PlaneWave Interface 4 API

Introduction

PlaneWave Interface 4 (PWI4) is a software package that can control a variety of PlaneWave equipment, including L-series direct drive mounts and the PW1000 1-meter telescope system. These hardware devices natively provide a fairly basic set of capabilities, such as reading the current position of a motor and commanding a desired velocity. PWI4 extends these capabilities by performing astronomical coordinate transformations, measuring and applying pointing model corrections, calculating field rotation effects, following trajectories for a variety of targets including satellites, and much more.

User applications will generally communicate with PlaneWave equipment using PWI4 as an intermediary.

Interfaces

It is currently possible for applications to communicate with PWI4 in two ways:

1. ASCOM drivers (Windows-only)
2. PWI4-hosted HTTP server

ASCOM drivers

ASCOM provides a standard driver model for allowing Windows-based astronomy applications to communicate with astronomical devices, such as mounts, focusers, rotators, and cameras. A wide variety of applications can use ASCOM-compatible devices, including TheSkyX, MaxIm DL, FocusMax, ACP, CCDAutoPilot, Sequence Generator Pro, and more. The PWI4 ASCOM drivers allow all of these applications, along with any ASCOM-compatible applications developed in the future, to use PlaneWave equipment out of the box.

The ASCOM drivers are in fact fairly simple wrappers around the HTTP interface discussed below. Because ASCOM focuses on capabilities that are fairly common across all astronomical equipment, the PWI4 ASCOM drivers expose only a subset of the full capabilities provided by PWI4. For custom-developed applications, it may make more sense to use the HTTP/TCP interfaces directly.

For more details on working with ASCOM drivers, refer to https://ascom-standards.org/. Please contact PlaneWave Instruments for sample code that illustrates how to talk to the PWI4 ASCOM drivers.

HTTP server

When PWI4 is running, it hosts an HTTP (web) server that can be used to issue commands to PWI4 and read the status of the system. Essentially every programming language in common use is able to make HTTP requests, either using a built-in construct (e.g. urllib in Python, HttpURLConnection in Java, WebRequest in .NET) or a third-party library/tool (e.g. libcurl for C, curl for shell scripts). Users can

experiment with commands simply by entering URLs in a web browser, and can create a custom user interface to their telescope using HTML and Javascript.

HTTP Server request/response format

By default, the HTTP server is hosted on port 8220.

A request to PWI4 will consist of a URL with the following format:

http://host:port/subsystem/command?param1=value1&param2=value2&...

host: The hostname or IP address of the machine running PWI4. This will often be **localhost** for the common case where the client program is running on the same machine as PWI4.

port: The TCP port number that the PWI4 HTTP server is listening on. By default this is **8220**.

/subsystem/command: The command that is being sent to PWI4. In many cases both a subsystem and a command are specified (for example, **/mount/stop** and **/focuser/goto**), but in some cases no subsystem is specified (for example, **/status**).

param1=value1: For commands that take parameters, one or more parameters can be specified as name=value pairs. For example, the **/mount/gotoradec2000** command takes parameters named **ra** and **dec**, and each parameter takes a numeric value.

As a basic example, the following URL retrieves the status from a copy of PWI4 running on the local machine:

http://localhost:8220/status

Here is an example of a request with a two-part command (specifying both the device and the action) that takes parameters:

http://localhost:8220/mount/gotoradec2000?ra=10.5&dec=78.321

For most commands, the response consists of a series of keyword=value pairs separated by newlines (ASCII character 0x0A, commonly represented as “\n”). A sample response is shown below:

```
site.latitude_degs=33.8703888888889
site.longitude_degs=-118.247166666667
site.height_meters=50
site.lst_hours=1.93332583543415
mount.is_connected=true
mount.geometry=0
mount.ra_apparent_hours=1.93031065506901
mount.dec_apparent_degs=-11.1462797189469
mount.ra_j2000_hours=1.91505628146335
mount.dec_j2000_degs=-11.2348569497659
mount.azimuth_degs=180.062736897424
mount.altitude_degs=44.9999794781428
mount.is_slewing=false
mount.is_tracking=true
mount.field_angle_degs=0.0530766291438186
mount.field_angle_rate_degs_per_sec=0.00490314245167322
mount.axis0_is_enabled=true
mount.axis1_is_enabled=true
mount.axis0_rms_error_arcsec=0.00588810631231419
mount.axis1_rms_error_arcsec=3.8042986139608E-06
mount.axis0_dist_to_target_arcsec=-0.00226689620820303
mount.axis1_dist_to_target_arcsec=1.11493199359018E-06
mount.axis0_servo_error_arcsec=0
mount.axis1_servo_error_arcsec=0
focuser.is_enabled=false
focuser.position=0
focuser.is_moving=false
rotator.is_enabled=false
rotator.position_degs=0
rotator.is_moving=false
m3.port=0
```

Keywords in the response are generally divided into [category].[property] pairs. Values are generally one of the following types:

Floating point: Can be either standard decimal notation, such as “-1.234”, or scientific notation to express very large or very small values, such as “6.294E-08”.

