DCS-100 Control Library

Generated by Doxygen 1.9.2

1 DCS-100 Control Library 1

2 Usage with Python 3

3 Namespace Index 5

3.1 Packages . . . 5

4 Hierarchical Index 7

4.1 Class Hierarchy . . . 7

5 Class Index 9

5.1 Class List . . . 9

6 File Index 11

6.1 File List . . . 11

7 Namespace Documentation 13

7.1 AdvancedIllumination Namespace Reference . . . 13

7.1.1 Function Documentation . . . 13

7.1.1.1 DisconnectEventHandler() . . . 13

7.1.1.2 Dispose() [1/2] . . . 14

7.1.1.3 Dispose() [2/2] . . . 14

7.1.1.4 ErrorEventHandler() . . . 14

7.1.1.5 FaultEventHandler() . . . 15

7.1.1.6 WarningEventHandler() . . . 15

8 Class Documentation 17

8.1 ChannelConfig Struct Reference . . . 17

8.1.1 Detailed Description . . . 17

8.1.2 Property Documentation . . . 18

8.1.2.1 ContinuousMaximum . . . 18

8.1.2.2 MaxStrobeFrequency . . . 18

8.1.2.3 MinimumCurrent . . . 18

8.1.2.4 OperatingMode . . . 18

8.1.2.5 OutputCurrent . . . 18

8.1.2.6 PulseDelayMicros . . . 19

8.1.2.7 PulseWidthMicros . . . 19

8.1.2.8 StrobeMaximum . . . 19

8.1.2.9 TriggerEdge . . . 19

8.1.2.10 TriggerInput . . . 19

8.2 ChannelLimits Struct Reference . . . 19

8.2.1 Detailed Description . . . 20

8.2.2 Property Documentation . . . 20
8.2.2.1 Channel . . . 20
8.2.2.2 MaxCurrent . . . 20
8.2.2.3 MaxTriggerFrequency . . . 20
8.2.2.4 MinOffTime . . . 21
8.3 DCS100_DeviceConfiguration Class Reference . . . 21
8.3.1 Detailed Description . . . 21
8.3.2 Member Data Documentation . . . 21
8.3.2.1 Channel1 . . . 21
8.3.2.2 Channel2 . . . 22
8.3.2.3 Channel3 . . . 22
8.3.2.4 ProfileNumber . . . 22
8.3.2.5 TriggerMap . . . 22
8.4 AdvancedIllumination.DCS_100 Class Reference . . . 22
8.4.1 Detailed Description . . . 25
8.4.2 Member Enumeration Documentation . . . 25
8.4.2.1 Channel . . . 25
8.4.2.2 MappedTriggerInput . . . 25
8.4.2.3 Mode . . . 25
8.4.2.4 Trigger . . . 26
8.4.3 Constructor & Destructor Documentation . . . 26
8.4.3.1 DCS_100() . . . 26
8.4.4 Member Function Documentation . . . 27
8.4.4.1 Connect() . . . 27
8.4.4.2 Disconnect() . . . 27
8.4.4.3 GetConfiguration() . . . 27
8.4.4.4 GetProfileNames() . . . 28
8.4.4.5 GetProfileNumber() . . . 28
8.4.4.6 LoadProfile() . . . 28
8.4.4.7 OnDisconnection() . . . 30
8.4.4.8 OnError() . . . 30
8.4.4.9 OnFault() . . . 30
8.4.4.10 OnWarning() . . . 31
8.4.4.11 PulseChannel() . . . 31
8.4.4.12 SaveProfile() . . . 31
8.4.4.13 SetCommunicationTarget() . . . 32
8.4.4.14 SetCurrent() . . . 32
8.4.4.15 SetDeviceName() . . . 33
8.4.4.16 SetDeviceStaticIP() . . . 33

8.4.4.17 SetMode() . . . 34
8.4.4.18 SetProfileName() . . . 34
8.4.4.19 SetPulseDelay() . . . 34
8.4.4.20 SetPulseWidth() . . . 35
8.4.4.21 SetTriggerEdge() . . . 35
8.4.4.22 SetTriggerInput() . . . 36
8.4.5 Property Documentation . . . 36
8.4.5.1 DeviceType . . . 36
8.4.5.2 FirmwareVersion . . . 36
8.4.5.3 IsConnected . . . 36
8.4.5.4 Lighthead . . . 37
8.4.5.5 Name . . . 37
8.4.5.6 TargetIP . . . 37
8.4.5.7 Timeout . . . 37
8.4.6 Event Documentation . . . 37
8.4.6.1 Disconnection . . . 37
8.4.6.2 Error . . . 38
8.4.6.3 Fault . . . 38
8.4.6.4 Warning . . . 38
8.5 DCS_Information Class Reference . . . 38
8.5.1 Detailed Description . . . 39
8.5.2 Member Function Documentation . . . 39
8.5.2.1 Equals() . . . 39
8.5.2.2 GetHashCode() . . . 40
8.5.2.3 ToString() . . . 40
8.5.3 Property Documentation . . . 40
8.5.3.1 FirmwareVersion . . . 40
8.5.3.2 IP_Address . . . 40
8.5.3.3 Lighthead . . . 41
8.5.3.4 Name . . . 41
8.5.3.5 Type . . . 41
8.5.3.6 WebConfigEnabled . . . 41
8.6 DisconnectEventArgs Class Reference . . . 41
8.6.1 Detailed Description . . . 42
8.6.2 Property Documentation . . . 42
8.6.2.1 IncidentTime . . . 42
8.7 ErrorEventArgs Class Reference . . . 42
8.7.1 Detailed Description . . . 43
8.7.2 Property Documentation . . . 43

8.7.2.1 IncidentChannel . . . 43
8.7.2.2 Message . . . 43
8.8 FaultEventArgs Class Reference . . . 43
8.8.1 Detailed Description . . . 44
8.8.2 Property Documentation . . . 44
8.8.2.1 FaultedChannel . . . 44
8.9 TriggerMapping Struct Reference . . . 44
8.9.1 Detailed Description . . . 44
8.9.2 Property Documentation . . . 44
8.9.2.1 Edge . . . 45
8.9.2.2 Input . . . 45
8.10 WarningEventArgs Class Reference . . . 45
8.10.1 Detailed Description . . . 45
8.10.2 Property Documentation . . . 46
8.10.2.1 IncidentChannel . . . 46
8.10.2.2 Message . . . 46

9 File Documentation **47**
9.1 DCS_100.cs File Reference . . . 47
9.1.1 Enumeration Type Documentation . . . 48
9.1.1.1 DeviceType . . . 48
9.2 python.md File Reference . . . 48
9.3 readme.md File Reference . . . 48

Index **49**

Chapter 1

DCS-100 Control Library

This SDK allows easy interfacing between the Advanced illumination DCS-100E and DCS-103 Strobe Controllers and code written for the .NET Framework.

This libraries in this SDK require .NET Framework 4.5+, as it relies on asynchronous methods added in that release.

