MSS - Mission Support System Documentation

Release 1.8.1

M. Rautenhaus, J. Ungermann, J.-U. Grooß, T. Breuer, R. Bauer

2019-07-18

Contents

CHAPTER 1

Mission Support System Usage Guidelines

Welcome to the Mission Support System software for planning atmospheric research flights. This document is intended to point you into the right direction in order to get the software working on your computer.

Please read the reference documentation

> Rautenhaus, M., Bauer, G., and Doernbrack, A.: A web service based tool to plan atmospheric research flights, Geosci. Model Dev., 5, 55-71, doi:10.5194/gmd-5-55-2012, 2012.

and the paper's Supplement (which includes a tutorial) before using the application. The documents are available at:

> http://www.geosci-model-dev.net/5/55/2012/gmd-5-55-2012.pdf http://www.geosci-model-dev.net/5/55/2012/gmd-5-55-2012-supplement.pdf

For copyright information, please see the files NOTICE and LICENSE, located in the same directory as this README file.

> When using this software, please be so kind and acknowledge its use by citing the above mentioned reference documentation in publications, presentations, reports, etc. that you create. Thank you very much.

1.1 Mission Support Dependencies

1.1.1 Mission Support System (MSS)

license APACHE-2.0, see LICENSE for details.

> When using this software, please acknowledge its use by citing the reference documentation in any publication, presentation, report, etc. you create:
>
> Rautenhaus, M., Bauer, G., and Doernbrack, A.: A web service based tool to plan atmospheric research flights, Geosci. Model Dev., 5, 55-71, doi:10.5194/gmd-5-55-2012, 2012.
>
> Thank you.

Below is a list of software packages that are used in the MSS or from which parts are used in the MSS (see the LICENSE file for the terms and conditions that apply to these components):

OWSLib

Copyright (c) 2006, Ancient World Mapping Center

Package for working with OGC map, feature, and coverage services. Obtained from SVN (http://svn.gispython.org/svn/gispy/OWSLib/trunk), revision 1672, on 2010/08/11.

NOTE: The file "wms.py" has been modified for the MSS. Changes are marked with "(mss)". We renamed it to "ogcwms.py" (2017-04-28)

We also did a PEP8 review of the ogcwms.py and adopted it to the recent 0.14 version https://pypi.python.org/pypi/OWSLib/0.14.0

NETCDF4-PYTHON

Copyright: 2008 by Jeffrey Whitaker

mslib/netCDF4tools.py contains substantial parts taken from the netcdf4-python library – Python/numpy interface to netCDF

https://code.google.com/p/netcdf4-python/

1.2 Installation

Current Releases of mss is based on *python 3*.

mss-1.7.6 was the last version with python2* support.

1.2.1 Install distributed version by conda

Anaconda provides an enterprise-ready data analytics platform that empowers companies to adopt a modern open data science analytics architecture.

The Mission Support Web Map Service (mss) is available as anaconda package on the channel.

conda-forge

The conda-forge packages are based on defaults and other conda-forge packages. This channel conda-forge has builds for osx-64, linux-64, win-64

The conda-forge github organization uses various automated continuos integration build processes.

conda-forge channel

Please add the channel conda-forge to your defaults:

```
$ conda config --add channels conda-forge
$ conda config --add channels defaults
```

The last channel added gets on top of the list. This gives the order: First search in default packages then in conda-forge.

You must install mss into a new environment to ensure the most recent versions for dependencies (On the Anaconda Prompt on Windows, you have to leave out the 'source' here and below).

```
$ conda create -n mssenv mss python=3
$ conda activate mssenv
$ mss
```

For updating an existing MSS installation to the current version, it is best to install it into a new environment. If an existing environment shall be updated, it is important to update all packages in this environment.

```
$ conda activate mssenv
$ conda update --all
$ mss
```

For further details *Configuration of mss*

1.2.2 Server based installation using miniconda

For a wms server setup you may want to have a dedicated user running mswms or the apache2 wsgi script. We suggest to create a mss user.

- create a mss user on your system
- login as mss user
- create a *src* directory in /home/mss
- cd src
- get miniconda for Python 3
- set execute bit on install script
- execute script, enable environment in .bashrc
- login again or export PATH="/home/mss/miniconda3/bin:$PATH"
- python –version should tell Python 3.X.X
- conda install -c conda-forge mss

For a simple test you could start the builtin standalone server by *mswms*. It should tell:

```
serving on http://127.0.0.1:8081
```

Pointing a browser to http://localhost:8081/?service=WMS&request=GetCapabilities&version=1.1.1 shows the generated XML data the mss app will use.

If you want to look on some data, we provide a demo data set by the program *Simulated Data and its configuration*.

For further configuration see *Apache server setup* or *Standalone server setup*.

1.2.3 Installation based on Docker

Since 1.7.4 mss is on the docker hub.

Build settings are based on the stable branch. Our latest is any update in the stable repo.

You can start server and client by loading the image

```
$ xhost +local:docker
$ docker run -ti --rm -e DISPLAY=$DISPLAY -v /tmp/.X11-unix/:/tmp/.X11-unix dreimark/
↪mss:latest  /bin/bash
$ mss &
$ mswms
```

If you want both server and ciient interact

```
$  xhost +local:docker
$  docker run -d --net=host -ti --rm -e DISPLAY=$DISPLAY -v /tmp/.X11-unix/:/tmp/.X11-
↪unix dreimark/mss:latest mss

$ docker run -d --net=host  dreimark/mss:latest:latest
$ curl "http://localhost/?service=WMS&request=GetCapabilities&version=1.1.1"
```

1.3 mss - User Interface

The executable for the user interface application is "**mss**". A short description of how to start the program is given by the –help option. Warnings about CDAT features are due to the NAppy package and can be ignored. The program should open the main window of the user interface, from which you can open further windows, including top view, side view and so on.

Configuration for the user interface is located in "mss_settings.json". In this file, you can specify, for instance, the default WMS URLs for the WMS client, the size of the local image cache (the MSUI caches retrieved WMS images to accelerate repeated retrievals), or the predefined locations that the user can select in the table view.

A few options influencing the appearance of the displayed plots and flight tracks (colours etc.) can be set directly in the user interface (top view and side view).

1.3.1 Configuration of mss

For storage capabilities mss uses the *pyfilesystem <http://pyfilesystem2.readthedocs.io>* approach. The default data dir is predefined as a directory: *~/mssdata* which is the same as *osfs://~/mssdata*.

PyFilesystem can open a filesystem via an *FS URL*, which is similar to a URL you might enter in to a browser. FS URLs are useful if you want to specify a filesystem dynamically, such as in a conf file or from the command line.

Format

FS URLs are formatted in the following way:

```
<protocol>://<username>:<password>@<resource>
```

The components are as follows:

- `<protocol>` Identifies the type of filesystem to create. e.g. `osfs`, `ftp`.

- <username> Optional username.
- <password> Optional password.
- <resource> A *resource*, which may be a domain, path, or both.

Here are a few examples:

```
osfs://~/projects
osfs://c://system32
ftp://ftp.example.org/pub
mem://
ftp://[user[:password]@]host[:port]/[directory]
webdav://[user[:password]@]host[:port]/[directory]
ssh://[user[:password]@]host[:port]/[directory]
```

Settings file

This file includes configuration settings central to the entire Mission Support User Interface (mss). Among others, define

- available map projections
- vertical section interpolation options
- the lists of predefined web service URLs
- predefined waypoints for the table view

If you don't have a mss_settings.json then default configuration is in place.

Store this mss_settings.json in a path, e.g. "$HOME/.config/mss"

The file could be loaded by the File Load Configuration dialog or by the environment variable MSS_SETTINGS pointing to your mss_settings.json.

