

---

# **oTree Virtual Machine Manager Documentation**

*Release 0.2.2*

**Tobias Raabe**

**Mar 19, 2019**



---

# Contents

---

<b>1</b>	<b>oTree Virtual Machine Manager</b>	<b>3</b>
1.1	Overview . . . . .	3
1.2	Features . . . . .	3
1.3	Tutorials . . . . .	4
1.4	Contributing . . . . .	4
1.5	Credits . . . . .	4
<b>2</b>	<b>Installation</b>	<b>5</b>
2.1	Preconfigured Systems . . . . .	5
2.2	Requirements . . . . .	5
2.3	Stable release . . . . .	6
2.4	From sources . . . . .	6
2.5	Recommendations for Additional Software . . . . .	7
<b>3</b>	<b>Usage</b>	<b>9</b>
3.1	General usage . . . . .	9
3.2	First run . . . . .	9
3.3	Command line options . . . . .	10
<b>4</b>	<b>End User Desktop</b>	<b>11</b>
4.1	Starting a new Project as End User . . . . .	12
4.2	Running Experiments as End User . . . . .	12
<b>5</b>	<b>Developer Interface</b>	<b>13</b>
5.1	Commands . . . . .	13
<b>6</b>	<b>Contributing</b>	<b>15</b>
6.1	Types of Contributions . . . . .	15
6.2	Get Started! . . . . .	16
6.3	Pull Request Guidelines . . . . .	17
6.4	Tips . . . . .	17
<b>7</b>	<b>Credits</b>	<b>19</b>
7.1	Development Lead . . . . .	19
7.2	Contributors . . . . .	19
<b>8</b>	<b>History</b>	<b>21</b>

8.1	0.2.2 (2017-07-21)	21
8.2	0.2.1 (2017-07-16)	21
8.3	0.2.0 (2017-07-16)	21
8.4	0.1.1 (2017-03-20)	21
8.5	0.1.0 (2017-03-20)	21
<b>9</b>	<b>Indices and tables</b>	<b>23</b>
	<b>Python Module Index</b>	<b>25</b>

Contents:



---

## oTree Virtual Machine Manager

---

oTree Virtual Machine Manager helps to manage user accounts.

- Free software: MIT license
- Documentation: <https://otree-virtual-machine-manager.readthedocs.io>.

### 1.1 Overview

**oTree Virtual Machine Manager** is a complement to the **oTree Virtual Machine Image** provided by Felix Albrecht and Holger Gerhardt.

Since doing research is time-consuming enough, this tool ensures that administrators of an **oTree** server do not waste their time on creating fully equipped user accounts and similar tedious tasks. Everything breaks down to a single commandline interface.

Managing an oTree server with multiple experiments running parallel has never been easier.

### 1.2 Features

**Create users** Creates a fully equipped experimenter account (clear project structure, virtual environment, graphical or point-and-click solutions to many oTree-related commands, samba access).

**Back up user** Creates a database and/or home folder backup for users upon account closure so that nothing gets lost.

**Remove user** Removes otree-server user accounts.

**Behind the scenes** Handles port configuration for multiple parallel user accounts on a single virtual host.

## 1.3 Tutorials

You can find a series of tutorial videos on [Youtube](#).

### For administrators:

- [Installation of ovmm](#)
- [Adding a user account with ovmm](#)
- [Restoring a database](#)

### For end users:

- [Using ovmm as an end user](#)
- [Resetting the database and backups as end user](#)

## 1.4 Contributing

[Contributions](#) are welcome and they are greatly appreciated! Every little bit helps, and credit will always be given.

## 1.5 Credits

This package was created with [Cookiecutter](#) and the [audreyr/cookiecutter-pypackage](#) project template.

As **oTree Virtual Machine Manager** provides useful commands to administrators of the **oTree Virtual Machine Image**, it requires superuser rights to perform its actions. Therefore, install `ovmm` under the provided administrator account.

## 2.1 Preconfigured Systems

We created a Virtualbox image for oTree users based on **ovmm** and on “Ubuntu 16.04.2 Server 64bit” and comes with three desktops *Gnome*, *LXDE*, and *fluxbox* to allow it to work on systems with varying level RAM.