Example code: (C#)

```
using System;
using AdvancedIllumination;
namespace DCS_100_Example
{
    class Program
    {
        static void Main(string[] args)
        {
            //Get list of online devices
            var devices = DCS_100.GetActiveDevices(true);
            if(devices.Count > 0)
            {
                //Connect to first device
                DCS_100 dcs = new DCS_100(devices[0].IP_Address);
                dcs.Connect();

                //Set mode on first channel to continuous
                dcs.SetMode(DCS_100.Channel.One, DCS_100.Mode.Continuous);
                //Set current on first channel to 50 mA
                dcs.SetCurrent(DCS_100.Channel.One, 50);
                //Set mode to strobe
                dcs.SetMode(DCS_100.Channel.One, DCS_100.Mode.Strobe);
                //Set current to 500 mA
                dcs.SetCurrent(DCS_100.Channel.One, 500);
                //Set pulse width to 10 ms (parameter two is in microseconds)
                ChannelLimits lim = dcs.SetPulseWidth(DCS_100.Channel.One, 10000);
                //The ChannelLimits contain the channel's maximum current at the specified pulse width,
                // as well as the minimum off time and equivalent maximum trigger frequency to avoid
                // damaging the light. The DCS will ignore any attempt to set values beyond these limits.
                //Warnings and errors sent from the DCS can come at any time, so the DCS object has an event for
    them.
                // If a warning is received, the unit is most likely still operational and can still be
    controlled.
                dcs.Warning += Dcs_Warning;
                // If an error is received, the communication path may have been cut.
                dcs.Error += Dcs_Error;
                dcs.Disconnect();
            }
            //The DCS_100 class implements IDisposable, so for a temporary connection you can do this:
            using (DCS_100 dcs = new DCS_100("192.168.0.100"))
            {
                dcs.SetCurrent(DCS_100.Channel.One, 50);
            }
```

```
        }
        private static void Dcs_Error(object sender, ErrorEventArgs e)
        {
            Console.WriteLine("DCS Error!");
        }
        private static void Dcs_Warning(object sender, WarningEventArgs e)
        {
            Console.WriteLine("DCS Warning!");
        }
    }
}
```

Chapter 2

Usage with Python

The DCS-100 SDK can be used in Python code with either `IronPython` or `Python.NET`. The usage is a little different between the two.

IronPython

```
import os
import clr
#Change directory to the DCS SDK directory
os.chdir(r"C:\Path\To\DCS\SDK")
clr.AddReferenceToFileAndPath("DCS100.dll")
import AdvancedIllumination as ai #A shorter name for convenience
#Also more convenient
Channel = ai.DCS_100.Channel
Mode = ai.DCS_100.Mode
devices = ai.DCS_100.GetActiveDevices(True)
print ", ".join(map(lambda x: x.IP_Address, devices))
device = ai.DCS_100(devices[0].IP_Address) #Create the device
device.Connect() #Don't forget the call to Connect to initialize the connection
device.SetMode(Channel.One, Mode.Strobe) #Set strobe mode
device.SetPulseWidth(Channel.One, 500) #Set pulse width 500 us
device.SetCurrent(Channel.One, 200) #Set current to 200 mA
device.PulseChannel(Channel.One) #Trigger a pulse on channel 1
```

Python.NET

```
import sys
import clr
#Change directory to the DCS SDK directory
sys.path.append(r"C:\Path\To\DCS\SDK")
clr.AddReference("DCS100")
import AdvancedIllumination as ai #A shorter name
#Also more convenient
Channel = ai.DCS_100.Channel
Mode = ai.DCS_100.Mode
devices = ai.DCS_100.GetActiveDevices(True)
print ", ".join(map(lambda x: x.IP_Address, devices));
device = ai.DCS_100(devices[0].IP_Address) #Create the device
device.Connect() #Don't forget the call to Connect to initialize the connection
device.SetMode(Channel.One, Mode.Strobe) #Set strobe mode
device.SetPulseWidth(Channel.One, 500) #Set pulse width 500 us
device.SetCurrent(Channel.One, 200) #Set current to 200 mA
device.PulseChannel(Channel.One) #Trigger a pulse on channel 1
```

Don't forget to install Python.NET with `pip install pythonnet` on the command line (assuming Python is in your `PATH`).

Note: versions of the SDK older than 1.0.740.147 may experience issues related to method overloads in Python.NET. To alleviate this, the SetPulseWidth and SetPulseDelay overloads that take a TimeSpan were removed from 1.0.740.147+.

Chapter 3

Namespace Index

3.1 Packages

Here are the packages with brief descriptions (if available):

AdvancedIllumination 13

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

- ChannelConfig . . . 17
- ChannelLimits . . . 19
- DCS100_DeviceConfiguration . . . 21
- DCS_Information . . . 38
- EventArgs
  - DisconnectEventArgs . . . 41
  - ErrorEventArgs . . . 42
  - FaultEventArgs . . . 43
  - WarningEventArgs . . . 45
- IDisposable
  - AdvancedIllumination.DCS_100 . . . 22
- TriggerMapping . . . 44

Chapter 5

Class Index

5.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

ChannelConfig
: Struct which represents the current configuration of a certain channel. 17

ChannelLimits
: Encapsulates the lighthead safety limits defined by Advanced illumination's Signatech© II technology. 19

DCS100_DeviceConfiguration
: Contains all the information that defines the state of a DCS-100E controller. 21

AdvancedIllumination.DCS_100
: Class representing one Advanced Illumination DCS-100 controller 22

DCS_Information
: Class that encapsulates basic information on a DCS-100E controller. 38

DisconnectEventArgs
: Additional information for disconnection events. 41

ErrorEventArgs
: Class which represents the details of an error signal event for the DCS-100E. 42

FaultEventArgs
: Class which represents the details of a fault event for the DCS-100E. 43

TriggerMapping
: Encapsulates information related to trigger mapping. 44

WarningEventArgs
: Class which represents the details of a warning signal event for the DCS-100E. 45

Chapter 6

File Index

6.1 File List

Here is a list of all files with brief descriptions:

DCS_100.cs 47

Chapter 7

Namespace Documentation

7.1 AdvancedIllumination Namespace Reference

Classes

- class DCS_100

  Class representing one Advanced Illumination DCS-100 controller

Functions

- delegate void FaultEventHandler (object sender, FaultEventArgs e)

  Delegate for the handler for the DCS_100.Fault event. Instantiate this delegate for your own event handler.
- delegate void WarningEventHandler (object sender, WarningEventArgs e)

  Delegate for the handler for the DCS_100.Warning event. Instantiate this delegate for your own event handler.
- delegate void ErrorEventHandler (object sender, ErrorEventArgs e)

  Delegate for the handler of the DCS_100.Error event. Instantiate this delegate for your own event handler.
- delegate void DisconnectEventHandler (object sender, DisconnectEventArgs de)

  Delegate for handling Disconnect events.
- virtual void Dispose (bool disposing)

  Disconnect from the target and dispose of the object.
- void Dispose ()

  Disconnect from the target and dispose of the object.

