IDeviceControl Proposal

# Objective

I make this proposal to improve ASCOM Platform flexibility and to enable innovation to continue between Platform releases. It documents and builds on suggestions from many people on the ASCOM Talk group. Peter Simpson

# Purpose

The interface enables:

* Native access to device controllers to enable device specific commands to be sent from client applications
* Device type Actions having well known names that perform common actions for particular types of device. To be an Action with a well known name, the action must be intended to be implemented by more than one driver otherwise it is really device specific.

# Implementation

The interface definition as a whole:

* will be implemented as a core part of the platform so that it is available to all
* is device independent (exploitation of course is device and device class dependent)
* is optional and can be implemented by client and driver authors at their discretion.

If this interface is implemented within a driver, all members must be present, it is not permissible to present just one or two members of this interface. Interface members whose functions are not supported must throw ASCOM not implemented exceptions.

When device control actions are supported by a driver, the Action, LastResult and SupportedActions members must all be functional i.e. all must work as described below and none may throw ASCOM not implemented exceptions.

If desired, a driver can implement this interface and make all methods throw ASCOM not implemented exceptions.

# Interface

interface IDeviceControl

{

string Action(string ActionName, string ActionParameters);

string LastResult { get; }

string[] SupportedActions { get; }

void CommandBlind(string Command, bool Raw);

bool CommandBool(string Command, bool Raw);

string CommandString(string Command, bool Raw);

}

### IDeviceControl.Action

**ActionName**:

A well known name agreed by interested parties that represents the action to be carried out.

E.g. suppose filter wheels start to appear with automatic wheel changers; new actions could be “QueryWheels” and “SelectWheel”. The former returning a formatted list of wheel names and the second taking a wheel name and making the change.

It is intended that the SupportedActions method will inform clients of driver capabilities, but the driver must still throw an ASCOM.ActionNotImplemented exception if it is asked to perform an action that it does not support.

**ActionParameters**

List of required parameters or empty string if none are required

**Returns**

A string response and sets the IDeviceControl.LastResult property

**Remarks**

This interface is intended for use in any current or future device type and to avoid name clashes, management of action names is important from day 1. A two-part naming convention will be adopted - DeviceType:UniqueActionName where:

* DeviceType is the same value as would be used by Chooser.DeviceType e.g. Telescope, Camera, Switch etc.
* UniqueActionName is a single word, or multiple words joined by underscore characters, that sensibly describes the action to be performed.

It is recommended that UniqueActionNames should be a maximum of 16 characters for legibility.

Should the same function and UniqueActionName be supported by more than one type of device, the reserved DeviceType of “General” will be used.

Action names will be case insensitive, so FilterWheel:SelectWheel, filterwheel:selectwheel and FILTERWHEEL:SELECTWHEEL will all refer to the same action.

### Property IDeviceControl.LastResult

**Returns**

A result code or message from the last action

**Remarks**

If supported, LastResult should return an ASCOM InvalidOperation exception if read before any action has be received and if read after IDeviceControl.CommandBlind.

Intentionally, there is no pre-conceived interpretation or standard for IDeviceControl.LastResult return values. This is to provide the maximum flexibility for client and driver authors to establish standards meaningful to their applications.

IDeviceControl.LastResult should never return the “null” / “nothing” value, it should always return empty string: “” or throw an exception if the last action had no textual message to return.

### Property IDeviceControl.SuportedActions

**Returns**

A string array of supported Action names.

**Remarks**

This is an aid to client authors and testers who would otherwise have to repeatedly poll the driver to determine its capabilities. Returned action names may be mixed case to enhance presentation but they will be recognised case insensitively in the IDeviceControl.Action method.

### IDeviceControl.CommandBlind

**Command**

The command string to be sent to the device

**Raw**

Boolean value indicating whether delimiters or framing normally added by the driver should be omitted (True) or included (False)

**Returns**

There is no return value.

**Remarks**

This is a fire and forget method that sends device specific commands to the device. IDeviceControl.LastResult is not set and any attempt to read it after this method should throw an exception.

Note that Raw is a mandatory parameter unlike ITelescopeV2 where it is optional.

If the Raw parameter is set True, the driver must not insert or append any delimiters and 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.

### IDeviceControl.CommandBool

**Command**

The command string to be sent to the device

**Raw**

Boolean value indicating whether delimiters or framing normally added by the driver should be omitted (True) or included (False)

**Returns**

True if the device response indicated success, else False

**Remarks**

This behaves the same as Telescope.CommandBool and is a synchronous call that sends device specific commands to the device, returns a boolean response and sets the IDeviceControl.LastResult property.

Note that Raw is a mandatory parameter unlike ITelescopeV2 where it is optional.

If the 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.

### IDeviceControl.CommandString

**Command**

The command string to be sent to the device

**Raw**

Boolean value indicating whether delimiters or framing normally added by the driver should be omitted (True) or included (False)

**Returns**

The response string from the device.

**Remarks**

This is a synchronous call that sends device specific commands to the device, returns a string response and sets the IDeviceControl.LastResult property.

Note that Raw is a mandatory parameter unlike ITelescopeV2 where it is optional.

If the Raw parameter is set True, the driver must not insert or append any delimiters and 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.