then you would need to run the server on multiple ports, e.g. 8000, 8001, etc.

Finally, if you use supervisor (or circus) as described above, each user should have their own conf file, with their personal parameters like virtualenv path, oTree project path, DATABASE\_URL and REDIS\_URL env vars, port number, etc.

**Next steps**[¶ℑ](https://otree.readthedocs.io/en/latest/server/ubuntu.html#next-steps)

See [Server deployment: final steps](https://otree.readthedocs.io/en/latest/server/next_steps.html#server-final-steps) for steps you should take before launching your study.