7.1.1 Function Documentation

7.1.1.1 DisconnectEventHandler()

```
delegate void AdvancedIllumination.DisconnectEventHandler (
            object sender,
            DisconnectEventArgs de )
```

Delegate for handling Disconnect events.

Parameters

sender	Source of the event.
de	Disconnect event information.

See also: Delegates

7.1.1.2 Dispose() [1/2]

```
void AdvancedIllumination.Dispose ( )
```

Disconnect from the target and dispose of the object.

7.1.1.3 Dispose() [2/2]

```
virtual void AdvancedIllumination.Dispose (
            bool disposing )  [protected], [virtual]
```

Disconnect from the target and dispose of the object.

Parameters

disposing	`true` if disposing of the object, `false` otherwise.

7.1.1.4 ErrorEventHandler()

```
delegate void AdvancedIllumination.ErrorEventHandler (
            object sender,
            ErrorEventArgs e )
```

Delegate for the handler of the DCS_100.Error event. Instantiate this delegate for your own event handler.

Parameters

sender	object that represents the sender
e	Event information, i.e. the channel that is faulted

See also: Delegates

7.1.1.5 FaultEventHandler()

```
delegate void AdvancedIllumination.FaultEventHandler (
            object sender,
            FaultEventArgs e )
```

Delegate for the handler for the DCS_100.Fault event. Instantiate this delegate for your own event handler.

Parameters

sender	object that represents the sender
e	Event information, i.e. the channel that is faulted

See also: Delegates

7.1.1.6 WarningEventHandler()

```
delegate void AdvancedIllumination.WarningEventHandler (
            object sender,
            WarningEventArgs e )
```

Delegate for the handler for the DCS_100.Warning event. Instantiate this delegate for your own event handler.

Parameters

sender	object that represents the sender
e	Event information, i.e. the channel that is faulted

See also: Delegates

Chapter 8

Class Documentation

8.1 ChannelConfig Struct Reference

Struct which represents the current configuration of a certain channel.

Properties

- int OutputCurrent `[getset]`

  Defines the output level (brightness) of the channel.
- DCS_100.Mode OperatingMode `[getset]`

  Defines the operating mode (strobe or continuous) of the channel.
- DCS_100.Trigger TriggerEdge `[getset]`

  Defines the trigger state of the channel, either rising or falling edge.
- DCS_100.Channel TriggerInput `[getset]`

  Specifies on which input this channel is triggering.
- uint PulseDelayMicros `[getset]`

  Defines the pulse delay (in microseconds) of the controller. Only applicable to strobe mode.
- uint PulseWidthMicros `[getset]`

  Defines the pulse width (in microseconds) of the controller. Only applicable to strobe mode.
- int ContinuousMaximum `[getset]`

  Defines the maximum possible current that this channel can sustain in continuous DCS_100.Mode.
- int StrobeMaximum `[getset]`

  Defines the maximum possible current that this channel can sustain in strobe DCS_100.Mode.
- double MaxStrobeFrequency `[getset]`

  Defines the maximum possible trigger frequency for this channel.
- int MinimumCurrent `[getset]`

  Defines the minimum possible current for this unit.

8.1.1 Detailed Description

Struct which represents the current configuration of a certain channel.

8.1.2 Property Documentation

8.1.2.1 ContinuousMaximum

```
int ChannelConfig.ContinuousMaximum  [get], [set]
```

Defines the maximum possible current that this channel can sustain in continuous DCS_100.Mode.

8.1.2.2 MaxStrobeFrequency

```
double ChannelConfig.MaxStrobeFrequency  [get], [set]
```

Defines the maximum possible trigger frequency for this channel.

8.1.2.3 MinimumCurrent

```
int ChannelConfig.MinimumCurrent  [get], [set]
```

Defines the minimum possible current for this unit.

8.1.2.4 OperatingMode

```
DCS_100.Mode ChannelConfig.OperatingMode  [get], [set]
```

Defines the operating mode (strobe or continuous) of the channel.

8.1.2.5 OutputCurrent

```
int ChannelConfig.OutputCurrent  [get], [set]
```

Defines the output level (brightness) of the channel.

8.1.2.6 PulseDelayMicros

```
uint ChannelConfig.PulseDelayMicros  [get], [set]
```

Defines the pulse delay (in microseconds) of the controller. Only applicable to strobe mode.

8.1.2.7 PulseWidthMicros

```
uint ChannelConfig.PulseWidthMicros  [get], [set]
```

Defines the pulse width (in microseconds) of the controller. Only applicable to strobe mode.

8.1.2.8 StrobeMaximum

```
int ChannelConfig.StrobeMaximum  [get], [set]
```

Defines the maximum possible current that this channel can sustain in strobe DCS_100.Mode.

8.1.2.9 TriggerEdge

```
DCS_100.Trigger ChannelConfig.TriggerEdge  [get], [set]
```

Defines the trigger state of the channel, either rising or falling edge.

8.1.2.10 TriggerInput

```
DCS_100.Channel ChannelConfig.TriggerInput  [get], [set]
```

Specifies on which input this channel is triggering.

The documentation for this struct was generated from the following file:

- DCS_100.cs

8.2 ChannelLimits Struct Reference

Encapsulates the lighthead safety limits defined by Advanced illumination's Signatech© II technology.

Properties

- DCS_100.Channel Channel [getset]

  The DCS_100.Channel for which these limits apply.
- ushort MaxCurrent [getset]

  Maximum current limit of the channel.
- double MaxTriggerFrequency [getset]

  Maximum trigger frequency for this lighthead.
- uint MinOffTime [getset]

  Minimum off time, or time between pulses, for safe operation of the light.

8.2.1 Detailed Description

Encapsulates the lighthead safety limits defined by Advanced illumination's Signatech© II technology.

8.2.2 Property Documentation

8.2.2.1 Channel

```
DCS_100.Channel ChannelLimits.Channel  [get], [set]
```

The DCS_100.Channel for which these limits apply.

8.2.2.2 MaxCurrent

```
ushort ChannelLimits.MaxCurrent  [get], [set]
```

Maximum current limit of the channel.

8.2.2.3 MaxTriggerFrequency

```
double ChannelLimits.MaxTriggerFrequency  [get], [set]
```

Maximum trigger frequency for this lighthead.

The maximum trigger frequency is determined by Advanced illumination's Signatech© II to protect the lighthead from damage. Any triggers at a frequency higher than this limit will be ignored by the controller.

8.2.2.4 MinOffTime

```
uint ChannelLimits.MinOffTime  [get], [set]
```

Minimum off time, or time between pulses, for safe operation of the light.

The minimum trigger off time is determined by Advanced illumination's Signatech© II to protect the lighthead from damage. Any triggers that recur sooner than this limit will be ignored by the controller.

The documentation for this struct was generated from the following file:

- DCS_100.cs

8.3 DCS100_DeviceConfiguration Class Reference

Contains all the information that defines the state of a DCS-100E controller.

