



Telescope Interface Standard 2.0 Programmer's Reference (revised 16-Jan-2008)

Please conduct discussions regarding this interface and its usage on the [ASCOM-Talk List](#).

[Click here to join ASCOM-Talk](#)

Introduction

This document describes the interface used by low-level telescope "driver" components as part of the Astronomy Common Object Model (ASCOM). Components that implement this interface can provide a way for programs to control various telescopes via a standard set of properties and methods.

This specification covers a *simple, low-level* telescope control interface for reading and writing coordinates, slewing, synchronizing and access to common controller functions. It does not provide for accessories such as focusers, flip mirrors, etc. Those sorts of things will have their own interfaces. The characteristics of this interface comply with the [ASCOM Quality Guidelines](#), assuring consistent behavior and compatibility with the widest possible variety of Windows Automation clients.

In addition to containing features, the 2.0 specification has been clarified in numerous areas. These changes appear with this background color when they apply to existing V1 properties and methods. New properties and methods in V2.0 are not marked with the change color. For a summary of the changes and additions since the previous standard (Telescope 1.4), please refer to the [Release Notes](#).

The revision of 16-Jan-2008 (in this color) clarifies the meaning of the [SideOfPier](#) property.

Implementation and Conventions

All of the methods and properties specified must be present in any conforming driver's interface. However, some methods and properties may not actually be implemented; they will raise a "not implemented" error when called or accessed.

Equatorial coordinates are in the system specified by the new [EquatorialSystem](#) property (usually local topocentric, and depending on the mount), hours for right ascension, and degrees for declination. Thus, "inner loop" issues such as mechanical compensation must be contained within the driver or it's underlying controller(s). Time is in Coordinated Universal Time (UTC).

Usage

To control a particular telescope type, a program would create an instance of the driver for that telescope type, then use the standard properties and methods described in this document to effect control of that scope. Thus any program or script that uses the standard driver interface automatically gains access to any telescope type for which a driver exists. For more information on ASCOM, see the [ASCOM Home Page](#).

For best results

This will display best if you have Internet Explorer 5 or later, text size set to "smaller", and the following fonts installed:

- Arial
- Verdana
- Lucida Console

Release Notes

ASCOM Telescope V2.0

This is the 2.0 version of the ASCOM Telescope Interface. The new properties, methods, and classes added in V2.0 are listed below, with hyperlinks to their descriptive pages in this document.

In addition to these new features, the specification has been clarified in numerous areas. These changes appear with this background color when they apply to existing V1 properties and methods. New properties and methods in V2.0 are not marked with the change color.

Important

There are a few minor incompatibilities with the V1 spec.

- In V2, setting the [Connected](#) property to True does not automatically unpark the telescope, nor does it explicitly turn on tracking. This may affect clients that use the V1 interface.
- The enumerated constants for the [AlignmentMode](#) property have been renamed. They now start with 'alg' instead of 'tele'. Most V1 drivers already use the 'alg' names.

For more information see the [ASCOM Initiative web site](#).

New Properties in 2.0

- [ApertureArea](#)
- [AtHome](#)
- [AtPark](#)
- [CanSetDeclinationRate](#)
- [CanSetGuideRates](#)
- [CanSetPierSide](#)
- [CanSetRightAscensionRate](#)
- [CanSlewAltAz](#)
- [CanSlewAltAzAsync](#)
- [CanSyncAltAz](#)
- [DoesRefraction](#)
- [DriverVersion](#)
- [EquatorialSystem](#)
- [GuideRateDeclination](#)
- [GuideRateRightAscension](#)
- [InterfaceVersion](#)
- [IsPulseGuiding](#)
- [SideOfPier](#)
- [TrackingRate](#)

- [TrackingRates](#)

New Methods in 2.0

- [AxisRates\(\)](#)
- [CanMoveAxis\(\)](#)
- [DestinationSideOfPier\(\)](#)
- [MoveAxis\(\)](#)
- [SlewToAltAz\(\)](#)
- [SlewToAltAzAsync\(\)](#)
- [SyncToAltAz\(\)](#)

New Classes in 2.0

- [Rate](#)

Rate Object

Describes a range of rates supported by the [MoveAxis\(\)](#) method (degrees/per second)

Remarks

These are contained within the [AxisRates](#) collection. They serve to describe one or more supported ranges of rates of motion about a mechanical axis.

It is possible that the [Rate.Maximum](#) and [Rate.Minimum](#) properties will be equal. In this case, the Rate object expresses a single discrete rate.

Both the [Rate.Maximum](#) and [Rate.Minimum](#) properties are always expressed in units of degrees per second.

Rate.Maximum Property

Required

 **Rate.Maximum** (read-only, Double)

The maximum rate (degrees per second)

Syntax

Rate.Maximum

The property syntax has these parts:


Part	Description
Value (Double)	The maximum rate (degrees per second)

Remarks

This must always be a positive number. It indicates the maximum rate in either direction about the axis.

Rate.Minimum Property

Required

 **Rate.Minimum** (read-only, Double)

The minimum rate (degrees per second)

Syntax

Rate.Minimum

The property syntax has these parts:

Part	Description
Value (Double)	The minimum rate (degrees per second)

Remarks

This must always be a positive number. It indicates the maximum rate in either direction about the axis.

Telescope Object

Represents the ASCOM low-level telescope driver interface.

Remarks

Use this interface to perform basic operations on a robotic telescope. It is designed to be used by client applications that wish to provide telescope control capabilities which are

device-independent. By using this interface, your application will be freed from dealing with the details of serial port control and low-level protocols in use by various telescopes.

Discovery API

Not all of the features in this interface are required to be supported by all drivers and telescopes. This permits support for a wide range of telescope types without the evils of least-common-denominator designs. **Telescope** includes properties that permit discovery of supported features and supported ranges of motion rates.

Many optional properties can be discovered by simply trying to read or set them. If a property is not supported, trying to read or set it it will raise an exception, which you can trap.

On the other hand, some features must be discovered by reading one of the Can____ properties, all of which *must* be supported. These are used to discover support for features (such as slewing capabilities) which would alter the state of the telescope if called directly for discovery.



These Can____ properties are intended to declare only that the telescope *supports* the particular capability. They are not intended to reflect the current state of the telescope/mount. Furthermore, connecting to the telescope may be required in order to determine the capabilities to reflect in the Can____ properties (an error will be raised if a Can____ property is not available). Once connected, however, the Can____ properties must *not* change.

Finally, the [MoveAxis\(\)](#) method supports rate discovery. To discover the supported rates for this feature, a collection of [Rate](#) objects is provided. Each Rate object contains a maximum and minimum rate value. They may be equal, in which case the Rate object specifies a single discrete supported rate. The collections may contain more than one Rate object. Each specifies a range of rates or a single discrete rate.

Concurrency

This interface does not provide for concurrency control. In the interest of keeping things simple at this level, the architecture assumes that client applications will refrain from performing conflicting operations.

Telescope.AlignmentMode Property

Optional

 **Telescope.AlignmentMode** (read-only, [AlignmentModes](#))

The alignment mode of the mount.

Syntax

Telescope.**AlignmentMode**

The property syntax has these parts:

Part	Description
Value (AlignmentModes)	The alignment mode of the mount.

Remarks

German equatorial mounts are distinct because they require a flip at the meridian.

Note that in V2, the names in the AlignmentModes enumeration have been changed from V1. They now start with 'alg' instead of 'tele'. Most V1 drivers already use the 'alg' names.

Symbolic Constants

The (symbolic) values for **AlignmentModes** are:

Constant	Value	Description
algAltAz	0	Altitude-Azimuth mount

algGermanPolar	2	German equatorial mount
algPolar	1	Polar mount other than German equatorial

Copyright © 2001-2008, The ASCOM Initiative

Telescope.Altitude Property

Optional

 **Telescope.Altitude** (read-only, Double)

The Altitude above the local horizon of the telescope's current position (degrees, positive up)

Syntax

Telescope.**Altitude**

The property syntax has these parts:

Part	Description
Value (Double)	The Altitude above the local horizon of the telescope's current position (degrees, positive up)

Remarks

All mounts must report equatorial coordinates.

