Operation

- See separate manual for initial setup

Login via SSH

```
# login
# normally ssh otree@dmd.zew.de, but we use shared key-pair
# you receive this key pair from Peter Buchmann
# save it to your computer to
#     %USERPROFILE%\.ssh\dmd-otree\
# now login
ssh -i %USERPROFILE%\.ssh\dmd-otree\dmd-otree.pem otree@dmd.zew.de

cd ~/otree/[your project]

# if VENV is not active
source ~/venv_otree/bin/activate
# you never need to leave it, but if you would
deactivate
```

Check if otree is running - terminate it

```
# check if otree is running from previous session
ps aux | grep otree

# stop previous otree
sudo /usr/bin/pkill otree
# or better stackoverflow.com/questions/18359433/
sudo /bin/kill $(ps aux | grep 'otree' | awk '{print $2}')
```

Operation of otree

- Since the usual tools to start and stop otree as service do not work
  we need a way to keep otree running after disconnecting the SSH session
- We also want to watch the log of the otree server,
  when we reconnect after a while
- You could try
  `nohup otree runprodserver --port=80 > /var/lib/otree/otree.log 2>&1 &`
  but this standard technique does not work
- Our solution:
  Using tmux
  `tmux` is a persistent shell
  to run the otree server
  persistent across ssh sessions
  and still observe `stdout` and `stderr`

```
# showing a list of tmux sessions:
tmux ls
# if this says
```

```
#    no server running on /tmp/tmux-1002/default
# then *create* a session
#
# if it says
#    sessotree: 1 windows (created Wed Nov 24 21:54:08 2021)
# then attach to existing session

# create new session and jump into it
tmux new -s sessotree

# attach to the one existing session
tmux a
# there should not be more than on session
# if you accidentally created more than one, delete the extra sessions


# detach from a session
CTRL+b d

# INSIDE the session: Run otree
# ---------------------------

# start dev server
otree devserver 8000
# stop with CTRL+C

# start prod server
sudo -E env "PATH=$PATH" otree runprodserver 80
# stop with CTRL+C

# open web browser
<http://dmd.zew.de>

# devserver would be
#   <http://dmd.zew.de:8000>
# but port 8000 is blocked by the ZEW firewall.
# Even from within ZEW via the in internal IP address
#   <http://192.168.2.82:8000>
# Thus devserver is pretty much useless.
#
# You can only check it via command line
#   wget http://193.196.11.82:8000/
#   wget http://localhost:8000/


# website login
ADMIN_USERNAME = 'admin'
ADMIN_PASSWORD = 'Crikey++1990'

# database reset
otree resetdb

# terminate ssh session
# *always* leave tmux session first CTRL+b d
exit
```

Setup for uploading experiment files

- Uploading is accomplished via SFTP
- There are several solutions for SFTP on windows
- WinSCP
  with automatic sync feature

or

- TotalCommander with SFTP plugin

or

- mobaxterm

SFTP plugin for TotalCommander - one time setup

- This does not work with a shared key pair
  this requires the appending of your *personal*
  `id_rsa.pub` to the server's
  `/home/otree/.ssh/authorized_keys` file
- Download TotalCommander-sftp Plugin.
- Extract ZIP file to `C:\totalcmd\plugins`
  (or maybe `C:\programs\totalcmd\plugins`)
- Start TotalCommander
- (assuming German language settings)
  Menü Konfigurieren - Einstellungen
  Plugins
  Dateisystem-Plugins - Konfigurieren
  Hinzufügen...
  Zur Datei `plugin_sftp.wfx` navigieren und auswählen.
- Nun findet sich im TotalCommander
  im Laufwerk-Dropdown oben rechts
  in der Liste ganz unten `Netzwerkumgebung`
  dann der erste Eintrag `[Secure FTP Connections]`
  dann `edit connections`
- Create and save a new session: `otree_server_01`
  - session: hostname (or IP address): dmd.zew.de
  - connection - data - auto-login-username: otree
  - SSH - Auth - Private key file for authentication:
  - .ppk file created from your local ~/.ssh/id_rsa and with your id_rsa.pub
  - Back to "Session" - Button save
  - Dialog schliessen

TotalCommander usage

- Connect to otree server fileystem
- Laufwerk-Dropdown ganz unten
  `Netzwerkumgebung`
- `[Secure FTP Connections]`
- `otree_server_01`
- (einmalig "Host key..." mit "ja" akzeptieren)
- Navigate to `/home/otree/otree/[your project]`

- Copy files back and forth...

Then see above for restarting the otree server...

Monitoring database during an experiment

```
sudo su - postgres
psql
select * from pg_stat_activity;
q
```

More database monitoring info
search for `Select...`

Monitoring server resources during experiment

```
htop
iotop
```

Static files - dynamic content

https://otree.readthedocs.io/en/latest/misc/advanced.html#static-files

- At the root of your otree project, there is a _static/ folder.
  Put a file there, for example `numbers.csv`
- Then, in your template, you can get the URL to that file with
  `{% static 'numbers.csv' %}`
- You can change the contents of that file from each python class

```
numbers = ["One\n", "Two\n", "Three\n", "Four\n", "Five\n"]
F = open("numbers.csv", "a")
F.writelines(numbers)
F.close()
```

- See separate manual for initial setup

Postgres caveat

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

Check redis

As of 2021-11, we dont seem to need redis anymore

```
redis-cli ping
# answer must be PONG
```

Using bg and fg

This is another way to run the server, but you lose the server log

```
# if you did not use tmux,
# you can bring the process into background
CTRL+Z
bg
# you can now exit the ssh shell with exit

# or bringing it into foreground again
fg
# stop with CTRL+C
```