Public Attributes

- ChannelConfig Channel1
  Information for Channel 1.
- ChannelConfig Channel2
  Information for Channel 2.
- ChannelConfig Channel3
  Information for Channel 2.
- TriggerMapping TriggerMap
  Information related to the trigger mapping setup.
- int ProfileNumber
  Indicates which profile number is currently loaded on the DCS-100E.

8.3.1 Detailed Description

Contains all the information that defines the state of a DCS-100E controller.

8.3.2 Member Data Documentation

8.3.2.1 Channel1

```
ChannelConfig DCS100_DeviceConfiguration.Channel1
```

Information for Channel 1.

8.3.2.2 Channel2

`ChannelConfig DCS100_DeviceConfiguration.Channel2`

Information for Channel 2.

8.3.2.3 Channel3

`ChannelConfig DCS100_DeviceConfiguration.Channel3`

Information for Channel 2.

8.3.2.4 ProfileNumber

`int DCS100_DeviceConfiguration.ProfileNumber`

Indicates which profile number is currently loaded on the DCS-100E.

8.3.2.5 TriggerMap

`TriggerMapping DCS100_DeviceConfiguration.TriggerMap`

Information related to the trigger mapping setup.

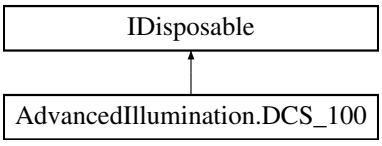
The documentation for this class was generated from the following file:

- DCS_100.cs

8.4 AdvancedIllumination.DCS_100 Class Reference

Class representing one Advanced Illumination DCS-100 controller

Inheritance diagram for AdvancedIllumination.DCS_100:



Public Types

- enum Mode { Off = 0 , Continuous = 1 , Strobe = 2 , Gated = 3 }

  Enumeration defining the operating mode for each channel
- enum Channel { One = 1 , Two , Three }

  Defines which channels are available with the DCS-100 controller
- enum Trigger { Rising = 1 , Falling = 0 }

  Define which trigger state a given channel is in; either rising edge ("Rising") or falling edge ("Falling")
- enum MappedTriggerInput { None = 0 , One , Two , Three }

  Defines which channel's input trigger used for mapping.

Public Member Functions

- DCS_100 (string where, ushort port=7777)

  Constructs a new instance of a DCS-100 controller with a given IP Address
- void Connect (ushort localPort=0)

  Attempts to connect to a DCS-100 over UDP. Note: the target device must be specified beforehand. See DCS_100(string, ushort) or SetCommunicationTarget(string) for how to do this.
- void Disconnect ()

  Instructs the target DCS-100E to accept connection requests from other library instances.
- ChannelLimits SetCurrent (Channel chan, ushort val)

  Sets output current on a channel.
- ChannelLimits SetMode (Channel chan, Mode m)

  Sets operating mode (either continuous or strobe) for a channel
- ChannelLimits SetPulseWidth (Channel chan, uint us)

  Set pulse width on channel, in raw microseconds for precision.
- void SetPulseDelay (Channel chan, uint us)

  Set delay between trigger and output pulse, in raw microseconds for precision.
- void SetCommunicationTarget (string target)

  Set remote device IP address for communication.
- void SetTriggerEdge (Channel chan, Trigger trig)

  Sets the trigger mode on the specified channel.
- void SetTriggerInput (Channel channel, Channel input)

  Maps the specified channel to follow the specified trigger input.
- void PulseChannel (DCS_100.Channel chan)

  Instructs the DCS-100 to strobe the given channel once.
- void SetDeviceName (string name)

  Sets the name of the device. The DCS-100 is limited to 31-character names.
- void SetDeviceStaticIP (string ip)

  Sets the device to a new static IP address.
- DCS100_DeviceConfiguration GetConfiguration ()

  Gets the current configuration (current level, pulse width, pulse delay, etc.) of all channels.
- string[ ] GetProfileNames ()

  Gets a list of the names of the DCS profiles.
- void SaveProfile (byte profileNumber)

  Saves the current configuration of the controller to the given profile number.
- void SetProfileName (string name)

Sets the name of the currently active profile.

- void LoadProfile (byte profileNumber)

  Loads the given profile from memory and sets the device's configuration to it.

- byte GetProfileNumber ()

  Get the number of the currently active configuration profile.

Protected Member Functions

- virtual void OnFault (FaultEventArgs e)

  Method that fires the DCS_100.Fault event.

- virtual void OnWarning (WarningEventArgs e)

  Method that fires the DCS_100.Warning event.

- virtual void OnError (ErrorEventArgs e)

  Method that fires the DCS_100.Error event.

- virtual void OnDisconnection (DisconnectEventArgs de)

  Raises the disconnect event.

Properties

- DeviceType DeviceType `[get]`

  Gets or sets the type of connected device

- bool IsConnected `[get]`

  Indicates whether or not the class instance is currently connected to a DCS-100E. ***true*** *if it is, and* ***false*** *otherwise.*

- string TargetIP `[getset]`

  Gets or sets the IP address of the target device.

- int Timeout `[getset]`

  Gets or sets the timeout, in milliseconds, for communications with the target device.

- string Name `[get]`

  The device's name. Null if not connected.

- string Lighthead `[get]`

  Gets the part number of the connected lighthead.

- int FirmwareVersion `[get]`

  Gets version of the firmware running on the DCS-100

Events

- FaultEventHandler Fault

  Event that indicates a fault warning was received from the connected DCS-100E unit.

- WarningEventHandler Warning

  Event that indicates a general warning message was received from the connected DCS-100E unit.

- ErrorEventHandler Error

  Event that indicates a general error message was received from the connected DCS-100E unit.

- DisconnectEventHandler Disconnection

  Event queue for all listeners interested in Disconnection events.

8.4.1 Detailed Description

Class representing one Advanced Illumination DCS-100 controller

8.4.2 Member Enumeration Documentation

8.4.2.1 Channel

`enum AdvancedIllumination.DCS_100.Channel`

Defines which channels are available with the DCS-100 controller

Enumerator

One	The first channel.
Two	The second channel.
Three	The third channel.

8.4.2.2 MappedTriggerInput

`enum AdvancedIllumination.DCS_100.MappedTriggerInput`

Defines which channel's input trigger used for mapping.

Enumerator

None	Defines no trigger mapping
One	Indicates Channel 1's trigger input.
Two	Indicates Channel 2's trigger input.
Three	Indicates Channel 3's trigger input.

8.4.2.3 Mode

`enum AdvancedIllumination.DCS_100.Mode`

Enumeration defining the operating mode for each channel

Enumerator

Off	Off mode, keeps the channel off.
Continuous	Continuous mode, always on DC current
Strobe	Strobe mode, fixed DC pulse for however long is set with SetPulsewidth. In strobe mode, the maximum current is 10A (10000 mA). The light only flashes for the specified pulse width, after the specified pulse delay. Each pulse is triggered by a pulse on the Channel's respective trigger pin.
Gated	Gated mode. When the respective trigger is held high, the channel is on. When it is low, the channel is off.

