Command-line interface: odoo-bin

Running the server

`-d <database>, --database <database>`

database(s) used when installing or updating modules. Providing a comma-separated list restrict access to databases provided in list.
For advanced database options, take a look below.

`-i <modules>, --init <modules>`

comma-separated list of modules to install before running the server (requires `-d`).

`-u <modules>, --update <modules>`

comma-separated list of modules to update before running the server (requires `-d`).

`--addons-path <directories>`

comma-separated list of directories in which modules are stored. These directories are scanned for modules.

`-c <config>, --config <config>`

provide an alternate configuration file

`-s, --save`

saves the server configuration to the current configuration file (`$HOME/.odoorc` by default, and can be overridden using `-c`).

`--without-demo`

disables demo data loading for modules installed comma-separated, use `all` for all modules.

`--test-enable`

runs tests after installing modules

`--test-tags 'tag_1,tag_2,...,-tag_n'`

select the tests to run by using tags.

`--screenshots`

Specify directory where to write screenshots when an HttpCase.browser_js test fails. It defaults to `/tmp/odoo_tests/db_name/screenshots`

`--screencasts`

Enable screencasts and specify directory where to write screencasts files. The **ffmpeg** utility needs to be installed to encode frames into a video file. Otherwise frames will be kept instead of the video file.

Database

-r <user>, --db_user <user>

database username, used to connect to PostgreSQL.

-w <password>, --db_password <password>

database password, if using password authentication (https://www.postgresql.org/docs/9.3/static/auth-methods.html#AUTH-PASSWORD).

--db_host <hostname>

host for the database server

- **localhost** on Windows
- UNIX socket otherwise

--db_port <port>

port the database listens on, defaults to 5432

--db-filter <filter>

hides databases that do not match **<filter>**. The filter is a regular expression (https://docs.python.org/3/library/re.html), with the additions that:

- **%h** is replaced by the whole hostname the request is made on.
- **%d** is replaced by the subdomain the request is made on, with the exception of **www** (so domain **odoo.com** and **www.odoo.com** both match the database **odoo**).
  These operations are case sensitive. Add option **(?i)** to match all databases (so domain **odoo.com** using **(?i)%d** matches the database **Odoo**).

Since version 11, it's also possible to restrict access to a given database listen by using the –database parameter and specifying a comma-separated list of databases
When combining the two parameters, db-filter supersedes the comma-separated database list for restricting database list, while the comma-separated list is used for performing requested operations like upgrade of modules.

```
$ odoo-bin --db-filter ^11.*$
```

Restrict access to databases whose name starts with 11

```
$ odoo-bin --database 11firstdatabase,11seconddatabase
```

Restrict access to only two databases, 11firstdatabase and 11seconddatabase

```
$ odoo-bin --database 11firstdatabase,11seconddatabase -u base
```

Restrict access to only two databases, 11firstdatabase and 11seconddatabase, and update base module on one database: 11firstdatabase. If database 11seconddatabase doesn't exist, the database is created and base modules is installed

```
$ odoo-bin --db-filter ^11.*$ --database 11firstdatabase,11seconddatabase -u base
```

Restrict access to databases whose name starts with 11, and update base module on one database: 11firstdatabase. If database 11seconddatabase doesn't exist, the database is created and base modules is installed

`--db-template <template>`

when creating new databases from the database-management screens, use the specified template database (https://www.postgresql.org/docs/9.3/static/manage-ag-templatedbs.html). Defaults to `template0`.

`--pg_path </path/to/postgresql/binaries>`

Path to the PostgreSQL binaries that are used by the database manager to dump and restore databases. You have to specify this option only if these binaries are located in a non-standard directory.

`--no-database-list`

Suppresses the ability to list databases available on the system

`--db_sslmode`

Control the SSL security of the connection between Odoo and PostgreSQL. Value should bve one of 'disable', 'allow', 'prefer', 'require', 'verify-ca' or 'verify-full' Default value is 'prefer'

Emails

`--email-from <address>`

Email address used as <FROM> when Odoo needs to send mails

`--smtp <server>`

Address of the SMTP server to connect to in order to send mails

`--smtp-port <port>`

`--smtp-ssl`

If set, odoo should use SSL/STARTSSL SMTP connections

`--smtp-user <name>`

Username to connect to the SMTP server

`--smtp-password <password>`

Password to connect to the SMTP server

Internationalisation

Use these options to translate Odoo to another language. See i18n section of the user manual. Option '-d' is mandatory. Option '-l' is mandatory in case of importation

`--load-language <languages>`

specifies the languages (separated by commas) for the translations you want to be loaded

`-l, --language <language>`

specify the language of the translation file. Use it with –i18n-export or –i18n-import

`--i18n-export <filename>`

export all sentences to be translated to a CSV file, a PO file or a TGZ archive and exit.

