

NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- <u>Main Page</u>
- <u>Related Pages</u>
- <u>Classes</u>

Programmer's Guide

2.31

Copyright(c) 2010-2016 NVIDIA Corporation. All rights reserved.

- Introduction
- NVWMI compatibility
- Version-specific Implementation Details
- Cooler and Thermal Events
- NVIDIA Performance Counters
- Using NVWMI
 - Using NVWMI with the PowerShell
 - Using NVWMI with the WMIC tool
 - Logging and tracing NVWMI activity

Introduction

1. General Purpose

NVWMI provider allows WMI clients to query and to monitor parameters of NVIDIA hardware. It is implemented as a decoupled WMI provider in a system service (nvwmi.exe in 32-bit or nvwmi64.exe in 64-bit flavors of

Windows).

2. General Requirements

- NVIDIA hardware
- operational NVIDIA display driver
- installed NVWMI provider

3. Version-specific

For detailed information about version-specific changes, please refer to the implementation Version-specific Implementation Details.

4. OS-specific

Windows XP has limited support. Supported classes are:

- System
- Gpu
- Board
- Cooler
- ThermalProbe
- CoolerEvent
- ThermalEvent

NVWMI Performance counter provider does not support Windows XP.

5. General Implementation Details

NVWMI implements several WMI providers inside \\.\root\cimv2\nv namespace:

- instance provider
- method provider
- property provider
- event provider
- performance counters provider

All classes are statically defined in the MOF file, there are no dynamically defined classes (e.g. no class provider).

Static vs. dynamic instances.

In WMI, there are two separate types of object instances - singletons and dynamic instances. If the MOF class is declared as a singleton, it has just one instance. The path to such an instance is always known to the WBEM engine. Therefore the exact path to the instance is not required to address the instance. All other classes have dynamic instances. This means that the user must specify either a precise path or WQL operator which addresses one or more instances.

classes **System**, **DisplayManager** and **ProfileManager** are declared as MOF singletons, all other classes have dynamic instancing. Thus addressing instances of any non-singleton class, requires either **where** WQL clause or instance path.

GPU handles

GPU handles are not guaranteed to persist across driver upgrades or hardware changes. Enabling or disabling hardware, installing new NVIDIA hardware, upgrading driver or changing SLI configuration might change value of any GPU handle.

Object reference

"Object reference" is a UTF-16 string which contains the path to the WMI object. References could be used to build WQL queries. References (i.e. object paths) could be used to address instances.

There is a potential for signed/unsigned mismatch, as the WMI COM interface **iwbemClassObject** requires that even uint32 MOF type is set as a variant of VI_I4 type. Conversely, when retrieving properties of WMI objects via **iwbemClassObject::Get**, supply CIMTYPE* pvtType to retrieve the correct type. Do not rely on VARIANT* pVal.

For more information about IWbemClassObject::Get, see <u>MSDN article</u> about IWbemClassObject::Get

Example of the object reference (relative to NV namespace):

DisplayMode.id=452288000

references instance of the **DisplayMode** class with id=452288000. **DisplayMode** instance with this id corresponds to the 1600x1200x32 Hz display mode (for versions 2.2, 2.3). Note that for classes with multiple keys, the reference string must contain all of them. Here is an example of **Gpu** instance reference:

Gpu.id=3137339648,uname="Quadro 5000"

When specifying references at command line, be sure to use quotation marks as appropriate (remember that reference is a string) - example of setting display mode on remote system **wks-1**

wmic /node:"wks-1" /namespace:nv path displaygrid where id=0 call se

6. WMI General Information

Please refer to the MSDN documentation on WMI providers for information about general functionality and use.

Links to relevant MSDN pages:

WMI Reference Creating WMI clients

> NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- <u>Main Page</u>
- <u>Related Pages</u>
- <u>Classes</u>

Legal Notice

Copyright (c) 2010-2016 NVIDIA Corporation. All rights reserved.

Notice

This software may not, in whole or in part, be copied through any means, mechanical, electromechanical, or otherwise, without the express permission of NVIDIA Corporation.

Information furnished is believed to be accurate and reliable. However, NVIDIA assumes no responsibility for the consequences of use of such information nor for any infringement of patents or other rights of third parties, which may result from its use. No License is granted by implication or otherwise under any patent or patent rights of NVIDIA Corporation. Specifications mentioned in the software are subject to change without notice.

NVIDIA Corporation products are not authorized for use as critical components in life support devices or systems without express written approval of NVIDIA Corporation.

Trademarks

NVIDIA and the NVIDIA logo are registered trademarks or trademarks of NVIDIA Corporation in the United States and/or other countries.

Microsoft, Windows, and the Windows logo are registered trademarks of Microsoft Corporation.

Other company and product names may be trademarks or registered trademarks of the respective companies with which they are associated.







NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>

Version-specific Implementation Details

Version 2.31

- Version 2.31 implements new properties of the VideoCodec class with data about video encoding statistics:
 - VideoCodec::encoderSessionsCount
 - VideoCodec::averageFps
 - VideoCodec::averageLatency
- Version 2.31 implements several performance counters with video encoding statistics:
 - Video Encoder Sessions
 - Video Encoder Average FPS
 - Video Encoder Average Latency (ms)

- Version 2.30 implements support of user-defined GPU utilization events. User-defined GPU utilization events are not persistent across service restarts. By default no GPU utilization events are defined.
- Version 2.30 implements Gpu::createUtilizationEvent method. This

method creates new utilization event in a given domain.

- Version 2.30 implements **Gpu::deleteUtilizationEvents** method. This method deletes previously created events in a given domain.
- Version 2.30 implements **Gpu::getAllUtilizationEvents** method. This method returns a string with all utilization events in a given domain.
- Version 2.30 implements support of user-defined filters of display modes. Custom display mode filters are not persistent across service restarts. By default display modes are not filtered.
- Version 2.30 implements **DisplayManager::createModeFilter** method. This method creates new custom display mode filter.
- Version 2.30 implements **DisplayManager::deleteModeFilter** method. This method deletes previously created custom display mode filter.
- Version 2.30 implements **DisplayManager::enumModeFilters** method. This method enumerates existing display mode filters.

Version 2.29

- Version 2.29 implements dithering control in the **Display** class. Following new properties are supported:
 - Display::ditherState
 - Display::ditherBits
 - Display::ditherMode
- Version 2.29 implements **Display::setDither** method. This method changes dithering parameters.
- Version 2.29 implements **DisplayManager::saveCustomTimings** method for saving custom timings. Custom timings are saved in XML file, compatible with NVIDIA Control Panel.
- Version 2.29 implements **DisplayManager::loadCustomTimings** method for loading custom timings.

Version 2.28

 Version 2.28 implements new class VideoCodec which contains information about VideoEngine encoder and decoder. Instances of VideoCodec class are accessible only as properties of the Gpu class.

- Version 2.28 implements new property of the **Gpu** class with data about VideoEngine:
 - Gpu::videoCodec
- Version 2.28 implements several performance counters for VideoEngine:
 - \circ % Video Encoder usage
 - % Video Decoder usage

New features of the version 2.28 were ported to the 2.27 and released as 2.27.2.

- Version 2.27 implements new class **PcieLink** which contains information about PCI-Express link to GPU. Instances of **PcieLink** class are accessible only as properties of the **Gpu** class.
- Version 2.27 implements new properties of the **Gpu** class with data about PCI-Express link to GPU:
 - Gpu::pcieDownstreamWidth
 - Gpu::pcieGpu
- Version 2.27 implements several performance counters for PCI-Express link to GPU:
 - % Bus Usage
 - % Video Usage
 - % FB Usage
 - PCI-E downstream width to GPU
 - PCI-E current width to GPU
 - PCI-E current speed to GPU Mbps
- Version 2.27 implements support of cloned displays in display profiles.
- Version 2.27 implements **DisplayManager::createClone** method. This method clones a screen of display into screens of other displays. Two modes supported:
 - basic clone identical display mode is set for all displays
 - smart clone source display with larger screen resolution is cloned into smaller resolution of a target display(s) with image panning

- Version 2.26 improves access to display functionality. All DisplayGrid instances are searched for matching Display and corresponding method of DisplayGrid is called when match is found.
- Version 2.26 implements **Display::setDisplayMode** method as a shortcut to the **DisplayGrid::setDisplayMode**.
- Version 2.26 implements **Display::setDisplayModeById** method as a shortcut to the **DisplayGrid::setDisplayModeById**.
- Version 2.26 implements **Display::setDisplayModeByRef** method as a shortcut to the **DisplayGrid::setDisplayModeByRef**.

- Version 2.25 implements support of licensable feature management in **System** and **Gpu** classes.
- Version 2.25 implements **Gpu::licensableFeatures** property. This property contains array of strings with names of all available GRID features to license.
- Version 2.25 implements **Gpu::licensableStatus** property. This property contains array of enumerated status codes of all available GRID features to license.
- Version 2.25 implements **Gpu::enableLicensedFeature** method. This method enables given features, based on available licenses on a server.
- Version 2.25 implements **Gpu::disableLicensedFeature** method. This method disables given features.
- Version 2.25 implements **System::licensingPort** property. This property contains networking port of a license server.
- Version 2.25 implements **System::licensingServer** property. This property contains network address of a license server.
- Version 2.25 implements **System::setLicensingServer** method. This method changes address and port of license server.
- Version 2.25 implements **System::unsetLicensingServer** method. This method resets address and port of license server and relinquish a license after reboot.

- Version 2.24 changed type of the **Board::serialNumber** property. For user readability it is changed to a string with human-readable serial number of a GPU board.
- Version 2.24 improved compatibility with Windows 10. Versions prior to 2.24 are not recommended for use under Windows 10.

Version 2.23

- Version 2.23 implements "Fan Speed (RPM)" performance counter.
- Version 2.23 changed type of several properties of the Gpu class from unsigned to signed. This change improves clarity of output when property is not supported in hardware and property value is set to -1. Following properties were modified:
 - Gpu::powerSampleCount
 - Gpu::powerSamplingPeriod
 - Gpu::percentGpuUsage
 - Gpu::percentGpuMemoryUsage

- Version 2.22 implements support of Windows gamma ramp in display profiles (class **DisplayProfile**) and system profiles (class **Profile** of type "system"=5).
- Version 2.22 implements **Display::saveGammaRamp** method. This method saves gamma ramp data of individual display.
- Version 2.22 implements **Display::setGammaRamp** method. This method applies previously saved gamma ramp data of individual display.
- Version 2.22 implements Display::setGammaRampBasic method. This method generates monochromatic gamma ramp for individual display from three input parameters - gamma, contrast and brightness.
- Version 2.22 implements **DisplayGrid::saveGammaRamp** method. This method saves Color Space Conversion data of all displays in a display grid.
- Version 2.22 implements **DisplayGrid::setGammaRamp** method.

This method applies Color Space Conversion data to all displays in a display grid.

- Version 2.22 implements **DisplayGrid::setGammaRampBasic** method. This method generates monochromatic gamma ramp for all displays in display grid from three input parameters - gamma, contrast and brightness.
- Version 2.22 implements **DisplayGridInfo::gammaRampFilePath** property. It contains path to the binary file with display grid gamma ramp data.
- Version 2.22 implements singleton class DesktopManager and several methods for manipulating virtual nView desktops -DesktopManager::getAllDesktops, DesktopManager::createDesktop, DesktopManager::editDesktop, DesktopManager::deleteDesktop.

Version 2.21

- Version 2.21 implements support of Color Space Conversion (CSC) in display profiles (class **DisplayProfile**) and system profiles (class **Profile** of type "system"=5).
- Version 2.21 implements **Display::saveCSC** method. This method saves Color Space Conversion data of individual display.
- Version 2.21 implements **Display::setCSC** method. This method applies previously saved Color Space Conversion data of individual display.
- Version 2.21 implements **DisplayGrid::saveCSC** method. This method saves Color Space Conversion data of all displays in a display grid.
- Version 2.21 implements **DisplayGrid::setCSC** method. This method applies Color Space Conversion data to all displays in a display grid.
- Version 2.21 implements **DisplayGridInfo::cscFilePath** property. It contains path to the binary file with display grid CSC data.

Version 2.20

• Version 2.20 implements **Display::getCurrentTiming** method. This

method returns a string with current display timing parameters.

- Version 2.20 implements **Gpu::fakeEDIDOnPort** method. This method fakes EDID on specified GPU port.
- Version 2.20 improves SyncTopology::id, SyncDelay::id, Sync::id, Gpu::id, Board::id and DisplayGrid::id properties by using more user-friendly values.
- Version 2.20 implements **Display::displayConnectorType** property.
- Version 2.20 implements **Gpu::nvapild** property.
- Version 2.20 implements **Gpu::archId** property. It contains numerical ID of the GPU architecture.
- Version 2.20 implements **Gpu::archName** property. It contains a string with the GPU architecture name (e.g. "Kepler", "Maxwell" etc.)
- Version 2.20 deprecates handle property in several classes. While handle is still available, id (WMI-only object identifier) is recommended for scripting and interactive commands and nvapild for interfacing with NVAPI.

Version 2.19

- Version 2.19 implements **System::setLogState** method. This method changes logging verbosity and other logging settings.
- Version 2.19 implements additional functionality in the **System** class. Following new properties are supported:
 - System::logTypes property. This property is a bitmap see System::setLogState for details.
 - **System::logFilter** property. This property is a bitmask.

- Version 2.18 implements additional functionality in the **Board** class. Following new properties are supported:
 - **Board::nvapild** property. It contains NVAPI **Board** ID, which was previously reported as **Board::id**.
 - **Board::chipSKU** SKU of the GPU chip(s) on this board
 - Board::chipSKUMod SKU modifier the GPU chip(s) on this board
 - **Board::project** project (board) number
 - Board::projectSKU project (board) SKU number

- Version 2.18 implements additional functionality in the **Cooler** class. Following new properties are supported:
 - **Cooler::id** unique object identifier.
 - **Cooler::minSpeed** Minimal cooler speed. Units are revolutions per minute (RPM) for fans.
 - **Cooler::maxSpeed** Maximal cooler speed. Units are revolutions per minute (RPM) for fans.
- Version 2.18 deprecates **Cooler::handle** property. While **Cooler::handle** is still available, **Cooler::id** should be used instead.
- Version 2.18 implements additional functionality in the **ThermalProbe** class. Following new properties are supported:
 - ThermalProbe::id unique object identifier.
 - **ThermalProbe::defaultMinTemperature** Minimal default temperature.
 - **ThermalProbe::defaultMaxTemperature** Maximal default temperature.
- Version 2.18 deprecates **ThermalProbe::handle** property. While **ThermalProbe::handle** is still available, **ThermalProbe::id** should be used instead.
- Version 2.18 implements **ProfileManager::deleteDesktopProfile** method.
- Version 2.18 implements nView version as a property of **System** class **System::vernViewDesktopManager**.
- Version 2.18 implements "% GPU Memory Usage" performance counter.
- Version 2.18 implements Gpu::percentGpuMemoryUsage property.
- Version 2.18 implements **Gpu::deviceInfo** property which reports the Device Id, Vendor Id, Sub-device Id, Revision Id.
- Version 2.18 implements **System::machineType** property which reports whether the system is Desktop or Mobile.
- Version 2.18 implements multiple **Display** grid support on single GPU.

• Version 2.17 implements user-friendly Display IDs. User-friendly display IDs are created from the GPU ordinal and the display ordinal

according to this formula:

id=(GPU #)*1000+(Display #);

where (GPU #) is a GPU ordinal as enumerated by NVAPI and starts from 1, (Display#) is a display ordinal in per-GPU enumeration by NVAPI, and also starts with 1.

- Version 2.17 implements Display::locus property. It contains string, representing user-friendly ID as "(GPU #).(Display #)". For example, a display, attached to 1st output of 1st GPU will have Display::locus="1.1" and Display::id=1001.
- Version 2.17 implements **Display::nvapild** property. It contains NVAPI **Display** ID, which was previously reported as **Display::id**.
- Version 2.17 implements class **Ecc. Gpu::ecc** property is a reference to an instance of the **Ecc** class. This feature is not supported if there are multiple GPU topologies enabled.
- Version 2.17 implements **DisplayManager::tryCustomTiming** method. This method validates the new Custom timing before creating/ applying, for the given displays.
- Version 2.17 implements **DisplayManager::createCustomTiming** method. This method creates the new Custom timing for the given displays.
- Version 2.17 implements **DisplayManager::deleteCustomTiming** method. This method deletes the existing Custom timing for the given displays.
- Version 2.17 implements **DisplayManager::editCustomTiming** method. This method deletes the existing Custom timing and creates a new Custom timing, for the given displays.
- Version 2.17 implements **DisplayManager::enumCustomTimings** method. This method enumerates all the existing Custom modes, for the given displays.

- Version 2.16 provides functionality related to **System** Profiles.
- Version 2.16 implements ProfileManager::currentProfileDesktop and ProfileManager::defaultProfileDesktop properties.
- Version 2.16 implements **ProfileManager::saveSystemProfile** method. This method saves the given **System Profile**.

- Version 2.16 implements **ProfileManager::applySystemProfile** method. This method applies the **System** Profiles.
- Version 2.16 implements ProfileManager::currentSystemProfile and ProfileManager::defaultSystemProfile properties.

- Version 2.15 enhances **DisplayManager::createDisplayGrids** method with support for default and custom topologies.
- Version 2.15 enhances **DisplayManager::validateDisplayGrids** method with support for default and custom topologies.
- Version 2.15 implements Gpu::fakeEDID method. This method fakes EDID on all connectors of a given output type and could be used to create virtual displays in headless configurations. For example, if one physical display is attached to the DisplayPort connector of a single-GPU board with 2 DisplayPort connectors then invoking Gpu::fakeEDID for DisplayPort output will override EDID for attached physical display plus a new virtual display will appear.
- Version 2.15 implements **DisplayManager::fakeEDIDAII** method. This method fakes EDID on all connectors of a given output type on all GPUs and could be used to create virtual displays in headless configurations.
- NOTE: This version of NVWMI will not be compatible to older version of nView. Some of the nView setting in Profile class are removed,refer Profile settings available in NVWMI for details. To apply nView setting the user must be logged in.
- Version 2.15 deprecates properties System::verSDIOutputDriver and System::verSDICaptureDriver.
- Version 2.15 deprecates DisplayManager::setEdidAll method.

- Version 2.14 implements **DisplayManager::createDisplayGrids** method. This method creates multiple display grids, with a single call.
- Version 2.14 implements **DisplayManager::validateDisplayGrids** method. This method validates multiple display grids, with a single call.

- Version 2.14 implements **DisplayManager::setGridPositions** method. This method repositions all display grids at once. Note that all active grids must be repositioned together.
- Version 2.14 implements support for passive stereo modes.
- Version 2.14 deprecates method ProfileManager::getAllApplicationProfiles(). This method has been superseded by getAllProfiles().
- Version 2.14 implements method ProfileManager::getAllProfiles(). This method lists all profiles of a given type as a string in format "\ <id1\> : \<profile name1\> ;\<id2\> : \<profile name2\>; ...". Profiles of type 3D Application Profile, 3D Global Profile, nView Profile, Display Profile, System Profile are supported

- Version 2.13 implements profile settings reflection mechanism by providing SettingsTable MOF class:
 - List all settings of a given profile type
 - Convert between profile setting name and profile setting ID
 - Obtain information about a profile setting by specifying either setting ID or setting name
 - **Setting** IDs and setting names could be used to modify or query profile setting values

See also:

PowerShell examples - Using profile settings reflection mechanism, table with profile settings and setting names -Profile settings available in NVWMI

- Version 2.13 implements
 DisplayManager::validateDisplayGridById method to aid user in display grid creation.
- Version 2.13 implements **DisplayGrid::setRotation** method. It rotates all displays in a display grid to a given angle. In this version only first element of an array with rotation angles is applied to all displays, in future versions per-display rotations will be supported.

- Version 2.12 implements classes **Sync** and **SyncTopology** with basic support of video signal synchronization on Sync-compatible hardware. Additional information about hardware compatibility see **NVIDIA Quadro Sync.**
- Version 2.12 implements **Board::serialNumber** property.

- Version 2.11 implements support for measuring GPU power consumption. Following properties reported on high-end Kepler Quadro boards:
 - Gpu::power in watts per sampling period
 - **Gpu::powerSampleCount** number of power sensor samples per iteration
 - **Gpu::powerSamplingPeriod** sampling period in milliseconds On unsupported hardware these properties are set to **0**.
- Version 2.11 improves hardware detection and differentiation.
- Version 2.11 improves **Gpu** class by reporting memory-related properties in megabytes rather than in kilobytes.

- Version 2.10 implements **ApplicationProfile** class with basic support for 3D application profiles.
 - Querying instances of ApplicationProfile class without specifying ApplicationProfile::id will list all active application profiles at the moment of query. Such query could be used to monitor profile activity on a system and to verify that appropriate profile is associated with application of interest.
 - Querying instances of ApplicationProfile class with specific ApplicationProfile::id will return an instance with matching id only. Such query could be used to examine or modify a specific application profile.
- Version 2.10 implements method
 ProfileManager::getAllApplicationProfiles(). It lists all application
 profiles of a given type as a string in format "\<id1\> : \<profile
 name1\> ;\<id2\> : \<profile name2\>; ...". Only profiles of type 0
 (3D application profiles) are supported at present.

- Version 2.10 implements display rotation support. Property Display::rotation reports current rotation. Following values are supported:
 - **0** No rotation
 - **1** rotated 90 degrees
 - **2** rotated 180 degrees
 - $\circ~$ 3 rotated 270 degrees

Use method **Display::setRotation()** to change display rotation.

- Version 2.10 implements additional properties in **DisplayProfile** class.
 - "positionCols" (32-bit unsigned) ID=0x586748C2, default value is 0. Display positions in columns of display grid (in pixels).
 - **"positionRows"** (32-bit unsigned) **ID=0x5879DDC6**, default value is **0**. Display positions in rows of display grid (in pixels).
 - **"primaryId"** (32-bit unsigned) **ID=0x58597B87**, no default value current value is always present. Display ID of the GDI primary display.
 - "rotation" (string with rotation values as unsigned integers, separated by semicolon ;) ID=0x58DECFA8, default value is "0".
 - "scaling" (string with scaling values as unsigned integers, separated by semicolon ;) ID=0x587B0428, default value is "0".

Version 2.9

- Version 2.9 improves performance of the profile framework
- Version 2.9 implements saving of an nView desktop profile in the method **ProfileManager::saveDesktopProfile**
- Version 2.9 supports saving a display EDID in the method Display::saveEDID

- Version 2.8 has revised API. Many classes, their properties and methods are renamed or moved into other classes.
 - update() method is removed from all classes
 - class NV class is split into classes System, DisplayManager

and **ProfileManager**

- methods NV::setScaling, NV::setEdid are refactored into DisplayManager::setScalingAll and DisplayManager::setEDIDAll
- class Mode is renamed **DisplayMode**
- per-row and and per-column property names in class
 DisplayGrid now start with either row or col prefix

- Version 2.7 implements additional functionality in **DisplayProfile** class. Following settings are supported:
 - "rows" (32-bit unsigned) ID=0x5822918D, default value is 1
 - "cols" (32-bit unsigned) ID=0x58E21BD4, default value is 1
 - "displayIds" (string with display IDs in hexadecimal notation separated by semicolon ;) ID=0x58B21E43, no default value current value is always present
 - "overlapCols" (string with overlap values as signed integers, separated by semicolon ;) ID=0x58EB619D, default value is "0"
 - **"overlapRows"** (string with overlap values as signed integers, separated by semicolon ;) **ID=0x58DD36C1**, default value is **"0"**
- Version 2.7 implements several methods for manipulating display profiles.
 - **ProfileManager::saveDisplayProfiles()**
 - ProfileManager::applyDisplayProfiles()
- Version 2.7 implements several methods for manipulating desktop profiles.
 - ProfileManager::loadDesktopProfile()
 - ProfileManager::lockDesktopProfile()
- Version 2.7 supports additional parameters for V-sync control. Method ProfileManager::setVsync() accepts following values:
 - 0 Passive
 - **1** Off
 - **2** On
 - **3** Adaptive (refresh rate)
 - **4** Adaptive (half refresh rate)
- Version 2.7 implements SDI capture driver version as a property of

NV class

• Version 2.7 allows to force EDID on inactive displays

Version 2.6

- Version 2.6 implements framework for generic profile support.
- Version 2.6 implements DisplayPath class and **DisplayProfile** class with limited functionality.
- Version 2.6 overhauls display-related classes.
 - Class Monitor renamed to Display
 - Class DisplaySource is deprecated.
 - Class DisplayTarget is deprecated.
 - Methods of DisplayTarget and DisplaySource classes are moved to **DisplayGrid** and **Display**.

Version 2.5

• Version 2.5 implements DisplayGrid::effectiveMode property. It reports bezel-corrected display dimensions while taking into account display grid layout, answering the question "how many unique pixels could you see?". Note that DisplayGrid::currentMode reports per-display video mode.

For example, for horizontal display grid 1x2 (1 row and 2 columns) with each display at 1600x1200 and with overlap -25 for the second display, DisplayGrid::currentMode (**DisplayGrid::displayModePhysical** in version 2.8 and later) will be reported as 1600x1200 whereas DisplayGrid::effectiveMode (**DisplayGrid::displayModeVirtual** in version 2.8 and later) will be 3175x1200. General formula is: *effective size = (per-display size)*(# of display for a given dimension)+(sum of all overlaps for a given dimension)*

- Version 2.4 implements several methods for defining display grids:
 - NV::createDisplayGridById() renamed
 DisplayManager::createDisplayGridById() in version 2.8 and later

- NV::createDisplayGridByRef() renamed
 DisplayManager::createDisplayGridByRef()in version 2.8 and later
- NV::createDisplayGridByName() renamed
 DisplayManager::createDisplayGridByName() in version 2.8 and later
- Version 2.4 implements **OverlapLimits** class.
- Version 2.4 implements several methods of forcing EDID:
 - NV::setEdid() renamed DisplayManager::setEDIDAll() in version 2.8 and later
 - Monitor::setEdid() renamed Display::setEDID() in version 2.8 and later
- Version 2.4 implements method for turning Vertical Synchronization (V-sync) on and off - NV::setVsync().

NOTE: Creating display grids on QuadroPlex is partially supported. Creation of some legitimate display grid configurations straddling more than one GPU might fail.

- Version 2.3 implements class **DisplayGrid** with basic support for multi-monitor and Mosaic configurations.
- Version 2.3 implements additional support for SELECT operator.
- Version 2.3 improves versioning by adding descriptions into binary version block and OS service records.
- Version 2.3 implements functionality differentiation between Quadro, NVS and GeForce. GeForce does not support advanced functionality. Note that for mixed configurations (GeForce and Quadro or NVS) functionality is equal to GeForce.
 - Basic classes (Version, NamedObject)
 - System class is supported
 - Gpu class is supported
 - Board class is supported
 - Cooler class is supported
 - ThermalProbe class is supported
 - Event classes are supported (CoolerEvent, ThermalEvent)
 - All other classes are supported only on Quadro (both desktop

and notebook) and NVS desktops

Version 2.2

- Version 2.2 implements several methods to change display parameters:
 - NV::setScaling() renamed **DisplayManager::setScalingAll()** in version 2.8 and later
 - DisplaySource::setMode() refactored to DisplayGrid::setMode() in version 2.8 and later
 - DisplayTarget::setMode() refactored to DisplayGrid::setMode() in version 2.8 and later
 - DisplayTarget::setScaling() deprecated and removed in version
 2.8 and later
- Version 2.2 implements additional properties and dynamic instancing of DisplaySource and DisplayTarget classes.
- Version 2.2 implements a Mode class (renamed **DisplayMode** in version 2.8 and later) which contains basic information about display modes.
- Version 2.2 improves logging with OS event log support. See Logging and tracing NVWMI activity.

Version 2.1

- Version 2.1 implements methods for static and dynamic instances.
- Version 2.1 implements a **Board** class to simplify diagnostic of multi-GPU and multi-board configurations.
- Version 2.1 implements many display-related classes Monitor, DisplaySource, DisplayTarget. Functionality is limited to read-only properties.

- Version 2.0 implements basic WQL support. Complex WQL operators not supported.
- Version 2.0 implements individual versioning of each MOF class as well as version-reporting of many software and hardware components of a system.

• Version 2.0 derives most of classes from **NamedObject** class to allow consistent addressing of objects.

Version 1.2

- Version 1.2 implements access to physical GPUs' thermal environment and provides basic information such as GPU and board names.
- Clients could monitor events from GPU coolers and thermal probes. Hardware that is capable of reporting this information is required.
- Version 1.2 doesn't support WQL queries and doesn't expose methods to modify current configuration. All properties are read-only.

NVIDIA

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





- <u>Main Page</u>
- <u>Related Pages</u>
- <u>Classes</u>

NVWMI compatibility

General information about NVWMI compatibility

NVWMI versions have inter-dependencies with the NVIDIA **Display** Driver Package versions. It is strongly recommended that customers follow the table below to choose a supported combination of NVWMI & NVIDIA **Display** Driver to guarantee functionality of all the NVWMI features of a specific version.

Note: If NVWMI gets installed on an earlier NVDIA Driver Package than the minimal recommended driver version there is a possibility that some of the NVWMI features/commands may not function and return a message indicating that the function is unsupported.

Please refer to the NVWMI release notes for details on which NVWMI/NVIDIA **Display** Driver versions add support for specific NVWMI features/commands.