/$HOME/.config/mss/mss_settings.json

```
{
    "data_dir": "~/mssdata",

    "layout": {
       "topview": [963, 702],
       "sideview": [913, 557],
       "tableview": [1236, 424],
       "immutable": false
     },

    "locations": {
        "EDMO": [48.08, 11.28],
        "Hannover": [52.37, 9.74],
        "Hamburg": [53.55, 9.99],
        "Juelich": [50.92, 6.36],
        "Leipzig": [51.34, 12.37],
        "Muenchen": [48.14, 11.57],
        "Stuttgart": [48.78, 9.18],
        "Wien": [48.20833, 16.373064],
        "Zugspitze": [47.42, 10.98],
        "Kiruna": [67.821, 20.336],
```

(continues on next page)

(continued from previous page)

```
        "Ny-Alesund": [78.928, 11.986]
    },

    "predefined_map_sections": {
        "01 Europe (cyl)": {"CRS": "EPSG:4326",
                            "map": {"llcrnrlon": -15.0, "llcrnrlat": 35.0,
                                    "urcrnrlon": 30.0, "urcrnrlat": 65.0}},
        "02 Germany (cyl)": {"CRS": "EPSG:4326",
                             "map": {"llcrnrlon": 5.0, "llcrnrlat": 45.0,
                                     "urcrnrlon": 15.0, "urcrnrlat": 57.0}},
        "03 Global (cyl)": {"CRS": "EPSG:4326",
                            "map": {"llcrnrlon": -180.0, "llcrnrlat": -90.0,
                                    "urcrnrlon": 180.0, "urcrnrlat": 90.0}},
        "04 Shannon (stereo)": {"CRS": "EPSG:77752350",
                                "map": {"llcrnrlon": -45.0, "llcrnrlat": 22.0,
                                        "urcrnrlon": 45.0, "urcrnrlat": 63.0}},
        "05 Northern Hemisphere (stereo)": {"CRS": "EPSG:77790000",
                                            "map": {"llcrnrlon": -45.0, "llcrnrlat":␣
↪0.0,
                                                    "urcrnrlon": 135.0, "urcrnrlat":␣
↪0.0}},
        "06 Southern Hemisphere (stereo)": {"CRS": "EPSG:77890000",
                                            "map": {"llcrnrlon": 45.0, "llcrnrlat": 0.
↪0,
                                                    "urcrnrlon": -135.0, "urcrnrlat":␣
↪0.0}}
    },

    "crs_to_mpl_basemap_table" : {
        "EPSG:4326": {"basemap": {"projection": "cyl"},
                      "bbox": "latlon"},
        "EPSG:77790000": {"basemap": {"projection": "stere", "lat_0": 90.0, "lon_0":␣
↪0.0},
                          "bbox": "latlon"}
    },

   "new_flighttrack_template": ["Kiruna", "Ny-Alesund"],
   "new_flighttrack_flightlevel": 250,

   "default_WMS": ["http://www.your-server.de/forecasts"],
   "default_VSEC_WMS": ["http://www.your-server.de/forecasts"],
   "WMS_login": {
                  "http://www.your-server.de/forecasts" : ["youruser", "yourpassword"]
                 },
   "num_interpolation_points": 201,
   "num_labels": 10,

   "filepicker_default": "default"
}
```

Flight track import/export

As the planned flight track has to be quickly communicated to different parties having different desired file formats, MSS supports a simple plugin system for exporting planned flights and importing changed files back in addition to the main FTML format. These filters may be accessed from the File menu of the Main Window.

MSS currently offers several import/export filters in the mslib.plugins.io module, which may serve as an example for the definition of own plugins. The CSV plugin is enabled by default. Enabling the experimental FliteStar text import plugin would require those lines in the UI settings file:

```
"import_plugins": {
    "FliteStar": ["txt", "mslib.plugins.io.flitestar", "load_from_flitestar"]
},
```

The dictionary entry defines the name of the filter in the File menu. The list specifies in this order the extension, the python module implementing the function, and finally the function name. The module may be placed in any location of the PYTHONPATH or into the configuration directory path.

An exemplary test file format that can be ex- and imported may be activated by:

```
"import_plugins": {
    "Text": ["txt", "mslib.plugins.io.text", "load_from_txt"]
},
"export_plugins": {
    "Text": ["txt", "mslib.plugins.io.text", "save_to_txt"]
},
```

The given plugins demonstrate, how additional plugins may be implemented. Please be advised that several attributes of the waypoints are automatically computed by MSS (for example all time and performance data) and will be overwritten after reading back the file.

Web Proxy

If you are in an area with a very low bandwith you may consider to use a squid web proxy and add those lines in your mss_settings pointing to the proxy server.

```
  "proxies": {
    "http": "http://yoursquidproxy:3128",
    "https": "http://yoursquidproxy:3128"
  }
```

Trajectory Tool

For accessing trajectory data based on NASA AMES format you need the nappy python module installed and can configure this view by

```
  "traj_nas_lon_identifier": ["GPS LON", "LONGITUDE"],
  "traj_nas_lat_identifier": ["GPS LAT", "LATITUDE"],
  "traj_nas_p_identifier": ["STATIC PRESSURE"]
```

Caching

For changing the default cache directory and behaviour to a named directory you can use these parameters. If you use shared directories you may have to solve access rights.

```
    "wms_cache": "/tmp/.cache/.mss/msui/wms_cache",
    "wms_cache_max_size_bytes": 20971520,
    "wms_cache_max_age_seconds": 432000,
```

Docking Widgets Configurations

Performance

MSS may also roughly estimate the fuel consumption and thus range of the aircraft neglecting weather conditions given a proper configuration file specifying the aircraft performance. Such a file may be loaded using the 'performance settings' button in Table View. The aircraft performance is specified using tables given in the JSON format. A basic configuration looks like the following file:

```
{
    "name": "DUMMY",
    "takeoff_weight": 91000,
    "fuel": 35000,
    "climb": [[0.00, 0.0, 0.0, 0.0, 0.0]],
    "descent": [[0.00, 0.0, 0.0, 0.0, 0.0]],
    "cruise": [[0.00, 0.00, 400, 2900.00]],
    "_comment1": "_comment fields are just for self-documentation!",
    "_comment2": "takeoff_weight: weight(lbs)",
    "_comment3": "fuel: weight(lbs)",
    "_comment4": "climb/descent: weight(lbs), altitude(ft), duration(min),␣
↪distance(nm), fuel(lbs)",
    "_comment5": "cruise: weight(lbs), altitude(ft), total air speed(nm/h), fuel␣
↪flow(lbs/h)"
}
```

This example file assumes a constant speed of 400 nm/h and a constant fuel consumption of 2900 lbs/h irrespective of flight level changes. The aircraft weight and available fuel are also given, but these may also be adjusted in the GUI after loading.

The columns of the cruise table are aircraft weight (lbs), aircraft altitude (feet), speed (nm/h), and fuel consumption (lbs/h). MSS bilinearily interpolates in aircraft weight and altitude and extrapolates assuming a constant behaviour outside the given data. The climb table specifies the aircraft performance when climbing up from 0 feet altitude, while the descent table specifies the behaviour when descending down to 0 feet altitude. The column headers are aircraft weight (lbs), aircraft altitude (feet), time spent (minutes), distance required (nm), and fuel consumed (lbs). To compute the required data for a flight level change, a bilinear interpolation in the table for current aircraft weight and the two involved altitudes is performed and the difference of the resulting value is used in the calculation.

Satellite Track Docking Widget

The TopView has a docking widget allowing the visualisation of satellite tracks. A web site to generate the data for such tracks is operated by NASA. The data can be downloaded as ASCII file that can be open by the docking widget. An example file is located at `docs/samples/satellite_tracks/satellite_predictor.txt`.

KML Overlay Docking Widget

The TopView has a docking widget that allows the visualization of KML files on top of the map. This feature currently does not support all features of KML, for example, external resources such as images are not supported. Some example KML files are located at `docs/samples/kml/line.kml` and `docs/samples/kml/folder.kml`.

Remote sensing Docking Widget

The TopView has a docking widget that allows the visualization of remote sensing related features. It may visualize the position of tangent points of limb sounders and can overlay the flight path with colours according to the relative position of sun, moon, and some planets (to either avoid or seek out alignments). Upon first starting the widget, it is thus necessary to download astronomic positional data (see here for more information). This is automatically performed by the skyfield python package, retrieving the data from public sources of JPL and other US services. The data is stored in the MSS configuration directory and may need to update irregularily.

File picker dialogue

MSS supports the use of a general file picker to access locations on remote machines facilitating collaboration on campaigns. To enable this feature apply

```
"filepicker_default": "fs",
```

to your configuration file. The allowed values are "qt" for QT-based dialogues, "fs" for fsfile_picker-based dialogues supporting remote locations, or "default" for the default dialogues. The default is currently identical to "qt", but may change in upcoming releases. The dialogues may also be configured more fine grained with the parameters of 'filepicker_flightrack for saving and loading flight tracks, 'filepicker_matplotlib' for saving figures, "filepicker_config" for loading json configuration files, "filepicker_performance" for loading performance data, "filepicker_satellitetrack" for loading satellite track data, and "filepicker_trajectories" for loading data in the trajectory tool. Additionally, the dialogue type may be configured for each export/import plugin individually by a fourth, optional, parameter.

data dir

With using the "filepicker_default": "fs" setting you can enable any implemented pyfilesystem2 fs url. Additional to the builtin fs urls we have added optional the webdavfs and sshfs service.

