
mpm Documentation

Release 0.25.1.dev9493927

Christian Fobel

Jan 17, 2018

Contents

1	Installation	3
2	Usage	5
2.1	Common flags	5
2.2	mpm install flags	5
2.3	mpm search flags	6
3	Examples	7
4	Documentation	9
5	Development	11
5.1	Project Modules	11
6	Indices and tables	15
	Python Module Index	17

Microdrop plugin manager inspired by `pip`.

CHAPTER 1

Installation

Install using pip:

```
pip install microdrop-package-manager
```

```
mpm install <plugin-name>[ (==|>|>=|<=)version] [<plugin-name>[ (==|>|>=|<=)version]...]
mpm install -r plugin_requirements.txt
mpm uninstall <plugin-name>
mpm freeze
mpm --help # Display detailed usage information
```

Use `mpm --help` for detailed usage information.

2.1 Common flags

`-l, --log-level`: Logging level (error, debug, info).

`-c, --config-file`: Microdrop config file (default= <Documents>\Microdrop\microdrop.ini).

`-d, --plugins-directory`: Microdrop plugins directory (default= <Documents>\Microdrop\plugins).

Note that the `--config-file`/`--plugins-directory` flags are used to locate the plugins directory to operate on.

If the `--config-file` flag is used, the plugin directory is read from the configuration file (relative paths are considered relative to the location of the configuration file).

The `--plugins-directory` flag sets the plugins directory explicitly.

2.2 mpm install flags

`-s, --server-url`: Microdrop plugin index URL (default= `http://microfluidics.utoronto.ca/update`)

`--no-on-install`: Do not run `on_plugin_install` hook after installing plugin

`-r, --requirements-file`: Requirements file (one line per plugin version descriptor)

2.3 mpm search flags

-s, --server-url: Microdrop plugin index URL (default= <http://microfluidics.utoronto.ca/update>)

CHAPTER 3

Examples

Install dmfc_control_board:

```
npm install dmfc_control_board
```

Install specific version of dmfc_control_board:

```
npm install "dmfc_control_board==1.1.0"
```

Uninstall dmfc_control_board:

```
npm uninstall dmfc_control_board
```

Install plugin from archive file:

```
npm install dmfc_control_board-1.1.0.tar.gz
```

Print list of installed plugins:

```
npm freeze
```

CHAPTER 4

Documentation

Documentation is available online [here](#) .

Project is hosted on [GitHub](#) .

Contents:

5.1 Project Modules

5.1.1 mpm Package

mpm Package

`mpm.pformat_dict` (*data*, *separator*=`' '`)

commands Module

Inspired by *pip*.

```
mpm install <plugin-name>[ (==|>|>=|<=)version] [<plugin-name>[ (==|>|>=|<=)version]...]
mpm install -r plugin_requirements.txt
mpm uninstall <plugin-name>
mpm freeze
```

`mpm.commands.extract_metadata` (*fileobj*)

Extract metadata from plugin archive file-like object (e.g., opened file, `StringIO`).

Parameters `fileobj` (*file-like*) – MicroDrop plugin archive file object.

Returns Metadata dictionary for plugin.

Return type `dict`

`mpm.commands.freeze` (*plugins_directory*)

Parameters `plugins_directory` (*str*) – Path to MicroDrop user plugins directory.

Returns List of package strings corresponding to installed plugin versions.

Return type list

`mpm.commands.get_plugins_directory` (*config_path=None, microdrop_user_root=None*)

Resolve plugins directory.

Plugins directory is resolved as follows, highest-priority first:

1. plugins directory specified in provided `config_path`.
2. plugins sub-directory of specified MicroDrop profile path (i.e., `microdrop_user_root`)
3. plugins sub-directory of parent directory of configuration file path specified using `MICRODROP_CONFIG` environment variable.
4. plugins sub-directory of MicroDrop profile path specified using `MICRODROP_PROFILE` environment variable.
5. Plugins directory specified in `<home directory>/MicroDrop/microdrop.ini`.
6. Plugins directory in default profile location, i.e., `<home directory>/MicroDrop/plugins`.

Parameters

- `config_path` (*str, optional*) – Configuration file path (i.e., path to `microdrop.ini`).
- `microdrop_user_root` (*str, optional*) – Path to MicroDrop user data directory.

Returns Absolute path to plugins directory.

Return type path

`mpm.commands.home_dir` ()

Returns Path to home directory (or Documents directory on Windows).

Return type *str*

`mpm.commands.install` (*plugin_package, plugins_directory, server_url='http://microfluidics.utoronto.ca/update/plugins/{}/json/'*)

Parameters

- `plugin_package` (*str*) – Name of plugin package hosted on MicroDrop plugin index. Version constraints are also supported (e.g., `"foo"`, `"foo==1.0"`, `"foo>=1.0"`, etc.) See [version specifiers](#) reference for more details.
- `plugins_directory` (*str*) – Path to MicroDrop user plugins directory.
- `server_url` (*str*) – URL of JSON request for MicroDrop plugins package index. See `DEFAULT_SERVER_URL` for default.