8.4.2.4 Trigger

```
enum AdvancedIllumination.DCS_100.Trigger
```

Define which trigger state a given channel is in; either rising edge ("Rising") or falling edge ("Falling")

Enumerator

Rising	Defines a rising-edge trigger
Falling	Defines a falling-edge trigger

8.4.3 Constructor & Destructor Documentation

8.4.3.1 DCS_100()

```
AdvancedIllumination.DCS_100.DCS_100 (
            string where,
            ushort port = 7777 )
```

Constructs a new instance of a DCS-100 controller with a given IP Address

Parameters

where	IP address for the controller (example: "192.168.1.55")
port	UDP port number to use to communicate with the DCS-100E. Defaults to 7777.

Exceptions

System.ArgumentException	Thrown when the given parameter 'where' is not a valid IP address string

8.4.4 Member Function Documentation

8.4.4.1 Connect()

```
void AdvancedIllumination.DCS_100.Connect (
            ushort localPort = 0 )
```

Attempts to connect to a DCS-100 over UDP. Note: the target device must be specified beforehand. See DCS_100(string, ushort) or SetCommunicationTarget(string) for how to do this.

Parameters

localPort	The local UDP port number to use.

If the parameter 'localPort' is equal to zero, then a new port is allocated at random from the IANA ephemeral port range 49152 to 65535. This is the default behavior if no port number is specified.

Exceptions

IOException	Thrown when connection over UDP fails, for any reason.

8.4.4.2 Disconnect()

```
void AdvancedIllumination.DCS_100.Disconnect ( )
```

Instructs the target DCS-100E to accept connection requests from other library instances.

8.4.4.3 GetConfiguration()

```
DCS100_DeviceConfiguration AdvancedIllumination.DCS_100.GetConfiguration ( )
```

Gets the current configuration (current level, pulse width, pulse delay, etc.) of all channels.

Returns

An array of ChannelConfig structures containing the configuration data. Index 0 corresponds to Channel 1, index 1 to Channel 2, and index 2 to Channel 3.

Exceptions

System.Xml.XmlException	Thrown when the XML sent from the controller does not contain the required information. This should never happen.

8.4.4.4 GetProfileNames()

```
string[] AdvancedIllumination.DCS_100.GetProfileNames ( )
```

Gets a list of the names of the DCS profiles.

Returns

An array of six (6) strings, the names of each profile number (0 to 5)

8.4.4.5 GetProfileNumber()

```
byte AdvancedIllumination.DCS_100.GetProfileNumber ( )
```

Get the number of the currently active configuration profile.

Returns

The number (0-5) of the currently active profile, as a byte (8-bit unsigned integer)

See also

SaveProfile

Exceptions

System.Exception	Thrown when the DCS-100 does not properly respond to the query.

8.4.4.6 LoadProfile()

```
void AdvancedIllumination.DCS_100.LoadProfile (
            byte profileNumber )
```

Loads the given profile from memory and sets the device's configuration to it.

Parameters

profileNumber	The profile number (0 through 5) to load.

See SaveProfile for more information.

Exceptions

System.ArgumentException	Thrown when the argument exceeds 5.

8.4.4.7 OnDisconnection()

```
virtual void AdvancedIllumination.DCS_100.OnDisconnection (
            DisconnectEventArgs de )  [protected], [virtual]
```

Raises the disconnect event.

Parameters

de	Event information to send to registered event handlers.

8.4.4.8 OnError()

```
virtual void AdvancedIllumination.DCS_100.OnError (
            ErrorEventArgs e )  [protected], [virtual]
```

Method that fires the DCS_100.Error event.

Parameters

e	Event arguments for the error. Indicates which channel has the error, etc.

8.4.4.9 OnFault()

```
virtual void AdvancedIllumination.DCS_100.OnFault (
            FaultEventArgs e )  [protected], [virtual]
```

Method that fires the DCS_100.Fault event.

Parameters

e	Event arguments for event. Indicates which channel is faulted.

8.4.4.10 OnWarning()

```
virtual void AdvancedIllumination.DCS_100.OnWarning (
            WarningEventArgs e )  [protected], [virtual]
```

Method that fires the DCS_100.Warning event.

Parameters

e	Event arguments for event. Indicates which channel has the warning, etc.

8.4.4.11 PulseChannel()

```
void AdvancedIllumination.DCS_100.PulseChannel (
            DCS_100.Channel chan )
```

Instructs the DCS-100 to strobe the given channel once.

Parameters

chan	The channel to strobe.

8.4.4.12 SaveProfile()

```
void AdvancedIllumination.DCS_100.SaveProfile (
            byte profileNumber )
```

Saves the current configuration of the controller to the given profile number.

Parameters

profileNumber	The profile number (0 through 24) to save.

There are twenty-four (24) configuration profiles available for use, numbered zero (0) through twenty-four (24). Each profile saves the following information:

- The current set on each channel
- The operating mode (continuous or strobe) of each channel
- The pulse delay on each channel
- The pulse width on each channel
- The trigger setting on each channel

Exceptions

System.ArgumentException	Thrown when the argument exceeds 24.

8.4.4.13 SetCommunicationTarget()

```
void AdvancedIllumination.DCS_100.SetCommunicationTarget (
            string target )
```

Set remote device IP address for communication.

Parameters

target	IP address of the target device

This function does not connect to the target. Use Connect(ushort) for that.

8.4.4.14 SetCurrent()

```
ChannelLimits AdvancedIllumination.DCS_100.SetCurrent (
            Channel chan,
            ushort val )
```

Sets output current on a channel.

Parameters

chan	Channel to set.
val	Output current value, in milliamps.

Returns

Updated information regarding the limits for safe operation of this light channel.

Exceptions

ArgumentOutOfRangeException	Thrown when the argument is beyond the maximum allowable current (10A in strobe mode)

8.4.4.15 SetDeviceName()

```
void AdvancedIllumination.DCS_100.SetDeviceName (
            string name )
```

Sets the name of the device. The DCS-100 is limited to 31-character names.

Parameters

name	The device's name

Exceptions

System.ArgumentException	Thrown when the device name exceeds 31 characters in length.

8.4.4.16 SetDeviceStaticIP()

```
void AdvancedIllumination.DCS_100.SetDeviceStaticIP (
            string ip )
```

Sets the device to a new static IP address.

Parameters

ip	The IP address to set, in a form like "192.168.24.55"

Exceptions

ArgumentException	Thrown when the given IP address is invalid.

This IP address will not take effect until the device is rebooted.

8.4.4.17 SetMode()

```
ChannelLimits AdvancedIllumination.DCS_100.SetMode (
            Channel chan,
            Mode m )
```

Sets operating mode (either continuous or strobe) for a channel

Parameters

chan	The channel to set
m	The mode to set

8.4.4.18 SetProfileName()

```
void AdvancedIllumination.DCS_100.SetProfileName (
            string name )
```

Sets the name of the currently active profile.