With setting the option "filepicker_default": "default" you can only access local storages.

```
"data_dir": "~/mssdata",
```

1.3.2 Example WMS Server

Some public accessible WMS Servers

- http://osmwms.itc-halle.de/maps/osmfree
- http://ows.terrestris.de/osm/service
- https://firms.modaps.eosdis.nasa.gov/wms

1.4 wms - Web Map Service

Once installation and configuration are complete, you can start the Web Map Service application (provided you have forecast data to visualise). The file "mswms" is an executable Python script starting up a Flask HTTP server with the WMS WSGI module. A short description of how to start the program is given by the –help option. The file "wms.wsgi" is intended to be used with an Apache web server installation.

We have a single method to use data for ECMWF, CLaMS, GWFC, EMAC, METEOSAT implemented. The data have to use for their parameters the CF attribute standard_name. A new method should be able to deal with any CF conforming file following a couple of simple additional requirements.

Per configuration you could register horizontal (*register_horizontal_layers*) or vertical layers (*register_vertical_layers*), give a basemap table for EPSG mapping (*epsg_to_mpl_basemap_table*) and at all how to access the data.

A few notes:

- The Flask WMS currently cannot run multithreaded (Apache does support multiple processes). This is due to that a single instance of the WSGI application handler class MSS_WMSResponse can create only one plot at a time (otherwise you get messed up plots when simultaneous requests occur). In the current implementation, only a single instance is passed to Flask (to do all the initialisation work only once. To extend the software to handle simultaneous requests would probably involve creating a "factory" of MSS_WMSResponse instances.. If you want to do this, check if/how Flask handles "worker" factories.
- Creating the capabilities document can take very long (> 1 min) if the forecast data files have to be read for the first time (the WMS program opens all files and tries to determine the available data and elevation ranges). A GetCapabilities request should return a document within a few seconds.
- A typical bottleneck for plot generation is when the forecast data files are located on a different computer than the WMS server. In this case, large amounts of data have to be transferred over the network. Hence, when possible, try to make sure the WMS runs on the same computer on which the input data files are hosted.

1.4.1 Configuration file of the wms server

Configuration for the Mission Support System Web Map Service (wms).

In this module the data organisation structure of the available forecast data is described. The class NWPDataAccess is subclassed for each data type in the system and provides methods to determine which file needs to be accessed for a given variable and time. The classes also provide methods to query the available initialisation times for a given variable, and the available valid times for a variable and a given initialisation time. As the latter methods need to open the NetCDF data files to determine the contained time values, a caching system is used to avoid re-opening already searched files.

Replace the name INSTANCE in the following examples by your service name.

The configuration file have to become added to the /home/mss/INSTANCE/config directory

/home/mss/config/mss_wms_settings.py

```
# -*- coding: utf-8 -*-
"""

    mss_wms_settings
    ~~~~~~~~~~~~~~~~

    Configuration module for programs accessing data on the MSS server.

    This file is part of mss.

    :copyright: 2008-2014 Deutsches Zentrum fuer Luft- und Raumfahrt e.V.
    :copyright: 2011-2014 Marc Rautenhaus
    :copyright: Copyright 2016-2019 by the mss team, see AUTHORS.
    :license: APACHE-2.0, see LICENSE for details.

    Licensed under the Apache License, Version 2.0 (the "License");
```

(continues on next page)

(continued from previous page)

```
    you may not use this file except in compliance with the License.
    You may obtain a copy of the License at

       http://www.apache.org/licenses/LICENSE-2.0

    Unless required by applicable law or agreed to in writing, software
    distributed under the License is distributed on an "AS IS" BASIS,
    WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
    See the License for the specific language governing permissions and
    limitations under the License.
"""

import os
import sys

# on a productions system you may want to limit the amout of tracebacks to 0
# sys.tracebacklimit = 0

# Configuration of Python's code search path
# If you already have set up the PYTHONPATH environment variable for the
# stuff you see below, you don't need to do a1) and a2).

# a1) Path of the directory where the mss code package is located.
# sys.path.insert(0, '/home/mss/INSTANCE/miniconda3/lib/python3.X/site-
↪packages')

# a2) Path of the directory where mss_wms_settings.py is located
MSSCONFIGPATH = os.path.abspath(os.path.normpath(os.path.dirname(sys.
↪argv[0])))
sys.path.insert(0, MSSCONFIGPATH)
os.chdir(MSSCONFIGPATH)

import mslib.mswms.dataaccess
from mslib.mswms import mpl_hsec_styles
from mslib.mswms import mpl_vsec_styles
import mslib.mswms


#
# SETTINGS                                    ###
#


# Paths to data directories. Process callables (the functions that are
# executed by the dispatcher system) can access these paths, hence
# simply define the key/value pairs that are required.
datapath = {
    "ecmwf": "/path/to/data/mss/grid/ecmwf/netcdf",
    "emac": "/path/to/data/mss/grid/emac/netcdf",
    "meteosat": "/path/to/data/mss/grid/meteosat/netcdf",
    "camsglobal": "/path/to/data/mss/grid/camsglobal/netcdf",
}


# Objects that let the user query the filename in which a particular
# variable can be found. Objects are instances of subclasses of␣
↪NWPDataAccess,
```

(continues on next page)

(continued from previous page)

```
# which provides the methods fc_filename() and full_fc_path().

data = {
    "ecmwf_NH_LL05": mslib.mswms.dataaccess.DefaultDataAccess(datapath["ecmwf
↪"], "NH_LL05"),
    #    "ecmwf_EUR_LL015": mslib.mswms.dataaccess.
↪DefaultDataAccess(datapath["ecmwf"], "EUR_LL015"),
    #    "meteosat_EUR_LL05": mslib.mswms.dataaccess.
↪DefaultDataAccess(datapath["meteosat"], "EUR_LL05"),
    #    "emac_GLOBAL_LL1125": mslib.mswms.dataaccess.
↪DefaultDataAccess(datapath["emac"]),
    #    "CAMSglb": mslib.mswms.dataaccess.DefaultDataAccess(datapath[
↪"camsglobal"]),

}

#
# HTTP Authentication                                    ###
#

# If you require basic HTTP authentication, set the following variable
# to True. Add usernames in the list "allowed:users". Note that the
# passwords are not specified in plain text but by their md5 digest.
enable_basic_http_authentication = False


#
# Basemap cache                                          ###
#

# Plotting coastlines on horizontal cross-sections requires usually the␣
↪parsing
# of the corresponding databases for each plot.
# A simple caching feature allows to reuse this data from previous plots␣
↪using
# the same bounding box and projection parameters, dramatically speeding up
# the plotting. 'basemap_cache_size' determines hows many sets of coastlines␣
↪shall
# be stored in memory and 'basemap_request_size' determines the length of␣
↪history
# used to determine, which data shall be purged first if the cache exceeds␣
↪its
# maximum size.
basemap_use_cache = False
basemap_request_size = 200
basemap_cache_size = 20

#
# Registration of horizontal layers.                     ###
#

# The following list contains tuples of the format (instance of
# visualisation classes, data set). The visualisation classes are
# defined in mpl_hsec.py and mpl_hsec_styles.py. Add only instances of
# visualisation products for which data files are available. The data
# sets must be defined in mss_config.py. The WMS will only offer
# products registered here.
```

(continues on next page)

