

OpenCable Specifications OCAP Extensions

OCAP Device Settings Extension

OC-SP-OCAP-DS-EXT-I02-090930

ISSUED

Notice

This OpenCable specification is the result of a cooperative effort undertaken at the direction of Cable Television Laboratories, Inc. for the benefit of the cable industry and its customers. This document may contain references to other documents not owned or controlled by CableLabs. Use and understanding of this document may require access to such other documents. Designing, manufacturing, distributing, using, selling, or servicing products, or providing services, based on this document may require intellectual property licenses from third parties for technology referenced in this document.

Neither CableLabs nor any member company is responsible to any party for any liability of any nature whatsoever resulting from or arising out of use or reliance upon this document, or any document referenced herein. This document is furnished on an "AS IS" basis and neither CableLabs nor its members provides any representation or warranty, express or implied, regarding the accuracy, completeness, noninfringement, or fitness for a particular purpose of this document, or any document referenced herein.

© Copyright 2006-2009 Cable Television Laboratories, Inc.
All rights reserved.

Document Status Sheet

Document Control Number:	OC-SP-OCAP-DS-EXT-I02-090930			
Document Title:	OCAP Device Settings Extension			
Revision History:	I01 – Released 4/16/07			
	I02 – Released 9/30/09			
Date:	September 30, 2009			
Status:	Work in Progress	Draft	Issued	Closed
Distribution Restrictions:	Author Only	GL/Member	GL/Member/ Vendor	Public

Key to Document Status Codes:

- Work in Progress** An incomplete document, designed to guide discussion and generate feedback that may include several alternative requirements for consideration.
- Draft** A document in specification format considered largely complete, but lacking review by Members and vendors. Drafts are susceptible to substantial change during the review process.
- Issued** A stable document, which has undergone rigorous member and vendor review and is suitable for product design and development, cross-vendor interoperability, and for certification testing.
- Closed** A static document, reviewed, tested, validated, and closed to further engineering change requests to the specification through CableLabs.

Trademarks:

CableLabs[®], DOCSIS[®], EuroDOCSIS[™], eDOCSIS[™], M-CMTS[™], PacketCable[™], EuroPacketCable[™], PCMM[™], CableHome[®], CableOffice[™], OpenCable[™], OCAP[™], CableCARD[™], M-Card[™], and DCAS[™] are trademarks of Cable Television Laboratories, Inc.

Contents

1	SCOPE	1
1.1	Introduction and Purpose.....	1
1.2	Requirements.....	1
2	REFERENCES	2
2.1	Normative References.....	2
2.2	Informative References.....	2
2.3	Reference Acquisition.....	2
3	TERMS AND DEFINITIONS	3
4	ABBREVIATIONS AND ACRONYMS	4
5	OVERVIEW	5
5.1	Audio Settings.....	5
5.1.1	Audio Outputs.....	5
5.1.2	Audio Clips.....	5
5.2	Video Output Port Settings.....	5
5.2.1	Output Resolutions/Formats.....	5
5.2.2	Output Aspect Ratio.....	6
5.2.3	Main Video Output Port.....	6
5.2.4	Dynamic configuration of video output port setting.....	6
5.2.5	Output Port Status.....	6
5.2.6	Connected Display Information.....	7
5.3	Zoom Modes.....	7
5.4	Security.....	7
5.5	Persistence.....	8
5.6	Minimum Keycode Set.....	8
5.7	System Property.....	8
5.8	Registry of Constants.....	8
6	OCAP DS API	10
ANNEX A	DEVICE SETTINGS API	11
	Package org.ocap.hardware.device.....	11
	Package org.ocap.hardware.device Description.....	11
	org.ocap.hardware.device Class AudioOutputPort.....	12
	org.ocap.hardware.device Class DynamicVideoOutputConfiguration.....	21
	org.ocap.hardware.device Class FeatureNotSupportedException.....	25
	org.ocap.hardware.device Interface FixedVideoOutputConfiguration.....	27
	org.ocap.hardware.device Interface HostSettings.....	28
	org.ocap.hardware.device Class OCSound.....	33
	org.ocap.hardware.device Interface VideoOutputConfiguration.....	36
	org.ocap.hardware.device Interface VideoOutputPortListener.....	37
	org.ocap.hardware.device Interface VideoOutputSettings.....	39
	org.ocap.hardware.device Class VideoResolution.....	43
	org.ocap.hardware.device Interface VideoZoomPreference.....	46
APPENDIX I	REVISION HISTORY	47