The images all contain a user account called `otreeadmin` that comes preconfigured with the set of end user commands provided by **ovmm**. The images further each contain an account `ovmmadmin` from where **ovmm** can be used to create more accounts.

You find the images here: [oTree Virtual Machine Images](#)

We also spent some time on trying to figure out what tools could be useful to the end users. Therefore all images with the following tools preinstalled.

**Chromium** Webbrowser

**Giggle** GUI for Git,

**Atom** Great free IDE with many plugins,

**pgAdmin 3** GUI for postgresSQL database systems

## 2.2 Requirements

The following components need to be present on your system to use `ovmm`:

**ftp** required for ftp access to the account

**git** version control

**nginx** webserver  
**postgresql** database server + development packages  
**pip** package manager for Python  
**python3-venv** virtual environment for Python 3  
**redis-server** remote dictionary server  
**samba** Windows share access  
**screen** detachable consoles  
**mailutils** sending mails if otree stops unexpectedly  
**ssh** remote shell access  
**ufw** firewall  
**xterm** Linux terminal emulator  
**zenity** GTK 3.0 dialog handler

In most Debian based Linux environments you can install these packages from the repositories like so:

```
$ sudo apt install ftp nginx samba screen mailutils ssh ufw postgresql git \
                    postgresql-server-dev-all python3-pip zenity \
                    python3-venv redis-server xterm
```

Please see the documentation of your Linux distribution for help.

## 2.3 Stable release

To install oTree Virtual Machine Manager, run this command in your terminal:

```
$ sudo pip3 install ovmm
```

This is the preferred method to install oTree Virtual Machine Manager, as it will always install the most recent stable release.

If you don't have [pip](#) installed, this [Python installation guide](#) can guide you through the process.

## 2.4 From sources

The sources for oTree Virtual Machine Manager can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/tobiasraabe/otree_virtual_machine_manager
```

Or download the [tarball](#):

```
$ curl -OL https://github.com/tobiasraabe/otree_virtual_machine_manager/tarball/
↪master
```

Once you have a copy of the source, you can install it with:

```
$ sudo python3 setup.py install
```

## 2.5 Recommendations for Additional Software

To enhance the working experience for the oTree end user we recommend to also install the following software packages.

**chromium** Chromium browser is the Open Source version of Chrome browser. As it is the most frequently used browser on the web it is the recommended testing environment for your oTree apps.

**conky** Desktop system monitor. ovmm statics provides a preconfigured conky configuration file which is unzipped into the user's \$HOME directory and provides helpful system information while running oTree.

**Atom** Open Source IDE with many features.

**pgAdmin III** Graphical user interface for PostgreSQL database servers. Helps with understanding oTree and data recovery if something goes wrong.

In Ubuntu and official derivatives you can install chromium, conky, and pgAdmin via the package manager, like so:

