



ESO / Sampo

PyMidas

Software User Manual

Document ref:	ESO-PyMidas-SUM
Version:	1.0
Authors:	Otto Solin
Acceptor:	Richard Hook
Released:	10 November 2005



	ESO / Sampo PyMidas Software User Manual	REF : ESO-PyMidas-SUM ISSUE : 1.0 DATE : 10.11.2005
---	---	---

TABLE OF CONTENTS

1.	INTRODUCTION	3
1.1	Purpose & Note on Support	3
1.2	Intended readers	3
1.3	Definitions, acronyms and abbreviations	3
1.4	References	4
1.4.1	Applicable documents.....	4
1.4.2	Reference documents	4
1.5	Document overview	4
2.	INSTALLATION INSTRUCTIONS	5
2.1	Requirements	5
2.2	Installation as an RPM package (only for Fedora Core 2/3 and Enterprise Linux 4 with Python 2.3)	5
2.3	Installing as a tar package	6
2.3.1	With root access.....	6
2.3.2	Without root access.....	7
3.	USING PYMIDAS	8
3.1	Starting PyMidas.....	8
3.2	PyMidas syntax	8
3.3	Extra commands.....	9
3.4	Python scripts	11
3.5	Other Python-based systems	11

	ESO / Sampo PyMidas Software User Manual	REF : ESO-PyMidas-SUM ISSUE : 1.0 DATE : 10.11.2005
---	---	---

1. INTRODUCTION

1.1 Purpose & Note on Support

This document is the user manual for the **PyMidas** software developed by the **Sampo** project. It is available online at: http://www.eso.org/sampo/pymidas/PyMidas_SW_User_Manual.pdf

The **Sampo** project is a three-year project looking at the future needs of the ESO user community in the areas of data reduction and analysis. **Sampo** will include several pilot projects to assess different approaches and technologies that will be needed to cope with data from ESO facilities during the next decade and beyond. **PyMidas** is the first **Sampo** pilot project.

PyMidas is an interface between Python (the dominant modern scripting language in astronomy) and MIDAS, the major ESO legacy general purpose data processing system. **PyMidas** will allow a user to exploit both the rich legacy of MIDAS software and the power of Python scripting in a unified interactive environment. **PyMidas** also allows the usage of other Python-based astronomical analysis systems such as PyRAF.

Important note: PyMidas, like other Sampo projects, is a pilot project to investigate possibilities and try things out. There is no formal support from ESO.


1.2 Intended readers

The intended readers of this document are the installers and users of PyMidas.

1.3 Definitions, acronyms and abbreviations

ADASS	Astronomical Data Analysis Software Systems conference
CSC	CSC Scientific Computing Ltd
Deliverable	See [AD1]
ESO	The European Southern Observatory
ESO/DMD	The Data Management and Operations Division (DMD) of ESO
ESO-MIDAS	Software system that provides general tools for image processing and data reduction with emphasis on astronomical applications
FAAG	The Finnish Astronomical Advisory Group
IRAF	Image Reduction and Analysis Facility
ON3.6	Opticon Network 3.6
PyMidas	Name of the SW product of the first pilot study of the Sampo project.
PyRAF	A new command language for running IRAF tasks. Based on the Python scripting language
SAC	Science Advisory Group
Sampo	The Finnish in-kind contribution to the entrance fee to ESO
STScI	Space Telescope Science Institute
TEKES	National Technology Agency of Finland

See also [AD1] for additional acronyms.

	ESO / Sampo PyMidas Software User Manual	REF : ESO-PyMidas-SUM ISSUE : 1.0 DATE : 10.11.2005
---	---	---

1.4 References

The documents below are available at the Sampo TWiki (username and password required)
<http://www.astro.helsinki.fi/sampo/twiki/bin/view.cgi/Main/WebHome>

1.4.1 Applicable documents

- [AD-1] Sampo PyMidas Project Plan, ver. 1.05, date: 14.03.2005
- [AD-2] White Paper (Draft)
- [AD-3] PyMidas Software Requirements Document (SRS) ESO-PyMidas-SRS 1.02, 07.04.2005.