(continued from previous page)

```
register_horizontal_layers = [
    # ECMWF standard surface level products.
    (mpl_hsec_styles.MPLBasemapHorizontalSectionStyle, ["ecmwf_NH_LL05"]),
    (mpl_hsec_styles.HS_CloudsStyle_01, ["ecmwf_EUR_LL015", "ecmwf_NH_LL05
↪"]),
    (mpl_hsec_styles.HS_MSLPStyle_01, ["ecmwf_EUR_LL015", "ecmwf_NH_LL05"]),
    (mpl_hsec_styles.HS_SEAStyle_01, ["ecmwf_NH_LL05"]),
    (mpl_hsec_styles.HS_SeaIceStyle_01, ["ecmwf_NH_LL05"]),
    (mpl_hsec_styles.HS_VIProbWCB_Style_01, ["ecmwf_EUR_LL015"]),

    # ECMWF standard pressure level products.
    (mpl_hsec_styles.HS_TemperatureStyle_PL_01, ["ecmwf_EUR_LL015"]),
    (mpl_hsec_styles.HS_GeopotentialWindStyle_PL, ["ecmwf_EUR_LL015"]),
    (mpl_hsec_styles.HS_RelativeHumidityStyle_PL_01, ["ecmwf_EUR_LL015"]),
    (mpl_hsec_styles.HS_EQPTStyle_PL_01, ["ecmwf_EUR_LL015"]),
    (mpl_hsec_styles.HS_WStyle_PL_01, ["ecmwf_EUR_LL015"]),
    (mpl_hsec_styles.HS_DivStyle_PL_01, ["ecmwf_EUR_LL015"]),

    # ECMWF standard model level products.
    (mpl_hsec_styles.HS_TemperatureStyle_ML_01, ["ecmwf_EUR_LL015"]),

    # ECMWF standard potential vorticity products.
    (mpl_hsec_styles.HS_PVTropoStyle_PV_01, ["ecmwf_EUR_LL015"]),

    # EMAC layers.
    # (mpl_hsec_styles.HS_EMAC_TracerStyle_ML_01, ["emac_GLOBAL_LL1125"]),
    # (mpl_hsec_styles.HS_EMAC_TracerStyle_SFC_01, ["emac_GLOBAL_LL1125"]),

    # Meteosat products.
    (mpl_hsec_styles.HS_Meteosat_BT108_01, ["meteosat_EUR_LL05"]),

    # MSS-Chem chemistry forecasts
    #
    # The MSS-Chem project (http://mss-chem.rtfd.io) provides an easy way to
    # download and prepare chemical weather forecasts from a range of␣
↪different
    # CTMs.  The plot style classes are called `HS_MSSChemStyle_XL_YYY_zzzzz`
    # and `VS_MSSChemStyle_XL_YYY_zzzzz` for horizontal and vertical␣
↪sections,
    # i.e., "maps" and "cross-sections", respectively.
    #
    # In the class name, X determines the type of vertical layering of the␣
↪model
    # data, and can be one of
    # - "P" for model data defined on pressure levels
    # - "M" for model data defined on model levels
    # - "A" for model data defined on altitude levels.
    #   CAUTION: In the latter case, the vertical axis in vertical section␣
↪plots
    #   (VS_MSSChemStyle_AL_YYY_zzzz) is only approximate, as the air␣
↪pressure
    #   can only be estimated using the barometric formula.
    #
    # YYY stands for the chemical species to be displayed (in UPPER CASE), e.
↪g.,
    # "NO2".
```

(continues on next page)

(continued from previous page)

```
    #
    # zzzzz determines the type of quantity, and can be one of
    # - mfrac -> mass fraction, kg/kg
    # - mconc -> mass concentration, kg/m3
    # - nfrac -> mole fraction, mol/mol
    # - nconc -> mole_concentration, mol/m3
    #
    # Putting this together, the following class provides a horizontal ("HS")
    # section of model-level ("ML") mass fractions ("mfrac") of nitrogen␣
↪dioxide
    # ("NO2"):

    # (mpl_hsec_styles.HS_MSSChemStyle_ML_NO2_mfrac, ["CAMSglb"]),
]


#
# Registration of vertical layers.                        ###
#

# The same as above, but for vertical cross-sections.

register_vertical_layers = [
    # ECMWF standard vertical section styles.
    (mpl_vsec_styles.VS_CloudsStyle_01, ["ecmwf_EUR_LL015"]),
    (mpl_vsec_styles.VS_HorizontalVelocityStyle_01, ["ecmwf_EUR_LL015"]),
    (mpl_vsec_styles.VS_PotentialVorticityStyle_01, ["ecmwf_EUR_LL015"]),
    (mpl_vsec_styles.VS_ProbabilityOfWCBStyle_01, ["ecmwf_EUR_LL015"]),
    (mpl_vsec_styles.VS_VerticalVelocityStyle_01, ["ecmwf_EUR_LL015"]),
    (mpl_vsec_styles.VS_RelativeHumdityStyle_01, ["ecmwf_EUR_LL015"]),
    (mpl_vsec_styles.VS_SpecificHumdityStyle_01, ["ecmwf_EUR_LL015"]),
    (mpl_vsec_styles.VS_TemperatureStyle_01, ["ecmwf_EUR_LL015"]),

    # EMAC layers.
    # (mpl_vsec_styles.VS_EMACEyja_Style_01, ["emac_GLOBAL_LL1125"]),

    # MSS-Chem chemistry forecasts
    # (mpl_vsec_styles.VS_MSSChemStyle_ML_NO2_mfrac, ["CAMSglb"]),
]


#
# Server settings.                                  ###
#

use_threadpool = False

# xml_template directory is a sub directory of mswms

base_dir = os.path.abspath(os.path.dirname(mslib.mswms.__file__))
xml_template_location = os.path.join(base_dir, "xml_templates")

# get_capabilities.pt
service_name = "OGC:WMS"
service_title = "Mission Support System Web Map Service"
service_abstract = "Your Abstract"
service_contact_person = "Your Name"
```

(continues on next page)

(continued from previous page)

```
service_contact_organisation = "Your Organization"
service_contact_position = "Your Position"
service_address_type = "postal"
service_address = "street"
service_city = "Your City"
service_state_or_province = ""
service_post_code = "12345"
service_country = "Germany"
service_email = "mail@example.com"
service_fees = "none"
service_access_constraints = "This service is intended for research purposes␣
↪only."


#
# EPSG Code Definitions for Matplotlib basemap                ###
#

# In this section you can define how EPSG codes are interpreted in
# terms of Matplotlib basemap parameters. If you require a new EPSG
# code, define it here.

# Table to translate EPSG codes to Matplotlib basemap projection parameters.
# Extend this table to add further EPSG codes.
# Also see: http://external.opengeospatial.org/twiki_public/bin/view/
#                  MetOceanDWG/MetCoordinateReferenceSystemDefinition
epsg_to_mpl_basemap_table = {
    # EPSG:4326, the standard cylindrical lat/lon projection.
    4326: {"projection": "cyl"},

    # Non-standard EPSG codes, specifically defined for MSS pruposes.
    # EPSG:777llLLL, north polar stereographic projections with lat_0=ll and
    # lon_0=LLL.
    77790000: {"projection": "stere", "lat_0": 90., "lon_0": 0.},
    77790010: {"projection": "stere", "lat_0": 90., "lon_0": 10.},
    77790015: {"projection": "stere", "lat_0": 90., "lon_0": 15.},
    77790340: {"projection": "stere", "lat_0": 90., "lon_0": -20.},
    77790105: {"projection": "stere", "lat_0": 90., "lon_0": -105.},

    77890000: {"projection": "spstere", "lat_0": 90., "lon_0": 0.},
    77890010: {"projection": "spstere", "lat_0": 90., "lon_0": 10.},
    77890015: {"projection": "spstere", "lat_0": 90., "lon_0": 15.},
    77890340: {"projection": "spstere", "lat_0": 90., "lon_0": -20.},
    77890105: {"projection": "spstere", "lat_0": 90., "lon_0": -105.}

    # Feel free to add other projections, e.g. a south polar projection
    # EPSG:778llLLL.
}
```

You have to adopt this file to your data.

1.4.2 Standalone server setup

For the standalone server *mswms* you need the path of your mss_wms_settings.py and other configuration files added to the PYTHONPATH. E.g.:

```
export PYTHONPATH=/home/mss/INSTANCE/config
```

For testing your server you can use the *Simulated Data and its configuration*

1.4.3 meteorological data

Data for the MSS server shall be provided in CF-compliant NetCDF format. Several specific data access methods are provided for ECMWF, Meteoc, and several other formats.

The prefered method "DefaultDataAccess" shall supplant most of these, but requires the data to be organised in the fashion described in the following (the others pose mostly the same requirements).

All data files belonging to one "set" shall have a common string in its name that can be used to uniquely identify all files of this set. Each set must share the same time, longitude, and latitude grid. Each set must use the same elevation layers for each type of vertical axis. Different data sets may be used to offer different geographical regions or results of different simulation models.

Each file of a set must contain only one or no vertical axis. If the data is required to be given on multiple vertical axis (such as providing data for horizontal plots on both pressure and theta levels), one (or more separate) file for each vertical axis type must be provided. All files for one axis type shall provide the same levels. If no vertical axis can be identified, it is assumed that the file contains 3-D data (time, lat, lon) such as, e.g., surface pressure or tropopause altitude.

The vertical coordinate variable is identified by the standard_name being one of the following names:

- atmosphere_hybrid_sigma_pressure_coordinate - "ml"
- atmosphere_pressure_coordinate - "pl"
- atmosphere_ertel_potential_vorticity_coordinate - "pv"
- atmosphere_altitude_coordinate - "al"
- atmosphere_potential_temperature_coordinate - "tl"

The two-letter abbreviation is used for brief identification in the plotting routines in addition to the standard_name of the variable to uniquely identify which data shall be used. The data shall be organized with the dimensions in the order of "time", "vertical coordinate", "latitudes", and "longitudes" (This is important to reduce disk access when generating the plots). Data variables are identified by their standard_name, which is expected to be CF compliant. Data variables should contain a "units" attribute that may be used by the plotting routines for checking and/or conversion. Please bear in mind that the vertical axis of all vertical sections is pressure in 'Pa'.