```
$ sudo apt-get install conky-all chromium-browser pgadmin3
```

It is recommended to obtain LightTable from the official website as the community packages are deprecated.

You can find the website here: <http://lighttable.com>



## 3.1 General usage

`ovmm` is a complement to the **oTree Virtual Machine Image** provided by Felix Albrecht and Holger Gerhardt. Therefore, until now it is only tested on this specific image. But, we do provide information (*Requirements*) on how to create a similar image which can use `ovmm`.

Always run `ovmm` with `sudo` or some commands might not work.

If you need more information on the commands, run `ovmm --help`

## 3.2 First run

1. When using `ovmm` for the first time, run `sudo ovmm initialise` to configure your system.

---

**Note:** The process leads you through the installation of required dependencies and creates necessary content files for `ovmm`.

1. Enter administrator's password.
2. Enter account information of the PostgreSQL superuser.

By default, the superuser for a PostgreSQL database is called `postgres`. You also have to set a password for this account. This password can be new and will be set to the user by request.

---

2. After running the command, there is a folder called `/home/<user>/ovmm_sources/` which contains `ovmm_conf.yml`. Perform the following checks:
  1. Check whether the login information for the PostgreSQL database is correctly set.
  2. Define the ranges of port numbers, e.g. `OVMM_DAPHNE_RANGE`. Port ranges are stored in lists. Insert distinct values or use list comprehension for bigger ranges.

3. Next, go back to your home folder. You should see a file called `nginx_template`. Read the commentary to adjust the file to your network configuration.

This file will be reused every time a new user is created. The process will fill out the user specific values and place it under `/etc/nginx/sites-avaialbe`. Then, a symlink will link this file to `/etc/nginx/sites-enabled`, so that **Nginx** takes care of it.

4. Forelast, try out one of the innocent lookup commands like `sudo ovmm count_user` or `list_user`. If they succeed, you have correctly set up `ovmm`.
5. Last, set up a test account and verify whether everything is working including *oTree* by running one of the example experiments.

## 3.3 Command line options

Command line options are entered used with `sudo ovmm <command>`

### 3.3.1 Commands

Since command expansion is implemented, the commands after the forward slash are also available.

**initialise / i** Adjust your system to the needs of `ovmm`. It installs Ubuntu dependencies as well as configuration files for the Administrator.

**Warning:**

1. This command should be executed on first run in advance of any other command.
2. An internet connection is needed

**add\_user / a** Add a new user to the configuration. Several prompts will ask you about user specific information. Please, fill out the forms in accordance with the provided examples.

**delete\_user / d** Delete a user from the configuration. Her account including her home folder is completely removed. Make sure you have a backup of all files in advance.

**backup\_user / b** Create a backup for a given user and save it in the administrator's home directory under `/home/<admin>/ovmm_sources/user_backups`. Choose from one of three options, `all`, `db`, `home`, whether you want to make a backup of the database or the home folder. `all` is a shortcut to run a backup of both, database and home folder.

**count\_user / c** Return the number of existing accounts and the number of possible, additional accounts.

**list\_user / l** Give a list of user names of all currently installed users.

**route\_port / r** Change nginx default config to route port 80 (HTTP) and port 443 (HTTPS) to specific user account in addition to user specific port. Intended for running experiments where router settings prevent the access to non-standard web-ports like 780x.

---

## End User Desktop

---

Apart from the standard otree commands we created a set of commands for the end user to improve usability. The commands are available via command line and via desktop launcher providing an easy to use GUI that guides the user through the several processes. The user has the following commands available:

- `otree_startproject` creates an otree project in the folder *Projects* in the user's home directory. The created folder is a basic otree project folder WITHOUT the example apps. It further creates either a symbolic link `venv` to the default virtual python environment of the user in the project folder or creates a specific virtual environment within the project folder called `venv`. The latter allows for specialized oTree versions for each project. Lastly it copies the `_rooms` directory from the folder *Templates* into the project folder. `_rooms` contains a preconfigured `econ101.txt` for 24 clients.
- `otree_link_experiment` is used to create a symbolic link to the nginx root location. The nginx webserver looks in a predefined location for a webserver root. The command handles the linking for you and creates a symlink to `$HOME/.oTree` which is where nginx looks for your project.
- `otree_restart` helps you with your starting requirements. It offer to run `otree collectstatic` for you as well as an `otree resetdb`. Before the database reset is executed `pg_dump` makes a backup of the current state of the database and stores a 7zipped version of it in `$HOME/DB_BackUps`. Finally it runs `otree runprodserver --port=YOUR_DAPHNE_PORT` or `otree runserver YOUR_DAPHNE_PORT` for you to ensure that you use the correct port. `otree_restart` also comes with a number of arguments that help you get started more quickly.

**-c** run `otree collectstatic`

**-h** help

**-l** run local (programming) server (`runserver`)

**-m** Send emails when otree stops running. (Your email is configured in `otree_envIRON_config`.)

**-p** skip queries and run `otree runprodserver --port=YOUR_PORT` directly

**-r** run `otree resetdb`

**-s** start otree in a detached screen (virtual console)

The commands are stored in `$HOME/.local/bin` for each user separately so that users can make adjustments for themselves if needed.

All three commands offer a console and a GUI based on GTK zenity dialogs. For calling the GUI three `*.desktop` are created in `$HOME/Desktop`.

When you call up a command the command will guide you through the process.

The below graphic provides an overview of the file structure created in the oTree user's home directory and the command relations.

The user should not run `otree runserver` (which is just for local testing) or `otree runprodserver` (because of the proxy settings).

## 4.1 Starting a new Project as End User

You need to complete 2 steps in order start a new project and connect it to the server root directory.

### Step 1)

1. Execute `otree_startproject`
2. Provide a project name.
3. Choose whether to create a **specialized virtual environment** for this project.

### Step 2)

1. Execute `otree_link_experiment`
2. Select the path to the project you want to run.

## 4.2 Running Experiments as End User

As End User you have to follow the following steps in order to run experiments.

1. Execute `otree_restart`
2. Choose whether to do a collect static or not.
3. Choose whether to do a database reset.
4. Watch the output if everything starts fine. ;-)

If you have trouble starting the experiment or you want to activate demo mode change the necessary parameters in the `otree_envIRON_config`. `otree_restart sources otree_envIRON_config` each time it is called. You don't need to manually source it to activate the new settings.

This part of the documentation covers all parts of **ovmm**.

### 5.1 Commands

The main functionality of **ovmm** resides in this module. You can see all the steps performed during execution.



Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

## 6.1 Types of Contributions

### 6.1.1 Report Bugs

Report bugs at [https://github.com/tobiasraabe/otree\\_virtual\\_machine\\_manager/issues](https://github.com/tobiasraabe/otree_virtual_machine_manager/issues).

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

### 6.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

### 6.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

## 6.1.4 Write Documentation

oTree Virtual Machine Manager could always use more documentation, whether as part of the official oTree Virtual Machine Manager docs, in docstrings, or even on the web in blog posts, articles, and such.

If you want to participate by writing docstrings, please, follow the guidelines for [NumPy Style Python Docstrings](#). For a complete example, follow this link ([Example NumPy Docstring](#)).

A more general tutorial for reStructuredText can be found [here](#).

## 6.1.5 Submit Feedback

The best way to send feedback is to file an issue at [https://github.com/tobiasraabe/otree\\_virtual\\_machine\\_manager/issues](https://github.com/tobiasraabe/otree_virtual_machine_manager/issues).

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

## 6.2 Get Started!

Ready to contribute? Here's how to set up *ovmm* for local development.

1. Fork the *ovmm* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/ovmm.git
```