Tables

Table 5-1 - Permissions Table Entry for Device Settings Extension.....7

1 SCOPE

1.1 Introduction and Purpose

This document defines a minimal profile for OCAP Host devices that support 'Device Settings' functionality. 'Device Settings' functionality is defined as a standardized software interface for allowing interoperable OCAP applications to change certain device-specific settings of the Host device. This platform is a modular extension to the OpenCable Application Platform ([OCAP]). The OCAP and OCAP DS Profiles were developed by Cable Television Laboratories, Inc. (CableLabs), in conjunction with representatives from a number of its member cable operating companies, as well as leading software firms.

The OCAP DS Profile is based on the OCAP 1.0 and subsequent profiles, and includes that document in its entirety.

The OCAP DS is an application profile that includes all required Application Program Interfaces (APIs), content and data formats, and protocols, up to the application level. Applications developed to the OCAP DS Profile will be executed on OpenCable-compliant Host devices. The OCAP DS Profile allows cable operators to deploy interoperable applications to manage certain device settings on OpenCable-compliant Host devices connected to their networks. This profile allows cable operators to have a single set of user interfaces for managing certain standard device parameters, such as master volume, across multiple Host device vendors.

The OCAP DS platform SHALL be applicable to a wide variety of hardware and operating systems to allow Consumer Electronics (CE) manufacturers flexibility in implementation. A primary objective in defining the OCAP DS is to enable competing implementations of the OCAP DS platform by CE manufacturers.

1.2 Requirements

Throughout this document, the words that are used to define the significance of particular requirements are capitalized. These words are:

"SHALL"	This word means that the item is an absolute requirement of this specification.
"SHALL NOT"	This phrase means that the item is an absolute prohibition of this specification.
"SHOULD"	This word means that there may exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before choosing a different course.
"SHOULD NOT"	This phrase means that there may exist valid reasons in particular circumstances when the listed behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
"MAY"	This word means that this item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because it enhances the product, for example; another vendor may omit the same item.

2 REFERENCES

2.1 Normative References

In order to claim compliance with this specification, it is necessary to conform to the following standards and other works as indicated, in addition to the other requirements of this specification. Notwithstanding, intellectual property rights may be required to use or implement such normative references.

[OCAP] OpenCable Application Platform Specification, Profile 1.1, OC-SP-OCAP1.1.2-090930, September 30, 2009, Cable Television Laboratories, Inc.

[HOST 2.1] OpenCable Host Device 2.1 Core Functional Requirements; OC-SP-HOST2.1-CFR-I09-090904, September 4, 2009, Cable Television Laboratories, Inc.

2.2 Informative References

None.

2.3 Reference Acquisition

Cable Television Laboratories, Inc., 858 Coal Creek Circle, Louisville, CO 80027;
Phone 303-661-9100; Fax 303-661-9199; Internet: <http://www.cablelabs.com/>

3 TERMS AND DEFINITIONS

This specification uses the following terms:

Device Settings	A set of device parameters that may be accessed or set using an OCAP application.
Video Output Port	One of the several video output ports defined by [OCAP] and represented by an the class <code>org.ocap.hardware.VideoOutputPort</code> .

4 ABBREVIATIONS AND ACRONYMS

This specification uses the following abbreviations:

DS	Device Setting
DVI	Digital Video Interface
E-EDID	Enhanced Extended Display Identification Data
HDMI	High-Definition Multimedia Interface
HD	High Definition
SD	Standard Definition

5 OVERVIEW

The Device Settings extension defines a set of APIs for accessing or modifying device parameters, such as master volume, HD settings, Audio output setting, etc. This functionality is typically controlled by device manufacturer applications. This extension allows OCAP applications to access or manage some of these parameters through a standard API, thus allowing a cable operator to deploy a consistent set of user-interfaces to display and manage these parameters.