Table of recommended version combinations

Minimal recommended

NVWMI version	Driver Release	version of the Display Driver package	Minimal nView version
2.00	Release 275	275.36	NA
2.02	Release 285	285.58	NA
2.04	Release 295	295.73	NA
2.06	Release 304	305.93	NA
2.07	Release 304	305.93	NA
2.09	Release 310	310.90	NA
2.10	Release 313	314.07	NA
2.12	Release 319	320.86	140.75
2.14	Release 325	320.86	140.75
2.15	Release 331	331.65	140.84
2.16.0	Release 331	331.65	140.84
2.16.1	Release 334	334.67	141.24
2.17	Release 337	337.50	141.24
2.18	Release 340	340.52	141.24

i	ii	1	
2.19	Release 343	340.52	141.24
2.20	Release 346	347.88	141.36
2.21	Release 349	348.27	146.33
2.22	Release 352	353.82	146.78
2.23	Release 352	353.82	146.78
2.24	Release 352	353.82	146.78
2.25	Release 361	361.75	146.78
2.26	Release 364	364.72	146.78
2.27	Release 367	368.39	147.00
2.28	Release 370	372.95	147.00
2.29	Release 375	375.63	147.00
2.30	Release 378	378.49	147.00
2.31	Release 384	387.39	147.00

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





Feb 2018

- <u>Main Page</u>
- <u>Related Pages</u>
- <u>Classes</u>

Cooler and Thermal Events

Cooler Events and Diagnostics

What is Fan Percent Speed? It is the current speed of the fan, normalized between 0-100%, where 0% means the fan is not spinning at all and 100% means the fan is spinning at its maximum possible speed.

The fan speed varies dynamically, without user or system intervention, within the minimum and maximum allowed speeds depending on GPU temperature. Normally, the fan is not required to spin at 100% as the fan is designed to maintain operational temperature at a lower speed. Therefore, the maximum allowed fan percent speed may be less than 100%. This also increases fan life and is acoustically better.

The fan is never supposed to be at 0% for desktop systems unless the system is shut down, in Standby mode (S3), or in Hibernate mode (S4).

Every NVIDIA GPU board has its own maximum and minimum fan speed percentage values.

Users can manually override the fan percent speed via NVAPI, however, GPU functionality may be affected if the user changes the speed beyond

the minimum or maximum allowed speeds.

What is Fan Mechanical Speed/RPM? The fan mechanical speed is the physical measurement of the number of fan rotations per minute (RPM). The board has a dedicated tachometer which measures this value. The relationship between fan speed percent and mechanical speed is roughly linear, but the actual mapping is dependent on the physical cooler such that different boards and coolers will produce different fan mechanical speeds for the same percent speeds.

How is Expected Fan Mechanical Speed calculated? The expected fan mechanical speed is the desired number of fan revolutions per minute (RPM) with respect to the current fan speed percent. Its calculated by linearly interpolating the current fan speed with minimum & maximum values of percent speed and RPM.

Below is the formula used for calculating expected RPM:

```
ERPM = MinRPM + ((CurPctSpeed - MinPctSpeed) * ((MaxRPM - MinRPM) /
(MaxPctSpeed - MinPctSPeed)))
```

Where:

ERPM > Expected fan mechanical speed (RPM) CurPctSpeed -> Current fan percent speed MinPctSpeed > Minimum fan percent speed MaxPctSpeed > Maximum fan percent speed MinRPM > Minimum RPMs with respect to Minimum fan percent speed MaxRPM > Maximum RPMs with respect to Maximum fan percent speed

What is Allowed Error Margin? It is the percent margin indicating the variation between electrical inputs and corresponding fan mechanical speed. Ideally mechanical speed should change linearly with electrical inputs, but based on statistical analysis, its slightly curved and is found to be in the range of 15%.

Description of Fan events & diagnostics:

1) *Normal* Fan is spinning as expected and designed, within the allowed error margin.

Diagnostics Fan is in working condition, no action is required.

2) *Warning* Fan is spinning either faster or slower than expected beyond the allowed error margin but less than 100% difference.

Diagnostics Check for any obstruction or dust and clean the fan. If fan continues to be in warning state, then fan mechanical components may be failing out.

3) *Critical* Fan is spinning either faster or slower than expected with greater than 100% difference from expected value. The fan may be stopped.

Diagnostics Check for any obstruction or dust and clean the fan. If fan continues to be in the Critical state, mechanical components have likely failed or will fail soon and operator needs to seek replacement.

Thermal Events and Diagnostics

What is Thermal Slowdown? Thermal slowdown is a mechanism which automatically reduces clocks when temperature reaches a critical point/threshold usually around 100C. This mechanism helps prevent thermal runaway and permanent damage to the chip/board.

Description of Thermal events & diagnostics:

1) *Normal* Temperature is within normal operating ranges i.e. more than 10C below the slowdown threshold temperature. *Diagnostics* GPU is functioning normally

2) *Warning* Temperature is within 10C of thermal slowdown threshold temperature.

Diagnostics Check if the fan is spinning properly (verify the fan event in addition) and that automatic variable fan control is enabled. Ensure that the system has proper ventilation and system fan is running properly. In addition, any running application should be closed to see if the temperature is returning to normal level after some time. If the warning is cleared by doing this (and normal event is generated), report this behavior to administrator.

3) *Critical* Temperature has reached thermal slowdown threshold temperature.

Diagnostics Check if the temperature is reduced after closing the application(s) running on the system, if the temperature is reduced (and normal event is generated) then report this matter to administrator and stop using that application until its fixed. If the problem persists even without any application running on that system then immediately stop using that GPU and contact administrator for further help. Note that GPU will engage in built-in slow down mechanism, which may reduce the temperature leading to generation of warning or normal event If this happens, please report this behavior to administrator. But if the temperature keeps on rising, then build-in shutdown mechanism will trigger to prevent damage.

NVIDIA

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>

Using NVWMI

General Information about NVWMI Clients

NVWMI implements WMI provider interfaces and general guidelines for WMI provider usage applies. See MSDN article <u>Using WMI</u>. Recommendations for selecting a language for NVWMI client implementation:

• WinScript

Both VisualBasic Script and JavaScript could be used. See MSDN article <u>Scripting API for WMI</u> for details. Implementation complexity is low for simple scripts, but quickly rise with development. Not recommended for any complex clients.

PowerShell

Recommended way to create NVWMI clients of low to moderate complexity. See MS TechNet articles <u>Get-WmiObject</u> and <u>Invoke-WmiMethod</u>. There is a useful utility for exploring WMI objects - <u>WMI Explorer</u>.

Moderate implementation complexity.

• WMIC - WMI Console WMIC is a standard utility for MS Windows since Windows XP. It has command-line interface and could operate in batch or interactive mode. See **Using NVWMI with the WMIC tool** for details. Should not be used in applications.

Recommended as a tool for quick testing only - see also **WMIC Known limitations**.

- .NET/C#/Managed C++ Recommended for complex NVWMI client applications. See MSDN article <u>WMI .NET Overview</u>. High implementation complexity.
- DCOM/C++ Recommended for complex NVWMI client applications. See MSDN article <u>COM API for WMI</u>.

Highest implementation complexity.

Profile data portability

Data of system profiles (see SystemProfile) and display profiles (see **DisplayProfile**) is stored as UTF-16 text files in "%ProgramData%\NVIDIA Corporation\Drs". Data of 3D profiles (see **ApplicationProfile**, **Profile**) is stored in binary format in the same location. Note that %ProgramData% is hidden directory.

NVWMI supports profile data transfer from one system to another with following limitations:

- system components are identical: MB, system BIOS, GPU boards, V-BIOS, CPU, RAM, any additional PCI-E device
- GPU boards are plugged into identical PCIe slots. Ordinal of PCIe slot and data width must match (e.g. if a system1 has Board1(Gpu1) in slot0 with 16 lanes then system2 must have Board1(Gpu1) in slot0 with 16 lanes)
- Monitors are identical and attached in same order via identical connectors
- Running identical OS and NVIDIA driver stack

Known issues

Windows 10-specific

Smart clone is not supported in Windows 10 as of version 2.27.

Windows 7-specific

Using method **DisplayManager::createDisplayGrids** may not produce desired results when layout key is specified. **Display** ordering might be off for 1xN grids, when N>3. This is OS-specific limitation. If you encounter this problem, please use following steps:

- configure desired display configuration using NVIDIA Control Panel
- save it as a set of display profiles using method ProfileManager::saveDisplayProfiles
- when you need to set or restore saved display configuration, use **ProfileManager::applyDisplayProfiles**

Diagnostic and troubleshooting

Controlling NVWMI Service

-In Services GUI application

- browse running services
- locate and select "NVIDIA WMI Provider"
- click "Stop Service" button on the toolbar or "Stop" link in the left pane

You might pause and start the NVWMI service in a similar way. -In command shell (CMD prompt)

To stop NVWMI service:

sc stop nvwmi

To start NVWMI service:

sc start nvwmi

To query whether NVWMI service is running:

sc query nvwmi

Logging off and logging on in interactive local session (not RDP), OS restart will restart NVWMI as well.

Using NVWMI logging to diagnose issues

See **Examples of Useful Logging configurations** . Enabling errors and warnings in the OS event log is recommended to diagnose and triage hardware or driver issues against NVWMI.

NVIDIA <u>Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.</u>





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>

NVIDIA Performance Counters

Introduction

NVWMI provides performance counters with information about the state of NVIDIA hardware in real-time. Available performance counters are described in the manifest file %System%\nvPerfProvider.man.

Performance counters are accessible using standard OS tools such as **Microsoft Performance Monitor** or programmatically. Microsoft supplies the Performance Monitor tool in all supported versions of the OS.

Read <u>About Performance Counters</u> for general outline.

For more technical details see <u>Overview of Windows Performance</u> <u>Monitor</u>.

Available NVIDIA GPU Counters

- Available Memory MB Amount of free memory, available for GPU in megabytes
- Total Memory MB Total amount of memory, available for GPU in megabytes

- Virtual Memory MB Amount of virtual memory, available for GPU in megabytes
- Memory Clock MHz On-board memory clock in megahertz
- Core Clock MHz GPU core clock in megahertz
- % Cooler rate GPU cooler rate in percents
- Temperature C GPU temperature (Celsius)
- Power Consumption mW Consumed power over sampling interval in milliwatts. Duration of a sampling interval (in ms) stored in Gpu::powerSamplingPeriod property
- % GPU Usage The percentage of time where the GPU is considered busy in the last 1 second interval
- % GPU Memory Usage The percentage of GPU Memory currently being utilized by all running applications
- Fan Speed GPU fan speed rotations per minute
- % Bus Usage PCI-Express Bus Usage in percents
- % Video Usage Video Usage in percents
- % FB Usage Frame Buffer Usage in percents
- PCI-E downstream width to GPU Width of the PCI-Express downstream link to GPU (number of lanes)
- PCI-E current width to GPU PCI-Express current width of link to GPU (number of lanes)
- PCI-E current speed to GPU Mbps Current speed of the PCI-E link to GPU (Megabits per second)
- % Video Encoder usage GPU Video Engine encoder usage in percents
- % Video Decoder usage GPU Video Engine decoder usage in percents
- Video Encoder Sessions Number of GPU Video Encoding Sessions
- Video Encoder Average FPS Video Encoder frames per seconds as a trailing average
- Video Encoder Average Latency (ms) Video Encoder average latency in milliseconds

Working with Windows Performance Monitor

1. Monitoring the counter graph

- Run perfmon.exe
- Browse to Performance->Monitoring Tools->Performance Monitor in the navigation bar
- Select item **Add counters** from context menu (right-click in graph area)
- Select target system (local or remote)
- Browse the list to find NVIDIA GPU
- Expand NVIDIA GPU item
- Select any counter from the list, multiple selections supported
- Select the instance of that counter from the instance list below
- Add the counter instance and press OK
- Choose appropriate graph scale and sampling rate

2. Creating a custom counter data set

- Run perfmon.exe
- Select Data Collector Sets
- Select **User Defined node** and bring up its context menu (by right click)
- Select New, Data collector Set
- Select Create data logs, Performance counter and click on Next
- Select Add
- Select desired counters (for example, Processor Time), Add and click on OK
- Select Next
- Browse to a folder, select **Next**
- Click on Change button to specify desired credentials under Run as
- Select Save, Close and click on Finish
- The user data collector is created
- Bring up context menu (right click on the created data set) and select **Start**
- Properties could be changed at any time via context menu (right click)
- To stop collecting data, select **Stop** in the context menu

Diagnostics and troubleshooting

When NVWMI system service is not properly installed, NVIDIA performance counters are also not available. See **Controlling NVWMI Service** for service troubleshooting.

The manifest file **nvPerfProvider.man** must be properly registered and enabled.

Manual installation

- C:\>lodctr /M:nvPerfProvider.man Registers counters by writing GUID to [HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Perflib_V2Providers]
- C:\>lodctr /E:nvWmi64.exe Enables performance counters. For 32bit OS substitute nvWmi.exe
- sc start nvwmi Starts the service, counters should be available.

Attention:

NVWMI service is presumed to be stopped

Manual uninstallation

- C:\>unlodctr /M:nvPerfProvider.man Removes counters GUID from [HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\Perflib_V2Providers]
- C:\>lodctr /D:nvWmi64.exe Disables performance counters. For 32bit OS substitute nvWmi.exe
- **sc stop nvwmi** Stops the service, counters will not be provided anymore.

Attention:

NVWMI service is presumed to be running

NVIDIA

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>

Logging and tracing NVWMI activity

NVWMi equipped with robust logging and tracing capabilities. Clients can select logging types and control logging levels. Supported logging types:

- MS Windows Event log
- Debugger message stream
- File log

Note that for logging types or logging filter changes to be in effect, NVWMI service restart is required. See **Controlling NVWMI Service** stopping and starting NVWMI.

Logging Types

Logging types are controlled by the registry value HKLM\SOFTWARE\NVIDIA Corporation\NVWMI\LogTypes.

It is a bitmask, several values might be combined to enable several logging types at once. Supported types are:

• Debug stream.

Any application, capable of capturing Win32 OutputDebugString. For example - DebugView, WinDbg, Visual Studio debugger.

LogTypes=1

• File log.

%ALLUSERSPROFILE%\nvwmi.log will be produced with records in format
<date> <timestamp> : <message text>

LogTypes=2

• MS OS event log.

Could be examined by Event Viewer - standard utility of MS Windows. Or any other application which is capable of attaching to the MS Windows event log. Event Viewer requires manual refresh (hotkey **F5**) to monitor events. NVWMI events are logged in **Application** log. Event source is **NVWMI**.

LogTypes=4

Setting LogType to 0 will disable logging completely (silent mode).

Configuring Logging Filter

Logging filter is controlled by the registry value HKLM\SOFTWARE\NVIDIA Corporation\NVWMI\LogFilter.

It is a bitmask, several values might be combined to filter out messages of several levels at once.

LogFilter=1 - traces

LogFilter=0x10 - errors

LogFilter=0x100 - warnings

LogFilter=0x1000 - function entries

LogFilter=0x10000 - informational messages

LogFilter=0x11111 will disable message filtering (all messages will be logged),

LogFilter=0 will filter out all messages (silent).

Examples of Useful Logging configurations

This registry script will enable logging to OS event log. Only error messages will be logged, all other messages will be discarded:

[HKEY_LOCAL_MACHINE\SOFTWARE\NVIDIA Corporation\NVWMI] "LogFilter"=dword:00000010 "LogTypes"=dword:00000004

Registry script to enable logging to the debug stream and OS event log. All messages will be logged (maximum verbosity):

[HKEY_LOCAL_MACHINE\SOFTWARE\NVIDIA Corporation\NVWMI] "LogFilter"=dword:00111111 "LogTypes"=dword:0000005

Disable all logging:

[HKEY_LOCAL_MACHINE\SOFTWARE\NVIDIA Corporation\NVWMI] "LogTypes"=dword:00000000

NVIDIA

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>

Using NVWMI with the WMIC tool

Examples of WMIC command-lines

You could set default namespace to root\cimv2\nv (otherwise path <object> won't work) or specify /namespace:nv at the command line:

wmic:root\cli>/namespace:nv

List all **System** singleton properties:

wmic:root\cli>path System get *

Default verb is get, it could be skipped, so command

wmic:root\cli>path System

will produces same output.

List several properties for all GPUs (see **Gpu**) in a system - in this example there are 2 Quadro K5000 GPUs :

wmic:root\cli>path Gpu get id,nvapiId,name,uname

Produces this output:

id name nvapiId uname

1	Quadro	K5000	768	Quadro	K5000	(1	of	2)
2	Quadro	K5000	1024	Quadro	K5000	(2	of	2)

List several properties for a given GPU (see **Gpu**) in a system - in this example GPU is identified by id, power consumption and current GPU load is listed (with several 3D applications running):

```
wmic:root\cli>path Gpu where id=1 get power, percentGpuUsage
percentGpuUsage power
12 44.797
```

Invoking WMI methods in WMIC

Be aware of a difference between singleton class and classes with dynamic instances as it affects the way WMI methods are called in general. See **static vs. dynamic instances** in the Introduction. Calling methods of the **System** class (singleton class, other singleton classes are **DisplayManager** and **ProfileManager**) without any parameters:

wmic /namespace:nv path System call info

Calling methods of the **ProfileManager** class (singleton) with input parameter (in this example - global profile name is "Autodesk Mudbox - compatible"):

wmic /namespace:nv path ProfileManager call setCurrentProfile3D "Aut

Calling methods of any other class (non-singleton) - general format:

```
wmic /namespace:nv path <class name> where <instance selection claus</pre>
```

Calling methods of any other class (non-singleton) - example of calling **Gpu::info** :

wmic /namespace:nv path Gpu where ordinal=1 call info

Getting some properties

wmic /namespace:nv path <class name> where <instance selection claus</pre>

For example, listing just uname and id property of all **Gpu** instances (note that where clause is skipped as it is query for all instances)

wmic /namespace:nv path Gpu get uname,id

Note that where clause is a system-dependent. Example demonstrates calling **Gpu::info** on a system with Quadro 5000 GPU:

wmic /namespace:nv path gpu where name="Quadro 5000" call info

Calling a method of MOF class with dynamic instances and input parameters is performed in similar way. Example demonstrates changing scaling mode to produce image centered on a screen without any scaling (scaling mode value=3) for the instance of the **DisplayGrid** class with unique name "DisplayGrid (1 of 4)"

```
wmic /namespace:nv path DisplayGrid where uname="DisplayGrid (1 of 4
```

The following is an example of calling a method with multiple input parameters using 'named parameters' convention.

```
wmic /namespace:nv path DisplayManager call createDisplayGridById ro
```

The following is an example of creating a desktop using **Profile** class. String expected for creating desktop is

"DesktopName;PerMonitorBackground;WallpaperPath;PerMonitorStyle;" where PerMonitorBackground=1 means to have different desktop background per display (0 means to have same background across all displays), PerMonitorStyle means how background appears on desktop (0 = Center, 1 = Tile, 2 = Stretch)

For 2 display and different background per display the input string is "DesktopName;PerMonitorBackground;WallpaperPath for Display1,PerMonitorStyle for Display1;WallpaperPath for Display2,PerMonitorStyle for Display2" and so on for other displays

This example is for a single display and creates a desktop named "Test" with Style as Tile

wmic /namespace:nv path profile where id=1465693279 call setStringVa

where id is retrieved by querying profile id that corresponds to nView Global

This example is for a single display and modifies the image for a desktop named "Test" and changes Style to Stretch

wmic /namespace:nv path profile where id=1465693279 call setStringVa

Any property could be used to select instance (or multiple instances at once). However, due to WMIC limitations it is not possible to specify numeric values with negative values in a shell command-line, as WMIC treats '-' as an argument separator. See WMIC Known limitations.

How to create new 3D application profile in WMIC

The following is an example of creating 3D application profile and modifying its settings in WMIC. "My profile" used as an example of profile name for executable "myApp.exe". **Setting** "OpenGL video-editing mode" used as an example of typical 3D setting. For a list of all available profile settings see

Create new application profile by invoking **ProfileManager::createProfile** method

wmic /namespace:nv path ProfileManager call createProfile name="My p

Retrieve setting ID from setting name by invoking SettingTable::getIdFromName method

wmic /namespace:nv path SettingTable where type=0 call getIdFromName

Change setting value "OpenGL video-editing mode" (id=552469172) to 1 (e.g. "enabled") by calling **ApplicationProfile::setValueById**. For a list of all available profile settings see **Profile settings available in NVWMI**.

wmic /namespace:nv path ApplicationProfile where name="My profile" c

Query current settings

wmic /namespace:nv path ApplicationProfile where name="My profile" g

In output you should see a line

{} 4 Setting.id=552469172 Setting Setting 2 5

Specifying arrays in WMIC

Some methods take arrays as input parameters. Arrays in WMIC are enclosed in braces () and individual elements are separated by comma. **WARNING**: Usage of any other separator (e.g. space) will lead to undefined results.

Specifying multiple executables in the array of strings for **ApplicationProfile::addApplications** and **ApplicationProfile::removeApplications** :

wmic /namespace:nv path ApplicationProfile where name="Adobe Photosh
wmic /namespace:nv path ApplicationProfile where name="Adobe Photosh

Specifying multiple display grids in the array of strings for DisplayManager::validateDisplayGrids and DisplayManager::createDisplayGrids :

4 displays, connected to one GPU, windows desktop extended on all 4 :

wmic /namespace:nv path DisplayManager call validateDisplayGrids gri
wmic /namespace:nv path DisplayManager call createDisplayGrids grids

4 displays, connected to one GPU, 2 Mosaics 1x2 in 1920x1080, 32-bit pixels at 60 Hz:

wmic /namespace:nv path DisplayManager call validateDisplayGrids gri
wmic /namespace:nv path DisplayManager call createDisplayGrids grids

WMIC Common tasks

Examples of WMIC command lines for several common tasks are given below.

Determine Video BIOS version

Determine Video BIOS version of all GPUs in the system:

wmic /namespace:nv path Gpu get verVBIOS

Determine Video BIOS version of the specific GPU:

wmic /namespace:nv path Gpu where id=1 get verVBIOS

Sample output - 70.0.30.0.a is a string representation of the video BIOS version, **Version::strValue** :

{} 3 Version.orderedValue=10 Version Version 2 10 70.0

Note that GPU could be selected by operator WHERE by any property (e.g. name, uname, id etc.). If a system has single GPU, WHERE operator could be skipped for properties, but it is required for calling a method. Alternative way of obtaining human-readable information about given GPU using **Gpu::info()** method:

wmic /namespace:nv path Gpu where id=1 call info

Search for "Video BIOS version:" in the output.

Determine Display Driver version

Similar approach applies to the common task of determining the display driver version. Following WMIC commands:

wmic /namespace:nv path System get verDisplayDriver

or

wmic /namespace:nv path System call info

Will produce display driver version in human-readable format. Also note that output of **System::info** method will contain exact Video BIOS version string in case when all GPUs in a system are running the same version. Otherwise Video BIOS version will be reported as "mixed".

Determine Current Display Mode

Query properties of class **DisplayGrid** to determine current display mode settings, such as resolution, color depth, refresh frequency:

DisplayGrid::displayModeVirtual contains "effective" screen dimensions, which are taking into account overlaps between individual physical displays:

wmic /namespace:nv path DisplayGrid get displayModeVirtual

DisplayGrid::displayModePhysical contains "raw" physical screen dimensions of individual physical display:

wmic /namespace:nv path DisplayGrid get displayModePhysical

When only one display is active or multiple displays are not combined into display grids, **DisplayGrid::displayModeVirtual** has identical values to the **DisplayGrid::displayModePhysical**.

wmic /namespace:nv path DisplayGrid where id=1 call info

In the output, "current grid mode:" corresponds to the **DisplayGrid::displayModePhysical** and "effective grid mode:" corresponds to the **DisplayGrid::displayModeVirtual**.

Change Current Display mode

Example of invoking method DisplayGrid::setDisplayMode :

wmic /namespace:nv path DisplayGrid where id=1 call setDisplayMode w

Cloning a Display

Cloning is useful for displaying identical images on several displays. It could be achieved in NVWMI by creating a **DisplayGrid** and overlapping displays at display dimensions.

Example of horizontally cloned displays (assuming 2 displays attached to the same GPU, both in display mode with screen width 1920):

 create DisplayGrid 1x2, physical display mode is 1920x1080, 32 bits per pixel, at 60 Hz refresh rate

wmic /namespace:nv path DisplayManager call createDisplayGrids c

• set columnar overlap at display width

wmic /namespace:nv path DisplayGrid where id=1 call setOverlapCc

Note that resolution of displays might differ. When resolutions are different, result will be either clone-to-fit or clone-to-pan. Overlap value must match exactly a dimension (width for horizontal clone, height - for vertical clone) for one of displays.

WMIC Useful options

• execute WMIC command on remote system:

/node:<system name>

Note that if the system name contains a hyphen (-) then it must be escaped or the system name enclosed in single quotes ('):

wmic /node:'wks-1' /namespace:nv path gpu

will get all instances of the **Gpu** class on the system with Windows name **wks-1**

• trace execution of the WMI command

/trace:on

 record WMIC commands including their output with timestamps to an XML file

/record:<file path>

• select format of the WMI command output

/format:<format specifier>

Specifies a key word or XSL file name to format the data. Command

wmic process list /format /?

will list all available formats.

• list all properties with their names when using GET * verb

wmic /namespace:nv path <classname> get * \value

See also WMIC documentation on MSDN

WMIC Known limitations

- WMIC doesn't allow specifying negative numbers. A minus character ('-') is treated as a switch. This prevents some legitimate commands from producing correct output. A slash ('/') and backslash ('\') are also special characters and should be either escaped or a string with these characters should be quoted literally (enclosed in apostrophes '). For example, be sure to quote computer name in **node:** parameter if it contains any special characters.
- WMIC is case-insensitive for class names and properties.
- WMIC doesn't display properties which contain array of object references (for example, **Board::gpus**).

NVIDIA

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>

Using NVWMI with the PowerShell

Examples of PowerShell command-lines

PowerShell has cmdlet Get-WmiObject to directly access and manipulate WMI objects. See MSDN article <u>Using the Get-WMiObject Cmdlet</u>. GWMI and gwmi are recognized as built-in aliases for Get-WmiObject.

Other helpful cmdlets to examine WMI objects are Get-member, Selectobject, Sort-object and Where-object. For more tips of WMI scripting in PowerShell see Microsoft's TechNet article <u>Scripting for WMI</u>

Listing **Gpu::id** property for all **Gpu** instances (note that namespace is case-insensitive):

Get-WmiObject -namespace "root\cimv2\nv" -query "select id from Gpu"

WMI methods invocation:

\$instance=gwmi -namespace "root\cimv2\nv" -class system
\$result=\$instance.invokeMethod("info",\$null,\$null) # \$result will be
\$result # print a result

Alternate way of accessing WMI objects - retrieving an instance of the **System** singleton, calling method **System::info()** and storing result:

```
$instance = [wmiclass]"\\.\root\cimv2\nv:system"
$result=$instance.invokeMethod("info",$null) # note that invokeMetho
$result # print a result
```

Shorter version:

```
([wmiclass]"\\.\root\cimv2\nv:system").invokeMethod("info",$null)
```

Note that it works only for singleton classes (System, DisplayManager, ProfileManager).

Running WQL query which selects certain property from WMI object(s).

```
$qr = Get-WmiObject -namespace "root\cimv2\nv" -Query "select uname
foreach ($t in $qr) {$t.uname}
```

In case of more than one GPUs \$qr will contain more than one element.

Selecting all instances and then iterating:

```
$gpus = Get-WmiObject -namespace "root\cimv2\nv" -class gpu
foreach($o in $gpus) # obtain an instance
{
    # print some properties
    "id="+$o.id+" uname="+$o.uname+" GPU clock="+$o.gpuCoreClockCurr
    # invoke info method
    $o.invokeMethod("info",$null)
}
```

Using profile settings reflection mechanism

Querying all setting names of a given profile type (0 - 3D application profile):

\$qr = Get-WmiObject -namespace "root\cimv2\nv" -Query "select settin \$qr.settingNames[0] # print a name of a setting, \$qr.settingNames is

Querying all setting IDs of a given profile type (0 - 3D application profile):

```
$qr = Get-WmiObject -namespace "root\cimv2\nv" -Query "select settin
$qr.settingIds[0] # print an ID of a setting, $qr.settingIds is an a
```

Obtaining brief information about a profile setting (3D setting with id=545898348 is "Antialiasing - Line gamma")

\$table=gwmi -namespace "root\cimv2\nv" -class SettingTable -filter "
\$table.invokeMethod("infoById",545898348) # get information about th
\$table.invokeMethod("infoByName","Antialiasing - Line gamma") # get

Creating multiple display grids

Creating 4 display grids, single display per each, 4 displays attached to a single GPU

```
$dm = Get-WmiObject -namespace "root\cimv2\nv" -Class DisplayManager
[array]$grids_4_1x1 = @("rows=1;cols=1;layout=1.1","rows=1;cols=1;l
$res = Invoke-WmiMethod -Path $dm.__PATH -Name createDisplayGrids -A
if($res.ReturnValue -eq $true) {"grids are valid"} else {"grids are
```

Creating 1 display grid from 4 displays, 2 displays per column and 2 displays per row, 4 displays attached to a single GPU

\$dm = Get-WmiObject -namespace "root\cimv2\nv" -Class DisplayManager [array]\$grids_1_2x2 = @("rows=2;cols=2;layout=1.1 1.2 1.3 1.4") \$res = Invoke-WmiMethod -Path \$dm.__PATH -Name createDisplayGrids -A if(\$res.ReturnValue -eq \$true) {"success"} else {"failure"}

NVIDIA

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.