It is assumed that forecast data is given from one initialisation time onward for several time steps into the future. For each file, the init time is determined by the units attribute of the "time" variable. The time variable is identified by its standard_name being "time". The date given after "since" is interpreted as the init time such that the numerical value of "0" were the init time (which need not be present in the file). For example, if the units field of "time" contains "hours since 2012-10-17T12:00:00.000Z", 2012-10-17T12Z would be the init time. Data for different time steps may be contained in one file or split over several ones.

An exemplary header for a file containing ozone on a vertical pressure coordinate and a 3-D tropopause would look as follows:

```
netcdf example_ASIA {
dimensions:
        press = 13 ;
        lat = 51 ;
        lon = 141 ;
        time = 12 ;
```

(continues on next page)

(continued from previous page)

```
variables:
        float press(press) ;
                press:units = "Pa" ;
                press:positive = "down" ;
                press:standard_name = "atmosphere_pressure_coordinate" ;
        float lat(lat) ;
                lat:units = "degrees_north" ;
                lat:standard_name = "latitude" ;
        float lon(lon) ;
                lon:units = "degrees_east" ;
                lon:standard_name = "longitude" ;
        float time(time) ;
                time:units = "hours since 2012-10-17T12:00:00Z" ;
                time:standard_name = "time" ;
        float O3(time, press, lat, lon) ;
                O3:units = "mol/mol" ;
                O3:standard_name = "mole_fraction_of_ozone_in_air" ;
        float tropopause(time, lat, lon) ;
                tropopause:units = "Pa" ;
                tropopause:standard_name = "tropopause_air_pressure" ;
}
```

1.4.4 Apache server setup

Install mod_wsgi

On some distributions an old mod_wsgi is shipped and have to become replaced by a version compatible to the conda environment.

At current state we have to use pip to install mod_wsgi into the INSTANCE environment:

```
# Instal `mod_wsgi`
$ pip install mod_wsgi

# Find the full path to installed `mod_wsgi`
$ which mod_wsgi-express

# Install and register the `mod_wsgi` module with Apache
$ sudo /full/path/to/installed/mod_wsgi-express install-module
```

Setup a /etc/apache2/mods-available/wsgi_express.conf:

```
WSGIPythonHome "/home/mss-demo/miniconda3/envs/demo/"
```

Setup a /etc/apache2/mods-available/wsgi_express.load:

```
LoadModule wsgi_module "/usr/lib/apache2/modules/mod_wsgi-py37.cpython-37m-x86_64-
↪linux-gnu.so"
```

Enable the new module by a2enmod and reload the apache2 server

One Instance

Our examples are based on the following directories located in the home directory of the mss user. INSTANCE is a placeholder for your service name:

```
.
├── INSTANCE
|   ├── config
|   |   └── mss_wms_settings.py
|   |   └── mss_wms_auth.py
|   ├── log
|   |   └── mss_error.log
|   └── wsgi
|       ├── auth.wsgi
|       └── wms.wsgi
├── miniconda3
|   ├── bin
|   ├── conda-bld
|   ├── conda-meta
|   ├── envs
|   |   └── instance
|   ├── etc
|   ├── include
|   ├── lib
|   ├── LICENSE.txt
|   ├── pkgs
|   ├── share
|   ├── ssl
|   └── var
```

Configuration of apache mod_wsgi.conf

One posibility to setup the PYTHONPATH environment variable is by adding it to your mod_wsgi.conf. Alternativly you could add it also to wms.wsgi.

> WSGIPythonPath /home/mss/INSTANCE/config:/home/mss/miniconda3/envs/instance/lib/python3.X/site-packages

By this setting you override the PYTHONPATH environment variable. So you have also to add the site-packes directory of your miniconda or anaconda installation besides the config file path.

If your server hosts different instances by different users you want to setup this path in mss_wms_setting.py.

Configuration of wsgi for wms

You can setup a vhost for this service.

/home/mss/INSTANCE/wsgi/wms.wsgi

```
import os
import sys

sys.path.insert(0, '/home/mss/INSTANCE/config')
sys.path.insert(0, '/home/mss/miniconda3/envs/instance/lib/python3.7/site-
↪packages/mslib')
sys.path.insert(0, '/home/mss/INSTANCE/wsgi')
```

(continues on next page)

(continued from previous page)

```
if os.getenv("PROJ_LIB") is None or os.getenv("PROJ_LIB") == "PROJ_LIB":
    os.environ["PROJ_LIB"] = '/home/mss/miniconda3/envs/instance/share/proj'

sys.stdout = sys.stderr
import logging
logging.basicConfig(stream=sys.stderr)

from mslib.mswms.wms import app as application
```

Configuration of wsgi auth

As long as you have only one instance of the server running you can use this method to restrict access.

To restrict access to your data use this script.

/home/mss/INSTANCE/wsgi/auth.wsgi

```
import sys
sys.path.extend(['/home/mss/INSTANCE/config'])

import mss_wms_auth
import hashlib

def check_password(environ, username, password):
    for u, p in mss_wms_auth.allowed_users:
        if (u == username) and (p == hashlib.md5(password).hexdigest()):
            return True
    return False
```

This needs also a configuration **/home/mss/INSTANCE/config/mss_wms_auth.py** script.

```
# -*- coding: utf-8 -*-
"""

    mss_wms_auth
    ~~~~~~~~~~~~

    Configuration module for authentication to the MSS server.

    This file is part of mss.

    :copyright: 2008-2014 Deutsches Zentrum fuer Luft- und Raumfahrt e.V.
    :copyright: 2011-2014 Marc Rautenhaus
    :copyright: Copyright 2016-2017 by the mss team, see AUTHORS.
    :license: APACHE-2.0, see LICENSE for details.

    Licensed under the Apache License, Version 2.0 (the "License");
    you may not use this file except in compliance with the License.
    You may obtain a copy of the License at

       http://www.apache.org/licenses/LICENSE-2.0

    Unless required by applicable law or agreed to in writing, software
    distributed under the License is distributed on an "AS IS" BASIS,
```

(continues on next page)

(continued from previous page)

```
    WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
    See the License for the specific language governing permissions and
    limitations under the License.
"""

#
# HTTP Authentication                                   ###
#
#
# Use the following code to create a new md5 digest of a password (e.g. in
# ipython):
#     import hashlib; hashlib.md5("my_new_password").hexdigest()
allowed_users = [("mswms", "add_md5_digest_of_PASSWORD_here"),
                 ("add_new_user_here", "add_md5_digest_of_PASSWORD_here")]
```

At the moment you have many different instances with different users or different versions of mss you have to use basic auth of your webserver configuration.

Configuration of your site as vhost

You have to setup a webserver server site configuration file

/etc/apache2/sites-available/mss.yourserver.de.conf

```
<VirtualHost *:80>
    ServerName mss.yourserver.de
    #ServerAlias localwms
    LogLevel debug
    ServerAdmin webmaster@yourserver.de
    ErrorLog /home/mss/log/mss_error.log
    TransferLog /var/log/apache2/mss_access.log
    CustomLog /var/log/apache2/mss_custom.log combined


    #all wsgi scripts are in /home/mss/wsgi
    <Directory /home/mss/INSTANCE/wsgi>
        AuthType Basic
        AuthName "mss"
        AuthBasicProvider wsgi
        WSGIAuthUserScript /home/mss/INSTANCE/wsgi/auth.wsgi
        AuthDigestDomain / http://mss.yourserver.de/
        Require valid-user
    </Directory>

    #alternative without authentication
    #<Directory /home/mss/INSTANCE/wsgi>
    #    Require all granted
    #</Directory>

    WSGIScriptAlias / /home/mss/INSTANCE/wsgi/wms.wsgi
    WSGIDaemonProcess MSS python-home=/home/mss/miniconda3/env/instance/bin␣
↪user=mss group=mss home=/home/mss/INSTANCE/config processes=2 threads=1␣
↪deadlock-timeout=25 display-name=MSS
    # WSGI Options
```

(continues on next page)

(continued from previous page)

```
    # python-home: path where your environment python bin is located
    # home: where your config scripts are located
    # Please see the documentation here : http://code.google.com/p/modwsgi/
↪wiki/ConfigurationDirectives
    WSGIProcessGroup MSS

    # Python Simplified GIL State API ( ISSUE )
    # http://code.google.com/p/modwsgi/wiki/ApplicationIssues
    WSGIApplicationGroup %{GLOBAL}


    DocumentRoot /home/mss/htdocs

    # don't loose time with IP address lookups
    HostnameLookups Off

    # needed for named virtual hosts
    UseCanonicalName Off

    # configures the footer on server-generated documents
    ServerSignature On

</VirtualHost>
```