5.1 Audio Settings

5.1.1 Audio Outputs

Host devices MAY support multiple audio outputs that could simultaneously output separate audio streams. For example, in case of Host devices that support multiple physical screens, each of these physical screens may present a separate audio stream. An "Audio Output" represents the control point for such an audio output port. A single Audio Output may be associated with multiple VideoOutputPorts, if the audio for those video output ports is shared. All applications SHALL have a default Audio Output defined as the AudioOutputPort associated with the main VideoOutputPort for the application's default HScreen.

Host devices supporting the OCAP DS Profile SHALL support at least one Audio Output. Each VideoOutputPort SHALL be connected to at least one Audio Output.

The master volume level for an Audio Output is defined as the logical volume level, changing of which would proportionally change the output volume of all audio streams connected to the Audio Output that may be presented through OCAP APIs. These audio streams include, but are not limited to, audio associated with the following: video presentations for OCAP services, components of OCAP services, JMF media players, or audio clips controlled through the HSound API. Where control over audio output attributes doesn't make sense (for example audio volume levels for a IEEE-1394 video output port or digital encoding settings for an analog output), such settings MAY be ignored.

5.1.2 Audio Clips

Audio outputs corresponding to audio clips, presented through the HSound API or the JMF Player API, SHALL be routed to the default Audio Output for the calling application, unless specified otherwise through an instance of the OCSound API. When the volume level for an instance of the OCSound is changed, the relative volume of the audio clip – relative to the master volume of the Audio Output – SHALL be changed, without affecting the volume of any other audio streams being presented on that Audio Output.

5.2 Video Output Port Settings

The Video Settings API allows an application to access or modify the resolution, format and aspect ratio of the output video for various video output ports of the Host device.

5.2.1 Output Resolutions/Formats

This specification adds API definitions for the output resolutions/formats for video output ports, as defined in [HOST 2.1]. As specified in [HOST 2.1], a host device may support different sets of output resolutions/formats on different video output ports. The VideoOutputSettings.getSupportedConfigurations method SHALL return an array that includes specifications of supported output resolutions for a given video output port. The host device SHALL also support converting any of the required HScreen resolutions to the currently configured video output resolution

and format. Any stretching/scaling/cropping/zooming performed on an HScreen to convert it to a supported video output resolution is implementation-dependant.

The set of supported resolutions SHALL be limited to the intersection of those supported by the video output port and the connected display device, where information about the connected display device is known. For example, where Enhanced Extended Display Identification Data (E-EDID) information is communicated from the display to the Host via an HDMI or DVI interface, the returned set of configurations will be the set supported by the video output port filtered to only include those also supported by the display. Where information about the connected display cannot be known (as would be the case with an analog component video output port), the set of supported configurations SHALL not be filtered.

5.2.2 Output Aspect Ratio

The aspect ratio of a video output port SHALL be the aspect ratio of the current output configuration for that video output port.

5.2.3 Main Video Output Port

For each HScreen, one of the video output ports is designated as the main video output port. This is the video output port that is returned when an application calls the method `HostSettings.getMainVideoOutputPort(HScreen)`. An application with sufficient permission may change the main output port by calling the method `HostSettings.setMainVideoOutputPort(..)`. The main `VideoOutputPort` for an application's default HScreen may be considered the application's default `VideoOutputPort`.

When an application sets the output video resolution for a main video output port or changes the main video output port such that the output video configuration changes, the implementation SHALL ensure that the aspect ratio of the HScreen associated with the video output port matches the aspect ratio of the video output resolution of the video output port. A change in HScreen aspect ratio SHALL be reflected in configuration changes for each component full-screen HScreenDevice, and any change in configuration SHALL be signaled by delivery of appropriate HScreenConfigurationEvents. HScreenDevice configuration changes due to video output configuration changes for the main video output port are implementation-dependant beyond the requirement that they reflect the aspect ratio of the video output port. If one or more applications has reserved any of the screen devices, the implementation SHALL release those reservations by calling the `release()` method on the respective ResourceClients, prior to changing any of the HScreenConfigurations.

Note that the converse is not implicitly true. There is no requirement that HScreenDevice configuration changes implicitly affect video output configuration.

