IOTA: Iowa Optical Telescope Automation

## Introduction

IOTA is a collection of Python scripts and routines for controlling and automating telescope mounts, focusers, cameras, and filter wheels. Mounts and focusers are controlled via ASCOM drivers. The ASCOM standard defines a programming interface for various astronomical devices, and most equipment vendors supply ASCOM drivers with their hardware. Camera control is done via the MaxIm DL application, which supports a wide variety of CCD cameras and filter wheels. Since IOTA communicates with hardware through these widely-supported interfaces, the same code can be used to control a variety of telescope equipment without requiring any changes to the IOTA code.

## Dependencies

The following items need to be installed:

**Python 2.7 (32-bit)**<https://www.python.org/downloads/>  
The IOTA code is written against Python 2 rather than Python 3, and Python 2.7 is the most up-to-date of the 2.x versions as of this writing. The 32-bit version is required in order to communicate with various 32-bit applications, such as MaxIm DL.

**PyEphem (for Python 2.7)**  
<http://rhodesmill.org/pyephem/>  
PyEphem is a library that implements a variety of astronomical calculations. Features include applying precession/nutation calculations to RA/Dec coordinates and determining the positions of solar system bodies. PyEphem is based upon the libastro code from OCAAS/Talon, which was originally developed at the University of Iowa by Elwood Downey and was still used to automate the Rigel telescope through 2015. PyEphem supports target descriptions in the ephem database format, which makes it easier to integrate existing Talon scheduling tools with IOTA code.

**PyWin32 (for Python 2.7)**<http://sourceforge.net/projects/pywin32/files/pywin32/Build%20219/pywin32-219.win32-py2.7.exe/download>  
The programming interface for ASCOM drivers and MaxIm DL is based on a technology called COM (Component Object Model), which allows code written in one programming language to be easily called from another programming language. The PyWin32 library allows Python to communicate with COM servers .

**PyFITS (for Python 2.7)**<http://www.stsci.edu/institute/software_hardware/pyfits/Download>  
Used for working with FITS files.

**NumPy (for Python 2.7)**<https://sourceforge.net/projects/numpy/files/NumPy/1.10.0/>  
A dependency of PyFITS

**ASCOM Platform 6**<http://ascom-standards.org/>  
The ASCOM Platform installer provides a number of tools that are necessary for communicating with ASCOM drivers, such as the PlaneWave Focuser driver.

**MaxIm DL Pro**<http://www.cyanogen.com/maximdl_dl.php>

MaxIm DL is an application that allows for control of a wide variety of cameras and filter wheels.

**TheSkyX**  
<http://www.bisque.com/sc/>  
*Required only when controlling the Paramount ME*  
TheSkyX is used to communicate with the Paramount ME mount. It can optionally be used to control other ASCOM-compatible mounts such as the Mathis fork mount, but it is not required in these cases.

**ASCOM2XMount Adapter**  
<http://www.bisque.com/sc/media/p/106120.aspx>  
*Required only when controlling the Paramount ME*  
TheSkyX supplies a proprietary COM-based programming interface for controlling TheSky-managed mounts. The ASCOM2XMount adapter provides an ASCOM driver interface that converts ASCOM commands to TheSkyX commands, allowing TheSky-controlled mounts like the Paramount to operate just like any other ASCOM-compatible mount from IOTA’s point of view.

## Scripts

### slew\_grid.py

Generates a grid of RA-Dec points, slews the mount to each point, waits for some period of time, and optionally takes an image at each target to verify mount motion. A variety of configuration values allow for changing the size of the grid, whether or not to save images at each grid point, etc.

## Generating new identity key files for SSH login

Run **puttygen.exe**, located under the **putty** directory of the IOTA distribution.

Move the mouse over the window, as instructed

Click **Save private key** and save the file to (for example) **IOTA\config\iota\_gemini.ppk**

Under **Key comment**, enter something like: **IOTA login from Gemini computer (iota\_gemini.ppk)**

Log in to talon@deimos. Edit **~/.ssh/authorized\_keys**, and append the contents of the **Public key for pasting into OpenSSH authorized\_keys file** section from puttygen to the end of the **authorized\_keys** file.

## Building Talon Tools for Windows

Install Cygwin with the default packages, plus:

* gcc-core
* make

Check out talon source using subversions:

svn checkout svn://svn.skynet.unc.edu/talon/trunk talon

Compile:

cd talon/libastro

make libastro.a

cd ../libfits

make libfits.a

cd ../libmisc

[edit makefile and set ENABLE\_(FLI|SBIG|APOGEE) to “no”]

make libmisc.a

cd ../libfs

make libfs.a

cd ../libwcs

make libwcs.a

cd ../tools/wcs

[edit makefile and reorder library linking order]

Make