Telescope.ApertureArea Property

Optional

 **Telescope.ApertureArea** (read-only, Double)

The area of the telescope's aperture, taking into account any obstructions (square meters)

Syntax

Telescope.**ApertureArea**

The property syntax has these parts:

Part	Description
Value (Double)	The area of the telescope's aperture, taking into account any obstructions (square meters)

Remarks

This can be used by exposure calculators, as it includes the effects of obscuration by central obstructions, etc.

Telescope.ApertureDiameter Property

Optional

 **Telescope.ApertureDiameter** (read-only, Double)

The telescope's effective aperture diameter (meters)

Syntax

Telescope.**ApertureDiameter**

The property syntax has these parts:

Part	Description
Value (Double)	The telescope's effective aperture diameter (meters)

Remarks

This can be used in conjunction with [FocalLength](#) to provide focal ratio information.

Telescope.AtHome Property

Required

 **Telescope.AtHome** (read-only, Boolean)

True if the telescope is stopped in the Home position. Set only following a FindHome() operation, and reset with any slew operation..

Syntax

Telescope.**AtHome**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if the telescope is stopped in the Home position. Set only following a FindHome() operation, and reset with any slew operation..

Remarks

This property must be False if the telescope does not support homing.

Telescope.AtPark Property

Required

 **Telescope.AtPark** (read-only, Boolean)

True if the telescope has been put into the parked state by the Park() method. Set False by calling the Unpark() method.

Syntax

Telescope.**AtPark**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if the telescope has been put into the parked state by the Park() method. Set False by calling the Unpark() method.

Remarks

AtPark is True when the telescope is in the parked state. This is achieved by calling the Park method. When AtPark is true, the telescope movement is stopped (or restricted to a small safe range of movement) and all calls that would cause telescope movement (e.g. slewing, changing Tracking state) must not do so, and must raise an error. The telescope is taken out of parked state by calling the Unpark() method. If the telescope cannot be parked, then AtPark must always return False.

Telescope.Azimuth Property

Optional

 **Telescope.Azimuth** (read-only, Double)

The azimuth at the local horizon of the telescope's current position (degrees, North-referenced, positive East/clockwise).

Syntax

Telescope.**Azimuth**

The property syntax has these parts:

Part	Description
Value (Double)	The azimuth at the local horizon of the telescope's current position (degrees, North-referenced, positive East/clockwise).

Remarks

All mounts must report equatorial coordinates.

Telescope.CanFindHome Property

Required

 **Telescope.CanFindHome** (read-only, Boolean)

True if this telescope is capable of programmed finding its home position (FindHome() method)

Syntax

Telescope.**CanFindHome**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if this telescope is capable of programmed finding its home position (FindHome() method)

Remarks

May raise an error if the telescope is not connected.

Telescope.CanPark Property

Required

 **Telescope.CanPark** (read-only, Boolean)

True if this telescope is capable of programmed parking ([Park\(\)](#) method)

Syntax

Telescope.**CanPark**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if this telescope is capable of programmed parking (Park() method)

Remarks

May raise an error if the telescope is not connected.

Telescope.CanPulseGuide Property

Required

 **Telescope.CanPulseGuide** (read-only, Boolean)

True if this telescope is capable of software-pulsed guiding (via the [PulseGuide\(\)](#) method)

Syntax

Telescope.**CanPulseGuide**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if this telescope is capable of software-pulsed guiding (via the PulseGuide() method)

Remarks

May raise an error if the telescope is not connected.

Telescope.CanSetDeclinationRate Property

Required

 **Telescope.CanSetDeclinationRate** (read-only, Boolean)

True if the [DeclinationRate](#) property can be changed to provide offset tracking in the declination axis.

Syntax

Telescope.**CanSetDeclinationRate**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if the DeclinationRate property can be changed to provide offset tracking in the declination axis.

Remarks

May raise an error if the telescope is not connected.

Telescope.CanSetGuideRates Property

Required

 **Telescope.CanSetGuideRates** (read-only, Boolean)

True if the guide rate properties used for [PulseGuide\(\)](#) can be adjusted.

Syntax

Telescope.**CanSetGuideRates**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if the guide rate properties used for PulseGuide() can be adjusted.

Remarks

May raise an error if the telescope is not connected.

Telescope.CanSetPark Property

Required

 **Telescope.CanSetPark** (read-only, Boolean)

True if this telescope is capable of programmed setting of its park position ([SetPark\(\)](#) method)

Syntax

Telescope.**CanSetPark**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if this telescope is capable of programmed setting of its park position (SetPark() method)

Remarks

May raise an error if the telescope is not connected.

Telescope.CanSetPierSide Property

Required

 **Telescope.CanSetPierSide** (read-only, Boolean)

True if the [SideOfPier](#) property can be *set*, meaning that the mount can be forced to flip.

Syntax

Telescope.**CanSetPierSide**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if the SideOfPier property can be <i>set</i> , meaning that the mount can be forced to flip.

Remarks

This will always return False for mounts (non-German-equatorial) that do not have to be flipped.

May raise an error if the telescope is not connected.

Telescope.CanSetRightAscensionRate Property

Required

 **Telescope.CanSetRightAscensionRate** (read-only, Boolean)

True if the [RightAscensionRate](#) property can be changed to provide offset tracking in the right ascension axis.

Syntax

Telescope.**CanSetRightAscensionRate**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if the RightAscensionRate property can be changed to provide offset tracking in the right ascension axis.

Remarks

May raise an error if the telescope is not connected.

Telescope.CanSetTracking Property

Required

 **Telescope.CanSetTracking** (read-only, Boolean)

True if the [Tracking](#) property can be changed, turning telescope sidereal tracking on and off.

Syntax

Telescope.**CanSetTracking**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if the Tracking property can be changed, turning telescope sidereal tracking on and off.

Remarks

May raise an error if the telescope is not connected.

Telescope.CanSlew Property

Required

 **Telescope.CanSlew** (read-only, Boolean)

True if this telescope is capable of programmed slewing (synchronous or asynchronous) to equatorial coordinates

Syntax

Telescope.**CanSlew**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if this telescope is capable of programmed slewing (synchronous or asynchronous) to equatorial coordinates

Remarks

If this is true, then only the synchronous equatorial slewing methods are guaranteed to be supported. See the CanSlewAsync property for the asynchronous slewing capability flag. May raise an error if the telescope is not connected.

Telescope.CanSlewAltAz Property

Required

 **Telescope.CanSlewAltAz** (read-only, Boolean)

True if this telescope is capable of programmed slewing (synchronous or asynchronous) to local horizontal coordinates

Syntax

Telescope.**CanSlewAltAz**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if this telescope is capable of programmed slewing (synchronous or asynchronous) to local horizontal coordinates

Remarks

If this is true, then only the synchronous local horizontal slewing methods are guaranteed to be supported. See the CanSlewAltAzAsync property for the asynchronous slewing capability flag. May raise an error if the telescope is not connected.

Telescope.CanSlewAltAzAsync Property

Required

 **Telescope.CanSlewAltAzAsync** (read-only, Boolean)

True if this telescope is capable of programmed asynchronous slewing to local horizontal coordinates

Syntax

Telescope.**CanSlewAltAzAsync**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if this telescope is capable of programmed asynchronous slewing to local horizontal coordinates

Remarks

This indicates the the asynchronous local horizontal slewing methods are supported. If this is True, then CanSlewAltAz will also be true. May raise an error if the telescope is not connected.