NVWMI v2.31 API Reference Documentation NVIDIA



Feb 2018

- Main Page
- Related Pages
- Classes

Profile settings available in NVWMI

OpenGL Settings for 3D profiles

Setting ID (32-bit hex)	Setting ID (32-bit decimal)	Туре	Type of values	Setting name	Desc
0x2089BF6C	545898348	samples	32-bit unsigned integer	Antialiasing - Line gamma	N/A
0x2097C2F6	546816758	samples	32-bit unsigned integer	Deep color for 3D applications	N/A
0x206A6582	543843714	bitfields	32-bit unsigned integer	OpenGL default swap interval	Controls number (signals fi display to before re a frame (SwapInt OpenGL

					to force \ ON or O VSYNCN
0x206C4581	543966593	samples		OpenGL default swap interval fraction	Controls current s evaluate (un)sync with neg intervals in the rar 100%
0x20655CFA	543513850	samples	boolean	OpenGL default swap interval sign	Swap int are treat negative positive v dependir value (0- 1-negativ
0x209DF23E	547222078	samples	32-bit unsigned integer	Event Log Severity Threshold	This sett specifies severity logged ir Windows log by th OpenGL
0x20FF7493	553612435	samples	32-bit unsigned integer	Extension String version	N/A
0x201F619F	538927519	samples	32-bit unsigned integer	Buffer-flipping mode	N/A
			32-bit		

0x204D9A0C	541956620	-	unsigned integer	Force Stereo shuttering	N/A
0x20D0F3E6	550564838	samples	UTF-16 string	Preferred OpenGL GPU	Controls of the de render G Autosele the drive chooses a heurist favors th quadro. string is control p not mean set direc user. The gpu_affir always o the defau GPU.
0x208E55E3	546199011		32-bit unsigned integer	Maximum frames allowed	N/A
0x209AE66F	547022447		32-bit unsigned integer	Exported Overlay pixel types	N/A
0x206C28C4	543959236	-	32-bit unsigned integer	Enable overlay	N/A

P	J		JI	J	v
0x20797D6C	544832876	samples	32-bit unsigned integer	High level control of the rendering quality on OpenGL	N/A
0x20A29055	547524693	samples	32-bit unsigned integer	Unified back/depth buffer	N/A
0x2092D3BE	546493374	bitfields	32-bit unsigned integer	Enable NV_gpu_multicast extension	Controls NV_gpu_ operation supporte
0x20C1221E	549528094	bitfields	32-bit unsigned integer	Threaded optimization	N/A
0x202888C1	539527361	-	32-bit unsigned integer	Event Log Tmon Severity Threshold	This sett specifies severity logged ir tmonlog OpenGL
0x20FDD1F9	553505273	samples	32-bit unsigned	Triple buffering	N/A

	integer	
JJJJJJJ	 ļ	 II

Shared Direct3D and OpenGL Settings for 3D profiles

Setting ID (32-bit hex)	Setting ID (32-bit decimal)	Туре	Type of values	Setting name	Descriptio
0x10ECDB82	283958146	samples	32-bit unsigned integer	Antialiasing - Behavior Flags	Flags for altering hov the driver interprets 'Antialiasin(- Setting'
0x10FC2D9C	284962204	samples	unsigned	Antialiasing - Transparency Multisampling	N/A

0x107D639D	276652957	samples	unsigned	Antialiasing - Gamma correction	N/A
0x10D773D2	282555346	samples	32-bit unsigned integer	Antialiasing -	Controls method and number of antialising

		samples

0x10D48A85	282364549	samples	32-bit unsigned integer	Antialiasing - Transparency Supersampling	N/A
0x107EFC5B	276757595	samples	32-bit unsigned integer	Antialiasing - Mode	N/A
0x107AFC5B	276495451	samples	boolean	Antialiasing - SLI AA	N/A
0x101E61A9	270426537	samples	32-bit unsigned integer	Anisotropic filtering setting	Controls number of samples ar algorithm o anisotropic filtering
			32-bit	Anisotropic	

0x10D2BB16	282245910	samples	unsigned integer	filtering mode	N/A
0x1035DB89	271965065	samples	boolean	NVIDIA Predefined Ansel Usage	Empower an app profile to disallow Ansel
0x1075D972	276158834	samples	boolean	Enable Ansel	Toggle Ansel on off
0x1085DA8A	277207690	samples	boolean	Ansel flags for enabled applications	Temporar whitelistin of apps allowed to enable Ansel
0x104554B6	272979126			Application Profile Notification Popup Timeout	This settin specifies how many seconds to popup displaying information about white profile is being applied should appear. Z counts as disabled.
					Steam Applicatic

0x107CDDBC	276618684	samples		Steam Application ID	ID is used t identify which Steam application: are installe
0x10115C89	269573257	samples	32-bit unsigned integer	Battery Boost	Enables the Battery Boost functionality cap FPS fo DC mode only
0x106D5CFF	275602687	samples	boolean	Do not display this profile in the Control Panel	This setting indicates to the Control Panel that given profil should not be displayed.
0x10354FF8	271929336	samples	UTF-16 string	List of Universal GPU ids	Unicode string containing list of Universal GPU ids, item separator is ';'. Exposec in UI as 'CUDA - GPUs'.
0x10D1EF29	282193705	samples	UTF-16 string	Maximum GPU Power	Maximum power, that GPU is

					allowed to use
0x108F0841	277809217	samples	boolean	Export Performance Counters	N/A
0x1034CB89	271895433	samples	boolean		Empowers an app profile to disallow FXAA
0x1074C972	276089202	samples	boolean	Enable FXAA	Toggle FXAA on o off
0x1068FB9C	275315612	samples	boolean		Toggle FXAA Indicator oi or off
0x10287051	271085649	samples	boolean	SLI Indicator	Show or hide the SL on-screen indicator
0x10444444	272909380	samples	boolean	NVIDIA Quality	Toggle NVIDIA Quality upscaling o or off
0x10F9DC83	284810371	range	-	Maximum AA samples allowed for a given application	Maximum AA we are going to allow for a given application
0x1094F16F	278196591	samples	boolean	Display the PhysX indicator	N/A

0x1057EB71	274197361	range	-	Power management mode	Preferentia power state
0x103BCCB5	272354485	samples	boolean	No override of Anisotropic filtering	This setting tells our UI that it cannot override Anisotropic filtering for this application
0x10834FEE	277041134	bitfields	32-bit unsigned integer	Frame Rate Limiter	Framerate Limiter parameters bit 31 to enable and LOWBYTE set to frames/sec

0x10834FFF	277041151	bitfields		Frame Rate Limiter 2 Control	Framerate Limiter 2 controls
0x10834F01	277040897	bitfields	32-bit unsigned integer	Frame Rate Monitor	FRM parameter (deprecate

0x10834F05	277040901	bitfields	32-bit unsigned integer	Frame Rate Monitor Control	FRM contro parameters
0x10F9DC82	284810370	samples		Maximum resolution allowed for a given application	Maximum resolution we are going to allow for a given application
0x10F9DC80	284810368	bitfields	unsigned	Optimus flags for enabled applications	White list fc shim layer

0x10F9DC81	284810369	bitfields	unsigned	Enable application for Optimus	White list fc shim layer per application
0x10F9DC84	284810372		unsigned integer	Shilli Bondoring	Rendering Mode Options for shim layer per application

0x1033DCD1	271834321	samples	32-bit unsigned integer	Number of GPUs to use on SLI rendering mode	User visib exposed number o GPUs to u on SLI
0x1033DCD2	271834322	samples	32-bit unsigned integer	NVIDIA predefined number of GPUs to use on SLI rendering mode	Setting to indicate ir the Contro Panel how many GP to use by default or this SLI profile
0x1033DCD3	271834323	samples	32-bit unsigned integer	NVIDIA predefined number of GPUs to use on SLI rendering mode on DirectX 10	Setting to indicate in the Contro Panel how many GP to use by default on this SLI profile on DirectX10

0x1033CEC1	271830721		unsigned	NVIDIA predefined SLI mode	indicate in the Contro Panel whic SLI mode i active by default
0x1033CEC2	271830722	samples	32-bit unsigned integer	NVIDIA predefined SLI mode on DirectX 10	Setting to indicate in the Contro Panel whic SLI mode i active by default on DirectX 10
0x1033CED1	271830737	samples	32-bit unsigned integer	SLI rendering mode	User visibl exposed S Modes
0x10111133	269553971		U U U	Virtual Reality pre-rendered frames	N/A
0x1094F157	278196567	samples	boolean	Toggle the VRR global feature	N/A
0x1095F16F	278262127	samples	boolean	Display the VRR Overlay Indicator	N/A
0x1094F1F7	278196727	samples	32-bit unsigned	VRR requested	N/A

			integer	state	
0x10A879CF	279476687	samples	32-bit unsigned integer	G-SYNC	Profile- specific override. Overrides the VRR_MOD global setting
0x10A879AC	279476652	samples	32-bit unsigned integer	G-SYNC	Profile- specific override. Overrides the VRR_MOD global setting
0x1194F158	294973784	samples	32-bit unsigned integer	Enable G- SYNC globally	N/A
0x101AE763	270198627	samples	boolean	Flag to control smooth AFR behavior	Key to control smooth AF mode
0x10A879CE	279476686		32-bit unsigned integer	Variable refresh Rate	Controls enabling or disabling VRR for OGL or D3
0x10FDEC23	285076515	-	32-bit unsigned integer	Vsync - Behavior Flags	Flags for altering how the driver interprets VSYNC

-	0x11AE435C	296633180	samples	32-bit unsigned integer	eyes	Swaps image for the left eye with image for the right eye
	0x11E91A61	300489313		32-bit unsigned integer	Display mode	Display mode to us when stere is enabled

0x11112233	286335539		32-bit unsigned integer	Memory Allocation Policy	N/A
0x112493BD	287609789		32-bit unsigned integer	Stereo - Dongle Support	Control of the stereo dongle
0x11AA9E99	296394393	samples	32-bit unsigned integer	Stereo - Enable	Support of the stereo API for workstatior
0x11333333	288568115	-	32-bit unsigned integer	Stereo � swap mode	N/A

System Profile Settings

Setting ID (32-bit hex)	Setting ID (32-bit decimal)	Туре	Type of values	Setting name	Description	Value
0x5ABCCB6D	1522322285	samples	UTF- 16 string	Name of display profile	N/A	×
0x5A362416	1513497622	samples	UTF- 16 string	Name of global 3D profile	N/A	*
0x5A22291F	1512188191	samples	UTF- 16 string	Name of nView profile	N/A	×

Direct3D Settings for 3D profiles

Setting ID (32-bit hex)	Setting ID (32-bit decimal)	Туре	Type of values	Setting name	Descriptio
0x00667329	6714153	-	32-bit unsigned integer	Ambient Occlusion	Ambient occlu mode
0x00664339	6701881	-	unsigned	Predefined	Setting provide to CPL to indic whether the NVIDIA driver uses Ambient Occlusion or n
0x00638E8F	6524559	samples		Texture filtering - Driver Controlled LOD Bias	ignored if LODBIASADJ is set
0x00041807	268295	samples	boolean	Enable GTX950 specific features	Enable GTX95 specific featur Currently usec enable FRM fc GTX950
0x00B65E72	11951730	samples	boolean	Export Performance Counters for DX9 only	N/A
0x00DB1337	14357303	samples	UTF-16 string	ICafe Settings	Configure Scri for ICafe Logo display in XP (only).

0x00738E8F	7573135	range	unsigned	Texture filtering - LOD Bias	N/A
0x0098C1AC	10011052	samples	boolean	Enable sample interleaving (MFAA)	When enablec driver will char the sample pattern positio each frame or each refresh c to emulate a higher quality antialiasing
0x007BA09E	8102046	range	unsigned	Maximum pre-rendered frames	Set the maxim number of presents that (be queued bef CPU blocks.
0x00198FFF	1675263	samples	boolean	Shader Cache	Enables/Disat strategy
0x00E73211	15151633	samples		Texture filtering - Anisotropic sample optimization	N/A
0x0084CD70	8703344	samples	boolean	Texture filtering - Anisotropic filter optimization	N/A
0x002ECAF2	3066610	samples		Texture filtering - Trilinear optimization	N/A
				Texture	

0x0019BB68	1686376	samples	boolean	filtering - Negative LOD bias	N/A
0x00CE2691	13510289	samples	32-bit unsigned integer	Texture filtering - Quality	N/A
0x0064B541	6600001	samples	32-bit unsigned integer	Preferred refresh rate	Enables refres rate override f digital monitor that support 110Hz/120Hz
0x00AE785C	11434076	samples	32-bit unsigned integer	PowerThrottle	N/A
0x00AB8687	11241095	samples	32-bit unsigned integer		This regkey defines the va set in the VAB

0x00A879CF	11041231	samples	32-bit unsigned integer	Vertical Sync	N/A
0x005A375C	5912412	samples	32-bit unsigned integer	Vertical Sync Tear Control	Tear control: if enabled and vsync is forced on, framerates clamped to 60 30/20/15) hz, l rates lower tha that run with vsync disabled

Display Profile Settings

Setting ID (32-bit hex)	Setting ID (32-bit decimal)	Туре	Type of values	Setting name	Descriț
0x5818A91D	1478011165	samples	UTF-16 string	cloneGroupIds	Clone g IDs in a
0x58E21BD4	1491213268	range	32-bit unsigned integer	cols	Numbe column: a displa grid
0x5822D5EE	1478677998	samples	UTF-16 string	gridCSC	File pat Color S Conver: data for display
0x58D3388F	1490237583	samples	UTF-16 string	gridGammaRamp	File pat Gamma Ramp c for a dis grid
0x58BA728B	1488614027	range	32-bit unsigned integer	positionCol	Display GDI col position pixels)
0x584B70FE	1481339134		32-bit unsigned integer	positionRow	Display GDI rov position pixels)

0x58D4B0B4	1490333876	samples	32-bit unsigned integer	gridScaling	Display scaling mode
0x58B21E43	1488068163	samples	UTF-16 string	displayIds	Display in a gr
0x58C7B07C	1489481852	samples	UTF-16 string	displayMode	Mode (display
0x58DECFA8	1490997160	samples	UTF-16 string	rotation	Per-dis rotatio
0x5822918D	1478660493	range	32-bit unsigned integer	rows	Numbe rows ir display
			UTF-16		Distand betwee display columr Positiv

0x58EB619D	1491820957	samples	string	overlapCols	number indicate overlap negativ gap (in pixels)
0x58DD36C1	1490892481	samples	UTF-16 string	overlapRows	Distanc betwee displays row. Positive number indicate overlap negativo gap (in pixels)

Profile for nView Settings

Setting ID (32-bit hex)	Setting ID (32-bit decimal)	Туре	Type of values	Setting name	D
0x596265E8	1499620840	samples	UTF-16 string	Active Desktop Name	The name (desktop
0x5971500B	1500598283	samples	UTF-16 string	Add a desktop	Add a new specified pr desktop na flag;path to wallpaper option. Per be 0 or 1, w are 0 - cent stretch. Exa MyDesktop
0x59AFFA30	1504705072	samples	UTF-16 string	Desktop Properties	Get the pro name and v the specifie
0x59E85D5D	1508400477	samples	UTF-16 string	List of Available Desktops	List with all
0x5932B06F	1496494191	samples	UTF-16 string	Desktop - Modify Properties	Modify prop Format is <i>d</i> <i>monitor fla</i> <i>with wallpa</i> <i>image,wall</i> monitor flaç wallpaper o center, 1 - t Example: <i>MyDeskto</i>
			UTF-16	Desktop	

0x5935FCD2	1496710354	samples	string	Remove	Remove a (
0x59224B67	1495419751	samples	32-bit unsigned integer	Enable Menu Collapse Desktop	Enable/Disa Option Coll desktop
0x591C2281	1495016065	samples	32-bit unsigned integer	Enable Desktop Active Notification	Enable/Disa desktop in \ notification
0x599EE73D	1503586109	samples	32-bit unsigned integer	Enable Desktop Allow Different Resolutions	Enable/Disa to use diffe

0x59FB742F	1509651503	samples		Enable Desktop Name	Enable/Disa name when
0x59D055DC	1506825692	samples	32-bit unsigned integer	Enable Desktop Switching Speed	Enable/Disa desktop sw
0x599A2B4E	1503275854	samples	32-bit unsigned integer	Enable Desktop Multiple Desktops	Enable/Disa desktops
0x59F95C1D	1509514269	-	-	Enable Menu Individual settings	Enable/Disa Option Indiv

0x590CE624	1494017572		32-bit unsigned integer	Enable Menu Lock Title Bar	
0x59D49F0B	1507106571	samples			Enable/Disa Option nVie Max)
0x592A2071	1495933041		32-bit unsigned integer	Enable Menu Send Application Desktop	Enable/Disa Option Sen desktop n

0x59873C	39 1502035001	samples	32-bit unsigned integer	Enable Menu Send Application Display	Enable/Disa Option Sen display n
0x59718C	A3 1500613795	samples	32-bit unsigned integer	Enable Menu Send Window Desktop	Enable/Disa Option Sen desktop n
0x59F487	56 1509197654	samples	32-bit unsigned integer	Enable Menu Send Window Display	Enable/Disa Option Sen monitor n
			32-bit		

0x5999A457	1503241303			Enable Menu Always Top	Enable/Disa Option 'Alw
0x59CF13F4	1506743284	samples	linteger	Enable nView Options System Menus	Enable/Disរ options to s
0x590B712C	1493922092	samples	32-bit unsigned integer	Enable Title Bar Collapse	Enable/Disa Collapse to
0x599F9564	1503630692	-	32-bit unsigned integer	Enable Title Bar Maximize	Enable/Disa Maximize

0x59F14C39	1508985913	samples	32-bit unsigned integer	Enable Title Bar Next Display	Enable/Disa Next displa
0x59BBCC16	1505479702	samples	32-bit unsigned integer	Enable Title Bar nView Options	Enable/Disa nView optic
0x59F9425B	1509507675		32-bit unsigned integer	Enable Menu Transparent	Enable/Disa Option Trar

0x5951917B	1498517883	samples	32-bit unsigned integer	Enable Menu Visible All Desktops	Enable/Disa Option Visil
0x5971E144	1500635460	samples	32-bit unsigned integer	Show Gridline Editor feature	Show/Hide feature
0x59C8134B	1506284363	samples	32-bit unsigned integer	Show Hotkey Save Workspace State	Show/Hide will save wo current disp manageme application
0x597D8B59	1501399897	-	-		
0x59414AE2	1497451234	samples	32-bit unsigned integer	Show Hotkey Display Grid	

0x59987658	1503163992	samples	32-bit unsigned integer	Show Hotkey Activate Desktop	Show/Hide will Activat
0x5915308A	1494560906	samples	32-bit unsigned integer	Show Hotkey Adjust Display Gamma	Show/Hide adjust disp
0x59CFA545	1506780485	samples		Show Hotkey Toggle Always Top	Show/Hide Toggle alwa
0x5933274D	1496524621	samples	32-bit unsigned integer	Show Hotkey Adjust Display Brightness	Show/Hide adjust disp
0x590FC640	1494206016	samples		Show Hotkey Collapse All Windows	Show/Hide Collapse al
			32-bit	Show Hotkey	

0x59C3330F	1505964815	-	unsigned integer	Collapse To Desktop	Show/Hide Collapse to
0x591AE39E	1494934430	-	32-bit unsigned integer	Show Hotkey Adjust Display Contrast	Show/Hide adjust disp
0x592CEC41	1496116289		32-bit unsigned integer	Show Hotkey Desktop Lock	
0x5910E51F	1494279455	-	32-bit unsigned integer		Show/Hide gamma bri contrast to
0x59FEA9B8	1509861816		32-bit unsigned integer	Show Hotkey Display Gather All Monitor1	Show/Hide Send all wi
0x596F125A	1500451418		32-bit unsigned integer	Show Hotkey Load Profile	

0x592C5C9D	1496079517	samples	32-bit unsigned integer	Show Hotkey Max Restore	
0x59F10483	1508967555	samples	-	Show Hotkey Rotate Display	Show/Hi will rotati desktop) specify
0x593A2A63	1496984163	samples	unsigned	Show Hotkey Run Application	Show/Hi will run a applicati
0x595EB7DA	1499379674	samples		Show Hotkey Task Toggle All Applications	Show/Hi will allow between desktops
0x596C172F	1500256047	samples	32-bit unsigned integer	Show Hotkey Task Switcher Toggle Desktop Applications	Show/Hi allows to applicati

0x596BF876	1500248182		32-bit unsigned integer	Show Hotkey Task Switcher Toggle Desktops	Show/Hide allows to to
0x59F512C8	1509233352	samples	32-bit unsigned integer	Show Hotkey nViewToolbar Toggle	
0x597DDC7F	1501420671	samples	32-bit unsigned integer	Show Hotkey Restore Workspace State	Show/Hide will restore state incluc desktop ma open applic
0x59EEC40F	1508819983	-	-	Show Hotkey Run Control Panel	Show/Hide will open th panel
0x59A0B829	1503705129		32-bit unsigned integer	Show Hotkey Save Profile	Show/Hide will save pr
			32-bit	Show Hotkey	

0x59979785	1503106949	samples	-	Move Window Desktop	Show/Hide will move w
0x59AE7814	1504606228	samples	32-bit unsigned integer	Show Hotkey Move Window Display	Show/Hide will move w
0x5989E671	1502209649	samples	32-bit unsigned integer	Show Hotkey Move Window Next Display	Show/Hide will move w display
0x591C9979	1495046521	samples	-	Show Hotkey Desktop Name	Show/Hide will show de
0x5908B520	1493742880		•	Show Hotkey nView Options	Show/Hide will Show n
0x5944AE99	1497673369	-	32-bit unsigned integer	Show Hotkey Toggle Clone	Show/Hide will toggle c off

0x5917CCEE	1494732014	-	32-bit unsigned integer	Show Hotkey Toggle LCD Scaling	Show/Hide will toggle L off
0x5900FA12	1493236242	-	32-bit unsigned integer	Show Hotkey Toggle Output	Show/Hide will Switch 1 device
0x59D5D836	1507186742	samples	32-bit unsigned integer	Show Hotkey Toggle Transparency	Show/Hide will Toggle 1
0x5993EB80	1502866304	samples	unsigned		Show/Hide will show aı window
0x59400FAE	1497370542	-	32-bit unsigned integer	Show Hotkey Zoom Type	Show/Hide toggles zoo between Cı Glass and I

0x59D169D7	1506896343	samples		Profile Current	The name
0x59ED4CAE	1508723886	samples	UTF-16 string	Profile Delete	Delete the from nView
0x593C6EED	1497132781	samples	32-bit unsigned integer	Profile Load	Load the se nView Desl current nVie Manager se replaced w
0x59EF38A2	1508849826	samples	32-bit unsigned integer	Profile Lock	Lock the se
0x5980CD81	1501613441	samples	UTF-16 string	Profile Save	Save curre Manager se profile
0x59A27815	1503819797	samples	unsigned	Show Applications feature	Show/Hide feature
0x598170AB	1501655211	samples	unsigned	Show Virtual Desktop Editor feature	Show/Hide Editor featu
			32-bit	Show Hotkey	Show/Hide

0x59014748	1493256008	samples	-	Manager feature	feature
0x59A7D007	1504169991	samples		Show Profile Manager feature	Show/Hide feature
0x5935F891	1496709265	samples		Show User Interface feature	Show/Hide feature
0x5972BABB	1500691131	samples	32-bit unsigned integer	Show Window Manager feature	Show/Hide feature
0x59F49C46	1509203014	samples	32-bit unsigned integer	Show Zoom feature	Show/Hide

NVIDIA

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- <u>Main Page</u>
- <u>Related Pages</u>
- <u>Classes</u>

Third-Party Notice

Rapid XML is used under the following MIT license:

Copyright (c) 2006, 2007 Marcin Kalicinski

Permission is hereby granted, free of charge, to any person obtainin of this software and associated documentation files (the "Software") in the Software without restriction, including without limitation th to use, copy, modify, merge, publish, distribute, sublicense, and/or of the Software, and to permit persons to whom the Software is furni subject to the following conditions:

The above copyright notice and this permission notice shall be inclu copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABIL FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SH THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARIS OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALIN IN THE SOFTWARE.

> NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>

Application Class Reference

Application, associated with an application profile. More...

List of all members.

Public Member Functions

string info ()

Formats basic information about this **Application** into a human-readable string.

Public Attributes

Version verClass

uint32 **id** Unique id of an application, used as a key. string name User-friendly name of an application. string subPaths Unique subpaths to identify an application, separated by ';'. string launcher When non-empty, subpath to a launcher. boolean isPredefined If true, application is predefined and cannot be removed. boolean isSupported If true, profile is predefined and cannot be removed.

Detailed Description

Application, associated with an application profile.

Member Function Documentation

string info ()

Formats basic information about this **Application** into a human-readable string.

Member Data Documentation

uint32 id

Unique id of an application, used as a key.

boolean isPredefined

If true, application is predefined and cannot be removed.

boolean isSupported

If true, profile is predefined and cannot be removed.

string launcher

When non-empty, subpath to a launcher.

string **name**

User-friendly name of an application.

string subPaths

Unique subpaths to identify an application, separated by ';'.

Version verClass

Version of Application class

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- <u>Main Page</u>
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>

ApplicationProfile Class Reference

Application profile. More...

List of all members.

Public Member Functions

boolean setValueById ([in]uint32 settingId,[in]uint32 value) Set the 32-bit value by a setting ID.
boolean setBinaryValueById ([in]uint32 settingId,[in]uint8 value[]) Set the binary value by a setting ID.
boolean setStringValueById ([in]uint32 settingId,[in]string value) Set the string value by a setting ID.
boolean restoreSettings ([in]uint32 settingIds[]) Restore or delete settings, specified by IDs. Predefined settings get restored and non-predefined settings get deleted.

string info ()

formats basic information about a profile into a humanreadable string

boolean **addApplications** ([in]string appNames[]) Add applications to a **Profile**.

boolean **removeApplications** ([in]string appNames[]) Remove applications from a profile. Only non-predefined applications can be removed.

Public Attributes

Version verClass uint32 id Unique id of a profile, used as a key. string name Name of a profile. boolean isPredefined If true, profile is predefined and cannot be removed. boolean isSupported If true, profile is supported on this system. Setting settings [] Array of settings for a profile. Application applications [] uint32 type Application Profile type. Possible values are: 0 - 3D Application, 2 - nView Application. DATETIME startTime Time when a profile was activated.

Detailed Description

Application profile.

Member Function Documentation

boolean addApplications ([in] string appNames[])

Add applications to a **Profile**.

Parameters:

appNames Array of Application names. In WMIC array of strings is: ("app1.exe", "app2.exe", "app3.exe")

string info ()

formats basic information about a profile into a human-readable string

boolean removeApplications ([in] string appNames[])

Remove applications from a profile. Only non-predefined applications can be removed.

Parameters:

appNames Array of Application names. In WMIC array of strings is: ("app1.exe", "app2.exe", "app3.exe")

boolean restoreSettings ([in] uint32 settingIds[])

Restore or delete settings, specified by IDs. Predefined settings get restored and non-predefined settings get deleted.

Parameters:

settingIds Array of Setting IDs

boolean setBinaryValueById ([in] uint32 *settingId*, [in] uint8 *value*[])

Set the binary value by a setting ID.

Parameters:

[in] uint32 value

Set the 32-bit value by a setting ID.

Parameters:

settingId Setting ID value 32-bit value

Member Data Documentation

)

Application applications[]

Array of applications associated to the Application Profile

uint32 id

Unique id of a profile, used as a key.

boolean isPredefined

If true, profile is predefined and cannot be removed.

boolean isSupported

If true, profile is supported on this system.

string name

Name of a profile.

Setting settings[]

Array of settings for a profile.

DATETIME **startTime**

Time when a profile was activated.

uint32 type

Application Profile type. Possible values are: 0 - 3D **Application**, 2 - nView **Application**.

Version verClass

Version of a profile class

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

Board Class Reference

Represents a board with NVIDIA GPU(s). More...

List of all members.

Public Member Functions

string info ()

Formats basic information about **Board** instance into a human-readable string.

Public Attributes

Version ver

uint32 **id** Unique id of named object, used as a key.

string name Name of an object. string **uname** Unique name of an object. sint32 ordinal Ordinal number of named object among objects with same names. sint32 count Total number of named objects with same name. string serialNumber uint32 nvapild **Board** ID from NVAPI. string chipSKU SKU of the GPU chip. string chipSKUMod SKU modifier of the GPU chip. string **project** Project(Board) number. string projectSKU Project(Board) SKU number. Gpu ref gpus [] Array of references to GPUs installed on this board. ThermalProbe ref thermalProbes [] Array of references to thermal probes of this board. Cooler ref coolers [] Array of references to coolers of this board.

Detailed Description

Represents a board with NVIDIA GPU(s).

Member Function Documentation

```
string info ( )
```

Formats basic information about **Board** instance into a human-readable string.