Parameters

name	The name of the profile

Exceptions

System.ArgumentException	Thrown when the profile name is too long ($>$ 15 characters)

8.4.4.19 SetPulseDelay()

```
void AdvancedIllumination.DCS_100.SetPulseDelay (
            Channel chan,
            uint us )
```

Set delay between trigger and output pulse, in raw microseconds for precision.

Parameters

chan	Channel to set
us	Pulse delay value (microseconds)

Exceptions

System.ArgumentOutOfRangeException	Thrown when pulse delay given exceeds 10 milliseconds.

8.4.4.20 SetPulseWidth()

```
ChannelLimits AdvancedIllumination.DCS_100.SetPulseWidth (
            Channel chan,
            uint us )
```

Set pulse width on channel, in raw microseconds for precision.

Parameters

chan	Channel number
us	Pulse width, in microseconds

Returns

The current, in milliamps, to which the controller is limited for lighthead protection.

Exceptions

System.ArgumentOutOfRangeException	Thrown when specified pulse width exceeds 65 milliseconds.

8.4.4.21 SetTriggerEdge()

```
void AdvancedIllumination.DCS_100.SetTriggerEdge (
            Channel chan,
            Trigger trig )
```

Sets the trigger mode on the specified channel.

Parameters

chan	The channel to set
trig	The trigger edge (rising or falling) to set

8.4.4.22 SetTriggerInput()

```
void AdvancedIllumination.DCS_100.SetTriggerInput (
            Channel channel,
            Channel input )
```

Maps the specified channel to follow the specified trigger input.

Parameters

channel	The channel whose input is to be mapped
input	The trigger input to which the channel will be mapped

8.4.5 Property Documentation

8.4.5.1 DeviceType

```
DeviceType AdvancedIllumination.DCS_100.DeviceType  [get]
```

Gets or sets the type of connected device

8.4.5.2 FirmwareVersion

```
int AdvancedIllumination.DCS_100.FirmwareVersion  [get]
```

Gets version of the firmware running on the DCS-100

8.4.5.3 IsConnected

```
bool AdvancedIllumination.DCS_100.IsConnected  [get]
```

Indicates whether or not the class instance is currently connected to a DCS-100E. **true** if it is, and **false** otherwise.

8.4.5.4 Lighthead

```
string AdvancedIllumination.DCS_100.Lighthead  [get]
```

Gets the part number of the connected lighthead.

8.4.5.5 Name

```
string AdvancedIllumination.DCS_100.Name  [get]
```

The device's name. Null if not connected.

8.4.5.6 TargetIP

```
string AdvancedIllumination.DCS_100.TargetIP  [get], [set]
```

Gets or sets the IP address of the target device.

8.4.5.7 Timeout

```
int AdvancedIllumination.DCS_100.Timeout  [get], [set]
```

Gets or sets the timeout, in milliseconds, for communications with the target device.

8.4.6 Event Documentation

8.4.6.1 Disconnection

```
DisconnectEventHandler AdvancedIllumination.DCS_100.Disconnection
```

Event queue for all listeners interested in Disconnection events.

8.4.6.2 Error

`ErrorEventHandler AdvancedIllumination.DCS_100.Error`

Event that indicates a general error message was received from the connected DCS-100E unit.

8.4.6.3 Fault

`FaultEventHandler AdvancedIllumination.DCS_100.Fault`

Event that indicates a fault warning was received from the connected DCS-100E unit.

A fault may mean one or more of the following:

- Driver is over voltage
- Channel disconnected
- Driver is over temperature
- Channel is over current

8.4.6.4 Warning

`WarningEventHandler AdvancedIllumination.DCS_100.Warning`

Event that indicates a general warning message was received from the connected DCS-100E unit.

The documentation for this class was generated from the following file:

- DCS_100.cs

8.5 DCS_Information Class Reference

Class that encapsulates basic information on a DCS-100E controller.

Public Member Functions

- override string ToString ()
  Converts the object into a string.
- override int GetHashCode ()
  Returns the hash code for this DCS_Information object.
- override bool Equals (object arg)
  Determines if this device is equivalent to another.

Properties

- string IP_Address [getset]

  Contains the device's IP address, as a string.
- string Name [getset]

  The device's name.
- string Lighthead [get]

  The part number of the lighthead to which the DCS-100E is attached.
- string FirmwareVersion [get]

  The version number of the firmware.
- bool WebConfigEnabled [getset]

  Indicates whether or not the web configuration page is enabled for this unit.
- DeviceType Type [get]

  The type of device that is connected.

8.5.1 Detailed Description

Class that encapsulates basic information on a DCS-100E controller.

This class cannot be instantiated by the end-user. A list of instances is returned by the method DCS_100.GetActive↩ Devices.

8.5.2 Member Function Documentation

8.5.2.1 Equals()

```
override bool DCS_Information.Equals (
            object arg )
```

Determines if this device is equivalent to another.

Parameters

arg	The other object to compare for equality

Returns

true if devices are equivalent, false otherwise

This function considers DCS-100 devices equivalent if they share the same name and IP address.

8.5.2.2 GetHashCode()

```
override int DCS_Information.GetHashCode ( )
```

Returns the hash code for this DCS_Information object.

This method considers DCS-100s the same if they share the same IP address and name.

Returns

The unique hash code for this object

8.5.2.3 ToString()

```
override string DCS_Information.ToString ( )
```

Converts the object into a string.

The string is formatted as "[IP address]: [Device name], [ligthead] <version>", e.g. "192.168.24.55: Production Line 1, AL116-WHIC1 <003>

Returns

A string representing the basic information of this DCS-100E.

8.5.3 Property Documentation

8.5.3.1 FirmwareVersion

```
string DCS_Information.FirmwareVersion  [get]
```

The version number of the firmware.

8.5.3.2 IP_Address

```
string DCS_Information.IP_Address  [get], [set]
```

Contains the device's IP address, as a string.

Changing the value of this property will set the device to use that static IP.

8.5.3.3 Lighthead

```
string DCS_Information.Lighthead  [get]
```

The part number of the lighthead to which the DCS-100E is attached.

An example of an Advanced illumination lighthead part number is "AL295-RGBC1".

8.5.3.4 Name

```
string DCS_Information.Name  [get], [set]
```

The device's name.

Changing the value of this property will not only change the value in the code, but will also change the name of the physical device.

8.5.3.5 Type

```
DeviceType DCS_Information.Type  [get]
```

The type of device that is connected.

8.5.3.6 WebConfigEnabled

```
bool DCS_Information.WebConfigEnabled  [get], [set]
```

Indicates whether or not the web configuration page is enabled for this unit.

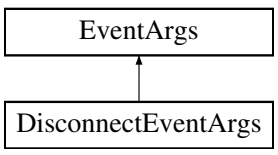
The documentation for this class was generated from the following file:

- DCS_100.cs

8.6 DisconnectEventArgs Class Reference

Additional information for disconnection events.

Inheritance diagram for DisconnectEventArgs:



Properties

- DateTime IncidentTime `[getset]`

  Gets the time the connection was lost.

8.6.1 Detailed Description

Additional information for disconnection events.