3. Install your local copy into a virtualenv. Assuming you have *virtualenvwrapper* installed, this is how you set up your fork for local development:

```
$ mkvirtualenv ovmm
$ cd ovmm/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you're done making changes, check that your changes pass all the tests defined with *tox*:

```
$ tox
```

You can test separately by typing:

```
$ tox -e ${TOXENV}
```

The following commands test python code and documentation:

```
$ tox -e flake8
$ tox -e doc8
```

If you want to lint code and documentation, type:

```
$ tox -e linters
```

To get flake8, tox, doc8, restructuredtext\_lint just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

## 6.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.4, 3.5 and 3.6, and for PyPy. Check [https://travis-ci.org/tobiasraabe/otree\\_virtual\\_machine\\_manager/pull\\_requests](https://travis-ci.org/tobiasraabe/otree_virtual_machine_manager/pull_requests) and make sure that the tests pass for all supported Python versions.

## 6.4 Tips

To run a subset of tests:

```
$ py.test tests.test_otree_virtual_machine_manager
```



### 7.1 Development Lead

- Felix Albrecht <f.albrecht@uni-bonn.com> (enduser commands/statics, documentation, director)
- Tobias Raabe <tobiasraabe@uni-bonn.de> (ovmm core)

### 7.2 Contributors

None yet. Why not be the first?



### **8.1 0.2.2 (2017-07-21)**

- Hotfix for v0.2.1 due to error on initialise.

### **8.2 0.2.1 (2017-07-16)**

- Re-released v0.2.0.

### **8.3 0.2.0 (2017-07-16)**

- Added `route_port` command to change standard ports to a different user.

### **8.4 0.1.1 (2017-03-20)**

- Alpha release. Re-released on PyPI :).

### **8.5 0.1.0 (2017-03-20)**

- Alpha release. Released on PyPI.



## CHAPTER 9

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



**O**

`ovmm.commands`, 13



O

ovmm.commands (*module*), 13