Member Data Documentation

string chipSKU

SKU of the GPU chip.

string chipSKUMod

SKU modifier of the GPU chip.

Cooler ref coolers[]

Array of references to coolers of this board.

sint32 count

Total number of named objects with same name.

Gpu ref gpus[]

Array of references to GPUs installed on this board.

uint32 id

Unique id of named object, used as a key.

string name

Name of an object.

uint32 nvapild

Board ID from NVAPI.

sint32 ordinal

Ordinal number of named object among objects with same names.

string project

Project(Board) number.

string projectSKU

Project(Board) SKU number.

string serialNumber

Board Serial Number stored in the InfoROM

ThermalProbe ref thermalProbes[]

Array of references to thermal probes of this board.

string **uname**

Unique name of an object.

Version ver

Version of named object

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

Cooler Class Reference

Represents a cooler (fan, liquid system etc.). More...

List of all members.

Public Member Functions

string info ()

Formats basic information about the **Cooler** object in a system into a human-readable string.

Public Attributes

Version verClass uint32 id Unique ID. uint32 handle Unique ID - deprecated, please use 'id'. uint32 coolerType The type of cooler present on the card (fan, water or liquid). Possible values are: 0 - unknown, 1 - fan, 2 - water, 3 - liquid. uint32 percentCoolerRate Current percentage speed of a cooler fan (or liquid flow rate %). uint32 fanSpeed The tachometer reading for fan speed in rpm. uint32 maxSpeed Maximal fan speed in rpm. uint32 minSpeed Minimal fan speed in rpm. uint32 coolerLevel **Cooler** fan speed event level. Possible values are: 0 - unknown, 1 - normal, 2 - warning, 3 - critical.

Detailed Description

Represents a cooler (fan, liquid system etc.).

Member Function Documentation

string info ()

Formats basic information about the **Cooler** object in a system into a human-readable string.

Member Data Documentation

uint32 coolerLevel

Cooler fan speed event level. Possible values are: 0 - unknown, 1 -

normal, 2 - warning, 3 - critical.

uint32 coolerType

The type of cooler present on the card (fan, water or liquid). Possible values are: 0 - unknown, 1 - fan, 2 - water, 3 - liquid.

uint32 fanSpeed

The tachometer reading for fan speed in rpm.

uint32 handle

Unique ID - deprecated, please use 'id'.

uint32 id

Unique ID.

uint32 maxSpeed

Maximal fan speed in rpm.

uint32 minSpeed

Minimal fan speed in rpm.

uint32 percentCoolerRate

Current percentage speed of a cooler fan (or liquid flow rate %).

Version verClass

Cooler class version

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

CoolerEvent Class Reference

Represents GPU cooler events. More...

List of all members.

Public Attributes

uint32 handleGpu uint32 coolerIndex Cooler index in array of coolers on a given GPU. uint32 coolerLevel Cooler level to identify the cooler zone. Possible values are: 0 unknown, 1 - normal, 2 - warning, 3 - critical.

Detailed Description

Represents GPU cooler events.

Member Data Documentation

uint32 coolerIndex

Cooler index in array of coolers on a given GPU.

uint32 coolerLevel

Cooler level to identify the cooler zone. Possible values are: 0 - unknown, 1 - normal, 2 - warning, 3 - critical.

uint32 handleGpu

Unique identification of the NVIDIA GPU

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>

DesktopManager Class Reference

Management of nView desktops. More ...

List of all members.

Public Member Functions

```
string getAllDesktops ()
```

Get a string with names of all Virtual Desktops. Desktop names are separated by semicolon. Returns empty string when Desktop Manager is disabled.

- boolean **createDesktop** ([in]string name,[in]string backgrounds) Create Virtual Desktop for nView Desktop Manager.
- boolean **editDesktop** ([in]string name,[in]string backgrounds) Edit properties of existing Virtual Desktop for nView Desktop Manager.

boolean deleteDesktop ([in]string name)

Delete Virtual Desktop from nView Desktop Manager.

Public Attributes

Version verClass

Detailed Description

Management of nView desktops.

Member Function Documentation

boolean createDesktop ([in] string *name*, [in] string *backgrounds*)

Create Virtual Desktop for nView Desktop Manager.

Parameters:

name nView desktop name backgrounds string with per-display array of paths to the file with background image and style, separated by semicolon. Default style is "center" (0). for example: "C:\ProgramData\Documents\My Pictures\Sample Pictures\Jellyfish.jpg#1;C:\ProgramData\Documents\My Pictures\Sample Pictures\Koala.jpg#0"

boolean deleteDesktop ([in] string name)

Delete Virtual Desktop from nView Desktop Manager.

Parameters:

name nView desktop name

boolean editDesktop ([in] string *name*, [in] string *backgrounds*)

Edit properties of existing Virtual Desktop for nView Desktop Manager.

Parameters:

name nView desktop name backgrounds string with per-display array of paths to the file with background image and style, separated by semicolon. Default style is "center" (0). for example: "C:\ProgramData\Documents\My Pictures\Sample Pictures\Jellyfish.jpg#1;C:\ProgramData\Documents\My Pictures\Sample Pictures\Koala.jpg#0"

string getAllDesktops ()

Get a string with names of all Virtual Desktops. Desktop names are separated by semicolon. Returns empty string when Desktop Manager is disabled.

Member Data Documentation

Version verClass

Object version

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>

Display Class Reference

Represents physical display. More...

List of all members.

Public Member Functions

boolean restoreNativeDisplayMode () Restore the display mode to native.
boolean saveEDID ([in]string filePath) save EDID into specified file (in text and binary format).
boolean setEDID ([in]string filePath) Override EDID with specified data. File can be text or a
boolean setScaling ([in, ValueMap{"0","1","2","3","5","6","7"}, Values{"Default","Closest","Native","ScanoutToNative",", uint32 scaling) Change scaling mode of this display. boolean **setRotation** ([in, ValueMap{"0","1","2","3"}, Values{"No uint32 **rotation**) Rotate display.

string getCurrentTiming ()

Get display's current timing into a human-readable string boolean **saveGammaRamp** ([in]string filePath)

Save display Gamma Ramp into specified file (binary fo boolean **setGammaRamp** ([in]string filePath)

Override display Gamma Ramp data from specified file

- boolean **setGammaRampBasic** ([in]real32 brightness,[in]real32 Calculate and set Gamma Ramp from basic color contro
- boolean saveCSC ([in]string filePath)

Save Color Space Conversion data into specified file (bi boolean **setCSC** ([in]string filePath)

Override Color Space Conversion data from specified fil boolean **setDisplayMode** ([in]uint32 width,[in]uint32 height,[in]re

Change display mode of a display grid, which contains t boolean **setDisplayModeById** ([in]uint32 id)

Change display mode of a display grid, which contains t boolean **setDisplayModeByRef** ([in]**DisplayMode** ref mode)

Change display mode of a display grid, which contains t

boolean **setDither** ([in, ValueMap{"0","1","2"}, Values{"Default","I bit","10 bit"}] uint32 bits,[in, ValueMap{"0","1","2","3","4"} Values{"SpatialDynamic","SpatialStatic","SpatialDynami Set dithering parameters.

string info ()

Formats basic information about this display into a huma

Public Attributes

Version ver uint32 id

Unique id of named object, used as a key.

string **name**

Name of an object.

string uname

Unique name of an object.

sint32 ordinal

Ordinal number of named object among objects with sar

sint32 count

Total number of named objects with same name.

string locus

uint32 nvapild

NVAPI Display ID.

Version verFirmware

Firmware version of this display.

string make

Manufacturer of this display.

string model

Model of this display.

DisplayMode displayModeNative

Native display mode.

uint32 scaling

Display scaling mode - Default, Closest, Native, Scanou ScanoutToClosest. Possible values are: 0 - Default, 1 - (AspectScanoutToClosest, 7 - ScanoutToClosest.

uint32 rotation

rotation angle. Possible values are: 0 - None, 1 - 90 deg uint8 EDID []

Raw EDID of this display.

uint32 EDIDSize

Size of the EDID in bytes.

sint32 gpuConnectorType

Type of physical connector on a GPU side. Possible vali S-Video, 0x13 - TV HDTV Component, 0x14 - TV SCAR 4120, 0x18 - PcPod HDTV-YPrPb, 0x19 - PcPod S-vide Composite, 0x30 - DVI-i, 0x31 - DVI-d, 0x32 - ADC, 0x3 external, 0x47 - DP internal, 0x48 - DP mini, external, 0> LFH DP 2.

sint32 displayConnectorType

Type of physical connector on a display side. Possible v S-Video, 0x13 - TV HDTV Component, 0x14 - TV SCAR 4120, 0x18 - PcPod HDTV-YPrPb, 0x19 - PcPod S-vide Composite, 0x30 - DVI-i, 0x31 - DVI-d, 0x32 - ADC, 0x3 external, 0x47 - DP internal, 0x48 - DP mini, external, 0x LFH DP 2.
boolean isActive True if display is active.
sint32 ditherState dithering state. Possible values are: -1 - N/A, 0 - Default sint32 ditherBits dithering to specified number of bits per channel. Possit sint32 ditherMode dithering mode. Possible values are: -1 - N/A, 0 - Spatia - Temporal
DisplayMode ref displayModes [] Array of references to the Display modes.

Detailed Description

Represents physical display.

Member Function Documentation

```
string getCurrentTiming ( )
```

Get display's current timing into a human-readable string.

string info ()

Formats basic information about this display into a human-readable string.

```
boolean restoreNativeDisplayMode ()
```

Restore the display mode to native.

boolean saveCSC ([in] string filePath)

Save Color Space Conversion data into specified file (binary format). Using file path with '.csc' extension is recommended.

Parameters:

filePath Path to the file with CSC data. File will be in binary format, specifying '.csc' extension explicitly is recommended.

boolean saveEDID ([in] string filePath)

save EDID into specified file (in text and binary format). File path with extension '.bin' is recommended.

Parameters:

filePath Path to the file with EDID data. File can be a binary or a text file, file path with extension '.bin' is recommended.

boolean saveGammaRamp ([in] string filePath)

Save display Gamma Ramp into specified file (binary format). Using file path with '.gr' extension is recommended.

Parameters:

filePath Path to the file with Gamma Ramp. File will be in binary format, specifying '.gr' extension explicitly is recommended.

boolean setCSC ([in] string filePath)

Override Color Space Conversion data from specified file (binary format). Using file path with '.csc' extension is recommended.

Parameters:

filePath Path to the file with CSC data. Using file path with '.csc' extension is recommended.

boolean setDisplayMode ([in] uint32 width,

[in] uint32 height,
[in] real32 refreshRate,
[in] uint32 depth
)

Change display mode of a display grid, which contains this display, by specifying a display mode properties.

Parameters:

width	Width of the display grid (in pixels)
height	Height of the display grid (in pixels)
refreshRate	Refresh rate of the display grid (in Hz)
depth	Color depth of the display grid (in bits per pixel)

boolean setDisplayModeById ([in] uint32 id)

Change display mode of a display grid, which contains this display, by specifying a display mode ID.

Parameters:

id DisplayMode ID

boolean setDisplayModeByRef ([in] DisplayMode ref mode)

Change display mode of a display grid, which contains this display, by specifying a reference to a display mode.

Parameters:

mode A reference to the display mode to be applied

boolean setDither ([in, ValueMap{"0","1","2"}, Values{"Default","Enable","Disable"] [in, ValueMap{"0","1","2"}, Values{"6 bit","8 bit","10 bit"}] uint32 [in, ValueMap{"0","1","2","3","4"}, Values{"SpatialDynamic","SpatialStatic","SpatialDynamic2x2"," uint32)

Set dithering parameters.

Parameters:

state New dithering state. Requiredbits Dither to specified number of bits per channel. Optionalmode Dithering mode. Optional

boolean setEDID ([in] string filePath)

Override EDID with specified data. File can be text or a binary file. File path with extension '.bin' is recommended.

Parameters:

filePath Path to the file with EDID data. This parameter must be empty string "" for removing the forced EDID.

boolean setGammaRamp ([in] string filePath)

Override display Gamma Ramp data from specified file (binary format). Using file path with '.gr' extension is recommended.

Parameters:

filePath Path to the file with Gamma Ramp. Using file path with '.gr' extension is recommended.

boolean setGammaRampBasic ([in] real32 brightness, [in] real32 contrast, [in] real32 gamma)

Calculate and set Gamma Ramp from basic color controls.

Parameters:

brightness brightness: -0.2 .. +0.2 default: 0.0 *contrast* contrast: -0.2 .. +0.2 default: 0.0

gamma gamma: 0.3 .. 2.8 default: 1.0

boolean setRotation [in, ValueMap{"0","1","2","3"}, Values{"No (rotation","rotate 90 degrees","rotate 180 degrees","rotate 270 degrees"}] uint32

Rotate display.

Parameters:

rotation rotation to be set

```
boolean
setScaling [in, ValueMap{"0","1","2","3","5","6","7"},
( Values{"Default","Closest","Native","ScanoutToNative","Aspecular
uint32
```

Change scaling mode of this display.

Parameters:

scaling scaling mode to be set

Member Data Documentation

sint32 count

Total number of named objects with same name.

sint32 displayConnectorType

Type of physical connector on a display side. Possible values are: -1 -Unknown, 0 - VGA 15-pin, 0x10 - TV Composite, 0x11 - TV S-Video, 0x13 - TV HDTV Component, 0x14 - TV SCART, 0x16 - TV Composite SCART on EIAJ-4120, 0x17 - TV HDTV EIAJ-4120, 0x18 - PcPod HDTV-YPrPb, 0x19 - PcPod S-video, 0x1A - PcPod Composite, 0x20 - DVI-i TV S-Video, 0x21 - DVI-i TV Composite, 0x30 - DVI-i, 0x31 - DVI-d, 0x32 -ADC, 0x38 - Lfh DVI-i 1, 0x39 - Lfh DVI-i 2, 0x40 - SPWG, 0x41 - OEM, 0x46 - DP external, 0x47 - DP internal, 0x48 - DP mini, external, 0x61 - HDMI type A, 0x63 - HDMI type C mini, 0x64 - LFH DP 1, 0x65 - LFH DP 2.

DisplayMode displayModeNative

Native display mode.

DisplayMode ref displayModes[]

Array of references to the **Display** modes.

sint32 ditherBits

dithering to specified number of bits per channel. Possible values are: -1 - N/A, 0 - 6 bit, 1 - 8 bit, 2 - 10 bit

sint32 ditherMode

dithering mode. Possible values are: -1 - N/A, 0 - SpatialDynamic, 1 - SpatialStatic, 2 - SpatialDynamic2x2, 3 - SpatialStatic2x2, 4 - Temporal

sint32 ditherState

dithering state. Possible values are: -1 - N/A, 0 - Default, 1 - Enable, 2 - Disable

uint8 EDID[]

Raw EDID of this display.

uint32 EDIDSize

Size of the EDID in bytes.

sint32 gpuConnectorType

Type of physical connector on a GPU side. Possible values are: -1 - Unknown, 0 - VGA 15-pin, 0x10 - TV Composite, 0x11 - TV S-Video,

0x13 - TV HDTV Component, 0x14 - TV SCART, 0x16 - TV Composite SCART on EIAJ-4120, 0x17 - TV HDTV EIAJ-4120, 0x18 - PcPod HDTV-YPrPb, 0x19 - PcPod S-video, 0x1A - PcPod Composite, 0x20 - DVI-i TV S-Video, 0x21 - DVI-i TV Composite, 0x30 - DVI-i, 0x31 - DVI-d, 0x32 -ADC, 0x38 - Lfh DVI-i 1, 0x39 - Lfh DVI-i 2, 0x40 - SPWG, 0x41 - OEM, 0x46 - DP external, 0x47 - DP internal, 0x48 - DP mini, external, 0x61 -HDMI type A, 0x63 - HDMI type C mini, 0x64 - LFH DP 1, 0x65 - LFH DP 2.

uint32 id

Unique id of named object, used as a key.

boolean isActive

True if display is active.

string locus

locus of the display - <gpu#>.<output#>

string make

Manufacturer of this display.

string model

Model of this display.

string name

Name of an object.

uint32 nvapild

NVAPI **Display** ID.

sint32 ordinal

Ordinal number of named object among objects with same names.

uint32 rotation

rotation angle. Possible values are: 0 - None, 1 - 90 degree, 2 - 180 degree, 3 - 270 degree, 4 - Ignored

uint32 scaling

Display scaling mode - Default, Closest, Native, ScanoutToNative, AspectScanoutToNative, AspectScanoutToClosest, ScanoutToClosest. Possible values are: 0 - Default, 1 - Closest, 2 - Native, 3 -ScanoutToNative, 5 - AspectScanoutToNative, 6 -AspectScanoutToClosest, 7 - ScanoutToClosest.

string uname

Unique name of an object.

Version ver

Version of named object

Version verFirmware

Firmware version of this display.

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

DisplayGrid Class Reference

Represents physical displays, organized into a regular grid (rows by columns). <u>More...</u>

List of all members.

Public Member Functions

- boolean **setOverlapRow** ([in]sint32 index,[in]sint32 overlap) Change per-row overlap of the display grid item with given index.
- boolean **setOverlapCol** ([in]sint32 index,[in]sint32 overlap) Change per-col overlap of the display grid item with given index.
- boolean **setOverlaps** ([in]sint32 index,[in]sint32 overlapRow, [in]sint32 overlapCol)

Change overlap of the display grid item with given

index.

boolean setDisplayMode ([in]uint32 width,[in]uint32 height,
[in]real32 refreshRate,[in]uint32 depth)
Change display mode of this display grid by specifying
a display mode properties.
boolean setDisplayModeById ([in]uint32 id)
Change mode of this display grid by specifying a display mode ID.
boolean setDisplayModeByRef ([in]DisplayMode ref
gridMode)
Change mode of this display grid by specifying a
reference to a display mode.
boolean setRotation ([in]uint32 rotation[])
Change rotation of the displays in this grid by
specifying the per-display rotation mode. Currently only
common mode for all displays in a grid is supported.
boolean saveGammaRamp ([in]string filePath)
Save Gamma Ramp data from all displays into
specified file (binary format). Using file path with '.grg'
extension is recommended.
boolean setGammaRamp ([in]string filePath)
Override Gamma Ramp data for all displays from
specified file (binary format). Using file path with '.grg'
extension is recommended.
boolean setGammaRampBasic ([in]real32 brightness,[in]real32 contrast,[in]real32 gamma)
Calculate and set Gamma Ramp from basic color
controls.
boolean saveCSC ([in]string filePath)
Save Color Space Conversion data into specified file
(binary format). Using file path with '.cscg' extension is
recommended.
boolean setCSC ([in]string filePath)
Override Color Space Conversion data from specified
file (binary format). Using file path with '.cscg' extension
is recommended.

string info ()

Formats basic information about this **Display** grid into a human-readable string.

Public Attributes

Version ver

uint32 id

Unique id of named object, used as a key.

string name

Name of an object.

string uname

Unique name of an object.

sint32 ordinal

Ordinal number of named object among objects with same names.

sint32 count

Total number of named objects with same name.

- uint32 rows
- uint32 cols

Number of columns in the grid.

sint32 overlapRows []

Distance between displays per row. Positive number indicates overlap, negative - gap (in pixels).

sint32 overlapCols []

Distance between displays per column. Positive number indicates overlap, negative - gap (in pixels).

OverlapLimits overlapLimits

Overlap limits (in pixels) for this display grid.

DisplayMode displayModeVirtual

Virtual mode of this display grid (including overlaps).

DisplayMode displayModePhysical

Current mode of this display grid (per-display).

sint32 positionCol

Column-wise (horizontal) position of the display grid (in

pixels).

sint32 positionRow

Row-wise (vertical) position of the display grid (in pixels).

uint32 rotation []

per-display rotation modes in a grid. Possible values are: 0 - No rotation, 1 - rotate 90 degrees, 2 - rotate 180 degrees, 3 - rotate 270 degrees

DisplayMode ref displayModes []

Array of references to modes available on this display grid.

Display ref displays []

Array of references to displays bound into this display grid.

Gpu ref gpus []

Array of references to GPUs, driving displays on this display grid.

Detailed Description

Represents physical displays, organized into a regular grid (rows by columns).

Member Function Documentation

string info ()

Formats basic information about this **Display** grid into a human-readable string.

```
boolean saveCSC ([in] string filePath )
```

Save Color Space Conversion data into specified file (binary format). Using file path with '.cscg' extension is recommended.

Parameters:

filePath Path to the file with CSC data. File will be in binary format, specifying '.cscg' extension explicitly is recommended.

boolean saveGammaRamp ([in] string filePath)

Save Gamma Ramp data from all displays into specified file (binary format). Using file path with '.grg' extension is recommended.

Parameters:

filePath Path to the file with Gamma Ramp. File will be in binary format, specifying '.grg' extension explicitly is recommended.

boolean setCSC ([in] string *filePath*)

Override Color Space Conversion data from specified file (binary format). Using file path with '.cscg' extension is recommended.

Parameters:

filePath Path to the file with CSC data. Using file path with '.cscg' extension is recommended.

```
boolean setDisplayMode ( [in] uint32 width,
[in] uint32 height,
[in] real32 refreshRate,
[in] uint32 depth
)
```

Change display mode of this display grid by specifying a display mode properties.

Parameters:

width	Width of the display grid (in pixels)
height	Height of the display grid (in pixels)
refreshRate	Refresh rate of the display grid (in Hz)

depth Color depth of the display grid (in bits per pixel)

boolean setDisplayModeById ([in] uint32 id)

Change mode of this display grid by specifying a display mode ID.

Parameters:

id DisplayMode ID

boolean setDisplayModeByRef ([in] DisplayMode ref gridMode)

Change mode of this display grid by specifying a reference to a display mode.

Parameters:

gridMode A reference to the display mode to be applied

boolean setGammaRamp ([in] string filePath)

Override Gamma Ramp data for all displays from specified file (binary format). Using file path with '.grg' extension is recommended.

Parameters:

filePath Path to the file with Gamma Ramp. Using file path with '.grg' extension is recommended.

```
boolean setGammaRampBasic ( [in] real32 brightness,
[in] real32 contrast,
[in] real32 gamma
)
```

Calculate and set Gamma Ramp from basic color controls.

Parameters:

brightness	brightness: -0.2 +0.2 default: 0.0
contrast	contrast: -0.2 +0.2 default: 0.0
gamma	gamma: 0.3 2.8 default: 1.0

boolean setOverlapCol ([in] sint32 *index*, [in] sint32 *overlap*)

Change per-col overlap of the display grid item with given index.

Parameters:

- *index* Index of the column. Value -1 could be used to change overlap for all columns at once
- overlap Overlap. Positive number indicates overlap, negative gap (in pixels)

boolean setOverlapRow ([in] sint32 *index*, [in] sint32 *overlap*)

Change per-row overlap of the display grid item with given index.

Parameters:

- *index* Index of the column. Value -1 could be used to change overlap for all rows at once
- overlap Overlap. Positive number indicates overlap, negative gap (in pixels)

boolean setOverlaps ([in] sint32 index,

[in] sint32 overlapRow, [in] sint32 overlapCol
)

Change overlap of the display grid item with given index.

Parameters:

index Index of the item. Value -1 could be used to change overlap for all items at once *overlapRow* Overlap. Positive number indicates overlap, negative gap (in pixels) *overlapCol* Overlap. Positive number indicates overlap, negative - gap (in pixels)

boolean setRotation ([in] uint32 rotation[])

Change rotation of the displays in this grid by specifying the per-display rotation mode. Currently only common mode for all displays in a grid is supported.

Parameters:

rotation per-display rotation modes to be set in the grid. Currently only identical rotation angle for all displays in a grid is supported.
Example 1: for 1x4 grid, "setRotation 1" will apply "rotate 90 degrees" to all displays in the grid.
Example 2: specifying same angle for all displays is supported: "setRotation rotation=(1,1,1,1)"
Example 3: attempt to rotate to non-identical angles "setRotation rotation=(1,2,1,1)" will fail

Member Data Documentation

uint32 cols

Number of columns in the grid.

sint32 count

Total number of named objects with same name.

DisplayMode displayModePhysical

Current mode of this display grid (per-display).

DisplayMode ref displayModes[]

Array of references to modes available on this display grid.

DisplayMode displayModeVirtual

Virtual mode of this display grid (including overlaps).

Display ref displays[]

Array of references to displays bound into this display grid.

Gpu ref gpus[]

Array of references to GPUs, driving displays on this display grid.

uint32 id

Unique id of named object, used as a key.

string name

Name of an object.

sint32 ordinal

Ordinal number of named object among objects with same names.

sint32 overlapCols[]

Distance between displays per column. Positive number indicates overlap, negative - gap (in pixels).

OverlapLimits overlapLimits

Overlap limits (in pixels) for this display grid.

sint32 overlapRows[]

Distance between displays per row. Positive number indicates overlap,

negative - gap (in pixels).

sint32 positionCol

Column-wise (horizontal) position of the display grid (in pixels).

sint32 positionRow

Row-wise (vertical) position of the display grid (in pixels).

uint32 rotation[]

per-display rotation modes in a grid. Possible values are: 0 - No rotation, 1 - rotate 90 degrees, 2 - rotate 180 degrees, 3 - rotate 270 degrees

uint32 rows

Number of rows in the grid

string uname

Unique name of an object.

Version ver

Version of named object

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

DisplayGridInfo Class Reference

Information about a display grid. More...

List of all members.

Public Member Functions

string info ()

Formats basic information about the display grid information into a human-readable string.

Public Attributes

Version verClass

uint32 id

Unique id of a display grid information, used as a key.

uint32 rows Number of rows in the display grid. uint32 cols Number of columns in the display grid. uint32 primaryId ID of the primary display. uint32 displayIds [] Array of display IDs (Display.id). string **unames** string of unique display names (**Display.uname**), separated by semicolon. Order corresponds to displayIds property string cscFilePath string, containing a file path to CSC data string gammaRampFilePath path to the file which contains desktop gamma ramps for every display in the grid

Detailed Description

Information about a display grid.

Member Function Documentation

string info ()

Formats basic information about the display grid information into a human-readable string.

Member Data Documentation

uint32 cols

Number of columns in the display grid.

string cscFilePath

string, containing a file path to CSC data

uint32 displayIds[]

Array of display IDs (Display.id).

string gammaRampFilePath

path to the file which contains desktop gamma ramps for every display in the grid

uint32 id

Unique id of a display grid information, used as a key.

uint32 primaryId

ID of the primary display.

uint32 rows

Number of rows in the display grid.

string **unames**

string of unique display names (**Display.uname**), separated by semicolon. Order corresponds to displayIds property

Version verClass

Version of a display grid information

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>

DisplayManager Class Reference

Management of display-related task. More...

List of all members.

Public Member Functions

- boolean setScalingAll ([in, ValueMap{"0","1","2","3","5","6","7"},
 - Values{"Default","Closest","Native","ScanoutToNative","AspectSc uint32 scaling)

Change scaling mode of all display targets in a system.

- boolean **validateDisplayGridById** ([in]uint32 cols,[in]uint32 rows,[in]uint3 Validate the display grid by specifying display IDs, before creating validateDisplayGrids method.
- boolean **createDisplayGridById** ([in]uint32 cols,[in]uint32 rows,[in]uint32 Create a display grid by specifying display IDs. Creates only a sir Will disable all previous grids.

- boolean **createDisplayGridByName** ([in]uint32 cols,[in]uint32 rows,[in]str Create a display grid by specifying unique display names. Create method. Will disable all previous grids.
- boolean **createDisplayGridByRef** ([in]uint32 cols,[in]uint32 rows,[in]strinç Create a display grid by specifying references to the **Display** clas createDisplayGrids method. Will disable all previous grids.
- boolean **validateDisplayGrids** ([in]string grids[]) Validate multiple display grid topologies.
- boolean **createDisplayGrids** ([in]string grids[]) Create multiple display grid topologies.
- boolean **setGridPositions** ([in]uint32 positionCol[],[in]uint32 positionRow Set desktop positions of all active display grids. All active display taskbar) must be positioned at (0,0).
- boolean **fakeEDIDAII** ([in]string filePath,[in, ValueMap{"-1","0","1","2","3"," Values{"unknown","unintialized","VGA","Component","S-Video","I Fake given EDID across all GPUs, on all or only specified display boolean **tryCustomTiming** ([in]string timing)

Try the new custom timing without applying it.

boolean createCustomTiming ([in]string timing)

Create the new Custom timing. This will only save the new timing boolean **editCustomTiming** ([in]uint32 modeld,[in]string newTiming)

- Modify the existing Custom timing. This will only save the new tin boolean **deleteCustomTiming** ([in]string timing)
 - Delete the existing Custom timing. This will permanently delete the string enumCustomTimings ([in]uint32 displays[])
 - Enumerates all existing Custom timing for the given displays.
- boolean **saveCustomTimings** ([in]uint32 displays[],[in]string filePath)

Saves all existing Custom Timings for the given displays to given boolean **loadCustomTimings** ([in]uint32 displays[],[in]string filePath)

- Loads all existing Custom Timings into the given displays from giboolean createClone ([in, ValueMap{"0","1"}, Values{"basic","smart"}] uint Clone displays. Note that all displays must be connected to the s
- boolean **createModeFilter** ([in]uint32 displays[],[in]string filter)

Create display mode filter for given displays.

boolean deleteModeFilter ([in]uint32 displays[])

Create display mode filter for given displays.

string enumModeFilters ([in]uint32 displays[])

Enumerate display mode filters.

string info ()

Formats basic information about DisplayManager objects in a sy

Public Attributes

Version verClass

Detailed Description

Management of display-related task.