Telescope.CanSlewAsync Property

Required

 **Telescope.CanSlewAsync** (read-only, Boolean)

True if this telescope is capable of programmed asynchronous slewing to equatorial coordinates

Syntax

Telescope.**CanSlewAsync**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if this telescope is capable of programmed asynchronous slewing to equatorial coordinates

Remarks

This indicates the the asynchronous equatorial slewing methods are supported. If this is True, then CanSlew will also be true. May raise an error if the telescope is not connected.

Telescope.CanSync Property

Required

 **Telescope.CanSync** (read-only, Boolean)

True if this telescope is capable of programmed synching to equatorial coordinates

Syntax

Telescope.**CanSync**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if this telescope is capable of programmed synching to equatorial coordinates

Remarks

May raise an error if the telescope is not connected.

Telescope.CanSyncAltAz Property

Required

 **Telescope.CanSyncAltAz** (read-only, Boolean)

True if this telescope is capable of programmed synching to local horizontal coordinates

Syntax

Telescope.**CanSyncAltAz**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if this telescope is capable of programmed synching to local horizontal coordinates

Remarks

May raise an error if the telescope is not connected.

Telescope.CanUnpark Property

Required

 **Telescope.CanUnpark** (read-only, Boolean)

True if this telescope is capable of programmed unparking ([Unpark\(\)](#) method)

Syntax

Telescope.**CanUnpark**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if this telescope is capable of programmed unparking (Unpark() method)

Remarks

If this is true, then CanPark will also be true. May raise an error if the telescope is not connected.

Telescope.Connected Property

Required

 **Telescope.Connected** (read-write, Boolean)

True if telescope connected, False otherwise

Syntax

Telescope.**Connected** [= *Boolean*]

The property syntax has these parts:

Part	Description
Value (Boolean)	True if telescope connected, False otherwise

Remarks

Set this property to True to connect to the telescope. Raises an error if there is a problem connecting.

Some Telescope properties and methods will raise errors if the scope is not connected.

Important

In V2, setting the Connected property to True does not automatically unpark the telescope, nor does it explicitly turn on tracking. This may affect clients that use the V1 interface.

Telescope.Declination Property

Required

 **Telescope.Declination** (read-only, Double)

The declination (degrees) of the telescope's current equatorial coordinates, in the coordinate system given by the [EquatorialSystem](#) property

Syntax

Telescope.**Declination**

The property syntax has these parts:

Part	Description
Value (Double)	The declination (degrees) of the telescope's current equatorial coordinates, in the coordinate system given by the EquatorialSystem property

Remarks

Reading the property will raise an error if the value is unavailable.

Telescope.DeclinationRate Property

Optional

 **Telescope.DeclinationRate** (read-write, Double)

The declination tracking rate (arcseconds per second, default = 0.0)

Syntax

Telescope.**DeclinationRate** [= *Double*]

The property syntax has these parts:

Part	Description
Value (Double)	The declination tracking rate (arcseconds per second, default = 0.0)

Remarks

This property, together with [RightAscensionRate](#), provides support for "offset tracking". Offset tracking is used primarily for tracking objects that move relatively slowly against the equatorial coordinate system. It also may be used by a software guiding system that controls rates instead of using the [PulseGuide\(\)](#) method.

NOTES:

- The property value represents an offset from zero motion.
- If [CanSetDeclinationRate](#) is False, this property will always return 0.
- To discover whether this feature is supported, test the [CanSetDeclinationRate](#) property.

- The supported range of this property is telescope specific, however, if this feature is supported, it can be expected that the range is sufficient to allow correction of guiding errors caused by moderate misalignment and periodic error.
- If this property is non-zero when an equatorial slew is initiated, the telescope *should* continue to update the slew destination coordinates at the given offset rate. This will allow precise slews to a fast-moving target with a slow-slewing telescope. When the slew completes, the TargetRightAscension and [TargetDeclination](#) properties *should* reflect the final (adjusted) destination. This is not a required feature of this specification, however it is desirable.

Telescope.Description Property

Required

 **Telescope.Description** (read-only, String)

The long description of the telescope.

Syntax

Telescope.**Description**

The property syntax has these parts:

Part	Description
Value (String)	The long description of the telescope.

Remarks

This string may contain line endings and may be hundreds to thousands of characters long. It is intended to display detailed information on the telescope itself. See the [DriverInfo property](#) for descriptive info on the driver itself.

NOTE: this string should not be over 1000 characters in length, as applications may use popup boxes to display Description. Older versions of Windows have string length limitations in (e.g.) `MessageBox()`, which will cause an application failure if the string is too long.

Telescope.DoesRefraction Property

Optional

 **Telescope.DoesRefraction** (read-write, Boolean)

True if the telescope or driver applies atmospheric refraction to coordinates.

Syntax

Telescope.**DoesRefraction** [= *Boolean*]

The property syntax has these parts:

Part	Description
Value (Boolean)	True if the telescope or driver applies atmospheric refraction to coordinates.

Remarks

If this property is True, the coordinates sent to, and retrieved from, the telescope are *unrefracted*.

NOTES:

- If the driver does not know whether the attached telescope does its own refraction, and if the driver does not itself calculate refraction, this property (if implemented) must raise an error when read.
- Writing to this property is optional. Often, a telescope (or its driver) calculates refraction using standard atmospheric parameters. If the client wishes to calculate a more accurate refraction, then this property could be set to False

and these client-refracted coordinates used. If disabling the telescope or driver's refraction is not supported, the driver must raise an error when an attempt to set this property to False is made.

- Setting this property to True for a telescope or driver that does refraction, or to False for a telescope or driver that does not do refraction, shall not raise an error. It shall have no effect.

Telescope.DriverInfo Property

Required

 **Telescope.DriverInfo** (read-only, String)

Descriptive and version information about this ASCOM Telescope driver

Syntax

Telescope.**DriverInfo**

The property syntax has these parts:

Part	Description
Value (String)	Descriptive and version information about this ASCOM Telescope driver

Remarks

This string may contain line endings and may be hundreds to thousands of characters long. It is intended to display detailed information on the ASCOM driver, including version and copyright data.. See the [Description](#) property for descriptive info on the telescope itself. To get the driver version in a parseable string, use the [DriverVersion](#) property.

Telescope.DriverVersion Property

Required

 **Telescope.DriverVersion** (read-only, String)

A string containing *only* the major and minor version of the driver. This must be in the form "n.n".

Syntax

Telescope.**DriverVersion**

The property syntax has these parts:


Part	Description
Value (String)	A string containing <i>only</i> the major and minor version of the driver. This must be in the form "n.n".

Remarks

Not to be confused with the [InterfaceVersion](#) property, which is the version of this specification supported by the driver (currently 2).

Telescope.EquatorialSystem Property

Required

 **Telescope.EquatorialSystem** (read-only, [EquatorialCoordinateType](#))

Equatorial coordinate system used by this telescope.

Syntax

`Telescope.EquatorialSystem`

The property syntax has these parts:

Part	Description
Value (EquatorialCoordinateType)	Equatorial coordinate system used by this telescope.

Remarks

Most amateur telescopes use local topocentric coordinates. This coordinate system is simply the apparent position in the sky (possibly uncorrected for atmospheric refraction) for "here and now", thus these are the coordinates that one would use with digital setting circles and most amateur scopes.

More sophisticated telescopes use one of the standard reference systems established by professional astronomers. The most common is the Julian Epoch 2000 (J2000). These instruments apply corrections for precession, nutation, aberration, etc. to adjust the coordinates from the standard system to the pointing direction for the time and location of "here and now".

Symbolic Constants

The (symbolic) values for **EquatorialCoordinateType** are:

Constant	Value	Description
equLocalTopocentric	1	Local topocentric; this is the most common for amateur scopes.
equJ2000	2	J2000 equator/equinox, ICRS reference frame
equJ2050	3	J2050 equator/equinox, ICRS reference frame
equOther	0	Custom or unknown equinox and/or reference frame
equB1950	4	B1950 equinox, FK4 reference frame

Telescope.FocalLength Property

Optional

 **Telescope.FocalLength** (read-only, Double)

The telescope's focal length, meters

Syntax

Telescope.**FocalLength**

The property syntax has these parts:

Part	Description
Value (Double)	The telescope's focal length, meters

Remarks

This property may be used by clients to calculate telescope field of view and plate scale when combined with detector pixel size and geometry.