Enable it with a2ensite mss.yourserver.de.conf

Many Instances

If you want to setup many instances we suggest to use a similiar proxy based configuration

```
<VirtualHost *:80>

ProxyRequests Off
ProxyPreserveHost On
RewriteEngine On
RequestHeader add X-SSL off
RewriteRule ^/demo/(.*) http://127.0.0.1/demo/$1 [P,L]


ServerName proxy_demo.yourserver.de
DocumentRoot /var/www/html
ProxyPreserveHost On
ProxyPass /demo http://127.0.0.1/demo

</VirtualHost>
```

and if you need authentication then use a Location based AuthType Basic

```
<VirtualHost 127.0.0.1:80>

    #<Directory /home/mss/DEMO/wsgi>
    #    Require all granted
    #</Directory>
```

(continues on next page)

(continued from previous page)

```
     <Location /demo>
        AuthType Basic
        AuthName "mss"
        AuthDigestDomain /
        AuthUserFile /home/mss/DEMO/config/apache_users
        <Limit GET>
            Require valid-user
        </Limit>
      </Location>
    # You can define different hostname for the WMS VirtualHost here
    # For example, you can add localwms to your /etc/hosts file along with␣
↪localhost
    # and use the 'localwms' name like in the template here.
    ServerName proxy_demo.yourserver.de
    RemoteIPHeader X-Forwarded-For
    RemoteIPInternalProxy 127.0.0.0/8
    LogLevel error
    ServerAdmin admin@email
    ErrorLog /home/mss/DEMO/log/mss_error.log
    TransferLog /var/log/apache2/mss_access.log
    CustomLog /var/log/apache2/mss_custom.log combined

    # WSGI Options
    # home: Initial working directory of the script, make sure you change it.
    # python-path : Directories to search for Modules, make sure you change␣
↪it as well.
    # Please see the documentation here : http://code.google.com/p/modwsgi/
↪wiki/ConfigurationDirectives
    WSGIScriptAlias /demo /home/mss/DEMO/wsgi/wms.wsgi

    WSGIDaemonProcess MSSDEMO python-home="/home/mss/miniconda3/envs/demo"␣
↪python-path="/home/mss/miniconda3/envs/demo/lib/python3.7/site-packages/"␣
↪home="/home/mss/DEMO/config"  user=mss group=mss processes=2 threads=1␣
↪deadlock-timeout=25 display-name=MSSDEMO

    WSGIProcessGroup MSSDEMO

    # Python Simplified GIL State API ( ISSUE )
    # http://code.google.com/p/modwsgi/wiki/ApplicationIssues
    WSGIApplicationGroup %{GLOBAL}

    # don't loose time with IP address lookups
    HostnameLookups Off

    # needed for named virtual hosts
    UseCanonicalName Off

    # configures the footer on server-generated documents
    ServerSignature On

</VirtualHost>
```

For further informations on apache2 server setup read https://httpd.apache.org/docs/2.4/howto/

1.5 Simulated Data and its configuration

We provide demodata by executing the demodata programm. This creates in your home directory data files and also the needed server configuration files. The program creates 70MB of examples. This script does not overwrite an existing mss_wms_settings.py.

```
mss
├── mss_wms_auth.py
├── mss_wms_settings.py
└── testdata
    ├── 20121017_12_ecmwf_forecast.ALTITUDE_LEVELS.EUR_LL015.036.ml.nc
    ├── 20121017_12_ecmwf_forecast.CC.EUR_LL015.036.ml.nc
    ├── 20121017_12_ecmwf_forecast.CIWC.EUR_LL015.036.ml.nc
    ├── 20121017_12_ecmwf_forecast.CLWC.EUR_LL015.036.ml.nc
    ├── 20121017_12_ecmwf_forecast.EMAC.EUR_LL015.036.ml.nc
    ├── 20121017_12_ecmwf_forecast.P_derived.EUR_LL015.036.ml.nc
    ├── 20121017_12_ecmwf_forecast.PRESSURE_LEVELS.EUR_LL015.036.pl.nc
    ├── 20121017_12_ecmwf_forecast.ProbWCB_LAGRANTO_derived.EUR_LL015.036.ml.nc
    ├── 20121017_12_ecmwf_forecast.ProbWCB_LAGRANTO_derived.EUR_LL015.036.sfc.nc
    ├── 20121017_12_ecmwf_forecast.PV_derived.EUR_LL015.036.ml.nc
    ├── 20121017_12_ecmwf_forecast.PVU.EUR_LL015.036.pv.nc
    ├── 20121017_12_ecmwf_forecast.Q.EUR_LL015.036.ml.nc
    ├── 20121017_12_ecmwf_forecast.SEA.EUR_LL015.036.sfc.nc
    ├── 20121017_12_ecmwf_forecast.SFC.EUR_LL015.036.sfc.nc
    ├── 20121017_12_ecmwf_forecast.T.EUR_LL015.036.ml.nc
    ├── 20121017_12_ecmwf_forecast.THETA_LEVELS.EUR_LL015.036.tl.nc
    ├── 20121017_12_ecmwf_forecast.U.EUR_LL015.036.ml.nc
    ├── 20121017_12_ecmwf_forecast.V.EUR_LL015.036.ml.nc
    └── 20121017_12_ecmwf_forecast.W.EUR_LL015.036.ml.nc
```

Before starting the standalone server you should add the path where the server config is to your python path. e.g.

```
$ export PYTHONPATH=~/mss
```

Detailed server configuration *mss_wms_settings.py* for this demodata

```
# -*- coding: utf-8 -*-
"""

    mss_wms_settings
    ~~~~~~~~~~~~~~~~

    Configuration module for programs accessing data on the MSS server.

    This file is part of mss.

    :copyright: 2008-2014 Deutsches Zentrum fuer Luft- und Raumfahrt e.V.
    :copyright: 2011-2014 Marc Rautenhaus
    :copyright: Copyright 2017 Jens-Uwe Grooss, Joern Ungermann, Reimar Bauer
    :copyright: Copyright 2017-2019 by the mss team, see AUTHORS.
    :license: APACHE-2.0, see LICENSE for details.

    Licensed under the Apache License, Version 2.0 (the "License");
    you may not use this file except in compliance with the License.
    You may obtain a copy of the License at

       http://www.apache.org/licenses/LICENSE-2.0
```

(continues on next page)

(continued from previous page)

```
    Unless required by applicable law or agreed to in writing, software
    distributed under the License is distributed on an "AS IS" BASIS,
    WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
    See the License for the specific language governing permissions and
    limitations under the License.
"""

import os
import sys

# on a productions system you may want to limit the amout of tracebacks to 0
# sys.tracebacklimit = 0

# Configuration of Python's code search path
# If you already have set up the PYTHONPATH environment variable for the
# stuff you see below, you don't need to do a1) and a2).

# a1) Path of the directory where the mss code package is located.
# sys.path.insert(0, '/home/mss/INSTANCE/miniconda3/lib/python3.X/site-
↪packages')

# a2) Path of the directory where mss_wms_settings.py is located
#MSSCONFIGPATH = os.path.abspath(os.path.normpath(os.path.dirname(sys.
↪argv[0])))
#sys.path.insert(0, MSSCONFIGPATH)
#os.chdir(MSSCONFIGPATH)

import mslib.mswms.dataaccess
from mslib.mswms import mpl_hsec_styles
from mslib.mswms import mpl_vsec_styles
import mslib.mswms


# Configuration for mss_wms_settings accessing data on the MSS server.
# This is the data organisation structure of demodata.


#service_name = "OGC:WMS"
#service_title = "Mission Support System Web Map Service"
#service_abstract = "Your Abstract"
#service_contact_person = "Your Name"
#service_contact_organisation = "Your Organization"
#service_address_type = "postal"
#service_address = "street"
#service_city = "Your City"
#service_state_or_province = ""
#service_post_code = "12345"
#service_country = "Germany"
#service_fees = "none"
#service_access_constraints = "This service is intended for research␣
↪purposes only."


#
# HTTP Authentication
#
```

(continues on next page)