1.4.2 Reference documents


- [RD-1] Python Style Guide (To Be Written in parallel with the User Manual and ADD)
- [RD-2] Integration Test Plan (To Be Written in parallel with the ADD)

1.5 Document overview

The document structure is as follows:

Chapter 2 contains the installation instructions for the PyMidas software.

Chapter 3 presents usage of PyMidas with examples.

	ESO / Sampo PyMidas Software User Manual	REF : ESO-PyMidas-SUM ISSUE : 1.0 DATE : 10.11.2005
---	---	---

2. INSTALLATION INSTRUCTIONS

There are three different ways to install PyMidas: as an RPM package (this requires root access) or as tar package with or without root access. The latter (tar package without root access) is normally preferred, particularly for a private or test installation.

Note that PyMidas is expected to be bundled with MIDAS itself, starting with a release of MIDAS in Autumn 2005. It will also be distributed as part of the ESO Scisoft bundle (<http://www.eso.org/scisoft>).

Note also that the filenames given below are examples only and will change as newer versions become available. Installers are recommended to check the web pages for the most recent instructions and download files.

2.1 Requirements

PyMidas requires following supporting packages. If you do not have them installed we suggest you follow the order in which they are listed below

- ESO-MIDAS 05SEP-p11.1 or later (for installation see below)
- Readline (the platform must support readline)
- Python - v2.3 or later (with the readline module enabled)
- Distutils (<http://www.python.org/sigs/distutils-sig/download.html>)

Some platforms have most of these packages already installed in their system directories. To test whether your Python installation has all the modules which are needed, start Python and try to import them (e.g. `import readline`). If you don't get an `ImportError`, this means that readline and distutils are already installed on your system. Distutils is normally part of Python development package.

2.2 Installation as an RPM package (only for Fedora Core 2/3 and Enterprise Linux 4 with Python 2.3)

Installing PyMidas as an RPM package has been successfully tested in the environment mentioned in the title above. Installing the RPM package requires root access.

- 1) PyMidas cannot be installed unless MIDAS version 05SEP-p11.1 or later is available. So if needed download this unofficial package of MIDAS

<http://www.eso.org/sampo/pymidas/midas-local-05SEP-p11.1.i386.rpm>

Next install MIDAS. Open a new shell and login as root.


```
> cd your download directory
> su
> rpm -i midas-local-05SEP-p11.1.i386.rpm
```

- 2) Download PyMidas

<http://www.eso.org/sampo/pymidas/pymidas-1.0.3-1.i386.rpm>

- 3) Install PyMidas. If not already done in step 1) open a new shell and login as root

```
> cd your download directory
> su
```

	ESO / Sampo PyMidas Software User Manual	REF : ESO-PyMidas-SUM ISSUE : 1.0 DATE : 10.11.2005
---	---	---

```
> rpm -i pymidas-1.0-3.i386.rpm
```

- 4) Run PyMidas with the command 'pymidas' in the new shell you opened in step 1) or 3).

2.3 Installing as a tar package

2.3.1 With root access

- 1) PyMidas cannot be installed on a system unless MIDAS version 05SEP-p11.1 or later is available. So if necessary download this unofficial package of MIDAS from

<http://www.eso.org/sampo/pymidas/midas-05SEP-p11.1.tar.gz>

- 2) Download PyMidas from

<http://www.eso.org/sampo/pymidas/pymidas-1.0.3.tar.gz>

- 3) Login as root

```
> su
```

Set the following environment variables

```
- MIDASHOME=/usr/local/midas or wherever MIDAS is installed
- MIDVERS=05SEPp11.1
- LD_LIBRARY_PATH = $MIDASHOME/$MIDVERS/lib
```

Example: to set an environment variable in the tcsh shell, type the following command

```
> setenv LD_LIBRARY_PATH $MIDASHOME/$MIDVERS/lib
```

- 4) Follow applicable parts of MIDAS installation instructions

<http://www.eso.org/projects/esomidas/doc/install/installunix/installunix.html>

- 5) Unpack the installation package and login as root

```
> tar -zxvf pymidas-1.0.3.tar.gz
> su
> cd pymidas-1.0.3
```



- 6) For extra installation options type

```
> python setup.py -help
```

Default installaton locations are

- o /usr/bin/pymidas --> python PyMidas startup script
- o <default python site-packages>/pymidas --> site packages