Telescope.GuideRateDeclination Property

Optional

 **Telescope.GuideRateDeclination** (read-write, Double)

The current Declination movement rate offset for telescope guiding (degrees/sec)

Syntax

Telescope.**GuideRateDeclination** [= *Double*]

The property syntax has these parts:

Part	Description
Value (Double)	The current Declination movement rate offset for telescope guiding (degrees/sec)

Remarks

This is the rate for both hardware/relay guiding and the [PulseGuide\(\)](#) method.

NOTES:

- To discover whether this feature is supported, test the [CanSetGuideRates](#) property.
- The supported range of this property is telescope specific, however, if this feature is supported, it can be expected that the range is sufficient to allow correction of guiding errors caused by moderate misalignment and periodic error.
- If a telescope does not support separate guiding rates in Right Ascension and

Declination, then it is permissible for [GuideRateRightAscension](#) and GuideRateDeclination to be tied together. In this case, changing one of the two properties will cause a change in the other.

- Mounts must start up with a known or default declination guide rate, and this property must return that known/default guide rate until changed.

Telescope.GuideRateRightAscension Property

Optional

 **Telescope.GuideRateRightAscension** (read-write, Double)

The current Right Ascension movement rate offset for telescope guiding (degrees/sec)

Syntax

Telescope.**GuideRateRightAscension** [= *Double*]

The property syntax has these parts:

Part	Description
Value (Double)	The current Right Ascension movement rate offset for telescope guiding (degrees/sec)

Remarks

This is the rate for both hardware/relay guiding and the [PulseGuide\(\)](#) method.

NOTES:

- To discover whether this feature is supported, test the [CanSetGuideRates](#) property.
- The supported range of this property is telescope specific, however, if this feature is supported, it can be expected that the range is sufficient to allow correction of guiding errors caused by moderate misalignment and periodic error.
- If a telescope does not support separate guiding rates in Right Ascension and

Declination, then it is permissible for GuideRateRightAscension and [GuideRateDeclination](#) to be tied together. In this case, changing one of the two properties will cause a change in the other.

- Mounts must start up with a known or default right ascension guide rate, and this property must return that known/default guide rate until changed.

Telescope.InterfaceVersion Property

Required

 **Telescope.InterfaceVersion** (read-only, Integer)

The version of this interface. Will return 2 for this version.

Syntax

Telescope.**InterfaceVersion**

The property syntax has these parts:

Part	Description
Value (Integer)	The version of this interface. Will return 2 for this version.

Remarks

Clients can detect legacy V1 drivers by trying to read this property. If the driver raises an error, it is a V1 driver. V1 did not specify this property. A driver may also return a value of 1. In other words, a raised error *or* a return value of 1 indicates that the driver is a V1 driver.

Telescope.IsPulseGuiding Property

Optional

 **Telescope.IsPulseGuiding** (read-only, Boolean)

True if a [PulseGuide\(\)](#) command is in progress, False otherwise

Syntax

`Telescope.IsPulseGuiding`

The property syntax has these parts:

Part	Description
Value (Boolean)	True if a PulseGuide() command is in progress, False otherwise

Remarks

Raises an error if the value of the [CanPulseGuide](#) property is false (the driver does not support the [PulseGuide\(\)](#) method).

Telescope.Name Property

Required

 **Telescope.Name** (read-only, String)

The short name of the telescope, for display purposes

Syntax

Telescope.**Name**

The property syntax has these parts:

Part	Description
Value (String)	The short name of the telescope, for display purposes

Remarks

Telescope.RightAscension Property

Required

 **Telescope.RightAscension** (read-only, Double)

The right ascension (hours) of the telescope's current equatorial coordinates, in the coordinate system given by the [EquatorialSystem](#) property

Syntax

Telescope.**RightAscension**

The property syntax has these parts:

Part	Description
Value (Double)	The right ascension (hours) of the telescope's current equatorial coordinates, in the coordinate system given by the EquatorialSystem property

Remarks

Reading the property will raise an error if the value is unavailable.

Telescope.RightAscensionRate Property

Optional

 **Telescope.RightAscensionRate** (read-write, Double)

The right ascension tracking rate *offset* from sidereal (seconds per *sidereal* second, default = 0.0)

Syntax

Telescope.**RightAscensionRate** [= *Double*]

The property syntax has these parts:

Part	Description
Value (Double)	The right ascension tracking rate <i>offset</i> from sidereal (seconds per <i>sidereal</i> second, default = 0.0)

Remarks

This property, together with [DeclinationRate](#), provides support for "offset tracking". Offset tracking is used primarily for tracking objects that move relatively slowly against the equatorial coordinate system. It also may be used by a software guiding system that controls rates instead of using the [PulseGuide\(\)](#) method.

NOTES:

- The property value represents an offset from the current selected [TrackingRate](#). If this property is zero, tracking will be at the selected [TrackingRate](#).

- If [CanSetRightAscensionRate](#) is False, this property must always return 0.
- To discover whether this feature is supported, test the [CanSetRightAscensionRate](#) property.
- The property value is in seconds of right ascension per *sidereal second*. To convert a given rate in (the more common) units of sidereal seconds per UTC (clock) second, multiply the value by 0.9972695677 (the number of UTC seconds in a sidereal second) then set the property. Please note that these units were chosen for the Telescope V1 standard, and in retrospect, this was an unfortunate choice. However, to maintain backwards compatibility, the units cannot be changed. A simple multiplication is all that's needed, as noted.
- The supported range of this property is telescope specific, however, if this feature is supported, it can be expected that the range is sufficient to allow correction of guiding errors caused by moderate misalignment and periodic error.
- If this property is non-zero when an equatorial slew is initiated, the telescope *should* continue to update the slew destination coordinates at the given offset rate. This will allow precise slews to a fast-moving target with a slow-slewing telescope. When the slew completes, the TargetRightAscension and [TargetDeclination](#) properties *should* reflect the final (adjusted) destination. This is not a required feature of this specification, however it is desirable.

Use the [Tracking property](#) to enable and disable sidereal tracking (if supported).

Telescope.SideOfPier Property

Optional

 **Telescope.SideOfPier** (read-write, [PierSide](#))

Indicates the pointing state of the mount. See Remarks.

Syntax

Telescope.**SideOfPier** [= [PierSide](#)]

The property syntax has these parts:

Part	Description
Value (PierSide)	Indicates the pointing state of the mount. See Remarks.

Remarks

For historical reasons, this property's name does not reflect its true meaning. The name will not be changed (so as to preserve compatibility), but the meaning has since become clear. In a future version of the standard, a synonym for this property will be added which better describes its meaning.

All conventional mounts have two pointing states for a given equatorial (sky) position. Mechanical limitations often make it impossible for the mount to position the optics at given HA/Dec in one of the two pointing states, but there are places where the same point can be reached in both pointing states (e.g., near the pole). In order to understand these pointing states, consider the following (thanks to Patrick Wallace for this info):

- All conventional telescope mounts have two axes nominally at right angles. For

an equatorial, the longitude axis is mechanical hour angle and the latitude axis is mechanical declination.