Member Function Documentation

boolean createClone	[in, ValueMap{"0","1"}, (Values{"basic","smart"}] uint32	type,
	[in] uint32	source,
	[in] uint32	targets[]
)	

Clone displays. Note that all displays must be connected to the same GPU.

Parameters:

type of clone - basic (same display resolution) or smart (pan and scan target to a source display)
 source display ID of clone source (Display.id)
 targets array of clone target display IDs (Display.id)

boolean createCustomTiming ([in] string timing)

Create the new Custom timing. This will only save the new timing and not

apply. To apply, call setDisplayMode.

Parameters:

timing String of custom timing parameters "displayLocus=1001 1002 2001 2002...; mode=1720 1000 32 60.45; override=6 possible override values: CURRENT=0, AUTO=1, EDID=2, DMT=3, DMT_RB=4, CVT=5, CVT_RB=6, GFT=7, EIA861=8, ANALOG_TV=9, CUST=10, NV_PSF=11, NV_ASPR=12, SDI=13. Note - not all override values may be supported. Default is CVT_RB Other parameters are optional and will be deduced from default

```
boolean createDisplayGridById ( [in] uint32 cols,
[in] uint32 rows,
[in] uint32 displayIds[]
)
```

Create a display grid by specifying display IDs. Creates only a single grid, for multigrid, please use createDisplayGrids method. Will disable all previous grids.

Parameters:

cols	number of columns in a grid (must be non-zero)
rows	number of rows in a grid (must be non-zero)
displayIds	array of display IDs (Display.id). Order is important. C
	convention for array - grid element at column=i and row=j
	must be at index=j*cols+i

```
boolean createDisplayGridByName ( [in] uint32 cols,
[in] uint32 rows,
[in] string displayNames
)
```

Create a display grid by specifying unique display names. Creates only a single grid, for multigrid, please use createDisplayGrids method. Will disable all previous grids.

Parameters:

colsnumber of columns in a grid (must be non-zero)rowsnumber of rows in a grid (must be non-zero)displayNamesstring of unique display names (Display.uname),separated by semicolon. Order is important. Cconvention for array - grid element at column=i androw=j must be at index=j*cols+i

```
boolean createDisplayGridByRef ( [in] uint32 cols,
[in] uint32 rows,
[in] string displayRefs
)
```

Create a display grid by specifying references to the **Display** class instances. Creates only a single grid, for multigrid, please use createDisplayGrids method. Will disable all previous grids.

Parameters:

cols	number of columns in a grid (must be non-zero)
rows	number of rows in a grid (must be non-zero)
displayRefs	string of references to Display instances, separated by semicolon. Order is important. C convention for array - grid element at column=i and row=j must be at index=j*cols+i

boolean createDisplayGrids ([in] string grids[])

Create multiple display grid topologies.

Parameters:

grids array of grid topologies to be created, each grid specified by a string with key=value pairs "rows=2;cols=2;stereo=0;layout=1.1 1.2 1.3 1.4;mode=1920 1200 32 60;rotation=0 0 0 0", "rows=1;cols=4;stereo=0;layout=2.1 2.2 2.3 2.4;mode=1920 1200 32 60;rotation=1 1 1 1", ... Only "rows" and "cols" are the mandatory keys in a string with grid specification Other parameters are optional and will be deduced from a default topology

Create display mode filter for given displays.

Parameters:

displays Array of display IDs (Display.id)

filter filter definition in format "<type>;<limits>;". Type could be <, > or =. Limits are given as <width>x<height>x<color depth><frequency>. Zero value means no filtering will occur for given limit. Example 1: ">;999x888x31@0" - all display modes with resolution more than 999x998 and color depth 32 or more, at any refresh rate will pass. Example 2: "=;1024x768x0@59.95" - only display mode 1024x768 with any color depth at refresh rate 59.95 Hz will pass.

boolean deleteCustomTiming ([in] string timing)

Delete the existing Custom timing. This will permanently delete the timing of the display.

Parameters:

timing String of custom timing parameters "displayLocus=1001 1002 2001 2002...; mode=1720 1000 32 60.45; override=CVT_RB possible override values: CURRENT=0, AUTO=1, EDID=2, DMT=3, DMT_RB=4, CVT=5, CVT_RB=6, GFT=7, EIA861=8, ANALOG_TV=9, CUST=10, NV_PSF=11, NV_ASPR=12, SDI=13. Note - not all override values may be supported. Default is CVT_RB Other parameters are optional and will be deduced from default boolean deleteModeFilter ([in] uint32 displays[])

Create display mode filter for given displays.

Parameters:

```
displays Array of display IDs (Display.id)
```

```
boolean editCustomTiming ( [in] uint32 modeld,
[in] string newTiming
)
```

Modify the existing Custom timing. This will only save the new timing and not apply. To apply, call setDisplayMode.

Parameters:

modeld	Existing custom timing source mode Id
newTiming	String of new custom timing parameters
	"displayLocus=1001 1002 2001 2002; mode=1720
	1000 32 60.45; override=6 possible override values:
	CURRENT=0, AUTO=1, EDID=2, DMT=3, DMT_RB=4,
	CVT=5, CVT_RB=6, GFT=7, EIA861=8,
	ANALOG_TV=9, CUST=10, NV_PSF=11,
	NV_ASPR=12, SDI=13. Note - not all override values
	may be supported. Default is CVT_RB Other
	parameters are optional and will be deduced from
	default

string enumCustomTimings ([in] uint32 displays[])

Enumerates all existing Custom timing for the given displays.

Parameters:

displays Array of display IDs (Display.id)

string enumModeFilters ([in] uint32 displays[])

Enumerate display mode filters.

Parameters:

displays Array of display IDs (Display.id)

```
boolean
fakeEDIDAll ( [in] string
```

file

[in, ValueMap{"-1","0","1","2","3","4","5","6","7","8","9"}, Values{"unknown","unintialized","VGA","Component","S-Video","HDMI","DVI","LVDS","DP","Composite","All"}] uint32

)

Fake given EDID across all GPUs, on all or only specified display outputs.

Parameters:

- *filePath* Full path to the file with the custom EDID. In order to remove the forced EDID, specify an empty string
- *output* **Display** output type for faking EDID. In order to force/remove the EDID on all display outputs, specify the output value "All"

string info ()

Formats basic information about **DisplayManager** objects in a system into a human-readable string.

```
boolean loadCustomTimings ( [in] uint32 displays[],
[in] string filePath
)
```

Loads all existing Custom Timings into the given displays from given XML file. File format is compatible with Control Panel.

Parameters:

displays Array of display IDs (**Display.id**) *filePath* Path to the file with Custom Timings boolean saveCustomTimings ([in] uint32 *displays*[], [in] string *filePath*)

Saves all existing Custom Timings for the given displays to given XML file. File format is compatible with Control Panel.

Parameters:

```
displays Array of display IDs (Display.id)
filePath Path to the file with Custom Timings
```

Set desktop positions of all active display grids. All active display grids have to be re-positioned together. Primary grid (with taskbar) must be positioned at (0,0).

Parameters:

positionCol	array of valid grid column positions in pixels. Order is
	important. Also, DisplayGrid::colOverlap (if any) should
	be considered.

positionRow array of valid grid row positions in pixels. Order is important. Also, DisplayGrid::rowOverlap (if any) should be considered.

boolean [in, ValueMap{"0","1","2","3","5","6","7"}, setScalingAll (Values{"Default","Closest","Native","ScanoutToNative","Asp uint32

Change scaling mode of all display targets in a system.

Parameters:

scaling Scaling mode to be set

boolean tryCustomTiming ([in] string timing)

Try the new custom timing without applying it.

Parameters:

timing String of custom timing parameters "displayLocus=1001 1002 2001 2002...; mode=1720 1000 32 60.45; override=6 possible override values: CURRENT=0, AUTO=1, EDID=2, DMT=3, DMT_RB=4, CVT=5, CVT_RB=6, GFT=7, EIA861=8, ANALOG_TV=9, CUST=10, NV_PSF=11, NV_ASPR=12, SDI=13. Note - not all override values may be supported. Default is CVT_RB Other parameters are optional and will be deduced from default

boolean validateDisplayGridById ([in] uint32 *cols*, [in] uint32 *rows*, [in] uint32 *displayIds*[])

Validate the display grid by specifying display IDs, before creating the grid. Validates only a single grid, for multigrid, please use validateDisplayGrids method.

Parameters:

colsnumber of columns in a grid (must be non-zero)rowsnumber of rows in a grid (must be non-zero)displayIdsarray of display IDs (**Display.id**). Order is important. C
convention for array - grid element at column=i and row=j
must be at index=j*cols+i

boolean validateDisplayGrids ([in] string grids[])

Validate multiple display grid topologies.

Parameters:

grids array of grid topologies to be validated, each grid specified by a string with key=value pairs

"rows=2;cols=2;stereo=0;layout=1.1 1.2 1.3 1.4;mode=1920 1200 32 60;rotation=0 0 0 0", "rows=1;cols=4;stereo=0;layout=2.1 2.2 2.3 2.4;mode=1920 1200 32 60;rotation=1 1 1 1", ... Only "rows" and "cols" are the mandatory keys in the string with grid specification Other parameters are optional and will be deduced from a default topology

Member Data Documentation

Version verClass

Object version

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

DisplayMode Class Reference

Represents display mode. More...

List of all members.

Public Member Functions

string info ()

Formats basic information about this display mode into a human-readable string.

Public Attributes

Version verClass

uint32 **id** Unique identification of the display mode. sint32 width Display mode width. sint32 height Display mode height. real32 refreshRate Display mode refresh rate. sint32 colorDepth Display mode color depth in bits per pixel (bpp).

Detailed Description

Represents display mode.

Member Function Documentation

string info ()

Formats basic information about this display mode into a humanreadable string.

Member Data Documentation

sint32 colorDepth

Display mode color depth in bits per pixel (bpp).

sint32 height

Display mode height.

uint32 id

Unique identification of the display mode.

real32 refreshRate

Display mode refresh rate.

Version verClass

Object version

sint32 width

Display mode width.

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

DisplayProfile Class Reference

Display profile. More...

List of all members.

Public Member Functions

boolean setValueById ([in]uint32 settingId,[in]uint32 value) Set the 32-bit value by a setting ID.
boolean setBinaryValueById ([in]uint32 settingId,[in]uint8 value[]) Set the binary value by a setting ID.
boolean setStringValueById ([in]uint32 settingId,[in]string value) Set the string value by a setting ID.
boolean restoreSettings ([in]uint32 settingIds[]) Restore or delete settings, specified by IDs. Predefined settings get restored and non-predefined settings get deleted.

string info ()

formats basic information about a profile into a humanreadable string

Public Attributes

Version verClass uint32 id Unique id of a profile, used as a key. string name Name of a profile. boolean isPredefined If true, profile is predefined and cannot be removed. boolean isSupported If true, profile is supported on this system. Setting settings [] Array of settings for a profile. uint32 type DisplayGridInfo grid information about a display grid

Detailed Description

Display profile.

Member Function Documentation

string info ()

formats basic information about a profile into a human-readable string

boolean restoreSettings ([in] uint32 settingIds[])

Restore or delete settings, specified by IDs. Predefined settings get restored and non-predefined settings get deleted.

Parameters:

```
settingIds Array of Setting IDs
```

```
boolean setBinaryValueById ( [in] uint32 settingId,
[in] uint8 value[]
)
```

Set the binary value by a setting ID.

Parameters:

```
settingId Setting ID
value Binary value as an array of bytes
```

boolean setStringValueById ([in] uint32 *settingId*, [in] string *value*)

Set the string value by a setting ID.

Parameters:

settingId Setting ID value String value

```
boolean setValueById ( [in] uint32 settingId,
[in] uint32 value
)
```

Set the 32-bit value by a setting ID.

Parameters:

settingId Setting ID

Member Data Documentation

DisplayGridInfo grid

information about a display grid

uint32 id

Unique id of a profile, used as a key.

boolean isPredefined

If true, profile is predefined and cannot be removed.

boolean isSupported

If true, profile is supported on this system.

string name

Name of a profile.

Setting settings[]

Array of settings for a profile.

uint32 type

Display Profile type. Possible values are: 4 - Display

Version verClass

Version of a profile class

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

• nvwmi.mof

NVIDIA

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

Ecc Class Reference

Represents the GPU Error Correction Code. This feature is not supported if there are multiple GPU topologies enabled. <u>More...</u>

List of all members.

Public Member Functions

boolean resetCounters ([in]boolean bResetCurrent,[in]boolean bResetAggregate) Resets memory error counters. This method will fail if option=0 (notSupported).
boolean setConfiguration ([in]boolean bEnable,[in]boolean bEnableImmediately) Updates the ECC memory configuration settings. This method will fail if option=0 (notSupported).
string info () Formats basic information about ECC objects into a human-readable string.

Public Attributes

Version verClass

uint32	id
	Unique identification of the NVIDIA GPU.
boolean	isSupported
	Is ECC memory feature supported?
boolean	isEnabled
	Is ECC memory setting enabled?
boolean	isWritable
	Is current ECC configuration stored in non-volatile memory?
boolean	isEnabledByDefault
	Is factory default ECC configuration enabled?
uint32	option
	Possible ECC memory configuration options. Possible values
	are: 0 - notSupported, 1 - deferred, 2 - immediate.
uint64	currentSingleBitErrors
	Number of single bit ECC errors detected since last boot.
uint64	currentDoubleBitErrors
	Number of double bit ECC errors detected since last boot.
uint64	aggregateSingleBitErrors
	Number of single bit ECC errors detected since last counter
	reset.
uint64	aggregateDoubleBitErrors
	Number of double bit ECC errors detected since last counter
	reset.

Detailed Description

Represents the GPU Error Correction Code. This feature is not supported if there are multiple GPU topologies enabled.

Member Function Documentation

```
string info ( )
```

Formats basic information about ECC objects into a human-readable string.

boolean resetCounters ([in] boolean *bResetCurrent*, [in] boolean *bResetAggregate*)

Resets memory error counters. This method will fail if option=0 (notSupported).

Parameters:

bResetCurrent Reset the current ECC error counters *bResetAggregate* Reset the aggregate ECC error counters

boolean setConfiguration ([in] boolean *bEnable*,

)

[in] boolean *bEnableImmediately*

Updates the ECC memory configuration settings. This method will fail if option=0 (notSupported).

Parameters:

bEnable Enable new ECC configuration setting *bEnableImmediately* New configuration setting should take effect immediately. This flag is valid only if option=2 (immediate).

Member Data Documentation

uint64 aggregateDoubleBitErrors

Number of double bit ECC errors detected since last counter reset.

uint64 aggregateSingleBitErrors

Number of single bit ECC errors detected since last counter reset.

uint64 currentDoubleBitErrors

Number of double bit ECC errors detected since last boot.

uint64 currentSingleBitErrors

Number of single bit ECC errors detected since last boot.

uint32 id

Unique identification of the NVIDIA GPU.

boolean isEnabled

Is ECC memory setting enabled?

boolean isEnabledByDefault

Is factory default ECC configuration enabled?

boolean isSupported

Is ECC memory feature supported?

boolean isWritable

Is current ECC configuration stored in non-volatile memory?

uint32 option

Possible ECC memory configuration options. Possible values are: 0 - notSupported, 1 - deferred, 2 - immediate.

Version verClass

Ecc class version

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>

EncoderSessions Class Reference

Represents encoder sessions info. More...

List of all members.

Public Member Functions

string info ()

Formats basic information about per process utilization into a human-readable string.

Public Attributes

Version verClass

uint32 id Unique ID. uint32 sessionsCount Total no of sessions. uint32 sessionId [] Session Id. uint32 processId [] Owning process ID. uint32 vgpulnstance [] Owning vGPU instance ID (only valid on vGPU hosts, otherwise zero). uint32 codecType [] Video encoder type.. Possible values are: 0 - Unknown, 1 -H264, 2 - HEVC. uint32 hResolution [] Current encode horizontal resolution. uint32 vResolution [] Current encode vertical resolution. uint32 averageEncodeFps [] Moving average encode frames per second. uint32 averageEncodeLatency [] Moving average encode latency in milliseconds.

Detailed Description

Represents encoder sessions info.

Member Function Documentation

string info ()

Formats basic information about per process utilization into a humanreadable string.

Member Data Documentation

uint32 averageEncodeFps[]

Moving average encode frames per second.

uint32 averageEncodeLatency[]

Moving average encode latency in milliseconds.

uint32 codecType[]

Video encoder type.. Possible values are: 0 - Unknown, 1 - H264, 2 - HEVC.

uint32 hResolution[]

Current encode horizontal resolution.

uint32 id

Unique ID.

uint32 processId[]

Owning process ID.

uint32 sessionId[]

Session Id.

uint32 sessionsCount

Total no of sessions.

Version verClass

EncoderSessions class version.

uint32 vgpulnstance[]

Owning vGPU instance ID (only valid on vGPU hosts, otherwise zero).

uint32 vResolution[]

Current encode vertical resolution.

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

Gpu Class Reference

Represents an NVIDIA GPU. More...

List of all members.

Public Member Functions

boolean enableLicensedFeature ([in]string feature) Enable licensed feature with given name.
boolean disableLicensedFeature ([in]string feature) Disable licensed feature with given name.
boolean fakeEDID ([in]string filePath,[in, ValueMap{"-1","0","1","2","3","4","5","6","7","8","9"} Values{"unknown","unintialized","VGA","Compone Video","HDMI","DVI","LVDS","DP","Composite","A uint32 output) Fake given EDID on all display outputs or on spec display outputs, of the given GPU.

boolean **fakeEDIDOnPort** ([in]string filePath,[in]uint32 port [in, ValueMap{"-1","0","1","2","3","4","5","6","7","8"} Values{"unknown","unintialized","VGA","Compone Video","HDMI","DVI","LVDS","DP","Composite"}] u output)

Fake given EDID on specified display output, of th given GPU port.

boolean **createUtilizationEvent** ([in]sint32 lower,[in]sint32 upper,[in, ValueMap{"-1","0","1","2","3"},

Values{"All","GPU","Frame Buffer","Video Engine", bus"}] sint32 domain)

Create utilization event with given bounds in given performance domain.

boolean deleteUtilizationEvents ([in,

ValueMap{"-1","0","1","2","3"},

Values{"All","GPU","Frame Buffer","Video Engine", bus"}] sint32 domain)

Delete all utilization events in given performance domain.

string getAllUtilizationEvents ([in,

ValueMap{"-1","0","1","2","3"},

Values{"All","GPU","Frame Buffer","Video Engine", bus"}] sint32 domain)

Get all utilization events in given performance dor string info ()

Formats basic information about this GPU into a human-readable string.

Public Attributes

Version ver

uint32 **id** Unique id of named object, used as a key. string **name** Name of an object.

string uname

Unique name of an object.

sint32 ordinal

Ordinal number of named object among objects wis same names.

sint32 count

Total number of named objects with same name.

uint32 handle

uint32 nvapild

NVAPI ID of the GPU.

uint32 productType

GPU product type. Possible values are: 0 - unknov GeForce, 2 - Quadro, 3 - NVS, 4 - Tesla.

uint32 memoryType

The type of memory associated with this GPU. Povalues are: 0 - unknown, 1 - SDRAM, 2 - DDR1, 3 DDR2, 4 - GDDR2, 5 - GDDR3, 6 - GDDR4, 7 - DI 8 - GDDR5, 9 - LPDDR2, 10 - GDDR5X, 11 - HBN - HBM2.

uint32 memorySizeVirtual

The virtual memory size in MB (Physical Video Me + **System** Video Memory + Shared **System** Memo

uint32 memorySizePhysical

The physical video memory size in MB.

uint32 memorySizeAvailable

The currently available physical video memory size MB.

uint32 memoryBusWidth

The width of the GPU's RAM memory bus in MB.

uint32 coreCount

The total number of Cores defined for a GPU.

sint32 memoryClockCurrent

The current memory clock frequency in MHz, 0 if r available, -1 if not supported in current environmer

sint32 gpuCoreClockCurrent

The current GPU clock frequency in MHz, 0 if not

available, -1 if not supported in current environmer PcieLink pcieGpu

Parameters of PCI-Express link to this GPU.

sint32 pcieDownstreamWidth

Width of the PCI-Express downstream link, in cour PCI-Express lanes.

VideoCodec videoCodec

Parameters of Video Engine encoder and decoder this GPU.

real32 power

The total GPU power drawn over last iteration of sampling in watts. -1 if not supported in current environment.

sint32 powerSampleCount

The number of power sensor samples per iteratior not supported in current environment.

sint32 powerSamplingPeriod

The period (in milliseconds) between the power samples. -1 if not supported in current environmer

sint32 percentGpuUsage

The percentage of GPU utilization. -1 if not suppor current environment.

sint32 percentGpuMemoryUsage

The percentage of GPU memory utilization. -1 if no supported in current environment.

ThermalProbe ref thermalProbes []

Array of thermal probes of this GPU.

Cooler ref coolers []

Array of references to coolers of this GPU.

Ecc ref ecc []

Array of references to Error Correction Code insta on this GPU.

string productName

The name of the graphics card to which this GPU belongs.

Version verVBIOS

The video BIOS version associated with this GPU.

string deviceInfe	
string deviceInfo The display adapter string comprising of Vendor Ic	4
	1
Device Id, Sub-device Id, Revision Id.	
string archName	
The name of the GPU architecture -	
'Curie', 'Tesla', 'Fermi', 'Tegra', 'Kepler', 'Maxwell', 'Volt)
sint32 archId	
numerical ID of the GPU architecture. Possible val	l
are: -1 - unknown, 0 - Legacy, 1 - Curie, 2 - Tesla,	
Fermi, 4 - Tegra, 5 - Kepler, 6 - Maxwell, 7 - Pasca	Э
Volta	
string licensableFeatures []	
Licensable features.	
sint32 licensableStatus []	
Status of licensable features. Possible values are:	
N/A, 0 - Disabled, 1 - Enabled.	
EncoderSessions ref encoderSessionsInfo []	
Array of references to encoder sessions info.	
ProcessUtilization ref processUtilization []	
Array of references to per process utilization info f	(
GPU.	

Detailed Description

Represents an NVIDIA GPU.

Member Function Documentation

boolean createUtilizationEvent	([in] sint32	lower,
	[in] sint32	upper,
	[in, ValueMap{"-1","0","1","2","3"},	
	Values{"All","GPU","Frame Buffer","Video Engine","PCIe bus"}]	domain

```
sint32
)
```

Create utilization event with given bounds in given performance domain.

Parameters:

- *lower* Lower bound. When utilization value drops below this value, event notification will be posted to the OS event log
- *upper* Upper bound. When utilization value grows above this value, event notification will be posted to the OS event log
- *domain* Performance domain of the GPU. "All" means that given bounds will trigger notification events for all domains

	[in, ValueMap{"-1","0","1","2","3"},	
boolean	Values{"All","GPU","Frame Buffer","Video Engine","PCIe bus"}]	domain)
deleteUtilizationEvents	⁽ Buffer","Video Engine","PCIe bus"}]	domain)
	sint32	

Delete all utilization events in given performance domain.

Parameters:

domain Performance domain of the GPU. "All" means that all event triggers for all domains will be removed

boolean disableLicensedFeature ([in] string feature)

Disable licensed feature with given name.

Parameters:

feature Licensed feature to disable

boolean enableLicensedFeature ([in] string feature)

Enable licensed feature with given name.

Parameters:

feature Licensable feature to enable

```
fakeEDID ([in] string
boolean
                                                                           filePa
            [in, ValueMap{"-1", "0", "1", "2", "3", "4", "5", "6", "7", "8", "9"},
            Values{"unknown","unintialized","VGA","Component","S-
                                                                           outpl
            Video", "HDMI", "DVI", "LVDS", "DP", "Composite", "All"}]
            uint32
           )
```

Fake given EDID on all display outputs or on specified display outputs, of the given GPU.

Parameters:

- *filePath* Full path to the file with the custom EDID. In order to remove the forced EDID, specify an empty string
- **Display** output type for faking EDID. In order to output force/remove the EDID on all display outputs, specify the output value "All"

boolean

fakeEDIDOnPort ^{([in]} string

)

```
[in] uint32
[in, ValueMap{"-1","0","1","2","3","4","5","6","7","8"},
Values{"unknown","unintialized","VGA","Component","
Video","HDMI","DVI","LVDS","DP","Composite"}] uint32
```

Fake given EDID on specified display output, of the given GPU port.

Parameters:

filePath	Full path to the file with the custom EDID. In order to
	remove the forced EDID, specify an empty string
portIndex	Physical location of the display port on the GPU.
output	Display output type for faking EDID.

	[in, ValueMap{"-1","0","1","2","3"},	
string	(Values{"All","GPU","Frame	domain)

getAllUtilizationEvents Buffer","Video Engine","PCIe bus"}] sint32

Get all utilization events in given performance domain.

Parameters:

domain Performance domain of the GPU. "All" means that all event triggers for all domains will be listed

string info ()

Formats basic information about this GPU into a human-readable string.

Member Data Documentation

sint32 archId

numerical ID of the GPU architecture. Possible values are: -1 - unknown, 0 - Legacy, 1 - Curie, 2 - Tesla, 3 - Fermi, 4 - Tegra, 5 - Kepler, 6 -Maxwell, 7 - Pascal, 8 - Volta

string archName

The name of the GPU architecture -'Curie','Tesla','Fermi','Tegra','Kepler','Maxwell','Volta' etc.

Cooler ref coolers[]

Array of references to coolers of this GPU.

uint32 coreCount

The total number of Cores defined for a GPU.

sint32 count

Total number of named objects with same name.

string deviceInfo

The display adapter string comprising of Vendor Id, Device Id, Subdevice Id, Revision Id.

Ecc ref ecc[]

Array of references to Error Correction Code instances on this GPU.

EncoderSessions ref encoderSessionsInfo[]

Array of references to encoder sessions info.

sint32 gpuCoreClockCurrent

The current GPU clock frequency in MHz, 0 if not available, -1 if not supported in current environment.

uint32 handle

same as NVAPI ID of the GPU. Unique ID - deprecated, please use 'id'

uint32 id

Unique id of named object, used as a key.

string licensableFeatures[]

Licensable features.

sint32 licensableStatus[]

Status of licensable features. Possible values are: -1 - N/A, 0 - Disabled, 1 - Enabled.

uint32 memoryBusWidth

The width of the GPU's RAM memory bus in MB.

sint32 memoryClockCurrent

The current memory clock frequency in MHz, 0 if not available, -1 if not supported in current environment.

uint32 memorySizeAvailable

The currently available physical video memory size in MB.

uint32 memorySizePhysical

The physical video memory size in MB.

uint32 memorySizeVirtual

The virtual memory size in MB (Physical Video Memory + **System** Video Memory + Shared **System** Memory).

uint32 memoryType

The type of memory associated with this GPU. Possible values are: 0 - unknown, 1 - SDRAM, 2 - DDR1, 3 - DDR2, 4 - GDDR2, 5 - GDDR3, 6 - GDDR4, 7 - DDR3, 8 - GDDR5, 9 - LPDDR2, 10 - GDDR5X, 11 - HBM1, 12 - HBM2.

string name

Name of an object.

uint32 nvapild

NVAPI ID of the GPU.

sint32 ordinal

Ordinal number of named object among objects with same names.

sint32 pcieDownstreamWidth

Width of the PCI-Express downstream link, in count of PCI-Express lanes.

PcieLink pcieGpu

Parameters of PCI-Express link to this GPU.

sint32 percentGpuMemoryUsage

The percentage of GPU memory utilization. -1 if not supported in current environment.

sint32 percentGpuUsage

The percentage of GPU utilization. -1 if not supported in current environment.

real32 power

The total GPU power drawn over last iteration of sampling in watts. -1 if not supported in current environment.

sint32 powerSampleCount

The number of power sensor samples per iteration. -1 if not supported in current environment.

sint32 powerSamplingPeriod

The period (in milliseconds) between the power samples. -1 if not supported in current environment.

ProcessUtilization ref processUtilization[]

Array of references to per process utilization info for this GPU.

string productName

The name of the graphics card to which this GPU belongs.

uint32 productType

GPU product type. Possible values are: 0 - unknown, 1 - GeForce, 2 - Quadro, 3 - NVS, 4 - Tesla.

ThermalProbe ref thermalProbes[]

Array of thermal probes of this GPU.

string uname

Unique name of an object.

Version ver

Version of named object

Version verVBIOS

The video BIOS version associated with this GPU.

VideoCodec videoCodec

Parameters of Video Engine encoder and decoder in this GPU.

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

• nvwmi.mof

NVIDIA <u>Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.</u>





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>

NamedObject Class Reference

Base class for all named NV objects. More...

List of all members.

Public Attributes

Version ver uint32 id Unique id of named object, used as a key. string name Name of an object. string uname Unique name of an object. sint32 ordinal Ordinal number of named object among objects with same names. sint32 **count** Total number of named objects with same name.

Detailed Description

Base class for all named NV objects.

Member Data Documentation

sint32 count

Total number of named objects with same name.

uint32 id

Unique id of named object, used as a key.

string name

Name of an object.

sint32 ordinal

Ordinal number of named object among objects with same names.

string uname

Unique name of an object.