5.2.4 Dynamic configuration of video output port setting

An OCAP host MAY support automatically setting the video output resolution for a video output port based on the video resolution/aspect ratio of the input video. This functionality MAY be used by applications to use the zoom-mode settings of a TV connected to the video output port instead of using the zoom-mode settings of the STB. This functionality is available only on a main Video Output Port. When resolution of the video output port is dynamically changed, the aspect ratio of the HScreen and the resolution of HScreen devices are also changed, according to the rules specified in Section 5.2.3 of this specification.

5.2.5 Output Port Status

Applications MAY subscribe to receive notification of changes to the status of a given video output port. Subscribed listeners SHALL be notified of status changes after the Host recognizes such changes. Where status changes cannot be recognized (for example, connection status may not be available depending upon the port type), listeners SHALL not be notified.

5.2.6 Connected Display Information

Depending upon the video output port type, information relating to the external display connected to the video output port MAY be retrieved. This includes connection status as well as display-specific attributes. For example, E-EDID information can be sent from the display to the Host via an HDMI or DVI video output port, in which case manufacturer-supplied information SHALL be provided.

5.3 Zoom Modes

Decoder format conversions represent video transformations that convert video in one source aspect ratio to the aspect ratio of the HScreen when the aspect ratios differ.

An application MAY set the decoder format conversion that is applied using the VideoFormatControl and BackgroundVideoPresentationControl JMF player controls. By default (that is for a newly-created JMF Player) and unless overridden by the application, the decoder format conversion is under platform control. When platform control is selected, the decoder format control specified by the HVideoConfiguration SHALL apply.

An application MAY set the decoder format conversion that is applied globally on a per-HVideoDevice basis by selecting an appropriate HVideoConfiguration. On Host devices implementing this specification, the ZOOM_MODE preference field in an HVideoConfigTemplate may be used to discover video configurations that specify a given decoder format conversion. The decoder format conversion of the current HVideoConfiguration SHALL apply as long as the DFC_PLATFORM is in force for associated JMF Players. Support via HVideoConfiguration SHALL be required for at least the same minimum set of decoder format conversions as is required for VideoFormatControl.

When video in one aspect ratio is transformed to another aspect ratio based on the ZOOM_MODE preference and displayed on a full-screen HVideoDevice, the method getDecoderFormatConversion() in the instance of VideoFormatControl corresponding to any associated Player SHALL return the value corresponding to the current configuration's HVideoConfigTemplate preference Object. The application MAY override this DFC value for the current player by setting another video transformation corresponding to one of the defined DFC_* constants (other than DFC_PLATFORM) using the VideoFormatControl and BackgroundVideoPresentationControl. Note: The effect of any such video transformations set will not outlive the lifetime of the JMF Player. Setting any video Transformation on a Player SHALL NOT affect the ZOOM_MODE setting for the HVideoConfiguration or HVideoDevice, but MAY affect the current configuration, per MHP 11.4.2.8.1.

5.4 Security

Calling some Java methods defined in this API requires the application to have MonitorAppPermission("deviceController"). In Host devices that implement the OCAP Device Settings Extension, the MonitorAppPermission javadoc SHALL be considered to contain an additional row in the permissions table as follows:

Table 5–1 - Permissions Table Entry for Device Settings Extension

deviceController	Allows access to the Device Settings API	Allows an application to access and modify certain device specific settings
------------------	--	---

In addition, the enumerated token value type of the name attribute of the ocap:monitorapplication element type defined in [OCAP] section 14.2.2.1.1 SHALL be considered to contain the "deviceController" value.

5.5 Persistence

Implementations of the Device Settings extension SHALL ensure that all settings controlled via the provided APIs are persisted in non-volatile memory such that those settings are restored following a reboot or power-cycle. The implementation SHALL be able to restore all settings controlled via the provided APIs to their implementation-dependent defaults as directed by the API.

5.6 Minimum Keycode Set

In Host devices that implement the OCAP Device Settings Extension, the Minimum Keycode Set table SHALL be considered to contain an additional row as follows:

Key	KeyEvent	Mandatory Ordinary Keycodes	System Keycodes
Zoom	VK_ZOOM		

In addition, this specification allows a sufficiently privileged application to change the “System Keycode” status of the volume up, down, and mute keys. The matching rows of the Minimum Keycode Set table SHALL be considered to be modified as follows:

Key	KeyEvent	Mandatory Ordinary Keycodes	System Keycodes
Volume_Down	VK_VOLUME_DOWN		If enabled via HostSettings API
Volume Up	VK_VOLUME_UP		If enabled via HostSettings API
Mute Volume	VK_MUTE		If enabled via HostSettings API

5.7 System Property

In Host devices that implement the OCAP Device Settings Extension, the “ocap.api.option.ds” Java system property SHALL be defined with a value corresponding to the implemented version of this specification.