- Sky coordinates and mechanical coordinates are two completely separate arenas. This becomes rather more obvious if your mount is an altaz, but it's still true for an equatorial.
- Both mount axes can in principle move over a range of 360 deg. This is distinct from sky HA/Dec, where Dec is limited to a 180 deg range (+90 to -90).
- Apart from practical limitations, any point in the sky can be seen in two mechanical orientations. To get from one to the other the HA axis is moved 180 deg and the Dec axis is moved through the pole a distance twice the sky codeclination (90 - sky declination).
- Mechanical zero HA/Dec is one of the two ways of pointing at the intersection of the celestial equator and the local meridian. Choose one, and once you're there, consider the two mechanical encoders zeroed.
- The two states are, then, (a) "normal", where the mechanical Dec is in the range ± 90 deg, and (b) "beyond the pole", where the mechanical Dec is outside that range.
- "Side of pier" is a *consequence* of the former definition, not something fundamental. Apart from mechanical interference, the telescope can move from one side of the pier to the other without the mechanical Dec having changed: you could track Polaris forever with the telescope moving from west of pier to east of pier or vice versa every 12h. Thus, SideOfPier is, in general, not a useful term (except perhaps in a loose, descriptive, explanatory sense).
- All this applies to a fork mount just as much as to a GEM, and it would be wrong to make the "beyond pole" state illegal for the former. You may not be able to get there if your camera hits the fork, but it's possible on some mounts. Whether this is useful depends on whether you're in Hawaii or Finland.

To first order, the relationship between sky and mechanical HA/Dec is as follows:

Normal state:

```
HA_sky = HA_mech
Dec_sky = Dec_mech
```

Beyond the pole

```
HA_sky = HA_mech + 12h, expressed in range  $\pm 12$ h
Dec_sky = 180d - Dec_mech, expressed in range  $\pm 90$ d
```

Astronomy software often needs to know which "side of the pier" (which pointing state) the mount is in. Examples include setting guiding polarities and calculating dome opening azimuth/altitude.

The meaning of SideOfPier, then is:

pierEast - Normal pointing state

pierWest - Beyond the pole pointing state

If the mount hardware reports neither the true pointing state (or equivalent) nor the

mechanical declination axis position (which varies from -180 to +180), a driver cannot calculate the pointing state, and **must not** implement SideOfPier.

If the mount hardware reports only the mechanical declination axis position (-180 to +180) then a driver can calculate SideOfPier as follows:

pierEast = abs(mechanical dec) <= 90 deg

pierWest = abs(mechanical Dec) > 90 deg

It is allowed (though not required) that this property may be written to force the mount to flip. Doing so, however, *may* change the right ascension of the telescope. During flipping, [Telescope.Slewing](#) must return True.

Symbolic Constants

The (symbolic) values for **PierSide** are:

Constant	Value	Description
pierEast	0	Mount on East side of pier (looking West)
pierWest	1	Mount on West side of pier (looking East)

Telescope.SiderealTime Property

Required

 **Telescope.SiderealTime** (read-only, Double)

The local apparent sidereal time from the telescope's internal clock (hours, sidereal)

Syntax

Telescope.**SiderealTime**

The property syntax has these parts:

Part	Description
Value (Double)	The local apparent sidereal time from the telescope's internal clock (hours, sidereal)

Remarks

It is required for a driver to calculate this from the system clock if the telescope has no accessible source of sidereal time.

Local Apparent Sidereal Time is the sidereal time used for pointing telescopes, and thus must be calculated from the Greenwich Mean Sidereal time, longitude, nutation in longitude and true ecliptic obliquity.

Telescope.SiteElevation Property

Optional

 **Telescope.SiteElevation** (read-write, Double)

The elevation above mean sea level (meters) of the site at which the telescope is located

Syntax

Telescope.**SiteElevation** [= *Double*]

The property syntax has these parts:

Part	Description
Value (Double)	The elevation above mean sea level (meters) of the site at which the telescope is located

Remarks

Setting this property will raise an error if the given value is outside the range -300 through +10000 metres. Reading the property will raise an error if the value has never been set or is otherwise unavailable.

Telescope.SiteLatitude Property

Optional

 **Telescope.SiteLatitude** (read-write, Double)

The *geodetic*(map) latitude (degrees, positive North, WGS84) of the site at which the telescope is located.

Syntax

Telescope.**SiteLatitude** [= *Double*]

The property syntax has these parts:

Part	Description
Value (Double)	The <i>geodetic</i> (map) latitude (degrees, positive North, WGS84) of the site at which the telescope is located.

Remarks

Setting this property will raise an error if the given value is outside the range -90 to +90 degrees. Reading the property will raise an error if the value has never been set or is otherwise unavailable.

Telescope.SiteLongitude Property

Optional

 **Telescope.SiteLongitude** (read-write, Double)

The longitude (degrees, positive East, WGS84) of the site at which the telescope is located.

Syntax

Telescope.**SiteLongitude** [= *Double*]

The property syntax has these parts:

Part	Description
Value (Double)	The longitude (degrees, positive East, WGS84) of the site at which the telescope is located.

Remarks

Setting this property will raise an error if the given value is outside the range -180 to +180 degrees. Reading the property will raise an error if the value has never been set or is otherwise unavailable. Note that West is negative!

Telescope.Slewing Property

Optional

 **Telescope.Slewing** (read-only, Boolean)

True if telescope is currently moving in response to one of the Slew methods or the MoveAxis() method, False at all other times.

Syntax

Telescope.**Slewing**

The property syntax has these parts:

Part	Description
Value (Boolean)	True if telescope is currently moving in response to one of the Slew methods or the MoveAxis() method, False at all other times.

Remarks

Reading the property will raise an error if the value is unavailable. If the telescope is not capable of asynchronous slewing, this property will always be False.

The definition of "slewing" excludes motion caused by sidereal tracking, PulseGuide(), RightAscensionRate, and DeclinationRate. It reflects only motion caused by one of the Slew commands, flipping caused by changing the [SideOfPier](#) property, or [MoveAxis\(\)](#).

Telescope.SlewSettleTime Property

Optional

 **Telescope.SlewSettleTime** (read-write, Integer)

Specifies a post-slew settling time (sec.).

Syntax

Telescope.**SlewSettleTime** [= *Integer*]

The property syntax has these parts:

Part	Description
Value (Integer)	Specifies a post-slew settling time (sec.).

Remarks

Adds additional time to slew operations. Slewing methods will not return, and the [Slewing](#) property will not become False, until the slew completes and the SlewSettleTime has elapsed. This feature (if supported) may be used with mounts that require extra settling time after a slew.

Telescope.TargetDeclination Property

Optional

 **Telescope.TargetDeclination** (read-write, Double)

The declination (degrees, positive North) for the target of an equatorial slew or sync operation

Syntax

Telescope.**TargetDeclination** [= *Double*]

The property syntax has these parts:

Part	Description
Value (Double)	The declination (degrees, positive North) for the target of an equatorial slew or sync operation

Remarks

Setting this property will raise an error if the given value is outside the range -90 to +90 degrees. Reading the property will raise an error if the value has never been set or is otherwise unavailable.

Telescope.TargetRightAscension Property

Optional

 **Telescope.TargetRightAscension** (read-write, Double)

The right ascension (hours) for the target of an equatorial slew or sync operation

Syntax

Telescope.**TargetRightAscension** [= *Double*]

The property syntax has these parts:

Part	Description
Value (Double)	The right ascension (hours) for the target of an equatorial slew or sync operation

Remarks

Setting this property will raise an error if the given value is outside the range 0 to 24 hours. Reading the property will raise an error if the value has never been set or is otherwise unavailable.

Telescope.Tracking Property

Required

 **Telescope.Tracking** (read-write, Boolean)

The state of the telescope's sidereal tracking drive.

Syntax

Telescope.**Tracking** [= *Boolean*]

The property syntax has these parts:

Part	Description
Value (Boolean)	The state of the telescope's sidereal tracking drive.

Remarks

Changing the value of this property will turn the sidereal drive on and off. However, some telescopes may not support changing the value of this property and thus may not support turning tracking on and off. See the [CanSetTracking](#) property.