Version ver

Version of named object

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

• nvwmi.mof

NVIDIA

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>

OverlapLimits Class Reference

Represents overlap limits of individual display in a display grid. More...

List of all members.

Public Member Functions

string info ()

Formats basic information about **OverlapLimits** instance into a human-readable string.

Public Attributes

Version verClass

uint32 **id** Unique identification of the overlap limits. sint32 minOverlapCol The minimum column overlap for each display (in pixels). sint32 maxOverlapCol The maximum column overlap for each display (in pixels). sint32 minOverlapRow The minimum row overlap for each display (in pixels). sint32 maxOverlapRow The maximum row overlap for each display (in pixels). sint32 minTotalCol The minimum total column overlap for the entire grid (in pixels). sint32 maxTotalCol The maximum total column overlap for the entire grid (in pixels). sint32 minTotalRow The minimum total row overlap for the entire grid (in pixels). sint32 maxTotalRow The maximum total row overlap for the entire grid (in pixels).

Detailed Description

Represents overlap limits of individual display in a display grid.

Member Function Documentation

string info ()

Formats basic information about **OverlapLimits** instance into a human-readable string.

Member Data Documentation

uint32 id

Unique identification of the overlap limits.

sint32 maxOverlapCol

The maximum column overlap for each display (in pixels).

sint32 maxOverlapRow

The maximum row overlap for each display (in pixels).

sint32 maxTotalCol

The maximum total column overlap for the entire grid (in pixels).

sint32 maxTotalRow

The maximum total row overlap for the entire grid (in pixels).

sint32 minOverlapCol

The minimum column overlap for each display (in pixels).

sint32 minOverlapRow

The minimum row overlap for each display (in pixels).

sint32 minTotalCol

The minimum total column overlap for the entire grid (in pixels).

sint32 minTotalRow

The minimum total row overlap for the entire grid (in pixels).

Version verClass

Object version

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

• nvwmi.mof

NVIDIA

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

PcieLink Class Reference

Represents parameters of PCI Express bus link. More...

List of all members.

Public Member Functions

string info ()

Formats basic information about this PCI-Express link into a human-readable string.

Public Attributes

Version verClass

sint32 maxSpeed

Maximum speed attainable on this link in Mbits per second.

sint32 maxGen Maximum supported generation of the PCIe bus protocol. sint32 maxWidth Maximum width, in # of PCI-Express lanes. sint32 curSpeed Current speed on this link in Mbits per second. sint32 curGen Currently negotiated generation of the PCIe bus protocol. sint32 curWidth Current width, in # of PCI-Express lanes.

Detailed Description

Represents parameters of PCI Express bus link.

Member Function Documentation

string info ()

Formats basic information about this PCI-Express link into a human-readable string.

Member Data Documentation

sint32 curGen

Currently negotiated generation of the PCIe bus protocol.

sint32 curSpeed

Current speed on this link in Mbits per second.

sint32 curWidth

Current width, in *#* of PCI-Express lanes.

sint32 maxGen

Maximum supported generation of the PCIe bus protocol.

sint32 maxSpeed

Maximum speed attainable on this link in Mbits per second.

sint32 maxWidth

Maximum width, in # of PCI-Express lanes.

Version verClass

Object version

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

• nvwmi.mof

NVIDIA

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

ProcessUtilization Class Reference

Represents per process GPU encoder utilization values. More...

List of all members.

Public Member Functions

string info ()

Formats basic information about per process utilization into a human-readable string.

Public Attributes

Version verClass

uint32 id Unique ID. uint32 processCount Total no of process. uint32 pid [] Process Id. string timeStamp [] CPU Timestamp. uint32 smUtilization [] SM Utilization value. uint32 memUtilization [] Mem Utilization value. uint32 encUtilization [] Enc Utilization value. uint32 decUtilization [] Dec Utilization value.

Detailed Description

Represents per process GPU encoder utilization values.

Member Function Documentation

string info ()

Formats basic information about per process utilization into a human-readable string.

Member Data Documentation

uint32 decUtilization[]

Dec Utilization value.

uint32 encUtilization[]

Enc Utilization value.

uint32 id

Unique ID.

uint32 memUtilization[]

Mem Utilization value.

uint32 pid[]

Process Id.

uint32 processCount

Total no of process.

uint32 smUtilization[]

SM Utilization value.

string timeStamp[]

CPU Timestamp.

Version verClass

ProcessUtilization class version.

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

• nvwmi.mof

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- <u>Main Page</u>
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- Class Index
- <u>Class Members</u>

Profile Class Reference

Global profile class. More...

List of all members.

Public Member Functions

boolean setValueById ([in]uint32 settingId,[in]uint32 value) Set the 32-bit value by a setting ID.
boolean setBinaryValueById ([in]uint32 settingId,[in]uint8 value[]) Set the binary value by a setting ID.
boolean setStringValueById ([in]uint32 settingId,[in]string value) Set the string value by a setting ID.
boolean restoreSettings ([in]uint32 settingIds[]) Restore or delete settings, specified by IDs. Predefined settings get restored and non-predefined settings get deleted.
string info () formats basic information about a profile into a human-readable string

Public Attributes

Version verClass uint32 id Unique id of a profile, used as a key. string name Name of a profile. uint32 type Type of global profile. Possible values are: 1 - 3D global, 3 nView global, 5 - system. boolean isPredefined If true, profile is predefined and cannot be removed. boolean isSupported If true, profile is supported on this system. Setting settings [] Array of settings for a profile.

Detailed Description

Global profile class.

Member Function Documentation

string info ()

formats basic information about a profile into a human-readable string

boolean restoreSettings ([in] uint32 settingIds[])

Restore or delete settings, specified by IDs. Predefined settings get

restored and non-predefined settings get deleted.

Parameters:

```
settingIds Array of Setting IDs
```

```
boolean setBinaryValueById ( [in] uint32 settingId,
[in] uint8 value[]
)
```

Set the binary value by a setting ID.

Parameters:

settingId **Setting** ID *value* Binary value as an array of bytes

boolean setStringValueById ([in] uint32 *settingId*, [in] string *value*)

Set the string value by a setting ID.

Parameters:

settingId Setting ID value String value

```
boolean setValueById ( [in] uint32 settingId,
[in] uint32 value
)
```

Set the 32-bit value by a setting ID.

Parameters:

settingId Setting ID value 32-bit value

Member Data Documentation

uint32 id

Unique id of a profile, used as a key.

boolean isPredefined

If true, profile is predefined and cannot be removed.

boolean isSupported

If true, profile is supported on this system.

string name

Name of a profile.

Setting settings[]

Array of settings for a profile.

uint32 type

Type of global profile. Possible values are: 1 - 3D global, 3 - nView global, 5 - system.

Version verClass

Version of a profile class

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

ProfileManager Class Reference

Represents root object for performing Profile related task. More...

List of all members.

Public Member Functions

string getAllProfiles ([in, ValueMap{"0","1","3","4","5"}, Values{"3D
Application Profile","3D Global Profile","NView Profile","Display
Profile","System Profile"}] uint32 type)
Get name and id of all profiles for a given profile type. The return
value is a string in format profile id: profile name e.g.profile1 id:
profile 1 name; profile2 id: profile 2 name;.
boolean restoreDefaults3D ()
Restore all profiles to default.
boolean setCurrentProfile3D ([in]string name)

Set current profile.

boolean setVSync ([in, ValueMap{"0","1","2","3","4"}, Values{"Passive","Off","On","Adaptive","Adaptive half"}] uint32 vsvncMode) Set mode of vertical synchronization in the current global profile. boolean loadDesktopProfile ([in]string name) Load a selected profile into nView Desktop Manager. boolean **lockDesktopProfile** ([in]string name,[in]uint32 lock) Lock a nView Desktop profile, once locked the setting in this profile will not be modified through nView Dekstop Manager. boolean saveDesktopProfile ([in]string name) Save nView Desktop Manager settings to the profile. boolean deleteDesktopProfile ([in]string name) Delete the nView Desktop profile. boolean saveSystemProfile ([in]string name) Save system profile. boolean applySystemProfile ([in]string name) Apply system profile. boolean saveDisplayProfiles ([in]string prefix) Save current display state in display profiles with given prefixes. One display profile per **DisplayGrid**. **DisplayGrid** might contain more than one physical display. boolean applyDisplayProfiles ([in]string prefix) Apply saved state in display profiles with given prefixes. One display profile per **DisplayGrid**. boolean createProfile ([in]string name,[in]uint32 type,[in]string params) Create a new **Profile**. boolean restoreProfile ([in]string name) Restore/Delete the Profile. Predefined Profile gets restored and non-predefined Profile gets deleted. string info () Formats basic information about Profile Manager into a humanreadable string.

Public Attributes

Version verClass

string currentProfile3D Current Global 3D profile (D3D, OGL). string defaultProfile3D Default Global 3D profile (D3D, OGL). string currentProfileDesktop Current Desktop Profile (nView). string defaultProfileDesktop Default Desktop Profile (nView). string currentSystemProfile Current system profile. string defaultSystemProfile Default system profile.

Detailed Description

Represents root object for performing **Profile** related task.

Member Function Documentation

boolean applyDisplayProfiles ([in] string prefix)

Apply saved state in display profiles with given prefixes. One display profile per **DisplayGrid**.

Parameters:

prefix Common prefix - e.g. "my display" Ordinals will be added per every display profile (e.g. "my display 1 of 4", "my display 2 of 4" etc.)

boolean applySystemProfile ([in] string name)

Apply system profile.

Parameters:

name Name of system profile to be applied

```
boolean createProfile ( [in] string name,
[in] uint32 type,
[in] string params
)
```

Create a new Profile.

Parameters:

name Name of the profile
 type Type of the profile. 0 - 3D application profile
 params Additional parameters separated by semicolon. For a 3D
 Application Profile specify a minimal path to the binary
 which uses Direct3D or OpenGL, e.g.
 "Application1.exe;Application2.exe". Note that white space at the beginning or end of a path will be trimmed and empty entries discarded.

boolean deleteDesktopProfile ([in] string name)

Delete the nView Desktop profile.

Parameters:

name Name of the existing profile which will be deleted

string getAllProfiles [in, ValueMap{"0","1","3","4","5"}, Values{"3D (Application Profile","3D Global Profile","NView type) Profile","Display Profile","System Profile"}] uint32

Get name and id of all profiles for a given profile type. The return value is a string in format profile id: profile name e.g.profile1 id: profile 1 name; profile2 id: profile 2 name;.

Parameters:

type Type of the profile

string info ()

Formats basic information about **Profile** Manager into a human-readable string.

```
boolean loadDesktopProfile ( [in] string name )
```

Load a selected profile into nView Desktop Manager.

Parameters:

name Name of existing profile to be loaded into nView Desktop Manager

boolean lockDesktopProfile ([in] string *name*, [in] uint32 *lock*)

Lock a nView Desktop profile, once locked the setting in this profile will not be modified through nView Dekstop Manager.

Parameters:

name Name of the existing profile which needs to be lockedlock Paramer to lock/unlock the profile. Pass 1 to indicate lock and pass 0 to unlock the profile

boolean restoreDefaults3D ()

Restore all profiles to default.

boolean restoreProfile ([in] string name)

Restore/Delete the **Profile**. Predefined **Profile** gets restored and nonpredefined **Profile** gets deleted.

Parameters:

name Profile name

boolean saveDesktopProfile ([in] string name)

Save nView Desktop Manager settings to the profile.

Parameters:

name Name of profile where the nView Desktop Manager settings will be saved

boolean saveDisplayProfiles ([in] string prefix)

Save current display state in display profiles with given prefixes. One display profile per **DisplayGrid**. **DisplayGrid** might contain more than one physical display.

Parameters:

prefix Common prefix - e.g. "my display". Ordinals will be added per every display profile (e.g. "my display 1 of 4", "my display 2 of 4" etc.)

boolean saveSystemProfile ([in] string name)

Save system profile.

Parameters:

name Name of system profile to be saved

boolean setCurrentProfile3D ([in] string name)

Set current profile.

Parameters:

name Name of existing profile to set as current

boolean setVSync [in, ValueMap{"0","1","2","3","4"}, (Values{"Passive","Off","On","Adaptive","Adaptive vsyncMode half"}] uint32 Set mode of vertical synchronization in the current global profile.

Parameters:

vsyncMode V-Sync mode to be applied to the current global profile values might be Application-Controlled (same as Passive) = 0, Off = 1, On = 2, Adaptive = 3 and Adaptive (half refresh rate) = 4

Member Data Documentation

string currentProfile3D

Current Global 3D profile (D3D, OGL).

string currentProfileDesktop

Current Desktop **Profile** (nView).

string currentSystemProfile

Current system profile.

string defaultProfile3D

Default Global 3D profile (D3D, OGL).

string defaultProfileDesktop

Default Desktop Profile (nView).

string defaultSystemProfile

Default system profile.

Version verClass

Profile Manager class version

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

Setting Class Reference

Profile setting. More...

List of all members.

Public Member Functions

string getStringValue ()

Get the string value for a setting.

string info ()

Formats basic information about the **Profile** setting object into a human-readable string.

Public Attributes

Version verClass

uint32 id Unique id of a setting, used as a key. uint32 **type** Type of value of the setting. Possible values are: 0 - invalid, 1 uint32, 2 - sint32, 3 - string, 4 - binary. uint8 **value** [] Value for the **Setting**.

Detailed Description

Profile setting.

Member Function Documentation

string getStringValue ()

Get the string value for a setting.

string info ()

Formats basic information about the **Profile** setting object into a human-readable string.

Member Data Documentation

uint32 id

Unique id of a setting, used as a key.

uint32 type

Type of value of the setting. Possible values are: 0 - invalid, 1 - uint32, 2 - sint32, 3 - string, 4 - binary.

uint8 value[]

Value for the **Setting**.

Version verClass

Version of a Setting class

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

SettingInfo Class Reference

Useful information about a profile setting. More...

List of all members.

Public Member Functions

string info ()

formats information about a setting into a human-readable string

Public Attributes

Version verClass uint32 id Unique id of a setting, used as a key. string name Setting name. string description Setting description.

Detailed Description

Useful information about a profile setting.

Member Function Documentation

string info ()

formats information about a setting into a human-readable string

Member Data Documentation

string description

Setting description.

uint32 id

Unique id of a setting, used as a key.

string name

Setting name.

Version verClass

Version of a SettingInfo class

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

• nvwmi.mof

NVIDIA

Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

SettingTable Class Reference

Table which describes profile settings of certain type. More...

List of all members.

Public Member Functions

uint32 getIdFromName ([in]string name) get setting ID from setting name string getNameFromId ([in]uint32 id) get setting name from setting ID string infoByName ([in]string name) formats information about given setting with given name string infoById ([in]uint32 id) formats information about given setting with given ID string info () formats basic information about all settings into a humanreadable string

Public Attributes

Version verClass

uint32 id Unique id of a setting table, used as a key.
uint32 type Profile type for settings in this table. Possible values are: 0 -3D Application, 1 - 3D Global, 2 - nView Application, 3 nView Global, 4 - Display, 5 - System.
uint32 settingIds [] Array of all setting IDs, described in this table.
string settingNames [] Array of all setting names, described in this table.
SettingInfo settings [] Array of settings, described in this table.

Detailed Description

Table which describes profile settings of certain type.

Member Function Documentation

uint32 getIdFromName ([in] string name)

get setting ID from setting name

Parameters:

name Setting name

string getNameFromId ([in] uint32 id)

get setting name from setting ID

Parameters:

id Setting ID

string info ()

formats basic information about all settings into a human-readable string

```
string infoById ([in] uint32 id)
```

formats information about given setting with given ID

Parameters:

id Setting ID

string infoByName ([in] string name)

formats information about given setting with given name

Parameters: name Setting name

Member Data Documentation

```
uint32 id
```

Unique id of a setting table, used as a key.

uint32 settingIds[]

Array of all setting IDs, described in this table.

string settingNames[]

Array of all setting names, described in this table.

SettingInfo settings[]

Array of settings, described in this table.

uint32 type

Profile type for settings in this table. Possible values are: 0 - 3D **Application**, 1 - 3D Global, 2 - nView **Application**, 3 - nView Global, 4 - **Display**, 5 - **System**.

Version verClass

Version of a SettingTable class

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

Sync Class Reference

Represents Sync-capable devices. More...

List of all members.

Public Member Functions

boolean setSyncStateByld ([in]uint32 syncDisplayIds[], [in]uint32 syncState[])
Setup the synchronization between displays with given IDs.
boolean setSyncStateByName ([in]string syncDisplayNames, [in]uint32 syncState[])
Setup the synchronization between displays with given Names.
boolean toggleSource ()
Toggle the sync signal source between internal and external house sync.

- boolean **setPolarity** ([in, ValueMap{"0","1","2"}, Values{"Rising Edge","Falling Edge","Both Edges"}] uint32 **polarity**) Set synchronization signal polarity.
- boolean **setVmode** ([in, ValueMap{"0","1","2","3","4"}, Values{"none","TTL","NTSC-PAL-SECAM","HDTV","composite"}] uint32 **vmode**) Set video mode type.
- boolean **setInterval** ([in]uint32 **interval**) Set number of pulses to wait between framelock signal generation.
- boolean **setInterlaceMode** ([in, ValueMap{"0","1"}, Values{"Disabled","Enabled"}] uint32 **interlaceMode**) Set interlace mode for a **Sync** device.
- boolean **setSyncSkew** ([in]uint32 numOfPixels,[in]uint32 numOfLines)

Set the amount of delay between the frame sync signal and the GPUs signal. **Sync** device must be in Client mode or if external Housesync signal present.

boolean **setStartupDelay** ([in]uint32 numOfPixels,[in]uint32 numOfLines)

Set the amount of delay the frame lock card should wait, until generating sync pulse. **Sync** device must be in Server mode.

string info ()

Formats basic information about **Sync** objects into a human-readable string.

Public Attributes

Version ver uint32 id Unique id of named object, used as a key. string name Name of an object. string uname

Unique name of an object.

sint32 ordinal

Ordinal number of named object among objects with same names.

sint32 count

Total number of named objects with same name.

Version verSyncFirmware

uint32 nvapild

NVAPI ID of the Sync device.

SyncTopology ref syncDisplays []

Array of references to SyncTopology objects.

boolean isSynced

Is sync enabled on this device?

boolean isHouseSync

Is house sync present?

boolean isStereoSynced

Is stereo enabled?

uint32 polarity

Polarity of a synchronization signal. Possible values are: 0 - Rising Edge, 1 - Falling Edge, 2 - Both Edges.

uint32 vmode

Video mode. Possible values are: 0 - none, 1 - TTL, 2 - NTSC-PAL-SECAM, 3 - HDTV, 4 - composite.

uint32 interval

Number of pulses to wait between framelock signal generation.

uint32 source

Source of a synchronization signal. Possible values are: 0 - VSync, 1 - HouseSync.

uint32 interlaceMode

Interlace mode for a **Sync** device. Possible values are: 0 - Disabled, 1 - Enabled.

real32 syncSignalRate

Sync signal refresh rate in Hz. If the house sync is present and selected source is HouseSync, then it will

be the house sync signal refresh rate. Otherwise, it will be an internal display signal refresh rate.

uint32 flStatus []

RJ-45 signal status. Possible values are: 0 - output, 1 - input, 2 - unused.

SyncDelay syncSkew

syncSkew object (The time delay between the frame sync signal and the GPUs signal). Available only in Client mode or if external Housesync signal is present.

SyncDelay startupDelay

startupDelay object (The amount of time the frame lock card should wait until generating sync pulse). Available only in Server mode.

Detailed Description

Represents Sync-capable devices.

Member Function Documentation

```
string info ( )
```

Formats basic information about **Sync** objects into a human-readable string.

```
boolean [in, ValueMap{"0","1"},
setInterlaceMode (Values{"Disabled","Enabled"}] interlaceMode )
uint32
```

Set interlace mode for a **Sync** device.

Parameters:

interlaceMode Interlace mode for a **Sync** device. Possible values are: 0 - Disabled, 1- Enabled.

boolean setInterval ([in] uint32 interval)

Set number of pulses to wait between framelock signal generation.

Parameters:

interval The number of pulses to skip for signal generation. 0 uses source, 1 gives half the frequency, and so on.

boolean [in, ValueMap{"0","1","2"}, Values{"Rising setPolarity (Edge","Falling Edge","Both Edges"}] uint32 polarity)

Set synchronization signal polarity.

Parameters:

```
polarity Possible values are: 0 - rising edge, 1 - falling edge, 2 - both
```

boolean setStartupDelay ([in] uint32 *numOfPixels*, [in] uint32 *numOfLines*)

Set the amount of delay the frame lock card should wait, until generating sync pulse. **Sync** device must be in Server mode.

Parameters:

numOfPixels Number of pixels to induce startup delay *numOfLines* Number of horizontal lines to induce startup delay

```
boolean setSyncSkew ( [in] uint32 numOfPixels,
[in] uint32 numOfLines
)
```

Set the amount of delay between the frame sync signal and the GPUs signal. **Sync** device must be in Client mode or if external Housesync signal present.

Parameters:

numOfPixels Number of pixels to induce sync skew

numOfLines Number of horizontal lines to induce sync skew

```
boolean setSyncStateById ( [in] uint32 syncDisplayIds[],
[in] uint32 syncState[]
)
```

Setup the synchronization between displays with given IDs.

Parameters:

```
syncDisplayIds Array of display IDs to
```

synchronize(**SyncTopology.id**), separated by comma. Order is important. For Mosaic topologies, user can specify either a master display or all displays in a Mosaic topology. If user does not specify any displays, the method will choose the best topology to synchronize. To un-synchronize the displays, user will have to pass all the displays with displaySyncState as UnSynced.

syncState Possible values are: 0 - UnSynced, 1 - Slave, 2 - Master

Setup the synchronization between displays with given Names.

Parameters:

syncDisplayNames Display Names to synchronize. String of unique display names (SyncTopology.uname), separated by semicolon. Order is important. For Mosaic topologies, user can specify either a master display or all displays in a Mosaic topology. If user does not specify any displays, the method will choose the best topology to synchronize. To un-synchronize the displays,

	user will have to pass all the displays with
	displaySyncState as UnSynced.
syncState	Possible values are: 0 - UnSynced, 1 - Slave, 2
	- Master

boolean
setVmode[in, ValueMap{"0","1","2","3","4"},
(Values{"none","TTL","NTSC-PAL-
SECAM","HDTV","composite"}] uint32vmode

Set video mode type.

Parameters:

vmode Possible values are: 0 - none, 1 - TTL, 2 -NTSC/PAL/SECAM, 3 - HDTV, 4 - composite

```
boolean toggleSource ()
```

Toggle the sync signal source between internal and external house sync.

Member Data Documentation

sint32 count

Total number of named objects with same name.

```
uint32 flStatus[]
```

RJ-45 signal status. Possible values are: 0 - output, 1 - input, 2 - unused.

uint32 id

Unique id of named object, used as a key.

uint32 interlaceMode

Interlace mode for a Sync device. Possible values are: 0 - Disabled, 1 -

Enabled.

uint32 interval

Number of pulses to wait between framelock signal generation.

boolean isHouseSync

Is house sync present?

boolean isStereoSynced

Is stereo enabled?

boolean isSynced

Is sync enabled on this device?

string **name**

Name of an object.

uint32 nvapild

NVAPI ID of the **Sync** device.

sint32 ordinal

Ordinal number of named object among objects with same names.

uint32 **polarity**

Polarity of a synchronization signal. Possible values are: 0 - Rising Edge, 1 - Falling Edge, 2 - Both Edges.

uint32 source

Source of a synchronization signal. Possible values are: 0 - VSync, 1 -

HouseSync.

SyncDelay startupDelay

startupDelay object (The amount of time the frame lock card should wait until generating sync pulse). Available only in Server mode.

SyncTopology ref syncDisplays[]

Array of references to **SyncTopology** objects.

real32 syncSignalRate

Sync signal refresh rate in Hz. If the house sync is present and selected source is HouseSync, then it will be the house sync signal refresh rate. Otherwise, it will be an internal display signal refresh rate.

SyncDelay syncSkew

syncSkew object (The time delay between the frame sync signal and the GPUs signal). Available only in Client mode or if external Housesync signal is present.

string uname

Unique name of an object.

Version ver

Version of named object

Version verSyncFirmware

Version of the Sync board firmware

uint32 vmode

Video mode. Possible values are: 0 - none, 1 - TTL, 2 - NTSC-PAL-

SECAM, 3 - HDTV, 4 - composite.

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

SyncDelay Class Reference

Represents the Sync Delay (Sync Skew or Startup Delay). More...

List of all members.

Public Member Functions

string info ()

Formats basic information about **SyncTopology** objects into a human-readable string.

Public Attributes

Version verClass uint32 id Unique identification of SyncDelay objects. uint32 delayType

Type of Sync Delay.. Possible values are: 0 - unknown, 1 - SyncSkew, 2 - StartupDelay.

uint32 minPixels

Minimum number of pixels required at current display mode to induce sync delay.

uint32 maxLines

Maximum number of lines supported at current display mode to induce sync delay.

uint32 numOfPixels

Sync delay to be induced in unit pixels.
uint32 numOfLines
Sync delay to be induced in unit horizontal lines.

Detailed Description

Represents the Sync Delay (Sync Skew or Startup Delay).

Member Function Documentation

string info ()

Formats basic information about **SyncTopology** objects into a human-readable string.

Member Data Documentation

uint32 delayType

Type of **Sync** Delay.. Possible values are: 0 - unknown, 1 - SyncSkew, 2 - StartupDelay.

uint32 id

Unique identification of **SyncDelay** objects.

uint32 maxLines

Maximum number of lines supported at current display mode to induce sync delay.

uint32 minPixels

Minimum number of pixels required at current display mode to induce sync delay.

uint32 numOfLines

Sync delay to be induced in unit horizontal lines.

uint32 numOfPixels

Sync delay to be induced in unit pixels.

Version verClass

SyncDelay class version

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

SyncEvent Class Reference

Represent Sync events. More...

List of all members.

Public Attributes

Version verClass

 uint32 syncDeviceId
 Unique sync device ID.
 uint32 eventType
 Sync event type. Possible values are: 0 - Unknown, 1 SyncLoss, 2 - SyncGain, 4 - StereoLoss, 8 - StereoGain, 0x10 HouseSyncGain, 0x20 - HouseSyncLoss, 0x40 - FlGain, 0x80 FlLoss, 0x10000 - SyncStateChange.

Detailed Description

Represent Sync events.

Member Data Documentation

uint32 eventType

Sync event type. Possible values are: 0 - Unknown, 1 - SyncLoss, 2 -SyncGain, 4 - StereoLoss, 8 - StereoGain, 0x10 - HouseSyncGain, 0x20 - HouseSyncLoss, 0x40 - FlGain, 0x80 - FlLoss, 0x10000 -SyncStateChange.

uint32 syncDeviceId

Unique sync device ID.

Version verClass

Version of the SyncEvent class

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





NVWMI v2.31 API Reference Documentation NVIDIA

Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

SyncTopology Class Reference

Represents the Sync Topology. More...

List of all members.

Public Member Functions

boolean isGPUSynced ()

Returns true if GPU is synchronized (in phase) with the **Sync** device.

string info ()

Formats basic information about **SyncTopology** objects into a human-readable string.

Public Attributes

Version ver uint32 id Unique id of named object, used as a key. string name Name of an object. string uname Unique name of an object. sint32 ordinal Ordinal number of named object among objects with same names. sint32 count Total number of named objects with same name. uint32 displaySyncState boolean isDisplayMasterable Can the current display be set as a sync master?

Detailed Description

Represents the **Sync** Topology.

Member Function Documentation

```
string info ( )
```

Formats basic information about **SyncTopology** objects into a human-readable string.

```
boolean isGPUSynced ()
```

Returns true if GPU is synchronized (in phase) with the Sync device.

Member Data Documentation

sint32 count

Total number of named objects with same name.

uint32 displaySyncState

Synchronization state of a display in the **Sync** topology. Possible values are: 0 - UnSynced, 1 - Slave, 2 - Master

uint32 id

Unique id of named object, used as a key.

boolean isDisplayMasterable

Can the current display be set as a sync master?

string name

Name of an object.

sint32 ordinal

Ordinal number of named object among objects with same names.

string uname

Unique name of an object.

Version ver

Version of named object

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

• nvwmi.mof

NVIDIA Copyright (c) 2010-2018 NVIDIA Corporation. All rights reserved.





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

System Class Reference

Represents root object for NV-related HW and SW in the system. More...

List of all members.

Public Member Functions

```
boolean setLicensingServer ([in]string address,[in]uint16 port)
Change address of a licensing server.
boolean unsetLicensingServer ()
Clear licensing server address and forfeit license after reboot.
boolean setnViewState ([in, ValueMap{"0","1"},
Values{"Disable","Enable"}] uint32 state)
Set nView Desktop Manager State.
boolean setLogState ([in, BitMap{"0","1","2"},
BitValues{"debugger","file","eventLog"}] uint32 types,[in]uint32
filter,[in]uint32 options)
```

Change logging settings.

```
string enableEncoderSessionsPerfCounter ()
```

Enable NVIDIA GPU Encoder Sessions performance counter. string disableEncoderSessionsPerfCounter ()

Disable NVIDIA GPU Encoder Sessions performance counter. string enableProcessUtilizationPerfCounter ()

Enable NVIDIA GPU Process Utilization performance counter. string disableProcessUtilizationPerfCounter ()

Disable NVIDIA GPU Process Utilization performance counter. string **info** ()

Formats basic information about **System** objects into a human-readable string.