8.6.2 Property Documentation

8.6.2.1 IncidentTime

```
DateTime DisconnectEventArgs.IncidentTime [get], [set]
```

Gets the time the connection was lost.

The incident time.

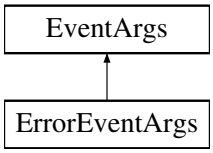
The documentation for this class was generated from the following file:

- DCS_100.cs

8.7 ErrorEventArgs Class Reference

Class which represents the details of an error signal event for the DCS-100E.

Inheritance diagram for ErrorEventArgs:



Properties

- string Message `[getset]`

  The actual error message received from the DCS-100E.
- DCS_100.? Channel IncidentChannel `[getset]`

  Indicates which channel has the error. If the error is not associated with a particular channel, it is null.

8.7.1 Detailed Description

Class which represents the details of an error signal event for the DCS-100E.

8.7.2 Property Documentation

8.7.2.1 IncidentChannel

```
DCS_100.?  Channel ErrorEventArgs.IncidentChannel  [get], [set]
```

Indicates which channel has the error. If the error is not associated with a particular channel, it is null.

8.7.2.2 Message

```
string ErrorEventArgs.Message  [get], [set]
```

The actual error message received from the DCS-100E.

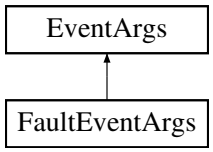
The documentation for this class was generated from the following file:

- DCS_100.cs

8.8 FaultEventArgs Class Reference

Class which represents the details of a fault event for the DCS-100E.

Inheritance diagram for FaultEventArgs:



Properties

- DCS_100.Channel FaultedChannel `[getset]`

  Indicates which channel has a fault condition.

8.8.1 Detailed Description

Class which represents the details of a fault event for the DCS-100E.

8.8.2 Property Documentation

8.8.2.1 FaultedChannel

```
DCS_100.Channel FaultEventArgs.FaultedChannel  [get], [set]
```

Indicates which channel has a fault condition.

The documentation for this class was generated from the following file:

- DCS_100.cs

8.9 TriggerMapping Struct Reference

Encapsulates information related to trigger mapping.

Properties

- DCS_100.MappedTriggerInput Input `[getset]`
  Defines which channel's trigger input is used for mapping, if any.
- DCS_100.Trigger Edge `[getset]`
  Defines which edge transition (rising or falling) is used for the mapped trigger.

8.9.1 Detailed Description

Encapsulates information related to trigger mapping.

8.9.2 Property Documentation

8.9.2.1 Edge

```
DCS_100.Trigger TriggerMapping.Edge  [get], [set]
```

Defines which edge transition (rising or falling) is used for the mapped trigger.

8.9.2.2 Input

```
DCS_100.MappedTriggerInput TriggerMapping.Input  [get], [set]
```

Defines which channel's trigger input is used for mapping, if any.

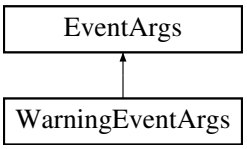
The documentation for this struct was generated from the following file:

- DCS_100.cs

8.10 WarningEventArgs Class Reference

Class which represents the details of a warning signal event for the DCS-100E.

Inheritance diagram for WarningEventArgs:



Properties

- DCS_100.? Channel IncidentChannel `[getset]`
  Indicates which channel has the warning.
- string Message `[getset]`
  The actual warning message received from the DCS-100E.

8.10.1 Detailed Description

Class which represents the details of a warning signal event for the DCS-100E.

8.10.2 Property Documentation

8.10.2.1 IncidentChannel

```
DCS_100.?  Channel WarningEventArgs.IncidentChannel  [get], [set]
```

Indicates which channel has the warning.

8.10.2.2 Message

```
string WarningEventArgs.Message  [get], [set]
```

The actual warning message received from the DCS-100E.

The documentation for this class was generated from the following file:

- DCS_100.cs

Chapter 9

File Documentation

9.1 DCS_100.cs File Reference

Classes

- class AdvancedIllumination.DCS_100

  Class representing one Advanced Illumination DCS-100 controller
- struct ChannelConfig

  Struct which represents the current configuration of a certain channel.
- struct TriggerMapping

  Encapsulates information related to trigger mapping.
- class DCS100_DeviceConfiguration

  Contains all the information that defines the state of a DCS-100E controller.
- struct ChannelLimits

  Encapsulates the lighthead safety limits defined by Advanced illumination's Signatech© II technology.
- class DCS_Information

  Class that encapsulates basic information on a DCS-100E controller.
- class FaultEventArgs

  Class which represents the details of a fault event for the DCS-100E.
- class WarningEventArgs

  Class which represents the details of a warning signal event for the DCS-100E.
- class ErrorEventArgs

  Class which represents the details of an error signal event for the DCS-100E.
- class DisconnectEventArgs

  Additional information for disconnection events.

Namespaces

- namespace AdvancedIllumination

Enumerations

- enum DeviceType { DCS100 , DCS103 }

  Enumeration indicating the type of DCS device

Functions

- delegate void AdvancedIllumination.FaultEventHandler (object sender, FaultEventArgs e)

  Delegate for the handler for the DCS_100.Fault event. Instantiate this delegate for your own event handler.
- delegate void AdvancedIllumination.WarningEventHandler (object sender, WarningEventArgs e)

  Delegate for the handler for the DCS_100.Warning event. Instantiate this delegate for your own event handler.
- delegate void AdvancedIllumination.ErrorEventHandler (object sender, ErrorEventArgs e)

  Delegate for the handler of the DCS_100.Error event. Instantiate this delegate for your own event handler.
- delegate void AdvancedIllumination.DisconnectEventHandler (object sender, DisconnectEventArgs de)

  Delegate for handling Disconnect events.
- virtual void AdvancedIllumination.Dispose (bool disposing)

  Disconnect from the target and dispose of the object.
- void AdvancedIllumination.Dispose ()

  Disconnect from the target and dispose of the object.

9.1.1 Enumeration Type Documentation

9.1.1.1 DeviceType

```
enum DeviceType
```

Enumeration indicating the type of DCS device

Enumerator

DCS100	A one-output, three channel Ethernet strobe controller.
DCS103	A three-output single channel Ethernet strobe controller.