Telescope.TrackingRate Property

Optional

 **Telescope.TrackingRate** (read-write, [DriveRates](#))

The current tracking rate of the telescope's sidereal drive

Syntax

Telescope.**TrackingRate** [= [DriveRates](#)]

The property syntax has these parts:

Part	Description
Value (DriveRates)	The current tracking rate of the telescope's sidereal drive

Remarks

Supported rates (one of the DriveRates values) are contained within the [TrackingRates](#) collection. Values assigned to TrackingRate must be one of these supported rates. If an unsupported value is assigned to this property, it will raise an error.

The currently selected tracking rate be further adjusted via the [RightAscensionRate](#) and [DeclinationRate](#) properties. These rate offsets are applied to the currently selected TrackingRate.

Mounts must start up with a known or default tracking rate, and this property must return that known/default tracking rate until changed. If the mount's current tracking rate cannot be determined (for example, it is a write-only property of the mount's protocol), it is permitted for the driver to force and report a default rate on connect. In this case, the

preferred default is Sidereal rate.

Symbolic Constants

The (symbolic) values for **DriveRates** are:

Constant	Value	Description
driveSidereal	0	Sidereal rate
driveLunar	1	Lunar rate
driveSolar	2	Solar rate
driveKing	3	King rate

Telescope.TrackingRates Property

Required

 **Telescope.TrackingRates** (read-only, Object)

Returns a collection of supported DriveRate values that describe the permissible values of the [TrackingRate](#) property for this telescope type.

Syntax

Telescope.**TrackingRates**

The property syntax has these parts:

Part	Description
Value (Object)	Returns a collection of supported DriveRate values that describe the permissible values of the TrackingRate property for this telescope type.

Remarks

At a minimum, this must contain an item for *driveSidereal*.

Telescope.UTCDate Property

Required

 **Telescope.UTCDate** (read-write, Date)

The UTC date/time of the telescope's internal clock

Syntax

Telescope.**UTCDate** [= *Date*]

The property syntax has these parts:

Part	Description
Value (Date)	The UTC date/time of the telescope's internal clock

Remarks

The driver must calculate this from the system clock if the telescope has no accessible source of UTC time. In this case, the property must not be writeable (this would change the system clock!) and will instead raise an error. However, it is permitted to change the telescope's internal UTC clock if it is being used for this property. This allows clients to adjust the telescope's UTC clock as needed for accuracy. Reading the property will raise an error if the value has never been set or is otherwise unavailable.

Telescope.AbortSlew() Method

Optional

Stops a slew in progress.

Syntax

Telescope.AbortSlew()

The method syntax has these parts:

Part	Description
Return (Nothing)	Does not return a value.

Remarks

Effective only after a call to [SlewToTargetAsync\(\)](#), [SlewToCoordinatesAsync\(\)](#), [SlewToAltAzAsync\(\)](#), or [MoveAxis\(\)](#). Does nothing if no slew/motion is in progress. Tracking is returned to its pre-slew state.

Raises an error if AtPark is true.

Telescope.AxisRates() Method

Required

Determine the rates at which the telescope may be moved about the specified axis by the MoveAxis() method.

Syntax

Telescope.AxisRates(*Axis*)

The method syntax has these parts:

Part	Description
Axis (TelescopeAxes)	The axis about which rate information is desired
Return (Object)	Collection of Rate objects describing the supported rates of motion that can be supplied to the MoveAxis() method for the specified axis.

Remarks

See the description of [MoveAxis\(\)](#) for more information.

This method must return an *empty* collection if MoveAxis is not supported.

Symbolic Constants

The (symbolic) values for **TelescopeAxes** are:

Constant	Value	Description

axisPrimary	0	Primary axis (e.g., Right Ascension or Azimuth)
axisSecondary	1	Secondary axis (e.g., Declination or Altitude)
axisTertiary	2	Tertiary axis (e.g. imager rotator/de-rotator)

Telescope.CanMoveAxis() Method

Required

True if the telescope can be controlled about the specified axis via the MoveAxis() method.

Syntax

Telescope.CanMoveAxis(*Axis*)

The method syntax has these parts:

Part	Description
<i>Axis</i> (TelescopeAxes)	The identifier for the axis to be tested
Return (Boolean)	True if the telescope can be controlled about the specified axis via the MoveAxis() method.

Remarks

See the description of [MoveAxis\(\)](#) for more information.

Symbolic Constants

The (symbolic) values for **TelescopeAxes** are:

Constant	Value	Description
axisPrimary	0	Primary axis (e.g., Right Ascension or Azimuth)
axisSecondary	1	Secondary axis (e.g., Declination or Altitude)

axisTertiary	2	Tertiary axis (e.g. imager rotator/de-rotator)
---------------------	---	--

Telescope.CommandBlind() Method

Optional

Send a string comand directly to the telescope without expecting response data.

Syntax

Telescope.CommandBlind(*Command* [, *Raw*])

The method syntax has these parts:

Part	Description
<i>Command</i> (String)	The command string to be sent to the telescope.
<i>Raw</i> (Boolean)	Bypass any delimiters or framing around the command (optional , default = False)
Return (Nothing)	Does not return a value.

Remarks

If the optional Raw parameter is set True, the driver must not insert or append any delimiters; this must send the unmodified raw string directly to the device. If the driver cannot support Raw=True, it must raise an error if Raw is set to True.

Important

If you use this feature of the Telescope driver interface, your application will be dependent on the low-level protocol used by the particular scope you are connected to. Thus your application will not work with any arbitrary type of

telescope.

Raises an error if there is a problem communicating with the telescope.

Copyright © 2001-2008, The ASCOM Initiative

Telescope.CommandBool() Method

Optional

Send a string comand to the telescope, returning a true/false response

Syntax

Telescope.CommandBool(*Command* [, *Raw*])

The method syntax has these parts:

Part	Description
<i>Command</i> (String)	The command string to be sent to the telescope
<i>Raw</i> (Boolean)	Bypass any delimiters or framing around the command (optional , default = False)
Return (Boolean)	True if the response indicated true or success, else False.

Remarks

If the optional Raw parameter is set True, the driver must not insert or append any delimiters; this must send the unmodified raw string directly to the device. If the driver cannot support Raw=True, it must raise an error if Raw is set to True.

Important

If you use this feature of the Telescope driver interface, your application will be dependent on the low-level protocol used by the particular scope you are connected to. Thus your application will not work with any arbitrary type of

telescope.

Raises an error if there is a problem communicating with the telescope.

The returned value is the Automation-compatible Boolean type, True or False. It is the responsibility of the driver implementing this interface to translate raw response data to True/False values for return. If you want to see the raw response string, see [CommandString\(\)](#).

Telescope.CommandString() Method

Optional

Send a string comand to the telescope, returning the response string

Syntax

Telescope.CommandString(*Command* [, *Raw*])

The method syntax has these parts:

Part	Description
<i>Command</i> (String)	The command string to be sent to the telescope
<i>Raw</i> (Boolean)	Bypass any delimiters or framing around the command (optional , default = False)
Return (String)	The response data from the telescope resulting from the sent command.

Remarks

If the optional Raw parameter is set True, the driver must not insert or append any delimiters; this must send the unmodified raw string directly to the device. Raw=True also indicates that any delimiters in the device's response string must be returned to the client. If the driver cannot support Raw=True, it must raise an error if Raw is set to True.

Important

If you use this feature of the Telescope driver interface, your application will

be dependent on the low-level protocol used by the particular scope you are connected to. Thus your application will not work with any arbitrary type of telescope.

Raises an error if there is a problem communicating with the telescope.

Telescope.DestinationSideOfPier() Method

Optional

Predict side of pier for German equatorial mounts

Syntax

Telescope.DestinationSideOfPier(*RightAscension*, *Declination*)

The method syntax has these parts:

Part	Description
<i>RightAscension</i> (Double)	The destination right ascension (hours).
<i>Declination</i> (Double)	The destination declination (degrees, positive North).
Return (PierSide)	The side of the pier on which the telescope <i>would be</i> on <i>if</i> a slew to the given equatorial coordinates is performed at the current instant of time .