Boolean: Can be either “true” or “false”

Integer: A simple series of digits with an optional sign, such as “180” or “-1”.

String: An arbitrary sequence of printable characters, such as “PlaneWave CDK700” or “-12:34:56.7”.

Name	Datatype	Description
site.latitude_degs	float	The configured latitude of the telescope, in degrees. Negative values are south of the equator.
site.longitude_degs	float	The configured longitude of the telescope, in degrees. Negative values are west of the prime meridian.
site.height_meters	float	The configured height (elevation) of the telescope

		above sea level, in meters.
site.lst_hours	float	The local mean sidereal time at the telescope.
mount.is_connected	boolean	True if PWI4 has an active connection to the mount.
mount.geometry	int	The geometry of configured mount. 0: Alt-Az 1: Equatorial Fork 2: German Equatorial
mount.ra_apparent_hours	float	The telescope's current position in hours of apparent right ascension. Pointing model corrections (if present) have been applied. Relative to a J2000 target, apparent coordinates account for precession, nutation, and annual aberration, but not refraction.
mount.dec_apparent_degs	float	The telescope's current position in degrees of apparent declination. Pointing model corrections (if present) have been applied. Relative to a J2000 target, apparent coordinates account for precession, nutation, and annual aberration, but not refraction.
mount.ra_j2000_hours	float	The telescope's current position in hours of J2000 right ascension. Pointing model corrections (if present) have been applied.
mount.dec_j2000_degs	float	The telescope's current position in degrees of J2000 declination. Pointing model corrections (if present) have been applied.
mount.azimuth_degs	float	The telescope's current position in degrees of azimuth. North is defined as 0 degrees, and East is 90 degrees.
mount.altitude_degs	float	The telescope's current position in degrees of altitude. 0 degrees is the horizon, and 90 degrees is zenith.
mount.is_slewing	Boolean	When the mount is first commanded to follow a new target, this flag reports true while the mount is moving to acquire the target. Once the target is acquired, or if the movement is stopped, this flag reports false.
mount.is_tracking	boolean	When the mount is trying to follow a target, this flag reports true. If mount.is_slewing is also true, then the mount has not yet acquired the target. When the mount is stopped, this flag reports false.
mount.field_angle_degs	float	The amount of field rotation induced by the geometry of the mount at the current telescope position. For equatorial fork mounts with good polar alignment, this number will be close to 0. For Alt-Az mounts
mount.field_angle_rate_degs_per_sec	float	

mount.axis0_is_enabled mount.axis1_is_enabled	boolean	
mount.axis0_rms_error_arcsec mount.axis1_rms_error_arcsec	float	
mount.axis0_dist_to_target_arcsec mount.axis1_dist_to_target_arcsec	float	
mount.axis0_servo_error_arcsec mount.axis1_servo_error_arcsec	float	
focuser.is_enabled	boolean	
focuser.position	float	
focuser.is_moving	boolean	
rotator.is_enabled	boolean	
rotator.position_degs	float	
rotator.is_moving	boolean	
m3.port	int	

TCP Server request/response format (SUBJECT TO CHANGE)

The TCP server is hosted on port 9220 by default.

The TCP server follows the same basic request and response structure as the HTTP server, but commands are sent and received in a slightly different way.

Example (where \n indicates a newline character, byte 0x0A):

OPEN localhost, port 8877
SEND: /status\n
SEND: \n
RECV: site.latitude_Degs=33.8703\nsite.longitude_degs=-118.24716\n…m3.port=0\n\n

(note: the socket remains open, so another request can be sent)

SEND: /v1/mount/gotoradec2000\n
SEND: ra=15.7\n
SEND: dec=85.3\n
SEND: \n
RECV: (status response, with blank line indicating end of message)

Commands

Command

Parameters

Description

Command		
Parameters		
Description		