Returns Path to directory of installed plugin and plugin package metadata dictionary.

Return type (path, dict)

`mpm.commands.install_fileobj` (*fileobj, plugin_path*)

Extract and install plugin from file-like object (e.g., opened file, `StringIO`).

Parameters

- `fileobj` (*file-like*) – MicroDrop plugin file object to extract and install.
- `plugin_path` (*path*) – Target plugin install directory path.

Returns Directory of installed plugin and metadata dictionary for plugin.

Return type (path, dict)

`mpm.commands.plugin_request(plugin_str)`

Extract plugin name and version specifiers from plugin descriptor string.

`mpm.commands.search(plugin_package, server_url='http://microfluidics.utoronto.ca/update/plugins/{}/json/')`

Parameters

- **plugin_package** (*str*) – Name of plugin package hosted on MicroDrop plugin index. Version constraints are also supported (e.g., "foo", "foo==1.0", "foo>=1.0", etc.) See [version specifiers](#) reference for more details.
- **server_url** (*str*) – URL of JSON request for MicroDrop plugins package index. See `DEFAULT_SERVER_URL` for default.

Returns Name of found plugin and mapping of version strings to plugin package metadata dictionaries.

Return type (str, OrderedDict)

`mpm.commands.uninstall(plugin_package, plugins_directory)`

Parameters

- **plugin_package** (*str*) – Name of plugin package hosted on MicroDrop plugin index.
- **plugins_directory** (*str*) – Path to MicroDrop user plugins directory.

hooks Module

`mpm.hooks.on_plugin_install(plugin_directory, ostream=<open file '<stdout>', mode 'w'>)`

Run `on_plugin_install` script for specified plugin directory (if available).

TODO Add support for Linux, OSX.

Parameters

- **plugin_directory** (*str*) – File system to plugin directory.
- **ostream** (*file-like*) – Output stream for status messages (default: `sys.stdout`).

Subpackages

bin Package

bin Package

`mpm.bin.main(args=None)`

`mpm.bin.parse_args(args=None)`

Parses arguments, returns (options, args).

`mpm.bin.validate_args(args)`

Apply custom validation and actions based on parsed arguments.

Parameters **args** (*argparse.Namespace*) – Result from `parse_args` method of `argparse.ArgumentParser` instance.

Returns Reference to input args, which have been validated/updated.

Return type `argparse.Namespace`

install_dependencies Module

`npm.bin.install_dependencies.install_dependencies` (*plugins_directory*, *ostream=<open file '<stdout>', mode 'w'>*)

Run `on_plugin_install` script for each plugin directory found in specified plugins directory.

Parameters

- **plugins_directory** (*str*) – File system path to directory containing zero or more plugin subdirectories.
- **ostream** (*file-like*) – Output stream for status messages (default: `sys.stdout`).

`npm.bin.install_dependencies.main` (*args=None*)

`npm.bin.install_dependencies.parse_args` (*args=None*)
Parses arguments, returns (*options*, *args*).

`npm.bin.install_dependencies.validate_args` (*args*)

CHAPTER 6

Indices and tables

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

m

- `npm`, [11](#)
- `npm.bin`, [13](#)
- `npm.bin.install_dependencies`, [14](#)
- `npm.commands`, [11](#)
- `npm.hooks`, [13](#)

E

`extract_metadata()` (in module `mpm.commands`), 11

F

`freeze()` (in module `mpm.commands`), 11

G

`get_plugins_directory()` (in module `mpm.commands`), 12

H

`home_dir()` (in module `mpm.commands`), 12

I

`install()` (in module `mpm.commands`), 12

`install_dependencies()` (in module `mpm.bin.install_dependencies`), 14

`install_fileobj()` (in module `mpm.commands`), 12

M

`main()` (in module `mpm.bin`), 13

`main()` (in module `mpm.bin.install_dependencies`), 14

`mpm` (module), 11

`mpm.bin` (module), 13

`mpm.bin.install_dependencies` (module), 14

`mpm.commands` (module), 11

`mpm.hooks` (module), 13

O

`on_plugin_install()` (in module `mpm.hooks`), 13

P

`parse_args()` (in module `mpm.bin`), 13

`parse_args()` (in module `mpm.bin.install_dependencies`), 14

`pformat_dict()` (in module `mpm`), 11

`plugin_request()` (in module `mpm.commands`), 13

S

`search()` (in module `mpm.commands`), 13

U

`uninstall()` (in module `mpm.commands`), 13

V

`validate_args()` (in module `mpm.bin`), 13

`validate_args()` (in module `mpm.bin.install_dependencies`), 14

module