9.2 python.md File Reference

9.3 readme.md File Reference

Index

AdvancedIllumination, 13
  DisconnectEventHandler, 13
  Dispose, 14
  ErrorEventHandler, 14
  FaultEventHandler, 14
  WarningEventHandler, 15
AdvancedIllumination.DCS_100, 22
  Channel, 25
  Connect, 27
  Continuous, 26
  DCS_100, 26
  DeviceType, 36
  Disconnect, 27
  Disconnection, 37
  Error, 37
  Falling, 26
  Fault, 38
  FirmwareVersion, 36
  Gated, 26
  GetConfiguration, 27
  GetProfileNames, 28
  GetProfileNumber, 28
  IsConnected, 36
  Lighthead, 36
  LoadProfile, 28
  MappedTriggerInput, 25
  Mode, 25
  Name, 37
  None, 25
  Off, 26
  OnDisconnection, 30
  One, 25
  OnError, 30
  OnFault, 30
  OnWarning, 31
  PulseChannel, 31
  Rising, 26
  SaveProfile, 31
  SetCommunicationTarget, 32
  SetCurrent, 32
  SetDeviceName, 33
  SetDeviceStaticIP, 33
  SetMode, 33
  SetProfileName, 34
  SetPulseDelay, 34
  SetPulseWidth, 35
  SetTriggerEdge, 35
  SetTriggerInput, 35
  Strobe, 26
  TargetIP, 37
  Three, 25
  Timeout, 37
  Trigger, 26
  Two, 25
  Warning, 38

Channel
  AdvancedIllumination.DCS_100, 25
  ChannelLimits, 20
Channel1
  DCS100_DeviceConfiguration, 21
Channel2
  DCS100_DeviceConfiguration, 21
Channel3
  DCS100_DeviceConfiguration, 22
ChannelConfig, 17
  ContinuousMaximum, 18
  MaxStrobeFrequency, 18
  MinimumCurrent, 18
  OperatingMode, 18
  OutputCurrent, 18
  PulseDelayMicros, 18
  PulseWidthMicros, 19
  StrobeMaximum, 19
  TriggerEdge, 19
  TriggerInput, 19
ChannelLimits, 19
  Channel, 20
  MaxCurrent, 20
  MaxTriggerFrequency, 20
  MinOffTime, 20
Connect
  AdvancedIllumination.DCS_100, 27
Continuous
  AdvancedIllumination.DCS_100, 26
ContinuousMaximum
  ChannelConfig, 18

DCS100
  DCS_100.cs, 48
DCS100_DeviceConfiguration, 21

Channel1, 21
Channel2, 21
Channel3, 22
ProfileNumber, 22
TriggerMap, 22
DCS103
DCS_100.cs, 48
DCS_100
AdvancedIllumination.DCS_100, 26
DCS_100.cs, 47
DCS100, 48
DCS103, 48
DeviceType, 48
DCS_Information, 38
Equals, 39
FirmwareVersion, 40
GetHashCode, 39
IP_Address, 40
Lighthead, 40
Name, 41
ToString, 40
Type, 41
WebConfigEnabled, 41
DeviceType
AdvancedIllumination.DCS_100, 36
DCS_100.cs, 48
Disconnect
AdvancedIllumination.DCS_100, 27
DisconnectEventArgs, 41
IncidentTime, 42
DisconnectEventHandler
AdvancedIllumination, 13
Disconnection
AdvancedIllumination.DCS_100, 37
Dispose
AdvancedIllumination, 14

Edge
TriggerMapping, 44
Equals
DCS_Information, 39
Error
AdvancedIllumination.DCS_100, 37
ErrorEventArgs, 42
IncidentChannel, 43
Message, 43
ErrorEventHandler
AdvancedIllumination, 14

Falling
AdvancedIllumination.DCS_100, 26
Fault
AdvancedIllumination.DCS_100, 38
FaultedChannel
FaultEventArgs, 44
FaultEventArgs, 43
FaultedChannel, 44
FaultEventHandler
AdvancedIllumination, 14
FirmwareVersion
AdvancedIllumination.DCS_100, 36
DCS_Information, 40

Gated
AdvancedIllumination.DCS_100, 26
GetConfiguration
AdvancedIllumination.DCS_100, 27
GetHashCode
DCS_Information, 39
GetProfileNames
AdvancedIllumination.DCS_100, 28
GetProfileNumber
AdvancedIllumination.DCS_100, 28

IncidentChannel
ErrorEventArgs, 43
WarningEventArgs, 46
IncidentTime
DisconnectEventArgs, 42
Input
TriggerMapping, 45
IP_Address
DCS_Information, 40
IsConnected
AdvancedIllumination.DCS_100, 36

Lighthead
AdvancedIllumination.DCS_100, 36
DCS_Information, 40
LoadProfile
AdvancedIllumination.DCS_100, 28

MappedTriggerInput
AdvancedIllumination.DCS_100, 25
MaxCurrent
ChannelLimits, 20
MaxStrobeFrequency
ChannelConfig, 18
MaxTriggerFrequency
ChannelLimits, 20
Message
ErrorEventArgs, 43
WarningEventArgs, 46
MinimumCurrent
ChannelConfig, 18
MinOffTime
ChannelLimits, 20
Mode
AdvancedIllumination.DCS_100, 25

Name
AdvancedIllumination.DCS_100, 37
DCS_Information, 41
None
AdvancedIllumination.DCS_100, 25

Off
AdvancedIllumination.DCS_100, 26
OnDisconnection
AdvancedIllumination.DCS_100, 30
One
AdvancedIllumination.DCS_100, 25
OnError
AdvancedIllumination.DCS_100, 30
OnFault
AdvancedIllumination.DCS_100, 30
OnWarning
AdvancedIllumination.DCS_100, 31
OperatingMode
ChannelConfig, 18
OutputCurrent
ChannelConfig, 18

ProfileNumber
DCS100_DeviceConfiguration, 22
PulseChannel
AdvancedIllumination.DCS_100, 31
PulseDelayMicros
ChannelConfig, 18
PulseWidthMicros
ChannelConfig, 19
python.md, 48

readme.md, 48
Rising
AdvancedIllumination.DCS_100, 26

SaveProfile
AdvancedIllumination.DCS_100, 31
SetCommunicationTarget
AdvancedIllumination.DCS_100, 32
SetCurrent
AdvancedIllumination.DCS_100, 32
SetDeviceName
AdvancedIllumination.DCS_100, 33
SetDeviceStaticIP
AdvancedIllumination.DCS_100, 33
SetMode
AdvancedIllumination.DCS_100, 33
SetProfileName
AdvancedIllumination.DCS_100, 34
SetPulseDelay
AdvancedIllumination.DCS_100, 34
SetPulseWidth
AdvancedIllumination.DCS_100, 35
SetTriggerEdge
AdvancedIllumination.DCS_100, 35
SetTriggerInput
AdvancedIllumination.DCS_100, 35
Strobe
AdvancedIllumination.DCS_100, 26
StrobeMaximum
ChannelConfig, 19

TargetIP
AdvancedIllumination.DCS_100, 37
Three
AdvancedIllumination.DCS_100, 25
Timeout
AdvancedIllumination.DCS_100, 37
ToString
DCS_Information, 40
Trigger
AdvancedIllumination.DCS_100, 26
TriggerEdge
ChannelConfig, 19
TriggerInput
ChannelConfig, 19
TriggerMap
DCS100_DeviceConfiguration, 22
TriggerMapping, 44
Edge, 44
Input, 45
Two
AdvancedIllumination.DCS_100, 25
Type
DCS_Information, 41

Warning
AdvancedIllumination.DCS_100, 38
WarningEventArgs, 45
IncidentChannel, 46
Message, 46
WarningEventHandler
AdvancedIllumination, 15
WebConfigEnabled
DCS_Information, 41