- 7) Install PyMidas in the default directory

 	ESO / Sampo PyMidas Software User Manual	REF : ESO-PyMidas-SUM ISSUE : 1.0 DATE : 10.11.2005
---	---	---

```
> python setup.py install
```

- 8) Run PyMidas

```
> pymidas
```

2.3.2 Without root access

- 1) PyMidas cannot be installed in system unless MIDAS version 05SEP-p11.1 or later is available. So if necessary download the unofficial version of MIDAS from

<http://www.eso.org/sampo/pymidas/midas-05SEP-p11.1.tar.gz>

- 2) Download PyMidas

<http://www.eso.org/sampo/pymidas/pymidas-1.0.3.tar.gz>

- 3) Set the following environment variables

- o MIDASHOME=/home/xxxx/midas or any place MIDAS is installed
- o MIDVERS=05SEPp11.1
- o LD_LIBRARY_PATH = \$MIDASHOME/\$MIDVERS/lib

Example: to set an environment variable in tcsh shell, type the following command

```
> setenv LD_LIBRARY_PATH $MIDASHOME/$MIDVERS/lib
```

- 4) Follow the applicable parts of MIDAS installation instructions

<http://www.eso.org/projects/esomidas/doc/install/installunix/installunix.html>

- 5) Unpack the installation package

```
> tar -zxvf pymidas-1.0.3.tar.gz
> cd pymidas-1.0.3
```

- 6) For extra installation options type

```
> python setup.py -help
```

- 7) Install PyMidas in your own directory instead of default directory


```
> python setup.py install --local=/home/xxxx/mydir
```

- 8) Run PyMidas

```
> setenv PYTHONPATH /mydir
> /home/xxxx/mydir/pymidas/pymidas
```

Note that the syntax has two pymidas levels. So if mydir=pymidas you run pymidas with

```
> /home/xxxx/pymidas/pymidas/pymidas
```

	ESO / Sampo PyMidas Software User Manual	REF : ESO-PyMidas-SUM ISSUE : 1.0 DATE : 10.11.2005
---	---	---

3. USING PYMIDAS

3.1 Starting PyMidas

PyMidas is started by simply typing 'pymidas' in your shell (and in the correct directory if the installation was done as tar package without root access, see chapter 2). PyMidas can be started also with options that you can view by typing 'pymidas -h' in your shell.

If you already have PyMidas installed but it fails to start check your environment variables (e.g. LD_LIBRARY_PATH and PYTHONPATH) against those mentioned in chapter 2 depending on the type of installation used.

3.2 PyMidas syntax

PyMidas has three ways of invoking MIDAS commands: a new syntax which is compatible with Python usage, a “mimic mode” which emulates the standard MIDAS syntax and a mode most suited to writing Python scripts to use MIDAS commands.

In normal use PyMidas uses a new syntax for MIDAS commands. In this syntax the character '/' is omitted and first letter of the extension following the character '/' is in upper case. The length of the extension is four letters for all commands. Example: the MIDAS command 'LOAD/IMAG myimage' becomes 'loadImag myimage'.

Secondly PyMidas has a mimic mode allowing usage of the original MIDAS syntax from the PyMidas prompt.

Mimic mode may be turned on with

```
PyMidas> .mimic 1
```

And then MIDAS commands may be executed using the standard syntax, for example

```
Midas 003> load/image myimage
```

Mimic mode may be switched off with

```
PyMidas> .mimic 0
```


The default when PyMidas is started is for mimic to be off. Finally MIDAS commands may be accessed from a Python script, and mixed easily with other Python packages using the following syntax:

```
PyMidas> midas.loadImag('myimage')
```

And this can also be used to execute any MIDAS command using pure MIDAS syntax as follows:

```
PyMidas> midas.do('load/image myimage')
```

More information about Python scripting with PyMidas is given below.

	ESO / Sampo PyMidas Software User Manual	REF : ESO-PyMidas-SUM ISSUE : 1.0 DATE : 10.11.2005
---	---	---

There are also a few executive commands, by typing

```
PyMidas> .help
```

you will get information.

By typing

```
PyMidas> .info
```

you will get a summary of general information on PyMidas.