Remarks

This is only relevant if [AlignmentMode](#) property is `algGermanPolar`. Non German mounts must raise an error.

This is useful for pre-correction of destination coordinates for a slew when mechanical error correction is in effect. It may also be used in a predictive scheduling application.

Symbolic Constants

The (symbolic) values for **PierSide** are:

Constant	Value	Description
pierEast	0	Mount on East side of pier (looking West)
pierWest	1	Mount on West side of pier (looking East)

Telescope.FindHome() Method

Optional

Locates the telescope's "home" position (synchronous)

Syntax

Telescope.FindHome()

The method syntax has these parts:

Part	Description
Return (Nothing)	Does not return a value.

Remarks

Returns only after the home position has been found. At this point the [AtHome](#) property will be True. Raises an error if there is a problem.

Raises an error if AtPark is true.

Telescope.MoveAxis() Method

Optional

Move the telescope in one axis at the given rate.

Syntax

Telescope.MoveAxis(*Axis*, *Rate*)

The method syntax has these parts:

Part	Description
<i>Axis</i> (TelescopeAxes)	The physical axis about which movement is desired
<i>Rate</i> (Double)	The rate of motion (deg/sec, + = clockwise) about the specified axis
Return (Nothing)	Does not return a value.

Remarks

This method supports control of the mount about its mechanical axes. The telescope will start moving at the specified rate about the specified axis and continue indefinitely. This method can be called for each axis separately, and have them all operate concurrently at separate rates of motion. The sign of the Rate parameter determines the direction of motion, with positive being clockwise and negative being counterclockwise.

Set the rate for an axis to zero to stop the motion about that axis. Tracking motion (if enabled, see note below) is suspended during this mode of operation. Raises an error if AtPark is true.

This must be implemented for the if the [CanMoveAxis](#) property returns True for the given axis.

Notes:

- The movement rate must be within the value(s) obtained from a [Rate](#) object in the the [AxisRates](#) collection. An out of range exception is raised the rate is out of range.
- The value of the [Slewing](#) property must be True if the telescope is moving about *any* of its axes as a result of this method being called.
- This can be used to simulate a handbox by initiating motion with the MouseDown event and stopping the motion with the MouseUp event.
- When the motion is stopped the scope will be set to the previous TrackingRate or to no movement, depending on the state of the [Tracking](#) property.
- It may be possible to implement satellite tracking by using the MoveAxis() method to move the scope in the required manner to track a satellite.

Symbolic Constants

The (symbolic) values for **TelescopeAxes** are:

Constant	Value	Description
axisPrimary	0	Primary axis (e.g., Right Ascension or Azimuth)
axisSecondary	1	Secondary axis (e.g., Declination or Altitude)
axisTertiary	2	Tertiary axis (e.g. imager rotator/de-rotator)

Telescope.Park() Method

Optional

Move the telescope to its park position, stop all motion (or restrict to a small safe range), and set AtPark to True.

Syntax

Telescope.Park()

The method syntax has these parts:

Part	Description
Return (Nothing)	Does not return a value.

Remarks

Raises an error if there is a problem communicating with the telescope or if parking fails.

Parking should put the telescope into a state where its pointing accuracy will not be lost if it is power-cycled (without moving it). Some telescopes *must* be power-cycled before unparking. Others may be unparked by simply calling the [Unpark\(\)](#) method.

Calling this with AtPark = True does nothing (harmless)

Telescope.PulseGuide() Method

Optional

Moves the scope in the given direction for the given interval or time at the rate given by the corresponding guide rate property (see Remarks)

Syntax

Telescope.PulseGuide(*Direction*, *Duration*)

The method syntax has these parts:

Part	Description
<i>Direction</i> (GuideDirections)	The direction in which the guide-rate motion is to be made
<i>Duration</i> (Long)	The duration of the guide-rate motion (milliseconds)
Return (Nothing)	Does not return a value.

Remarks

This method returns immediately if the hardware is capable of back-to-back moves, i.e. dual-axis moves. For hardware not having the dual-axis capability, the method returns only after the move has completed.

Raises an error if AtPark is true.

The [IsPulseGuiding](#) property must be be True during pulse-guiding.

The rate of motion for movements about the right ascension axis is specified by the [GuideRateRightAscension](#) property. The rate of motion for movements about the declination

axis is specified by the [GuideRateDeclination](#) property. These two rates may be tied together into a single rate, depending on the driver's implementation and the capabilities of the telescope.

Symbolic Constants

The (symbolic) values for **GuideDirections** are:

Constant	Value	Description
guideNorth	0	North (+ declination/elevation)
guideSouth	1	South (- declination/elevation)
guideEast	2	East (+ right ascension/azimuth)
guideWest	3	West (- right ascension/azimuth)

Telescope.SetPark() Method

Optional

Sets the telescope's park position to be its current position

Syntax

Telescope.SetPark()

The method syntax has these parts:

Part	Description
Return (Nothing)	Does not return a value.

Remarks

Raises an error if there is a problem.

Telescope.SetupDialog() Method

Required

Displays a setup dialog, allowing the user to set telescope-specific values such as baud rate, geodetic position, etc.

Syntax

Telescope.SetupDialog()


The method syntax has these parts:

Part	Description
Return (Nothing)	Does not return a value.

Remarks

If there are no setup items, a simple popup box with the driver name and version should be displayed, along with a message that no setup or configuration is required. Drivers may raise an error if the telescope is connected when this method is called (as a way of preventing config changes while the driver is active).

The SetupDialog must contain, at a minimum, two graphical elements:

- An ASCOM icon (shown on the right), which is hyperlinked to the ASCOM Initiative web site. When the mouse hovers over this icon, it should change to the standard web-link icon . A click on the icon must open a web browser to the ASCOM Initiative home page at <http://ASCOM-Standards.org/>. The image should also display a hover tool-tip that indicates that the image is a hyperlink to the ASCOM Initiative web site.



- A label that appears as [Help](#). This label must behave as a hyperlink (as described in the preceding item) to a local HTML document, included with the driver, and which describes its behavior and other information that may be useful to an end-user of the driver.

NOTE: The setup dialog typically includes selection of the serial (COM) port to be used for telescope communication. If this is implemented in the setup dialog, then it should include the ability to dynamically discover and offer for selection all COM ports on the system. If dynamic discovery is not included, then at least 32 COM ports should be offered for selection. This is necessary because most newer systems use USB-to-serial adapters, and these devices may appear on high-numbered COM ports. Again, dynamic discovery is preferred.

Telescope.SlewToAltAz() Method

Optional

Move the telescope to the given local horizontal coordinates, return when slew is complete

Syntax

Telescope.SlewToAltAz(*Azimuth*, *Altitude*)

The method syntax has these parts:

Part	Description
<i>Azimuth</i> (Double)	Target azimuth (degrees, North-referenced, positive East/clockwise).
<i>Altitude</i> (Double)	Target altitude (degrees, positive up)
Return (Nothing)	Does not return a value.

Remarks

This Method must be implemented if [CanSlewAltAz](#) returns True. Raises an error if the slew fails.

The slew may fail if the target coordinates are beyond limits imposed within the driver component. Such limits include mechanical constraints imposed by the mount or attached instruments, building or dome enclosure restrictions, etc.

The TargetRightAscension and TargetDeclination properties are not changed by this method.

Raises an error if AtPark is True, or if Tracking is True.

Copyright © 2001-2008, The ASCOM Initiative

Telescope.SlewToAltAzAsync() Method

Optional

Move the telescope to the given local horizontal coordinates, return immediately after starting the slew.

Syntax

Telescope.SlewToAltAzAsync(*Azimuth*, *Altitude*)

The method syntax has these parts:

Part	Description
<i>Azimuth</i> (Double)	Target azimuth (degrees, North-referenced, positive East/clockwise).
<i>Altitude</i> (Double)	Target altitude (degrees, positive up)
Return (Nothing)	Does not return a value.