`--i18n-import <filename>`

import a CSV or a PO file with translations and exit. The '-l' option is required.

`--i18n-overwrite`

overwrites existing translation terms on updating a module or importing a CSV or a PO file.

`--modules`

specify modules to export. Use in combination with –i18n-export

Advanced Options

Developer features

`--dev <feature,feature,...,feature>`

- `all` : all the features below are activated
- `xml` : read template qweb from xml file directly instead of database. Once a template has been modified in database, it will be not be read from the xml file until the next update/init.
- `reload` : restart server when python file are updated (may not be detected depending on the text editor used)
- `qweb` : break in the evaluation of qweb template when a node contains `t-debug='debugger'`
- `(i)p(u)db` : start the chosen python debugger in the code when an unexpected error is raised before logging and returning the error.

HTTP

`--no-http`

do not start the HTTP or long-polling workers (may still start cron (actions.html#reference-actions-cron) workers)

> ⚠ **Warning**
> has no effect if `--test-enable` is set, as tests require an accessible HTTP server

`--http-interface <interface>`

TCP/IP address on which the HTTP server listens, defaults to `0.0.0.0` (all addresses)

`--http-port <port>`

Port on which the HTTP server listens, defaults to 8069.

`--longpolling-port <port>`

TCP port for long-polling connections in multiprocessing or gevent mode, defaults to 8072. Not used in default (threaded) mode.

`--proxy-mode`

enables the use of `X-Forwarded-*` headers through Werkzeug's proxy support (http://werkzeug.pocoo.org/docs/contrib/fixers/#werkzeug.contrib.fixers.ProxyFix).

> ⚠ **Warning**
> proxy mode *must not* be enabled outside of a reverse proxy scenario

Logging

By default, Odoo displays all logging of level (https://docs.python.org/3/library/logging.html#logging.Logger.setLevel) `info` except for workflow logging (`warning` only), and log output is sent to `stdout` . Various options are available to redirect logging to other destinations and to customize the amount of logging output.

`--logfile <file>`

sends logging output to the specified file instead of stdout. On Unix, the file can be managed by external log rotation programs (https://docs.python.org/3/library/logging.handlers.html#watchedfilehandler) and will automatically be reopened when replaced

`--syslog`

logs to the system's event logger: syslog on unices (https://docs.python.org/3/library/logging.handlers.html#sysloghandler) and the Event Log on Windows (https://docs.python.org/3/library/logging.handlers.html#nteventloghandler). Neither is configurable

`--log-db <dbname>`

logs to the `ir.logging` model (`ir_logging` table) of the specified database. The database can be the name of a database in the "current" PostgreSQL, or a PostgreSQL URI (https://www.postgresql.org/docs/9.2/static/libpq-connect.html#AEN38208) for e.g. log aggregation.

`--log-handler <handler-spec>`

`LOGGER``:`*`LEVEL`* , enables `LOGGER` at the provided `LEVEL` e.g. `odoo.models:DEBUG` will enable all logging messages at or above `DEBUG` level in the models.

The colon `:` is mandatory

The logger can be omitted to configure the root (default) handler

If the level is omitted, the logger is set to `INFO`

The option can be repeated to configure multiple loggers e.g.

```
$ odoo-bin --log-handler :DEBUG --log-handler werkzeug:CRITICAL --log-handler odoo.field
```

`--log-request`

enable DEBUG logging for RPC requests, equivalent to `--log-handler=odoo.http.rpc.request:DEBUG`

`--log-response`

enable DEBUG logging for RPC responses, equivalent to `--log-handler=odoo.http.rpc.response:DEBUG`

`--log-web`

enables DEBUG logging of HTTP requests and responses, equivalent to `--log-handler=odoo.http:DEBUG`

`--log-sql`

enables DEBUG logging of SQL querying, equivalent to `--log-handler=odoo.sql_db:DEBUG`

`--log-level <level>`

Shortcut to more easily set predefined levels on specific loggers. "real" levels (`critical` , `error` , `warn` , `debug`) are set on the `odoo` and `werkzeug` loggers (except for `debug` which is only set on `odoo`).

