MetaDoc
Quick start guide

Bjørnar Grip Fjær

July 5, 2010

1 Introduction

This guide is ment to provide a quick introduction to using MetaDoc and the MetaDoc API (MAPI). For more complete information about MetaDoc, please refer to the MetaDoc Documentation [1].

2 Retrieving the MetaDoc client

Before you start you need a version of the MetaDoc client. You can download the latest version from the MetaDoc GitHub repository [2].

Extract the client to the wanted destination.

3 Setting up folders

The MetaDoc client uses two system folders, namely `/var/log/mapi/` and `/var/cache/mapi/` for log files and cache files, respectively. Make sure to create these folders, and make sure the user running the MetaDoc client has access to **read** and **write** to these folders.

4 Configuring the MetaDoc client

The MetaDoc client uses a configuration file called `metadoc.conf` in the same folder as the client itself. You can create a sample configuration file by running `main.py` without any parameters.

The configuration file contains the following settings:

host URI of a server responding to the MetaDoc server API [1].

key Location of file containing the private key for the client.

cert Location of file containing the certificate for the client key.

trailing_slash Whether the server uses a trailing slash on URIs, should be set to True for now.

valid Defaults to False to avoid the script from running with default configuration. Should be set to True or removed once properly configured.

Please note that file locations **must** be given as an absolute path.

The certificate file referenced by the **cert** key in the configuration must be availible at the server and related to your client properly, otherwise the server will refuse the connection.

5 Using the information

Once the actions detailed above have been performed, the MetaDoc client is ready to send and recieve information. However, it has to know what to do with the information it recieves, and how to gather information to send.

`doc/examples/` contains examples of customized functions that produce a shadow file from user data recieved, project user file from project data and project quota file from recieved allocation data. It also contains an example of a command line tool for adding events.

Please refer to the MetaDoc documentation [1] for more information on customizing the MetaDoc client.

6 Setting up crontab

The MetaDoc client can be set up as a crontab in order to syncronize sites with the server. Below, some examples of using the client in crontab is shown. The information passed between the client and server depends on the handles passed on script execution. See the MetaDoc documentation for a detailed explanation of each handle [1].

```
0  0  *  *  *  /path/to/metadoc/client/main.py -ecs -q -l error
```

Sends event, configuration and software data to the server at midnight every night. Makes sure the script executes quietly, passing no output to `stdout`, only to `stderr` if the script execution fails. Only logs error messages.

```
0  0  *  *  0  /path/to/metadoc/client/main.py -upa -q -l warning
```

Fetches user, project and allocation data once a week. Makes sure the script executes quietly and logs warning messages.

7 Logging

The MetaDoc client logs to `/var/log/mapi/`, log files are named `metadoc.client.YYYY-mm-dd.log`. The amount of information logged depends on the log level passed to **-l** on script execution. The possible log levels are **debug**, **info**, **warning**, **error** and **critical**. When **-l** is not passed, or passed with a log level not availible, the level is set to **warning**. See the MetaDoc documentation for more information on logging [1].

References

[1] *MetaDoc Documentation*, Bjørnar Grip Fjær, `http://bjornar.me/metadoc/doc.pdf`

[2] *MetaDoc Downloads*, `http://github.com/bjornarg/metadoc/downloads`