(continued from previous page)

```
# If you require basic HTTP authentication, set the following variable
# to True. Add usernames in the list "allowed:users". Note that the
# passwords are not specified in plain text but by their md5 digest.
#enable_basic_http_authentication = False


#xml_template directory is a sub directory of mswms
#base_dir = os.path.abspath(os.path.dirname(mslib.mswms.__file__))
#xml_template_location = os.path.join(base_dir, "xml_templates")

_datapath = r"/path/to/data/mss/testdata"

data = {
    "ecmwf_EUR_LL015": mslib.mswms.dataaccess.DefaultDataAccess(_datapath,
↪"EUR_LL015"),
}

epsg_to_mpl_basemap_table = {
    # EPSG:4326, the standard cylindrical lat/lon projection.
    4326: {"projection": "cyl"}
}

basemap_use_cache = True

#
# Registration of horizontal layers.
#

# The following list contains tuples of the format (instance of
# visualisation classes, data set). The visualisation classes are
# defined in mpl_hsec.py and mpl_hsec_styles.py. Add only instances of
# visualisation products for which data files are available. The data
# sets must be defined in mss_config.py. The WMS will only offer
# products registered here.
register_horizontal_layers = None
if mpl_hsec_styles is not None:
    register_horizontal_layers = [
        # ECMWF standard pressure level products.
        (mpl_hsec_styles.HS_TemperatureStyle_PL_01, ["ecmwf_EUR_LL015"]),
        (mpl_hsec_styles.HS_GeopotentialWindStyle_PL, ["ecmwf_EUR_LL015"]),
        (mpl_hsec_styles.HS_RelativeHumidityStyle_PL_01, ["ecmwf_EUR_LL015
↪"]),
        (mpl_hsec_styles.HS_EQPTStyle_PL_01, ["ecmwf_EUR_LL015"]),
        (mpl_hsec_styles.HS_WStyle_PL_01, ["ecmwf_EUR_LL015"]),
        (mpl_hsec_styles.HS_DivStyle_PL_01, ["ecmwf_EUR_LL015"]),
    ]


#
# Registration of vertical layers.
#
# The same as above, but for vertical cross-sections.
register_vertical_layers = None
if mpl_vsec_styles is not None:
    register_vertical_layers = [
        # ECMWF standard vertical section styles.
        (mpl_vsec_styles.VS_CloudsStyle_01, ["ecmwf_EUR_LL015"]),
```

(continues on next page)

(continued from previous page)

```
        (mpl_vsec_styles.VS_HorizontalVelocityStyle_01, ["ecmwf_EUR_LL015"]),
        (mpl_vsec_styles.VS_VerticalVelocityStyle_01, ["ecmwf_EUR_LL015"]),
        (mpl_vsec_styles.VS_RelativeHumdityStyle_01, ["ecmwf_EUR_LL015"]),
        (mpl_vsec_styles.VS_SpecificHumdityStyle_01, ["ecmwf_EUR_LL015"]),
        (mpl_vsec_styles.VS_TemperatureStyle_01, ["ecmwf_EUR_LL015"])
    ]
```

For setting authentication see *mss_wms_auth.py*

```
# -*- coding: utf-8 -*-
"""

    mss_wms_auth
    ~~~~~~~~~~~~

    Configuration module for authentication to the MSS server.

    This file is part of mss.

    :copyright: 2008-2014 Deutsches Zentrum fuer Luft- und Raumfahrt e.V.
    :copyright: 2011-2014 Marc Rautenhaus
    :copyright: Copyright 2016-2017 by the mss team, see AUTHORS.
    :license: APACHE-2.0, see LICENSE for details.

    Licensed under the Apache License, Version 2.0 (the "License");
    you may not use this file except in compliance with the License.
    You may obtain a copy of the License at

       http://www.apache.org/licenses/LICENSE-2.0

    Unless required by applicable law or agreed to in writing, software
    distributed under the License is distributed on an "AS IS" BASIS,
    WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
    See the License for the specific language governing permissions and
    limitations under the License.
"""

#
# HTTP Authentication                                   ###
#
#
# Use the following code to create a new md5 digest of a password (e.g. in
# ipython):
#     import hashlib; hashlib.md5("my_new_password").hexdigest()
allowed_users = [("mswms", "add_md5_digest_of_PASSWORD_here"),
                 ("add_new_user_here", "add_md5_digest_of_PASSWORD_here")]
```

1.6 Development

This chapter will get you started with MSS development.

MSS is written in Python.

Once a stable release is published we do only bug fixes in stable and release regulary new minor versions. If a fix needs a API change or it is likly more a new feature you have to make a pull request to the develop branch. Documentation

of changes is done by using our issue tracker.

When it is ready the developer version becomes the next stable.

1.6.1 Style guide

We generally follow pep8, with 120 columns instead of 79.

1.6.2 Output and Logging

When writing logger calls, always use correct log level (debug only for debugging, info for informative messages, warning for warnings, error for errors, critical for critical errors/states).

1.6.3 Building a development environment

If you want to contribute make a fork on bitbucket of mss. For building you have to use the conda recipe localy and install in a new environment.

Some of used packages are in the conda-forge channel located, so we have to add this channel to the default:

```
$ conda config --add channels conda-forge
```

Or add the channel by an editor to the .condarc config file:

```
$ more ~/.condarc
channels:
- defaults
- conda-forge
```

using a local meta.yaml recipe:

```
$ git clone https://bitbucket.org/yourfork/mss.git
$ cd mss
$ conda create -n mssdev python=3
$ conda activate mssdev
$ conda build .
$ conda install --use-local mss
$ mkdir "$HOME/.config/mss"
$ # cp mss_settings.json.sample to "$HOME/.config/mss/mss_settings.json"
$ conda remove mss
```

alternative get the whole package first:

```
$ conda create -n mssdev mss
$ conda activate mssdev
$ conda remove mss
```

Compare versions used in the meta.yaml between stable and develop branch and apply needed changes.

Add the path of your local cloned mss directory to $PYTHONPATH.

Add developer packages for running tests, activate your env and run:

```
$ conda install --file requirements.d/development.txt
```

On linux install *xvfb* from your linux package manager. This is used to run tests on a virtual display.

1.6.4 Setup demodata

Simulated Data and its configuration is provided by executing:

```
$(mssdev) python mslib/mswms/demodata.py
```

To use this data add the mss_wms_settings.py in your python path:

```
$(mssdev) :~/PycharmProjects/mss
$(mssdev) export PYTHONPATH="`pwd`:~/mss"
$(mssdev) python mslib/mswms/mswms.py
```

1.6.5 Running tests

We have implemented demodata as data base for testing. On first call of pytest a set of demodata becomes stored in a /tmp/mss* folder. If you have installed gitpython a postfix of the revision head is added.

```
$ pytest
```

Use the -v option to get a verbose result. By the -k option you could select one test to execute only.

A pep8 only test is done by *py.test –pep8 -m pep8* or *pytest –pep8 -m pep8*

Instead of running a ibrary module as a script by the -m option you may also use the pytest command.

```
$ pytest --cov mslib
```

This plugin produces a coverage report, example:

```
----------- coverage: platform linux, python 3.7.3-final-0 -----------
Name                                      Stmts   Miss Branch BrPart  Cover
-------------------------------------------------------------------------------
mslib/__init__.py                             2      0      0      0   100%
mslib/msui/__init__.py                       23      0      0      0   100%
mslib/msui/aircrafts.py                      52      1      8      1    97%
mslib/msui/constants.py                      12      2      4      2    75%
mslib/msui/flighttrack.py                   383    117    141     16    66%
```

Profiling can be done by e.g.:

```
$ python -m cProfile  -s time ./mslib/mswms/demodata.py > profile.txt
```

example:

```
/!\ existing server config: "mss_wms_settings.py" for demodata not overwritten!


To use this setup you need the mss_wms_settings.py in your python path e.g.
export PYTHONPATH=~/mss
         398119 function calls (389340 primitive calls) in 0.834 seconds

   Ordered by: internal time

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
       19    0.124    0.007    0.496    0.026 demodata.py:912(generate_file)
       19    0.099    0.005    0.099    0.005 {method 'close' of 'netCDF4._netCDF4.
↪Dataset' objects}
```

Setup mss_settings.json

On default all tests use default configuration defined in mslib.msui.MissionSupportSystemDefaultConfig. If you want to overwrite this setup and try out a special configuration add an mss_settings.json file to the testings base dir in your tmp directory.

1.6.6 Building the docs with Sphinx

The documentation (in reStructuredText format, .rst) is in docs/.

To build the html version of it, you need to have sphinx installed:

```
cd docs/
make html
```

Then point a web browser at docs/_build/html/index.html.