Odoo also provides debugging pseudo-levels which apply to different sets of loggers:

`debug_sql`

sets the SQL logger to `debug`

equivalent to `--log-sql`

`debug_rpc`

sets the `odoo` and HTTP request loggers to `debug` equivalent to `--log-level debug --log-request`

`debug_rpc_answer`
sets the `odoo` and HTTP request and response loggers to `debug`

equivalent to `--log-level debug --log-request --log-response`

> In case of conflict between `--log-level` and `--log-handler`, the latter is used

Multiprocessing

`--workers <count>`

if `count` is not 0 (the default), enables multiprocessing and sets up the specified number of HTTP workers (sub-processes processing HTTP and RPC requests).

> multiprocessing mode is only available on Unix-based systems

A number of options allow limiting and recycling workers:

`--limit-request <limit>`

Number of requests a worker will process before being recycled and restarted.
Defaults to *8196*.

`--limit-memory-soft <limit>`

Maximum allowed virtual memory per worker. If the limit is exceeded, the worker is killed and recycled at the end of the current request.
Defaults to *2048MiB*.

`--limit-memory-hard <limit>`

Hard limit on virtual memory, any worker exceeding the limit will be immediately killed without waiting for the end of the current request processing.
Defaults to *2560MiB*.

`--limit-time-cpu <limit>`

Prevents the worker from using more than <limit> CPU seconds for each request. If the limit is exceeded, the worker is killed.
Defaults to *60*.

`--limit-time-real <limit>`

Prevents the worker from taking longer than <limit> seconds to process a request. If the limit is exceeded, the worker is killed.
Differs from `--limit-time-cpu` in that this is a “wall time” limit including e.g. SQL queries.
Defaults to *120*.

`--max-cron-threads <count>`

number of workers dedicated to cron (actions.html#reference-actions-cron) jobs. Defaults to *2*. The workers are threads in multi-threading mode and processes in multi-processing mode. For multi-processing mode, this is in addition to the HTTP worker processes.

Configuration file

Most of the command-line options can also be specified via a configuration file. Most of the time, they use similar names with the prefix `-` removed and other `-` are replaced by `_` e.g. `--db-template` becomes `db_template`.

Some conversions don't match the pattern:

- `--db-filter` becomes `dbfilter`
- `--no-http` corresponds to the `http_enable` boolean
- logging presets (all options starting with `--log-` except for `--log-handler` and `--log-db`) just add content to `log_handler`, use that directly in the configuration file
- `--smtp` is stored as `smtp_server`
- `--database` is stored as `db_name`
- `--i18n-import` and `--i18n-export` aren't available at all from configuration files

The default configuration file is `$HOME/.odoorc` which can be overridden using `--config`. Specifying `--save` will save the current configuration state back to that file. The configuration items relative to the command-line are to be specified in the section `[options]`.

Here is a sample file:

```
[options]
db_user=odoo
dbfilter=odoo
```

Shell

Odoo command-line also allows to launch odoo as a python console environment. This enables direct interaction with the orm (orm.html#reference-orm) and its functionalities.

```
$ odoo_bin shell
```

`--shell-interface (ipython|ptpython|bpython|python)`

Specify a preferred REPL to use in shell mode.

Scaffolding

Scaffolding is the automated creation of a skeleton structure to simplify bootstrapping (of new modules, in the case of Odoo). While not necessary it avoids the tedium of setting up basic structures and looking up what all starting requirements are.
Scaffolding is available via the **odoo-bin scaffold** subcommand.

```
$ odoo_bin scaffold my_module /addons/
```

`name (required)`

the name of the module to create, may munged in various manners to generate programmatic names (e.g. module directory name, model names, …)

`destination (default=current directory)`

directory in which to create the new module, defaults to the current directory

`-t <template>`

a template directory, files are passed through jinja2 (http://jinja.pocoo.org) then copied to the **`destination`** directory
This will create module *my_module* in directory */addons/*.