Public Attributes

Version verClass Version verNVWMI Version of the NVWMI driver. Version verDisplayDriver Version of the NV display driver. Version verSDICaptureFirmware Firmware version of the SDI Capture board. Version verSDIOutputFirmware Firmware version of the SDI output board. uint32 logTypes log types. Default is 5 (e.g. debugger and OS event log). 0 disabled. uint32 logFilter bitmask for filtering out messages. Recommended values are 0 quiet, 16 - errors only, 272 - errors and warnings, 16777216 performance profiling uint32 logOptions reserved. Default is 0 uint16 licensingPort Network port on a licensing server. Default value is 0. When 0 is

specified, default server port value will be used. string **licensingServer**

Network address of licensing server.

Version verVBIOS

The video BIOS version associated with the GPU's.

Version vernViewDesktopManager

The version of nView Desktop Manager.

uint32 machineType

Type of machine whether desktop or mobile. Possible values are: 0 - Desktop, 1 - Mobile.

uint32 nViewState

nView Desktop Manager current state. Possible values are: 0 - Disable, 1 - Enable, 2 - Not Installed

Detailed Description

Represents root object for NV-related HW and SW in the system.

Member Function Documentation

string disableEncoderSessionsPerfCounter()

Disable NVIDIA GPU Encoder Sessions performance counter.

string disableProcessUtilizationPerfCounter ()

Disable NVIDIA GPU Process Utilization performance counter.

string enableEncoderSessionsPerfCounter ()

Enable NVIDIA GPU Encoder Sessions performance counter.

string enableProcessUtilizationPerfCounter ()

Enable NVIDIA GPU Process Utilization performance counter.

string info ()

Formats basic information about **System** objects into a human-readable string.

boolean setLicensingServer ([in] string address, [in] uint16 port)

Change address of a licensing server.

Parameters:

address Network address of licensing server

port	Network port on licensing server. This is optional
	parameter and will be set to default value of 0 when
	omitted

boolean setLogState	<pre>([in, BitMap{"0","1","2"}, (BitValues{"debugger","file","eventLog"}] uint32</pre>	types,
	[in] uint32	filter,
	[in] uint32	options
)	

Change logging settings.

Parameters:

- *types* enabled log types. Default is 5 (e.g. debugger and OS event log)
- *filter* bitmask for filtering out messages. Recommended values are 0 - quiet, 16 - errors only, 272 - errors and warnings, 16777216 - performance profiling

options optional parameter, reserved. Set to 0

boolean [in, ValueMap{"0","1"},

Set nView Desktop Manager State.

Parameters: state nView state to set

boolean unsetLicensingServer ()

Clear licensing server address and forfeit license after reboot.

Member Data Documentation

uint16 licensingPort

Network port on a licensing server. Default value is 0. When 0 is specified, default server port value will be used.

string licensingServer

Network address of licensing server.

uint32 logFilter

bitmask for filtering out messages. Recommended values are 0 - quiet, 16 - errors only, 272 - errors and warnings, 16777216 - performance profiling

uint32 logOptions

reserved. Default is 0

uint32 logTypes

log types. Default is 5 (e.g. debugger and OS event log). 0 - disabled.

uint32 machineType

Type of machine whether desktop or mobile. Possible values are: 0 - Desktop, 1 - Mobile.

uint32 nViewState

nView Desktop Manager current state. Possible values are: 0 - Disable, 1 - Enable, 2 - Not Installed

Version verClass

System root class version

Version verDisplayDriver

Version of the NV display driver.

Version vernViewDesktopManager

The version of nView Desktop Manager.

Version verNVWMI

Version of the NVWMI driver.

Version verSDICaptureFirmware

Firmware version of the SDI Capture board.

Version verSDIOutputFirmware

Firmware version of the SDI output board.

Version verVBIOS

The video BIOS version associated with the GPU's.

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

• nvwmi.mof

NVIDIA





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

ThermalEvent Class Reference

Represents GPU thermal events. More...

List of all members.

Public Attributes

uint32 handleGpu uint32 probeIndex Thermal probe index in array of probes on a given GPU. uint32 thermalLevel Thermal level to identify the temperature zone. Possible values are: 0 - unknown, 1 - normal, 2 - warning, 3 - critical.

Detailed Description

Represents GPU thermal events.

Member Data Documentation

uint32 handleGpu

Unique identification of the NVIDIA GPU

uint32 probeIndex

Thermal probe index in array of probes on a given GPU.

uint32 thermalLevel

Thermal level to identify the temperature zone. Possible values are: 0 - unknown, 1 - normal, 2 - warning, 3 - critical.

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

• nvwmi.mof





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

ThermalProbe Class Reference

Represents a thermal probe. More...

List of all members.

Public Member Functions

string info ()

Formats basic information about the Thermal object in a system into a human-readable string.

Public Attributes

Version verClass uint32 id Unique ID. uint32 handle

Unique ID - deprecated, please use 'id'.

uint32 type

The type of thermal probe controller type. Possible values are: -1 - unknown, 0 - none, 1 - gpuInternal, 2 - adm1032, 3 max6649, 4 - max1617, 5 - Im99, 6 - Im89, 7 - Im64, 8 - adt7473, 9 - sbmax6649, 10 - vbiosEvt, 11 - os.

sint32 temperature

The current temperature value of the thermal sensor in degrees Celsius.

sint32 defaultMinTemperature

Default minimum temperature value of the thermal sensor in degrees Celsius.

sint32 defaultMaxTemperature

Default maximum temperature value of the thermal sensor in degrees Celsius.

uint32 thermalLevel

current temperature event level. Possible values are: 0 - unknown, 1 - normal, 2 - warning, 3 - critical

Detailed Description

Represents a thermal probe.

Member Function Documentation

string info ()

Formats basic information about the Thermal object in a system into a human-readable string.

Member Data Documentation

sint32 defaultMaxTemperature

Default maximum temperature value of the thermal sensor in degrees Celsius.

sint32 defaultMinTemperature

Default minimum temperature value of the thermal sensor in degrees Celsius.

uint32 handle

Unique ID - deprecated, please use 'id'.

uint32 id

Unique ID.

sint32 temperature

The current temperature value of the thermal sensor in degrees Celsius.

uint32 thermalLevel

current temperature event level. Possible values are: 0 - unknown, 1 - normal, 2 - warning, 3 - critical

uint32 type

The type of thermal probe controller type. Possible values are: -1 - unknown, 0 - none, 1 - gpuInternal, 2 - adm1032, 3 - max6649, 4 - max1617, 5 - Im99, 6 - Im89, 7 - Im64, 8 - adt7473, 9 - sbmax6649, 10 - vbiosEvt, 11 - os.

Version verClass

Thermal probe class version

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

• nvwmi.mof

NVIDIA





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>

Version Class Reference

Internal version of NV objects. More...

List of all members.

Public Attributes

uint32 **orderedValue** uint32 **value** Raw version value. string **strValue** Human-readable string with version value.

Detailed Description

Internal version of NV objects.

Member Data Documentation

uint32 orderedValue

Ordered version value, guaranteed to grow monotonically

string strValue

Human-readable string with version value.

uint32 value

Raw version value.

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

• nvwmi.mof





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

VideoCodec Class Reference

Represents parameters of Video Engine encoder and decoder. More...

List of all members.

Public Member Functions

string info ()

Formats basic information about this PCI-Express link into a human-readable string.

Public Attributes

Version verClass sint32 percentEncoderUsage The percentage of Video Encoder utilization during sampling period.

sint32 percentDecoderUsage

The percentage of Video Decoder utilization during sampling period.

sint32 encoderSamplingPeriod

The period (in milliseconds) between Video Encoder samples. -1 if not supported in current environment.

sint32 decoderSamplingPeriod

The period (in milliseconds) between Video Decoder samples. -1 if not supported in current environment.

sint32 encoderSessionsCount

Count of active encoder sessions. 0 if not supported in current environment or when encoder inactive.

sint32 averageFps

Frames per second encoded, as a trailing average for all active sessions. -1 if not supported in current environment.

sint32 averageLatency

Encoding latency in milliseconds. -1 if not supported in current environment.

Detailed Description

Represents parameters of Video Engine encoder and decoder.

Member Function Documentation

string info ()

Formats basic information about this PCI-Express link into a human-readable string.

Member Data Documentation

sint32 averageFps

Frames per second encoded, as a trailing average for all active sessions. -1 if not supported in current environment.

sint32 averageLatency

Encoding latency in milliseconds. -1 if not supported in current environment.

sint32 decoderSamplingPeriod

The period (in milliseconds) between Video Decoder samples. -1 if not supported in current environment.

sint32 encoderSamplingPeriod

The period (in milliseconds) between Video Encoder samples. -1 if not supported in current environment.

sint32 encoderSessionsCount

Count of active encoder sessions. 0 if not supported in current environment or when encoder inactive.

sint32 percentDecoderUsage

The percentage of Video Decoder utilization during sampling period.

sint32 percentEncoderUsage

The percentage of Video Encoder utilization during sampling period.

Version verClass

Object version

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

• nvwmi.mof

NVIDIA





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- •
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>W</u>

Here is a list of all class members with links to the classes they belong to:

- a -

- addApplications() : ApplicationProfile
- aggregateDoubleBitErrors : Ecc
- aggregateSingleBitErrors : Ecc
- applications : ApplicationProfile
- applyDisplayProfiles() : ProfileManager
- applySystemProfile() : ProfileManager
- archld : Gpu
- archName : Gpu
- averageEncodeFps : EncoderSessions
- averageEncodeLatency : EncoderSessions
- averageFps : VideoCodec
- averageLatency : VideoCodec

NVIDIA





Feb 2018

- <u>Main Page</u>
- <u>Related Pages</u>
- <u>Classes</u>

Related Pages

Here is a list of all related documentation pages:

- Legal Notice
- Version-specific Implementation Details
- NVWMI compatibility
- Cooler and Thermal Events
- Using NVWMI
- NVIDIA Performance Counters
- Logging and tracing NVWMI activity
- Using NVWMI with the WMIC tool
- Using NVWMI with the PowerShell
- Profile settings available in NVWMI
- Third-Party Notice

NVIDIA





Feb 2018

- <u>Main Page</u>
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

Class List

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

Application Application, associated with an application profile **ApplicationProfile Application** profile Represents a board with NVIDIA GPU(s) Board Represents a cooler (fan, liquid system etc.) Cooler Represents GPU cooler events CoolerEvent DesktopManager Management of nView desktops **Display** Represents physical display Represents physical displays, organized into a **DisplayGrid** regular grid (rows by columns) DisplayGridInfo Information about a display grid **DisplayManager** Management of display-related task **DisplayMode** Represents display mode **DisplayProfile Display** profile

Ecc	Represents the GPU Error Correction Code. This feature is not supported if there are multiple GPU topologies enabled	
EncoderSessions	Represents encoder sessions info	
Gpu	Represents an NVIDIA GPU	
NamedObject	Base class for all named NV objects	
OverlapLimits	Represents overlap limits of individual display in a display grid	
PcieLink	Represents parameters of PCI Express bus link	
ProcessUtilization	Represents per process GPU encoder utilization values	
Profile	Global profile class	
ProfileManager	Represents root object for performing Profile related task	
Setting	Profile setting	
SettingInfo	Useful information about a profile setting	
SettingTable	Table which describes profile settings of certain type	
Sync	Represents Sync-capable devices	
SyncDelay	Represents the Sync Delay (Sync Skew or Startup Delay)	
SyncEvent	Represent Sync events	
SyncTopology	Represents the Sync Topology	
System	Represents root object for NV-related HW and SW in the system	
ThermalEvent	Represents GPU thermal events	
ThermalProbe	Represents a thermal probe	
Version	Internal version of NV objects	
VideoCodec	Represents parameters of Video Engine encoder and decoder	





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

Class Index

$\underline{A} | \underline{B} | \underline{C} | \underline{D} | \underline{E} | \underline{G} | \underline{N} | \underline{O} | \underline{P} | \underline{S} | \underline{T} | \underline{V}$

А	DesktopManager	EncoderSessions	ProcessUt
Application	Display	G	Profile
ApplicationProfile	DisplayGrid	Gpu	ProfileMan
В	DisplayGridInfo	Ν	S
Board	DisplayManager	NamedObject	Setting
С	DisplayMode	0	SettingInfc
Cooler	DisplayProfile	OverlapLimits	SettingTab
CoolerEvent	E	Р	Sync
D	Ecc	PcieLink	SyncDelay

A | B | C | D | E | G | N | O | P | S | T | V





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

Application Member List

This is the complete list of members for **Application**, including all inherited members.

idApplicationinfo()ApplicationisPredefinedApplicationisSupportedApplicationlauncherApplicationnameApplicationsubPathsApplicationverClassApplication





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

ApplicationProfile Member List

This is the complete list of members for **ApplicationProfile**, including all inherited members.

addApplications([in]string appNames[]) **ApplicationProfile** applications **ApplicationProfile** id **ApplicationProfile** info() **ApplicationProfile ApplicationProfile isPredefined isSupported ApplicationProfile ApplicationProfile** name **ApplicationProfile** removeApplications([in]string appNames[]) restoreSettings([in]uint32 settingIds[]) **ApplicationProfile** setBinaryValueById([in]uint32 settingId,[in]uint8 **ApplicationProfile** value∏) setStringValueById([in]uint32 settingId,[in]string **ApplicationProfile** value)

settings setValueById([in]uint32 settingId,[in]uint32 value) startTime type verClass

ApplicationProfile ApplicationProfile ApplicationProfile ApplicationProfile ApplicationProfile

NVIDIA





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

Board Member List

This is the complete list of members for **Board**, including all inherited members.

chipSKU	Board
chipSKUMod	Board
coolers	Board
count	Board
gpus	Board
id	Board
info()	Board
name	Board
nvapild	Board
ordinal	Board
project	Board
projectSKU	Board
serialNumber	Board

thermalProbes	Board
uname	Board
ver	Board





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

Cooler Member List

This is the complete list of members for **Cooler**, including all inherited members.

coolerLevel	Cooler
coolerType	Cooler
fanSpeed	Cooler
handle	Cooler
id	Cooler
info()	Cooler
maxSpeed	Cooler
minSpeed	Cooler
percentCoolerRate	Cooler
verClass	Cooler





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

CoolerEvent Member List

This is the complete list of members for **CoolerEvent**, including all inherited members.

coolerIndex CoolerEvent coolerLevel CoolerEvent handleGpu CoolerEvent





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

DesktopManager Member List

This is the complete list of members for **DesktopManager**, including all inherited members.

<pre>createDesktop([in]string name,[in]string backgrounds)</pre>	DesktopManager
deleteDesktop([in]string name)	DesktopManager
<pre>editDesktop([in]string name,[in]string backgrounds)</pre>	DesktopManager
getAllDesktops()	DesktopManager
verClass	DesktopManager





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

Display Member List

This is the complete list of members for **Display**, including all inherited members.

count displayConnectorType displayModeNative displayModes ditherBits ditherBits ditherMode ditherState EDID EDIDSize getCurrentTiming() gpuConnectorType id info()

isActive locus make model name nvapild ordinal restoreNativeDisplayMode() rotation saveCSC([in]string filePath) saveEDID([in]string filePath) saveGammaRamp([in]string filePath) scaling setCSC([in]string filePath) setDisplayMode([in]uint32 width,[in]uint32 height,[in]real32 refreshRate,[setDisplayModeById([in]uint32 id) setDisplayModeByRef([in]DisplayMode ref mode) setDither([in, ValueMap{"0","1","2"}, Values{"Default","Enable","Disable"}] bit","10 bit"}] uint32 bits,[in, ValueMap{"0","1","2","3","4"}, Values{"SpatialDynamic", "SpatialStatic", "SpatialDynamic2x2", "SpatialStat setEDID([in]string filePath) setGammaRamp([in]string filePath) setGammaRampBasic([in]real32 brightness,[in]real32 contrast,[in]real32 setRotation([in, ValueMap{"0","1","2","3"}, Values{"No rotation","rotate 90 uint32 rotation) setScaling([in, ValueMap{"0","1","2","3","5","6","7"}, Values{"Default","Closest","Native","ScanoutToNative","AspectScanoutTo uint32 scaling) uname ver verFirmware





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>

DisplayGrid Member List

This is the complete list of members for **DisplayGrid**, including all inherited members.

cols count displayModePhysical displayModeS displayModeVirtual displays gpus id info() name ordinal overlapCols overlapLimits DisplayGrid DisplayGrid

overlapRows	DisplayGrid
positionCol	DisplayGrid
positionRow	DisplayGrid
rotation	DisplayGrid
rows	DisplayGrid
<pre>saveCSC([in]string filePath)</pre>	DisplayGrid
<pre>saveGammaRamp([in]string filePath)</pre>	DisplayGrid
setCSC([in]string filePath)	DisplayGrid
<pre>setDisplayMode([in]uint32 width,[in]uint32 height, [in]real32 refreshRate,[in]uint32 depth)</pre>	DisplayGrid
setDisplayModeById([in]uint32 id)	DisplayGrid
<pre>setDisplayModeByRef([in]DisplayMode ref gridMode)</pre>	DisplayGrid
setGammaRamp([in]string filePath)	DisplayGrid
<pre>setGammaRampBasic([in]real32 brightness,[in]real32 contrast,[in]real32 gamma)</pre>	DisplayGrid
<pre>setOverlapCol([in]sint32 index,[in]sint32 overlap)</pre>	DisplayGrid
<pre>setOverlapRow([in]sint32 index,[in]sint32 overlap)</pre>	DisplayGrid
<pre>setOverlaps([in]sint32 index,[in]sint32 overlapRow, [in]sint32 overlapCol)</pre>	DisplayGrid
<pre>setRotation([in]uint32 rotation[])</pre>	DisplayGrid
uname	DisplayGrid
ver	DisplayGrid





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

DisplayGridInfo Member List

This is the complete list of members for **DisplayGridInfo**, including all inherited members.

cols	DisplayGridInfo
cscFilePath	DisplayGridInfo
displayIds	DisplayGridInfo
gammaRampFilePath	DisplayGridInfo
id	DisplayGridInfo
info()	DisplayGridInfo
primaryId	DisplayGridInfo
rows	DisplayGridInfo
unames	DisplayGridInfo
verClass	DisplayGridInfo





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

DisplayManager Member List

This is the complete list of members for **DisplayManager**, including all inherited members.

createClone([in, ValueMap{"0","1"}, Values{"basic","smart"}] uint32 type,[createCustomTiming([in]string timing) createDisplayGridById([in]uint32 cols,[in]uint32 rows,[in]uint32 displayId createDisplayGridByName([in]uint32 cols,[in]uint32 rows,[in]string display createDisplayGridByRef([in]uint32 cols,[in]uint32 rows,[in]string displayF createDisplayGridS([in]string grids[]) createModeFilter([in]uint32 displays[],[in]string filter) deleteCustomTiming([in]string timing) deleteModeFilter([in]uint32 displays[]) editCustomTiming([in]uint32 modeId,[in]string newTiming) enumCustomTimings([in]uint32 displays[]) fakeEDIDAII([in]string filePath,[in, ValueMap{"-1","0","1","2","3","4","5","6" Values{"unknown","unintialized","VGA","Component","S-Video","HDMI","E

info()

loadCustomTimings([in]uint32 displays[],[in]string filePath)
saveCustomTimings([in]uint32 displays[],[in]string filePath)
setGridPositions([in]uint32 positionCol[],[in]uint32 positionRow[])
setScalingAll([in, ValueMap{"0","1","2","3","5","6","7"},
Values{"Default","Closest","Native","ScanoutToNative","AspectScanoutTol
uint32 scaling)
tryCustomTiming([in]string timing)
validateDisplayGridById([in]uint32 cols,[in]uint32 rows,[in]uint32 display
validateDisplayGridS([in]string grids[])
verClass

NVIDIA





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

DisplayMode Member List

This is the complete list of members for **DisplayMode**, including all inherited members.

colorDepth	DisplayMode
height	DisplayMode
id	DisplayMode
info()	DisplayMode
refreshRate	DisplayMode
verClass	DisplayMode
width	DisplayMode





- <u>Main Page</u>
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

DisplayProfile Member List

This is the complete list of members for **DisplayProfile**, including all inherited members.

grid **DisplayProfile DisplayProfile** id **DisplayProfile** info() **DisplayProfile isPredefined DisplayProfile isSupported DisplayProfile** name DisplayProfile restoreSettings([in]uint32 settingIds[]) setBinaryValueById([in]uint32 settingId,[in]uint8 value[]) DisplayProfile **setStringValueById**([in]uint32 settingId,[in]string value) **DisplayProfile DisplayProfile** settings setValueById([in]uint32 settingId,[in]uint32 value) **DisplayProfile DisplayProfile** type

NVIDIA





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

Ecc Member List

This is the complete list of members for **Ecc**, including all inherited members.

aggregateDoubleBitErrors	Ecc
aggregateSingleBitErrors	Ecc
currentDoubleBitErrors	Ecc
currentSingleBitErrors	Ecc
id	Ecc
info()	Ecc
isEnabled	Ecc
isEnabledByDefault	Ecc
isSupported	Ecc
isWritable	Ecc
option	Ecc
resetCounters ([in]boolean bResetCurrent,[in]boolean bResetAggregate)	Ecc

setConfiguration([in]boolean bEnable,[in]booleanEccbEnableImmediately)EccverClassEcc

NVIDIA





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

EncoderSessions Member List

This is the complete list of members for **EncoderSessions**, including all inherited members.

averageEncodeFps	EncoderSessions
averageEncodeLatency	EncoderSessions
codecType	EncoderSessions
hResolution	EncoderSessions
id	EncoderSessions
info()	EncoderSessions
processId	EncoderSessions
sessionId	EncoderSessions
sessionsCount	EncoderSessions
verClass	EncoderSessions
vgpulnstance	EncoderSessions
vResolution	EncoderSessions





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

Gpu Member List

This is the complete list of members for **Gpu**, including all inherited members.

archid	Gpu
archName	Gpu
coolers	Gpu
coreCount	Gpu
count	Gpu
createUtilizationEvent ([in]sint32 lower,[in]sint32 upper,[in, ValueMap{"-1","0","1","2","3"}, Values{"All","GPU","Frame Buffer","Video Engine","PCIe bus"}] sint32 domain)	Gpu
deleteUtilizationEvents([in, ValueMap{"-1","0","1","2","3"}, Values{"All","GPU","Frame Buffer","Video Engine","PCIe bus"}] sint32 domain)	Gpu
deviceInfo disableLicensedFeature([in]string feature)	Gpu Gpu

ecc enableLicensedFeature([in]string feature) encoderSessionsInfo fakeEDID([in]string filePath,[in,	Gpu Gpu Gpu
ValueMap{"-1","0","1","2","3","4","5","6","7","8","9"}, Values{"unknown","unintialized","VGA","Component","S- Video","HDMI","DVI","LVDS","DP","Composite","All"}] uint32 output)	Gpu
fakeEDIDOnPort([in]string filePath,[in]uint32 portIndex,[in, ValueMap{"-1","0","1","2","3","4","5","6","7","8"}, Values{"unknown","unintialized","VGA","Component","S- Video","HDMI","DVI","LVDS","DP","Composite"}] uint32 output)	Gpu
getAllUtilizationEvents([in, ValueMap{"-1","0","1","2","3"}, Values{"All","GPU","Frame Buffer","Video Engine","PCIe bus"}] sint32 domain)	Gpu
gpuCoreClockCurrent	Gpu
handle	Gpu
id	Gpu
info()	Gpu
licensableFeatures	Gpu
licensableStatus	Gpu
memoryBusWidth	Gpu
memoryClockCurrent	Gpu
memorySizeAvailable	Gpu
memorySizePhysical	Gpu
memorySizeVirtual	Gpu
memoryТуре	Gpu
name	Gpu
nvapild	Gpu
ordinal	Gpu
pcieDownstreamWidth	Gpu
pcieGpu	Gpu
percentGpuMemoryUsage	Gpu

percentGpuUsage	Gpu
power	Gpu
powerSampleCount	Gpu
powerSamplingPeriod	Gpu
processUtilization	Gpu
productName	Gpu
productType	Gpu
thermalProbes	Gpu
uname	Gpu
ver	Gpu
verVBIOS	Gpu
videoCodec	Gpu





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

NamedObject Member List

This is the complete list of members for **NamedObject**, including all inherited members.

count NamedObject id NamedObject name NamedObject ordinal NamedObject uname NamedObject ver NamedObject





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- Class Index
- <u>Class Members</u>

OverlapLimits Member List

This is the complete list of members for **OverlapLimits**, including all inherited members.

id	OverlapLimits
info()	OverlapLimits
maxOverlapCol	OverlapLimits
maxOverlapRow	OverlapLimits
maxTotalCol	OverlapLimits
maxTotalRow	OverlapLimits
minOverlapCol	OverlapLimits
minOverlapRow	OverlapLimits
minTotalCol	OverlapLimits
minTotalRow	OverlapLimits
verClass	OverlapLimits





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

PcieLink Member List

This is the complete list of members for **PcieLink**, including all inherited members.

curGen	PcieLink
curSpeed	PcieLink
curWidth	PcieLink
info()	PcieLink
maxGen	PcieLink
maxSpeed	PcieLink
maxWidth	PcieLink
verClass	PcieLink





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

ProcessUtilization Member List

This is the complete list of members for **ProcessUtilization**, including all inherited members.

decUtilization	ProcessUtilization
encUtilization	ProcessUtilization
id	ProcessUtilization
info()	ProcessUtilization
memUtilization	ProcessUtilization
pid	ProcessUtilization
processCount	ProcessUtilization
smUtilization	ProcessUtilization
timeStamp	ProcessUtilization
verClass	ProcessUtilization





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

Profile Member List

This is the complete list of members for **Profile**, including all inherited members.

id	Profile
info()	Profile
isPredefined	Profile
isSupported	Profile
name	Profile
<pre>restoreSettings([in]uint32 settingIds[])</pre>	Profile
<pre>setBinaryValueById([in]uint32 settingId,[in]uint8 value[])</pre>	Profile
<pre>setStringValueById([in]uint32 settingId,[in]string value)</pre>	Profile
settings	Profile
<pre>setValueById([in]uint32 settingId,[in]uint32 value)</pre>	Profile
type	Profile
verClass	Profile





Feb 2018

- <u>Main Page</u>
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

ProfileManager Member List

This is the complete list of members for **ProfileManager**, including all inherited members.

applyDisplayProfiles([in]string prefix) ProfileManager applySystemProfile([in]string name) **ProfileManager** createProfile([in]string name,[in]uint32 type,[in]string ProfileManager params) currentProfile3D **ProfileManager** currentProfileDesktop ProfileManager currentSystemProfile ProfileManager defaultProfile3D ProfileManager defaultProfileDesktop ProfileManager defaultSystemProfile ProfileManager deleteDesktopProfile([in]string name) **ProfileManager** getAllProfiles([in, ValueMap{"0","1","3","4","5"}, Values{"3D Application Profile","3D Global ProfileManager

Profile", "Niew Profile", "Display Profile", "System Profile"}] uint32 type) info() **ProfileManager** loadDesktopProfile([in]string name) ProfileManager lockDesktopProfile([in]string name,[in]uint32 lock) **ProfileManager** restoreDefaults3D() ProfileManager restoreProfile([in]string name) ProfileManager saveDesktopProfile([in]string name) ProfileManager saveDisplayProfiles([in]string prefix) **ProfileManager** saveSystemProfile([in]string name) **ProfileManager** setCurrentProfile3D([in]string name) **ProfileManager** setVSync([in, ValueMap{"0","1","2","3","4"}, Values{"Passive","Off","On","Adaptive","Adaptive half"}] ProfileManager uint32 vsyncMode)

verClass

ProfileManager

NVIDIA





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

Setting Member List

This is the complete list of members for **Setting**, including all inherited members.

getStringValue()	Setting
id	Setting
info()	Setting
type	Setting
value	Setting
verClass	Setting





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

SettingInfo Member List

This is the complete list of members for **SettingInfo**, including all inherited members.

descriptionSettingInfoidSettingInfoinfo()SettingInfonameSettingInfoverClassSettingInfo





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

SettingTable Member List

This is the complete list of members for **SettingTable**, including all inherited members.

getIdFromName([in]string name)	SettingTable
getNameFromId([in]uint32 id)	SettingTable
id	SettingTable
info()	SettingTable
infoById([in]uint32 id)	SettingTable
<pre>infoByName([in]string name)</pre>	SettingTable
settingIds	SettingTable
settingNames	SettingTable
settings	SettingTable
type	SettingTable
verClass	SettingTable





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

Sync Member List

This is the complete list of members for **Sync**, including all inherited members.