PyMidas allows the results of an operation to be placed in a Python variable, more specifically a list of character strings. For example

```
myresult = midas.readTabl(mytable)
```

will write the text output from this command into a Python list. This information can then be extracted and used as required.

3.3 Extra commands

To ease Python scripting with PyMidas a few extra commands are available.

Command `getPix` gets and prints the value of a pixel in an image

```
midas.getPix(<image>,<pixel x-pos>,<pixel y-pos>)
```

Example

```
PyMidas> midas.getPix('sombrero.fits',5,8)
```

This will result in

```
line =      8, pixel =      5
      224.0000
```

Command `putPix` gives a value to a pixel in an image and opens the image

```
midas.putPix(<image>,<pixel x-pos>,<pixel y-pos>,value)
```

Example


```
PyMidas> midas.putPix('sombrero.fits',5,8,225)
```

The command `putKeyw` stores values into a MIDAS keyword in an easier way than MIDAS command `WRITE/KEYWORD`.

```
PyMidas> midas.putKeyw(<name>,value1,value2,...)
```

Example that creates keyword named `testKw` containing values 1.3 and 2.4.

```
PyMidas> midas.putKeyw('testKw',1.3,2.4)
```

	ESO / Sampo PyMidas Software User Manual	REF : ESO-PyMidas-SUM ISSUE : 1.0 DATE : 10.11.2005
---	---	---

Note that it is normally better to use Python variables and objects to hold intermediate values when writing PyMidas scripts, rather than MIDAS keywords.

Command `getDesc` gets and prints the value of a header keyword

```
midas.getDesc(<file>,<descriptor>)
```

Example

```
PyMidas> midas.getDesc('sombbrero.fits','IDENT')
```

This will result in


```
frame: sombrero.fits (data = R4, format = FITS)  
IDENT:                SOMBRERO NGC 4594
```

Command `putDesc` creates a header keyword and gives a value to it

```
midas.putDesc(<file>,<descriptor>,<value>)
```

Example

```
PyMidas> midas.putDesc('sombbrero.fits','TEST','mytest')
```

	ESO / Sampo PyMidas Software User Manual	REF : ESO-PyMidas-SUM ISSUE : 1.0 DATE : 10.11.2005
---	---	---

3.4 Python scripts

PyMidas allows scripting in Python to invoke MIDAS commands. This is best illustrated with some simple examples.

Example

Write the following simple script `testExec.py`.

```
midas.putKeyword('testKw',1.3,2.4)
midas.readKeyw('testKw')
```

Then type the following at the PyMidas prompt
PyMidas> `execfile("testExec.py")`

This will result in

```
keyword: TESTKW           type: real      no_elems: 2
          1.2             2.4
```

3.5 Other Python-based systems

PyMidas allows flexible mixing with other Python-based systems, in particular access to IRAF through PyRAF.

To import PyRAF type

```
PyMidas> from pyraf import iraf
```

Make sure PyRAF is correctly installed and available in your Python environment. It is usually also necessary to have set up an IRAF environment using the `mkiraf` command, and hence created and possibly customised a `login.cl` file, before starting PyMidas and loading the PyRAF package.

A Simple Example that uses PyRAF.

Write the following script `testFunc.py`.

```
from pymidas import midas
from pyraf import iraf

def test():
    image='sombbrero.fits'
    # sombrero as an example, use whatever you have

    result=midas.statistImag(image)
    max=float(result[15])

    iraf.imstat(image)
    iraf.imarith(image,'/',max,'qqq.fits')
    iraf.imstat('qqq.fits')
```

Then, at the PyMidas prompt, type

```
PyMidas> import testFunc
PyMidas> testFunc.test()
```

and note that this is a different syntax from the previous example.

This will result in

#	IMAGE	NPIX	MEAN	STDDEV	MIN	MAX
	sombrero.fits	250000	273.2	98.27	186.	1023.
	qqq.fits	250000	0.267	0.09606	0.1818	1.

If you modify your script during the same PyMidas session type instead of 'import testFunc'

```
PyMidas> reload(testFunc)
```

NOTE: if you create a background MIDAS process in a Python session, using 'from pymidas import midas', it can be terminated with the command 'midas.bye()' at the Python prompt.

Further examples of Python scripts using PyMidas and PyRAF are available, along with other information, from:

<http://www.eso.org/sampo/pymidas>