Remarks

This Method must be implemented if [CanSlewAltAzAsync](#) returns True. This method should only be implemented if the properties Altitude, Azimuth, Right Ascension, Declination and Slewing can be read while the scope is slewing. Raises an error if *starting* the slew fails.

Returns immediately after starting the slew. The client may monitor the progress of the slew by reading the Azimuth, Altitude, and Slewing properties during the slew. When the slew completes, Slewing becomes False.

The slew may fail if the target coordinates are beyond limits imposed within the driver

component. Such limits include mechanical constraints imposed by the mount or attached instruments, building or dome enclosure restrictions, etc.

The TargetRightAscension and TargetDeclination properties are not changed by this method.

Raises an error if AtPark is True, or if Tracking is True.

Copyright © 2001-2008, The ASCOM Initiative

Telescope.SlewToCoordinates() Method

Optional

Move the telescope to the given equatorial coordinates, return when slew is complete

Syntax

Telescope.SlewToCoordinates(*RightAscension*, *Declination*)

The method syntax has these parts:

Part	Description
<i>RightAscension</i> (Double)	The destination right ascension (hours). Copied to Telescope.TargetRightAscension.
<i>Declination</i> (Double)	The destination declination (degrees, positive North). Copied to Telescope.TargetDeclination.
Return (Nothing)	Does not return a value.

Remarks

This Method must be implemented if [CanSlew](#) returns True. Raises an error if the slew fails.

The slew may fail if the target coordinates are beyond limits imposed within the driver component. Such limits include mechanical constraints imposed by the mount or attached instruments, building or dome enclosure restrictions, etc. The target coordinates are copied to Telescope.TargetRightAscension and Telescope.TargetDeclination whether or not the slew succeeds.

Raises an error if AtPark is True, or if Tracking is False.

Telescope.SlewToCoordinatesAsync() Method

Optional

Move the telescope to the given equatorial coordinates, return immediately after starting the slew.

Syntax

Telescope.SlewToCoordinatesAsync(*RightAscension*, *Declination*)

The method syntax has these parts:

Part	Description
<i>RightAscension</i> (Double)	The destination right ascension (hours). Copied to Telescope.TargetRightAscension.
<i>Declination</i> (Double)	The destination declination (degrees, positive North). Copied to Telescope.TargetDeclination.
Return (Nothing)	Does not return a value.

Remarks

This Method must be implemented if [CanSlewAsync](#) returns True. Raises an error if *starting* the slew failed.

Returns immediately after starting the slew. The client may monitor the progress of the slew by reading the RightAscension, Declination, and Slewing properties during the slew. When the slew completes, Slewing becomes False.

The slew may fail to start if the target coordinates are beyond limits imposed within the

driver component. Such limits include mechanical constraints imposed by the mount or attached instruments, building or dome enclosure restrictions, etc. The target coordinates are copied to TargetRightAscension and TargetDeclination whether or not the slew succeeds.

Raises an error if AtPark is True, or if Tracking is False.

Copyright © 2001-2008, The ASCOM Initiative

Telescope.SlewToTarget() Method

Optional

Move the telescope to the TargetRightAscension and TargetDeclination coordinates, return when slew complete.

Syntax

Telescope.SlewToTarget()

The method syntax has these parts:

Part	Description
Return (Nothing)	Does not return a value.

Remarks

This Method must be implemented if [CanSlew](#) returns True. Raises an error if the slew fails.

The slew may fail if the target coordinates are beyond limits imposed within the driver component. Such limits include mechanical constraints imposed by the mount or attached instruments, building or dome enclosure restrictions, etc.

Raises an error if AtPark is True, or if Tracking is False.

Telescope.SlewToTargetAsync() Method

Optional

Move the telescope to the TargetRightAscension and TargetDeclination coordinates, returns immediately after starting the slew.

Syntax

Telescope.SlewToTargetAsync()

The method syntax has these parts:

Part	Description
Return (Nothing)	Does not return a value.

Remarks

This Method must be implemented if [CanSlewAsync](#) returns True. Raises an error if *starting* the slew failed.

Returns immediately after starting the slew. The client may monitor the progress of the slew by reading the RightAscension, Declination, and Slewing properties during the slew. When the slew completes, Slewing becomes False.

The slew may fail to start if the target coordinates are beyond limits imposed within the driver component. Such limits include mechanical constraints imposed by the mount or attached instruments, building or dome enclosure restrictions, etc.

Raises an error if AtPark is True, or if Tracking is False.

Telescope.SyncToAltAz() Method

Optional

Matches the scope's local horizontal coordinates to the given local horizontal coordinates.

Syntax

Telescope.SyncToAltAz(*Azimuth*, *Altitude*)

The method syntax has these parts:

Part	Description
<i>Azimuth</i> (Double)	Target azimuth (degrees, North-referenced, positive East/clockwise)
<i>Altitude</i> (Double)	Target altitude (degrees, positive up)
Return (Nothing)	Does not return a value.

Remarks

This must be implemented if the [CanSyncAltAz](#) property is True. Raises an error if matching fails.

Raises an error if AtPark is True, or if Tracking is True.

Telescope.SyncToCoordinates() Method

Optional

Matches the scope's equatorial coordinates to the given equatorial coordinates.

Syntax

Telescope.SyncToCoordinates(*RightAscension*, *Declination*)

The method syntax has these parts:

Part	Description
<i>RightAscension</i> (Double)	The corrected right ascension (hours). Copied to the TargetRightAscension property.
<i>Declination</i> (Double)	The corrected declination (degrees, positive North). Copied to the TargetDeclination property.
Return (Nothing)	Does not return a value.

Remarks

This must be implemented if the [CanSync](#) property is True. Sets TargetRightAscension to the given right ascension, and TargetDeclination to the given declination. Raises an error if matching fails.

Raises an error if AtPark is True, or if Tracking is False.

Telescope.SyncToTarget() Method

Optional

Matches the telescope's current equatorial coordinates to TargetRightAscension and TargetDeclination

Syntax

Telescope.SyncToTarget()

The method syntax has these parts:

Part	Description
Return (Nothing)	Does not return a value.

Remarks

This must be implemented if the [CanSync](#) property is True. Raises an error if matching fails.

Raises an error if AtPark is True, or if Tracking is False.

Telescope.Unpark() Method

Optional

Takes telescope out of the Parked state. The state of Tracking after unparking is undetermined. Valid only after Park().

Syntax

Telescope.Unpark()

The method syntax has these parts:

Part	Description
Return (Nothing)	Does not return a value.

Remarks

Applications must check and change Tracking as needed after unparking. Raises an error if unparking fails. Calling this with AtPark = False does nothing (harmless)

Telescope Run-Time Errors

The Telescope object raises trappable errors when it cannot continue its current operation. The Contents tab of this document lists the error messages sorted alphabetically (under Error Messages) and the Index tab lists them by hexadecimal code (under Error Codes). Clicking on either of these entries leads to a page with a brief description of the error condition.

Error codes are Automation/ActiveX compatible 32-bit values, based on FACILITY_ITF (a.k.a. vbObjectError for VBA) with an offset of 400 hex, and are thus compatible with all ActiveX scripting languages, Visual Basic, Visual Basic for Applications (VBA), etc. it is imperative that drivers raise Automation errors with values equal to 80040400 hex and higher.

Drivers are free to choose their error codes and messages, except for one.

All drivers must implement the error "xxx is not implemented in this driver" with error code 80040400 hex. Drivers must report the specific property or method not implemented in the description part of the error object.

(0x80040400) This is the first error message and code

This is a placeholder for the first error message. Each error message added to the list will have a code one greater than the previous. The base is hex 80040400, which is FACILITY_ITF plus 1024, preventing clashes with Visual Basic internal Automation codes.

Copyright © 2001-2008, The ASCOM Initiative