1.6.7 Update local stable branch

If you don't have a stable branch, create one first or change to that branch:

```
git checkout [-b] stable
git pull git@bitbucket.org:wxmetvis/mss.git stable
git push
```

1.6.8 Merging stable into develop

Bug fixes we have done in stable we need to merge regulary into develop too:

```
git checkout stable
git pull
git checkout develop
git pull
git merge stable
```

1.6.9 Creating a new release

- make sure all issues for this milestone are closed or moved to the next milestone
- update CHANGES.rst, based on git log
- check version number of upcoming release in CHANGES.rst
- verify that version.py, meta.yaml, MANIFEST.in and setup.py are complete
- for a new stable release merge from develop to stable
- tag the release:

  ```
  git tag -s -m "tagged/signed release X.Y.Z" X.Y.Z
  git push origin X.Y.Z
  ```

- create a release on anaconda conda-forge
- announce on:

- Mailing list
- Twitter (follow @ReimarBauer for these tweets)

1.7 Contributors

- Andreas Hilboll <hilboll@uni-bremen.de>
- Anveshan Lal <anveshanrx8@gmail.com>
- Christian Rolf <c.rolf@fz-juelich.de>
- Isabell Krisch <isabellkrisch@gmail.com>
- Jens-Uwe Grooß <j.-u.grooss@fz-juelich.de>
- Jörn Ungermann <j.ungermann@fz-juelich.de>
- Marc Rautenhaus <wxmetvis@posteo.de>
- Reimar Bauer <rb.proj@gmail.com>
- Shivashis Padhi <shivashispadhi@gmail.com>
- Thomas Breuer <t.breuer@fz-juelich.de>

1.8 License

(continues on next page)

(continued from previous page)

```
      including but not limited to software source code, documentation
      source, and configuration files.

      "Object" form shall mean any form resulting from mechanical
      transformation or translation of a Source form, including but
      not limited to compiled object code, generated documentation,
      and conversions to other media types.

      "Work" shall mean the work of authorship, whether in Source or
      Object form, made available under the License, as indicated by a
      copyright notice that is included in or attached to the work
      (an example is provided in the Appendix below).

      "Derivative Works" shall mean any work, whether in Source or Object
      form, that is based on (or derived from) the Work and for which the
      editorial revisions, annotations, elaborations, or other modifications
      represent, as a whole, an original work of authorship. For the purposes
      of this License, Derivative Works shall not include works that remain
      separable from, or merely link (or bind by name) to the interfaces of,
      the Work and Derivative Works thereof.

      "Contribution" shall mean any work of authorship, including
      the original version of the Work and any modifications or additions
      to that Work or Derivative Works thereof, that is intentionally
      submitted to Licensor for inclusion in the Work by the copyright owner
      or by an individual or Legal Entity authorized to submit on behalf of
      the copyright owner. For the purposes of this definition, "submitted"
      means any form of electronic, verbal, or written communication sent
      to the Licensor or its representatives, including but not limited to
      communication on electronic mailing lists, source code control systems,
      and issue tracking systems that are managed by, or on behalf of, the
      Licensor for the purpose of discussing and improving the Work, but
      excluding communication that is conspicuously marked or otherwise
      designated in writing by the copyright owner as "Not a Contribution."

      "Contributor" shall mean Licensor and any individual or Legal Entity
      on behalf of whom a Contribution has been received by Licensor and
      subsequently incorporated within the Work.

   2. Grant of Copyright License. Subject to the terms and conditions of
      this License, each Contributor hereby grants to You a perpetual,
      worldwide, non-exclusive, no-charge, royalty-free, irrevocable
      copyright license to reproduce, prepare Derivative Works of,
      publicly display, publicly perform, sublicense, and distribute the
      Work and such Derivative Works in Source or Object form.

   3. Grant of Patent License. Subject to the terms and conditions of
      this License, each Contributor hereby grants to You a perpetual,
      worldwide, non-exclusive, no-charge, royalty-free, irrevocable
      (except as stated in this section) patent license to make, have made,
      use, offer to sell, sell, import, and otherwise transfer the Work,
      where such license applies only to those patent claims licensable
      by such Contributor that are necessarily infringed by their
      Contribution(s) alone or by combination of their Contribution(s)
      with the Work to which such Contribution(s) was submitted. If You
      institute patent litigation against any entity (including a
      cross-claim or counterclaim in a lawsuit) alleging that the Work
```

(continues on next page)

(continued from previous page)

```
   or a Contribution incorporated within the Work constitutes direct
   or contributory patent infringement, then any patent licenses
   granted to You under this License for that Work shall terminate
   as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the
   Work or Derivative Works thereof in any medium, with or without
   modifications, and in Source or Object form, provided that You
   meet the following conditions:

   (a) You must give any other recipients of the Work or
       Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices
       stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works
       that You distribute, all copyright, patent, trademark, and
       attribution notices from the Source form of the Work,
       excluding those notices that do not pertain to any part of
       the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its
       distribution, then any Derivative Works that You distribute must
       include a readable copy of the attribution notices contained
       within such NOTICE file, excluding those notices that do not
       pertain to any part of the Derivative Works, in at least one
       of the following places: within a NOTICE text file distributed
       as part of the Derivative Works; within the Source form or
       documentation, if provided along with the Derivative Works; or,
       within a display generated by the Derivative Works, if and
       wherever such third-party notices normally appear. The contents
       of the NOTICE file are for informational purposes only and
       do not modify the License. You may add Your own attribution
       notices within Derivative Works that You distribute, alongside
       or as an addendum to the NOTICE text from the Work, provided
       that such additional attribution notices cannot be construed
       as modifying the License.

   You may add Your own copyright statement to Your modifications and
   may provide additional or different license terms and conditions
   for use, reproduction, or distribution of Your modifications, or
   for any such Derivative Works as a whole, provided Your use,
   reproduction, and distribution of the Work otherwise complies with
   the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise,
   any Contribution intentionally submitted for inclusion in the Work
   by You to the Licensor shall be under the terms and conditions of
   this License, without any additional terms or conditions.
   Notwithstanding the above, nothing herein shall supersede or modify
   the terms of any separate license agreement you may have executed
   with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade
   names, trademarks, service marks, or product names of the Licensor,
   except as required for reasonable and customary use in describing the
```

(continues on next page)

(continued from previous page)

```
   origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or
   agreed to in writing, Licensor provides the Work (and each
   Contributor provides its Contributions) on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
   implied, including, without limitation, any warranties or conditions
   of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A
   PARTICULAR PURPOSE. You are solely responsible for determining the
   appropriateness of using or redistributing the Work and assume any
   risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,
   whether in tort (including negligence), contract, or otherwise,
   unless required by applicable law (such as deliberate and grossly
   negligent acts) or agreed to in writing, shall any Contributor be
   liable to You for damages, including any direct, indirect, special,
   incidental, or consequential damages of any character arising as a
   result of this License or out of the use or inability to use the
   Work (including but not limited to damages for loss of goodwill,
   work stoppage, computer failure or malfunction, or any and all
   other commercial damages or losses), even if such Contributor
   has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing
   the Work or Derivative Works thereof, You may choose to offer,
   and charge a fee for, acceptance of support, warranty, indemnity,
   or other liability obligations and/or rights consistent with this
   License. However, in accepting such obligations, You may act only
   on Your own behalf and on Your sole responsibility, not on behalf
   of any other Contributor, and only if You agree to indemnify,
   defend, and hold each Contributor harmless for any liability
   incurred by, or claims asserted against, such Contributor by reason
   of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

   To apply the Apache License to your work, attach the following
   boilerplate notice, with the fields enclosed by brackets "[]"
   replaced with your own identifying information. (Don't include
   the brackets!)  The text should be enclosed in the appropriate
   comment syntax for the file format. We also recommend that a
   file or class name and description of purpose be included on the
   same "printed page" as the copyright notice for easier
   identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
```

(continues on next page)

(continued from previous page)

```
   distributed under the License is distributed on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
   See the License for the specific language governing permissions and
   limitations under the License.



********************************************************************************
********************************************************************************

MISSION SUPPORT SYSTEM THIRD PARTY COMPONENTS:

The Mission Support System includes a number of components with
separate copyright notices and license terms. Your use of the source
code for the these components is subject to the terms and
conditions of the following licenses.

OWSLib
======

Copyright (c) 2006, Ancient World Mapping Center
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

    * Redistributions of source code must retain the above copyright
      notice, this list of conditions and the following disclaimer.
    * Redistributions in binary form must reproduce the above copyright
      notice, this list of conditions and the following disclaimer in the
      documentation and/or other materials provided with the distribution.
    * Neither the name of the University of North Carolina nor the names of
      its contributors may be used to endorse or promote products derived
      from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.


netcdf4-python (netCDF4tools.py in the mslib directory)
=======================================================

copyright: 2008 by Jeffrey Whitaker.

Permission to use, copy, modify, and distribute this software and
its documentation for any purpose and without fee is hereby granted,
provided that the above copyright notice appear in all copies and that
both the copyright notice and this permission notice appear in
```

(continues on next page)

(continued from previous page)

```
supporting documentation.

THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE,
INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO
EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, INDIRECT OR
CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF
USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR
OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR
PERFORMANCE OF THIS SOFTWARE.
```

CHAPTER 2

Indices and tables

- genindex
- modindex
- search