Database Population

Odoo CLI supports database population features. If the feature is implemented on a given model (testing.html#reference-testing-populate-methods), it allows automatic data generation of the model's records to test your modules in databases containing non-trivial amounts of records.

```
$ odoo_bin populate
```

`--models`

list of models for which the database should be filled

`--size (small|medium|large)`

population size, the actual records number depends on the model's **`_populate_sizes`** attribute. The generated records content is specified by the **`_populate_factories()`** method of a given model (cf. the **`populate`** folder of modules for further details).

> 🡆 **See also**
> **Database population (testing.html#reference-testing-populate)**

Cloc

Odoo Cloc is a tool to count the number of relevant lines written in Python, Javascript or XML. This can be used as a rough metric for pricing maintenance of extra modules.

Command-line options

`-d <database>, --database <database>`

Process the code of all extra modules installed on the provided database, and of all server actions and computed fields manually created in the provided database.
The `--addons-path` option is required to specify the path(s) to the module folder(s).
If combined with `--path`, the count will be that of the sum of both options' results (with possible overlaps). At least one of these two options is required to specify which code to process.

```
$ odoo-bin cloc --addons-path=addons -d my_database
```

> **➡ See also**
>
> **With the --database option**

`-p <path>, --path <path>`

Process the files in the provided path.
If combined with `--database`, the count will be that of the sum of both options' results (with possible overlaps). At least one of these two options is required to specify which code to process.

```
$ odoo-bin cloc -p addons/account
```

Multiple paths can be provided by repeating the option.

```
$ odoo-bin cloc -p addons/account -p addons/sale
```

> **➡ See also**
>
> **With the --path option**

`--addons-path <directories>`

Comma-separated list of directories in which modules are stored. These directories are scanned for modules.
Required if the `--database` option is used.

`-c <directories>`

Specify a configuration file to use in place of the `--addons-path` option.

```
$ odoo-bin cloc -c config.conf -d my_database
```

`-v, --verbose`

Show the details of lines counted for each file.

Processed files

With the `--database` option

Odoo Cloc counts the lines in each file of extra installed modules in a given database. In addition, it counts the Python lines of server actions and custom computed fields that have been directly created in the database or imported.
Some files are excluded from the count by default:

- The manifest (`__manifest__.py` or `__openerp__.py`)
- The contents of the folder `static/lib`
- The tests defined in the folder `tests` and `static/tests`
- The migrations scripts defined in the folder `migrations`
- The XML files declared in the `demo` or `demo_xml` sections of the manifest

For special cases, a list of files that should be ignored by Odoo Cloc can be defined per module. This is specified by the `cloc_exclude` entry of the manifest:

```
"cloc_exclude": [
    "lib/common.py", # exclude a single file
    "data/*.xml",    # exclude all XML files in a specific folder
    "example/**/*",  # exclude all files in a folder hierarchy recursively
]
```

The pattern `**/*` can be used to ignore an entire module. This can be useful to exclude a module from maintenance service costs.
For more information about the pattern syntax, see glob (https://docs.python.org/3/library/pathlib.html#pathlib.Path.glob).

With the `--path` option

This method works the same as with the –database option if a manifest file is present in the given folder. Otherwise, it counts all files.

Identifying Extra Modules

To distinguish between standard and extra modules, Odoo Cloc uses the following heuristic: modules that are located (real file system path, after following symbolic links) in the same parent directory as the `base` , `web` or `web_enterprise` standard modules are considered standard. Other modules are treated as extra modules.

Error Handling

Some file cannot be counted by Odoo Cloc. Those file are reported at the end of the output.

Max file size exceeded

Odoo Cloc rejects any file larger than 25MB. Usually, source files are smaller than 1 MB. If a file is rejected, it may be:

- A generated XML file that contains lots of data. It should be excluded in the manifest.
- A JavaScript library that should be placed in the `static/lib` folder.

Syntax Error

Odoo Cloc cannot count the lines of code of a Python file with a syntax problem. If an extra module contains such files, they should be fixed to allow the module to load. If the module works despite the presence of those files, they are probably not loaded and should therefore be removed from the module, or at least excluded in the manifest via `cloc_exclude`.