flStatus Sync	С
id Syne	r
info() Syne	
interlaceMode Sync	С
interval Sync	С
isHouseSync Sync	С
isStereoSynced Sync	С
isSynced Sync	С
name Syne	С
nvapild Syne	С
ordinal Syne	С
polarity Sync	С

estintaria e Mede (fin.) (alu e Man ("O" "1")	
<pre>setInterlaceMode([in, ValueMap{"0","1"}, Values{"Disabled","Enabled"}] uint32 interlaceMode)</pre>	Sync
setInterval([in]uint32 interval)	Sync
<pre>setPolarity([in, ValueMap{"0","1","2"}, Values{"Rising Edge","Falling Edge","Both Edges"}] uint32 polarity)</pre>	Sync
<pre>setStartupDelay([in]uint32 numOfPixels,[in]uint32 numOfLines)</pre>	Sync
<pre>setSyncSkew([in]uint32 numOfPixels,[in]uint32 numOfLines)</pre>	Sync
<pre>setSyncStateById([in]uint32 syncDisplayIds[],[in]uint32 syncState[])</pre>	Sync
<pre>setSyncStateByName([in]string syncDisplayNames,[in]uint32 syncState[])</pre>	Sync
<pre>setVmode([in, ValueMap{"0","1","2","3","4"}, Values{"none","TTL","NTSC-PAL-SECAM","HDTV","composite"}] uint32 vmode)</pre>	Sync
source	Sync
startupDelay	Sync
syncDisplays	Sync
syncSignalRate	Sync
syncSkew	Sync
toggleSource()	Sync
uname	Sync
ver	Sync
verSyncFirmware	Sync
vmode	Sync

NVIDIA





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

SyncDelay Member List

This is the complete list of members for **SyncDelay**, including all inherited members.

delayType	SyncDelay
id	SyncDelay
info()	SyncDelay
maxLines	SyncDelay
minPixels	SyncDelay
numOfLines	SyncDelay
numOfPixels	SyncDelay
verClass	SyncDelay





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

SyncEvent Member List

This is the complete list of members for **SyncEvent**, including all inherited members.

eventType SyncEvent syncDeviceId SyncEvent verClass SyncEvent





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

SyncTopology Member List

This is the complete list of members for **SyncTopology**, including all inherited members.

count	SyncTopology
displaySyncState	SyncTopology
id	SyncTopology
info()	SyncTopology
isDisplayMasterable	SyncTopology
isGPUSynced()	SyncTopology
name	SyncTopology
ordinal	SyncTopology
uname	SyncTopology
ver	SyncTopology





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

System Member List

This is the complete list of members for **System**, including all inherited members.

disableEncoderSessionsPerfCounter()	System
disableProcessUtilizationPerfCounter()	System
enableEncoderSessionsPerfCounter()	System
enableProcessUtilizationPerfCounter()	System
info()	System
licensingPort	System
licensingServer	System
logFilter	System
logOptions	System
logTypes	System
machineType	System
nViewState	System
<pre>setLicensingServer([in]string address,[in]uint16 port)</pre>	System

<pre>setLogState([in, BitMap{"0","1","2"}, BitValues{"debugger","file","eventLog"}] uint32 types,[in]uint32 filter,[in]uint32 options)</pre>	System
<pre>setnViewState([in, ValueMap{"0","1"}, Values{"Disable","Enable"}] uint32 state)</pre>	System
unsetLicensingServer()	System
verClass	System
verDisplayDriver	System
vernViewDesktopManager	System
verNVWMI	System
verSDICaptureFirmware	System
verSDIOutputFirmware	System
verVBIOS	System





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

ThermalEvent Member List

This is the complete list of members for **ThermalEvent**, including all inherited members.

handleGpu ThermalEvent probeIndex ThermalEvent thermalLevel ThermalEvent





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

ThermalProbe Member List

This is the complete list of members for **ThermalProbe**, including all inherited members.

defaultMaxTemperature ThermalProbe		
defaultMinTemperature	ThermalProbe	
handle	ThermalProbe	
id	ThermalProbe	
info()	ThermalProbe	
temperature	ThermalProbe	
thermalLevel	ThermalProbe	
type	ThermalProbe	
verClass	ThermalProbe	

NVIDIA





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>

Version Member List

This is the complete list of members for **Version**, including all inherited members.

orderedValue Version strValue Version value Version





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- <u>Class Index</u>
- <u>Class Members</u>

VideoCodec Member List

This is the complete list of members for **VideoCodec**, including all inherited members.

averageFps	VideoCodec
averageLatency	VideoCodec
decoderSamplingPeriod	VideoCodec
encoderSamplingPeriod	VideoCodec
encoderSessionsCount	VideoCodec
info()	VideoCodec
percentDecoderUsage	VideoCodec
percentEncoderUsage	VideoCodec
verClass	VideoCodec





Feb 2018

- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- j
- •
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>
- ⊻

- a -

- addApplications() : **ApplicationProfile**
- applyDisplayProfiles() : ProfileManager
- applySystemProfile() : ProfileManager

- C -

- createClone() : DisplayManager
- createCustomTiming() : DisplayManager
- createDesktop() : DesktopManager
- createDisplayGridById() : DisplayManager
- createDisplayGridByName() : **DisplayManager**
- createDisplayGridByRef() : DisplayManager
- createDisplayGrids() : DisplayManager
- createModeFilter() : DisplayManager
- createProfile() : ProfileManager
- createUtilizationEvent() : Gpu

- d -

- deleteCustomTiming() : DisplayManager
- deleteDesktop() : DesktopManager
- deleteDesktopProfile() : ProfileManager
- deleteModeFilter() : DisplayManager
- deleteUtilizationEvents() : Gpu
- disableEncoderSessionsPerfCounter() : System
- disableLicensedFeature() : Gpu
- disableProcessUtilizationPerfCounter() : System

- e -

- editCustomTiming() : DisplayManager
- editDesktop() : DesktopManager
- enableEncoderSessionsPerfCounter() : System
- enableLicensedFeature() : Gpu
- enableProcessUtilizationPerfCounter() : System
- enumCustomTimings() : DisplayManager
- enumModeFilters() : DisplayManager

- f -

- fakeEDID() : Gpu
- fakeEDIDAII() : DisplayManager
- fakeEDIDOnPort() : Gpu

- g -

- getAllDesktops() : DesktopManager
- getAllProfiles() : ProfileManager
- getAllUtilizationEvents() : Gpu
- getCurrentTiming() : Display
- getIdFromName() : SettingTable
- getNameFromId() : SettingTable
- getStringValue() : Setting

- i -

- info(): PcieLink, VideoCodec, System, DisplayGrid, SettingTable, SettingInfo, DisplayProfile, ApplicationProfile, Profile, DisplayGridInfo, Application, Setting, ProfileManager, Sync, EncoderSessions, SyncDelay, SyncTopology, ProcessUtilization, Ecc, ThermalProbe, Cooler, DisplayMode, Display, Board, OverlapLimits, DisplayManager, Gpu
- infoById() : SettingTable
- infoByName() : SettingTable
- isGPUSynced() : SyncTopology

- | -

- loadCustomTimings() : DisplayManager
- loadDesktopProfile() : ProfileManager
- lockDesktopProfile() : ProfileManager

- r -

removeApplications() : ApplicationProfile

- resetCounters() : Ecc
- restoreDefaults3D() : ProfileManager
- restoreNativeDisplayMode() : Display
- restoreProfile() : ProfileManager
- restoreSettings(): DisplayProfile , ApplicationProfile , Profile

- S -

- saveCSC() : DisplayGrid , Display
- saveCustomTimings() : DisplayManager
- saveDesktopProfile() : ProfileManager
- saveDisplayProfiles() : ProfileManager
- saveEDID() : Display
- saveGammaRamp() : Display , DisplayGrid
- saveSystemProfile() : ProfileManager
- setBinaryValueById() : Profile , ApplicationProfile , DisplayProfile
- setConfiguration() : Ecc
- setCSC() : DisplayGrid , Display
- setCurrentProfile3D() : ProfileManager
- setDisplayMode() : DisplayGrid , Display
- setDisplayModeById() : DisplayGrid , Display
- setDisplayModeByRef() : DisplayGrid , Display
- setDither() : Display
- setEDID() : Display
- setGammaRamp() : DisplayGrid , Display
- setGammaRampBasic() : DisplayGrid , Display
- setGridPositions() : DisplayManager
- setInterlaceMode() : Sync
- setInterval() : Sync
- setLicensingServer() : System
- setLogState() : System
- setnViewState() : System
- setOverlapCol() : DisplayGrid
- setOverlapRow() : DisplayGrid
- setOverlaps() : DisplayGrid
- setPolarity() : Sync
- setRotation() : DisplayGrid , Display

- setScaling() : **Display**
- setScalingAll() : DisplayManager
- setStartupDelay() : Sync
- setStringValueById() : ApplicationProfile , Profile , DisplayProfile
- setSyncSkew() : Sync
- setSyncStateById() : Sync
- setSyncStateByName() : Sync
- setValueById() : ApplicationProfile , Profile , DisplayProfile
- setVmode() : Sync
- setVSync() : ProfileManager

- t -

- toggleSource() : Sync
- tryCustomTiming() : DisplayManager

- u -

• unsetLicensingServer() : System

- V -

- validateDisplayGridById() : DisplayManager
- validateDisplayGrids() : **DisplayManager**

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- a -

- aggregateDoubleBitErrors : Ecc
- aggregateSingleBitErrors : Ecc
- applications : ApplicationProfile
- archld : Gpu
- archName : Gpu
- averageEncodeFps : EncoderSessions
- averageEncodeLatency : EncoderSessions
- averageFps : VideoCodec
- averageLatency : VideoCodec

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- C -

- chipSKU : Board
- chipSKUMod : Board
- codecType : EncoderSessions
- colorDepth : DisplayMode
- cols : DisplayGrid , DisplayGridInfo
- coolerIndex : CoolerEvent
- coolerLevel : Cooler , CoolerEvent
- coolers : Gpu , Board
- coolerType : Cooler
- coreCount : Gpu
- count : Board , DisplayGrid , Display , SyncTopology , Sync , NamedObject , Gpu
- createClone() : DisplayManager
- createCustomTiming() : DisplayManager
- createDesktop() : DesktopManager
- createDisplayGridById() : DisplayManager
- createDisplayGridByName() : DisplayManager
- createDisplayGridByRef() : DisplayManager
- createDisplayGrids() : DisplayManager
- createModeFilter() : DisplayManager
- createProfile() : ProfileManager
- createUtilizationEvent() : Gpu
- cscFilePath : DisplayGridInfo
- curGen : PcieLink
- currentDoubleBitErrors : Ecc
- currentProfile3D : ProfileManager
- currentProfileDesktop : ProfileManager
- currentSingleBitErrors : Ecc
- currentSystemProfile : ProfileManager
- curSpeed : PcieLink

• curWidth : PcieLink

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>w</u>

- d -

- decoderSamplingPeriod : VideoCodec
- decUtilization : ProcessUtilization
- defaultMaxTemperature : ThermalProbe
- defaultMinTemperature : ThermalProbe
- defaultProfile3D : ProfileManager
- defaultProfileDesktop : ProfileManager
- defaultSystemProfile : ProfileManager
- delayType : SyncDelay
- deleteCustomTiming() : DisplayManager
- deleteDesktop() : DesktopManager
- deleteDesktopProfile() : ProfileManager
- deleteModeFilter() : DisplayManager
- deleteUtilizationEvents() : Gpu
- description : **SettingInfo**
- deviceInfo : Gpu
- disableEncoderSessionsPerfCounter() : System
- disableLicensedFeature() : Gpu
- disableProcessUtilizationPerfCounter() : System
- displayConnectorType : Display
- displayIds : **DisplayGridInfo**
- displayModeNative : Display
- displayModePhysical : DisplayGrid
- displayModes : DisplayGrid , Display
- displayModeVirtual : DisplayGrid
- displays : **DisplayGrid**
- displaySyncState : SyncTopology
- ditherBits : Display
- ditherMode : Display
- ditherState : Display





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>w</u>

- e -

- ecc : Gpu
- EDID : Display
- EDIDSize : Display
- editCustomTiming() : DisplayManager
- editDesktop() : **DesktopManager**
- enableEncoderSessionsPerfCounter() : System
- enableLicensedFeature() : Gpu
- enableProcessUtilizationPerfCounter() : System
- encoderSamplingPeriod : VideoCodec
- encoderSessionsCount : VideoCodec
- encoderSessionsInfo : Gpu
- encUtilization : ProcessUtilization
- enumCustomTimings() : DisplayManager
- enumModeFilters() : DisplayManager
- eventType : SyncEvent

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>W</u>

- f -

- fakeEDID() : Gpu
- fakeEDIDAII() : DisplayManager
- fakeEDIDOnPort() : Gpu
- fanSpeed : Cooler
- flStatus : Sync





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- g -

- gammaRampFilePath : DisplayGridInfo
- getAllDesktops() : DesktopManager
- getAllProfiles() : ProfileManager
- getAllUtilizationEvents() : Gpu
- getCurrentTiming() : Display
- getIdFromName() : SettingTable
- getNameFromId() : SettingTable
- getStringValue() : Setting
- gpuConnectorType : Display
- gpuCoreClockCurrent : Gpu
- gpus : Board , DisplayGrid
- grid : DisplayProfile

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>W</u>

- h -

- handle : Gpu , Cooler , ThermalProbe
- handleGpu : CoolerEvent , ThermalEvent
- height : **DisplayMode**
- hResolution : EncoderSessions

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>w</u>

- i -

- id : NamedObject, Gpu, OverlapLimits, Cooler, Setting, Application, ThermalProbe, DisplayGridInfo, Profile, DisplayGrid, Ecc, ApplicationProfile, DisplayProfile, ProcessUtilization, SettingInfo, SettingTable, Board, Display, SyncTopology, EncoderSessions, DisplayMode, SyncDelay, Sync
- info(): SettingTable , Display , DisplayMode , ApplicationProfile , Cooler , ThermalProbe , Ecc , Setting , ProcessUtilization , SyncTopology , EncoderSessions , SyncDelay , Sync , ProfileManager , Application , DisplayGridInfo , Profile , PcieLink , DisplayProfile , DisplayGrid , SettingInfo , VideoCodec , Gpu , System , Board , DisplayManager , OverlapLimits
- infoById() : SettingTable
- infoByName() : SettingTable
- interlaceMode : Sync
- interval : Sync
- isActive : Display
- isDisplayMasterable : SyncTopology
- isEnabled : Ecc
- isEnabledByDefault : Ecc
- isGPUSynced() : **SyncTopology**
- isHouseSync : Sync
- isPredefined : Profile , DisplayProfile , ApplicationProfile , Application
- isStereoSynced : Sync
- isSupported : ApplicationProfile , DisplayProfile , Profile , Application , Ecc
- isSynced : Sync
- isWritable : Ecc





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- <u>Class List</u>
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>w</u>

- | -

- launcher : Application
- licensableFeatures : Gpu
- licensableStatus : Gpu
- licensingPort : System
- licensingServer : System
- loadCustomTimings() : DisplayManager
- loadDesktopProfile() : ProfileManager
- lockDesktopProfile() : ProfileManager
- locus : Display
- logFilter : System
- logOptions : System
- logTypes : System

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>w</u>

- m -

- machineType : System
- make : Display
- maxGen : PcieLink
- maxLines : SyncDelay
- maxOverlapCol : OverlapLimits
- maxOverlapRow : OverlapLimits
- maxSpeed : Cooler , PcieLink
- maxTotalCol : OverlapLimits
- maxTotalRow : OverlapLimits
- maxWidth : PcieLink
- memoryBusWidth : Gpu
- memoryClockCurrent : Gpu
- memorySizeAvailable : Gpu
- memorySizePhysical : Gpu
- memorySizeVirtual : Gpu
- memoryType : Gpu
- memUtilization : **ProcessUtilization**
- minOverlapCol : OverlapLimits
- minOverlapRow : OverlapLimits
- minPixels : SyncDelay
- minSpeed : Cooler
- minTotalCol : OverlapLimits
- minTotalRow : OverlapLimits
- model : Display

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>W</u>

- n -

- name : NamedObject , Gpu , DisplayGrid , Application , Profile , Display , ApplicationProfile , DisplayProfile , SettingInfo , Board , SyncTopology , Sync
- numOfLines : SyncDelay
- numOfPixels : SyncDelay
- nvapild : Gpu , Board , Display , Sync
- nViewState : System

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>W</u>

- 0 -

- option : Ecc
- orderedValue : Version
- ordinal : Gpu , SyncTopology , Sync , Board , Display , DisplayGrid , NamedObject
- overlapCols : DisplayGrid
- overlapLimits : DisplayGrid
- overlapRows : DisplayGrid

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- p -

- pcieDownstreamWidth : Gpu
- pcieGpu : Gpu
- percentCoolerRate : Cooler
- percentDecoderUsage : VideoCodec
- percentEncoderUsage : VideoCodec
- percentGpuMemoryUsage : Gpu
- percentGpuUsage : **Gpu**
- pid : ProcessUtilization
- polarity : Sync
- positionCol : DisplayGrid
- positionRow : DisplayGrid
- power : Gpu
- powerSampleCount : Gpu
- powerSamplingPeriod : Gpu
- primaryId : DisplayGridInfo
- probeIndex : ThermalEvent
- processCount : ProcessUtilization
- processId : EncoderSessions
- processUtilization : Gpu
- productName : Gpu
- productType : Gpu
- project : Board
- projectSKU : Board

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>w</u>

- r -

- refreshRate : DisplayMode
- removeApplications() : ApplicationProfile
- resetCounters() : Ecc
- restoreDefaults3D() : ProfileManager
- restoreNativeDisplayMode() : Display
- restoreProfile() : **ProfileManager**
- restoreSettings() : ApplicationProfile , DisplayProfile , Profile
- rotation : Display , DisplayGrid
- rows : DisplayGrid , DisplayGridInfo

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>W</u>

- S -

- saveCSC() : DisplayGrid , Display
- saveCustomTimings() : DisplayManager
- saveDesktopProfile() : ProfileManager
- saveDisplayProfiles() : ProfileManager
- saveEDID() : Display
- saveGammaRamp() : Display , DisplayGrid
- saveSystemProfile() : ProfileManager
- scaling : Display
- serialNumber : Board
- sessionId : EncoderSessions
- sessionsCount : EncoderSessions
- setBinaryValueById() : DisplayProfile , Profile , ApplicationProfile
- setConfiguration() : Ecc
- setCSC() : DisplayGrid , Display
- setCurrentProfile3D() : ProfileManager
- setDisplayMode() : DisplayGrid , Display
- setDisplayModeById() : DisplayGrid , Display
- setDisplayModeByRef() : DisplayGrid , Display
- setDither() : Display
- setEDID() : Display
- setGammaRamp() : DisplayGrid , Display
- setGammaRampBasic() : Display , DisplayGrid
- setGridPositions() : DisplayManager
- setInterlaceMode() : Sync
- setInterval() : Sync
- setLicensingServer() : System
- setLogState() : System
- setnViewState() : System
- setOverlapCol() : DisplayGrid
- setOverlapRow() : DisplayGrid

- setOverlaps() : DisplayGrid
- setPolarity() : Sync
- setRotation() : DisplayGrid , Display
- setScaling() : Display
- setScalingAll() : DisplayManager
- setStartupDelay() : Sync
- setStringValueById() : Profile , ApplicationProfile , DisplayProfile
- setSyncSkew() : Sync
- setSyncStateById() : Sync
- setSyncStateByName() : Sync
- settingIds : **SettingTable**
- settingNames : SettingTable
- settings : SettingTable , ApplicationProfile , Profile , DisplayProfile
- setValueById() : ApplicationProfile , DisplayProfile , Profile
- setVmode() : Sync
- setVSync() : ProfileManager
- smUtilization : ProcessUtilization
- source : Sync
- startTime : ApplicationProfile
- startupDelay : Sync
- strValue : Version
- subPaths : Application
- syncDeviceId : SyncEvent
- syncDisplays : Sync
- syncSignalRate : Sync
- syncSkew : Sync

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- t -

- temperature : ThermalProbe
- thermalLevel : ThermalProbe , ThermalEvent
- thermalProbes : Gpu , Board
- timeStamp : ProcessUtilization
- toggleSource() : Sync
- tryCustomTiming() : **DisplayManager**
- type : Setting , DisplayProfile , ApplicationProfile , Profile , ThermalProbe , SettingTable

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>w</u>

- u -

- uname : NamedObject , Gpu , DisplayGrid , Sync , SyncTopology , Display , Board
- unames : DisplayGridInfo
- unsetLicensingServer() : System





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>W</u>

- V -

- validateDisplayGridById() : DisplayManager
- validateDisplayGrids() : DisplayManager
- value : Setting , Version
- ver : DisplayGrid , Display , NamedObject , SyncTopology , Sync , Gpu , Board
- verClass : DisplayManager , SettingInfo , SettingTable , OverlapLimits , SyncEvent , DisplayMode , DisplayProfile , ApplicationProfile , Profile , Cooler , Application , Setting , ThermalProbe , Ecc , ProcessUtilization , EncoderSessions , PcieLink , SyncDelay , ProfileManager , VideoCodec , System , DisplayGridInfo , DesktopManager
- verDisplayDriver : System
- verFirmware : **Display**
- vernViewDesktopManager : System
- verNVWMI : System
- verSDICaptureFirmware : System
- verSDIOutputFirmware : System
- verSyncFirmware : Sync
- verVBIOS : Gpu , System
- vgpulnstance : EncoderSessions
- videoCodec : Gpu
- vmode : Sync
- vResolution : EncoderSessions

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- W -

• width : **DisplayMode**





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- C -

- chipSKU : Board
- chipSKUMod : Board
- codecType : EncoderSessions
- colorDepth : DisplayMode
- cols : DisplayGrid , DisplayGridInfo
- coolerIndex : CoolerEvent
- coolerLevel : Cooler , CoolerEvent
- coolers : Gpu , Board
- coolerType : Cooler
- coreCount : Gpu
- count : Board , DisplayGrid , Gpu , Display , Sync , SyncTopology , NamedObject
- cscFilePath : DisplayGridInfo
- curGen : PcieLink
- currentDoubleBitErrors : Ecc
- currentProfile3D : ProfileManager
- currentProfileDesktop : ProfileManager
- currentSingleBitErrors : Ecc
- currentSystemProfile : ProfileManager
- curSpeed : PcieLink
- curWidth : PcieLink

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- d -

- decoderSamplingPeriod : VideoCodec
- decUtilization : ProcessUtilization
- defaultMaxTemperature : ThermalProbe
- defaultMinTemperature : ThermalProbe
- defaultProfile3D : ProfileManager
- defaultProfileDesktop : ProfileManager
- defaultSystemProfile : ProfileManager
- delayType : SyncDelay
- description : SettingInfo
- deviceInfo : Gpu
- displayConnectorType : Display
- displayIds : DisplayGridInfo
- displayModeNative : Display
- displayModePhysical : **DisplayGrid**
- displayModes : Display , DisplayGrid
- displayModeVirtual : DisplayGrid
- displays : DisplayGrid
- displaySyncState : SyncTopology
- ditherBits : Display
- ditherMode : Display
- ditherState : Display

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>W</u>

- e -

- ecc : Gpu
- EDID : Display
- EDIDSize : Display
- encoderSamplingPeriod : VideoCodec
- encoderSessionsCount : VideoCodec
- encoderSessionsInfo : Gpu
- encUtilization : ProcessUtilization
- eventType : SyncEvent

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>V</u>
- <u>W</u>

- f -

- fanSpeed : Cooler
- flStatus : Sync

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- g -

- gammaRampFilePath : DisplayGridInfo
- gpuConnectorType : Display
- gpuCoreClockCurrent : Gpu
- gpus : Board , DisplayGrid
- grid : DisplayProfile

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- h -

- handle : Gpu , Cooler , ThermalProbe
- handleGpu : CoolerEvent , ThermalEvent
- height : **DisplayMode**
- hResolution : EncoderSessions

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>V</u>
- <u>W</u>

- i -

- id : NamedObject, Gpu, OverlapLimits, Cooler, Setting, Application, ThermalProbe, DisplayGridInfo, Profile, DisplayGrid, Ecc, ApplicationProfile, DisplayProfile, ProcessUtilization, SettingInfo, SettingTable, Board, Display, SyncTopology, EncoderSessions, DisplayMode, SyncDelay, Sync
- interlaceMode : Sync
- interval : Sync
- isActive : **Display**
- isDisplayMasterable : SyncTopology
- isEnabled : Ecc
- isEnabledByDefault : Ecc
- isHouseSync : Sync
- isPredefined : Profile , ApplicationProfile , DisplayProfile , Application
- isStereoSynced : Sync
- isSupported : Application , Ecc , DisplayProfile , ApplicationProfile , Profile
- isSynced : Sync
- isWritable : Ecc

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- Class Index
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- | -

- launcher : Application
- licensableFeatures : Gpu
- licensableStatus : Gpu
- licensingPort : System
- licensingServer : System
- locus : Display
- logFilter : System
- logOptions : System
- logTypes : System

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- m -

- machineType : System
- make : Display
- maxGen : PcieLink
- maxLines : SyncDelay
- maxOverlapCol : OverlapLimits
- maxOverlapRow : OverlapLimits
- maxSpeed : Cooler , PcieLink
- maxTotalCol : OverlapLimits
- maxTotalRow : OverlapLimits
- maxWidth : PcieLink
- memoryBusWidth : Gpu
- memoryClockCurrent : Gpu
- memorySizeAvailable : Gpu
- memorySizePhysical : Gpu
- memorySizeVirtual : Gpu
- memoryType : Gpu
- memUtilization : ProcessUtilization
- minOverlapCol : OverlapLimits
- minOverlapRow : OverlapLimits
- minPixels : SyncDelay
- minSpeed : Cooler
- minTotalCol : OverlapLimits
- minTotalRow : OverlapLimits
- model : Display

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- n -

- name : NamedObject , Gpu , DisplayGrid , Application , Profile , Display , ApplicationProfile , DisplayProfile , SettingInfo , Board , SyncTopology , Sync
- numOfLines : SyncDelay
- numOfPixels : SyncDelay
- nvapild : Gpu , Board , Display , Sync
- nViewState : System

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- 0 -

- option : Ecc
- orderedValue : Version
- ordinal : Gpu , SyncTopology , Sync , Board , Display , DisplayGrid , NamedObject
- overlapCols : DisplayGrid
- overlapLimits : DisplayGrid
- overlapRows : DisplayGrid

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- p -

- pcieDownstreamWidth : Gpu
- pcieGpu : Gpu
- percentCoolerRate : Cooler
- percentDecoderUsage : VideoCodec
- percentEncoderUsage : VideoCodec
- percentGpuMemoryUsage : Gpu
- percentGpuUsage : **Gpu**
- pid : **ProcessUtilization**
- polarity : Sync
- positionCol : DisplayGrid
- positionRow : DisplayGrid
- power : Gpu
- powerSampleCount : Gpu
- powerSamplingPeriod : Gpu
- primaryId : DisplayGridInfo
- probeIndex : ThermalEvent
- processCount : ProcessUtilization
- processId : EncoderSessions
- processUtilization : Gpu
- productName : Gpu
- productType : Gpu
- project : Board
- projectSKU : Board

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- r -

- refreshRate : DisplayMode
- rotation : DisplayGrid , Display
- rows : DisplayGrid , DisplayGridInfo

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- <u>Functions</u>
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- [
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- ⊻
- <u>W</u>

- S -

- scaling : Display
- serialNumber : Board
- sessionId : EncoderSessions
- sessionsCount : EncoderSessions
- settingIds : SettingTable
- settingNames : SettingTable
- settings : ApplicationProfile , DisplayProfile , SettingTable , Profile
- smUtilization : ProcessUtilization
- source : Sync
- startTime : ApplicationProfile
- startupDelay : Sync
- strValue : Version
- subPaths : Application
- syncDeviceId : SyncEvent
- syncDisplays : Sync
- syncSignalRate : Sync
- syncSkew : Sync

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- Functions
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- •
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>W</u>

- t -

- temperature : ThermalProbe
- thermalLevel : ThermalProbe , ThermalEvent
- thermalProbes : Gpu , Board
- timeStamp : ProcessUtilization
- type : ThermalProbe , DisplayProfile , ApplicationProfile , SettingTable , Setting , Profile

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- Functions
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- •
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>W</u>

- u -

- uname : NamedObject , Gpu , Sync , SyncTopology , Display , DisplayGrid , Board
- unames : DisplayGridInfo





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- Functions
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- •
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>V</u>
- <u>W</u>

- V -

- value : Version , Setting
- ver : Gpu , SyncTopology , Sync , Board , NamedObject , DisplayGrid , Display
- verClass : DisplayMode , SyncEvent , Cooler , ThermalProbe , Ecc , PcieLink , SettingTable , SettingInfo , DisplayProfile , ProcessUtilization , Profile , DisplayGridInfo , EncoderSessions , VideoCodec , SyncDelay , ProfileManager , System , Setting , Application , DesktopManager , DisplayManager , ApplicationProfile , OverlapLimits
- verDisplayDriver : System
- verFirmware : **Display**
- vernViewDesktopManager : System
- verNVWMI : System
- verSDICaptureFirmware : System
- verSDIOutputFirmware : System
- verSyncFirmware : Sync
- verVBIOS : Gpu , System
- vgpulnstance : EncoderSessions
- videoCodec : Gpu
- vmode : Sync
- vResolution : EncoderSessions

NVIDIA





- Main Page
- <u>Related Pages</u>
- <u>Classes</u>
- Class List
- <u>Class Index</u>
- <u>Class Members</u>
- <u>All</u>
- Functions
- <u>Variables</u>
- <u>a</u>
- <u>C</u>
- <u>d</u>
- <u>e</u>
- <u>f</u>
- g
- <u>h</u>
- •
- •
- <u>m</u>
- <u>n</u>
- 0
- p
- <u>r</u>
- <u>S</u>
- <u>t</u>
- <u>U</u>

- <u>v</u>
- <u>W</u>

- W -

• width : DisplayMode