5.8 Registry of Constants

This subsection contains the registry of Java constants specific to this extension.

org.ocap.hardware.device.AudioOutputPort		
public static final int	COMPRESSION_HEAVY	3
public static final int	COMPRESSION_LIGHT	1
public static final int	COMPRESSION_MEDIUM	2
public static final int	COMPRESSION_NONE	0
public static final int	ENCODING_AC3	3
public static final int	ENCODING_DISPLAY	1
public static final int	ENCODING_NONE	0
public static final int	ENCODING_PCM	2
public static final int	STEREO_MODE_MONO	1
public static final int	STEREO_MODE_STEREO	2

org.ocap.hardware.device.AudioOutputPort		
public static final int	STEREO_MODE_SURROUND	3
org.ocap.hardware.device.HostSettings		
public static final int	RANGE_NARROW	1
public static final int	RANGE_NORMAL	2
public static final int	RANGE_WIDE	3
org.ocap.hardware.device.VideoResolution		
public static final int	SCANMODE_INTERLACED	1
public static final int	SCANMODE_PROGRESSIVE	2
public static final int	SCANMODE_UNKNOWN	0
org.ocap.hardware.device.VideoZoomPreference		
public static final int	ZOOM_MODE	8193

6 OCAP DS API

Host devices supporting the OCAP DS Profile SHALL implement APIs listed in Annex A of this specification.

Annex A Device Settings API

Package `org.ocap.hardware.device`

Provides extension classes and interfaces related to accessing and controlling certain device parameters that are normally controlled by a device manufacturer specific application.

See:

Description

Interface Summary

FixedVideoOutputConfiguration	An instance of this class represents a video output configuration defined by a fixed set of attributes.
HostSettings	System-level extensions to OCAP Host.
VideoOutputConfiguration	Describes a <code>VideoOutputConfiguration</code> supported by a <code>VideoOutputPort</code> .
VideoOutputPortListener	The callback interface to be implemented by application classes that wish to receive notification of changes to the status of a <code>VideoOutputPort</code> .
VideoOutputSettings	The <code>VideoOutputSettings</code> interface extends the functionality of the video outputs to support configuration of video output resolution.
VideoZoomPreference	An interface that defines the constants that can be used for specifying zoom mode preferences in an <code>HVideoConfigTemplate</code> .

Class Summary

AudioOutputPort	Represents an individually controllable audio output port of the host device.
DynamicVideoOutputConfiguration	An instance of this class may be used to represent a dynamic selection of video output configurations based upon specified input video.
OCSound	Extends the HAVi <code>HSound</code> class, adding additional configuration options.
VideoResolution	Specifies the attributes of a video stream.

Exception Summary

FeatureNotSupportedException	Thrown when an application attempts to query/set/get a feature not supported on the device.
-------------------------------------	---

Package `org.ocap.hardware.device` Description

Provides extension classes and interfaces related to accessing and controlling certain device parameters that are normally controlled by a device manufacturer specific application.

org.ocap.hardware.device Class AudioOutputPort

```
java.lang.Object
└─ org.ocap.hardware.device.AudioOutputPort
```

```
public class AudioOutputPort
extends java.lang.Object
```

Represents an individually controllable audio output port of the host device. A single instance of this class *MAY* represent the audio outputs for one or more video output ports depending upon the output port configuration of the device. If a host device is capable of outputting multiple AV streams to multiple output devices simultaneously, the device *MAY* have multiple instances of this class, each representing an individually controllable audio output.

The interpretation of audio gain measured in *level* and *decibel* scales is as for `javax.media.GainControl`.

The volume controlled via an instance of `java.media.GainControl`, if supported, *SHALL* be relative to the volume controlled by an appropriate instance of `AudioOutputPort`.

See Also:

```
GainControl, HostSettings.getAudioOutputs(),
HostSettings.setSystemVolumeKeyControl(boolean),
HostSettings.setSystemMuteKeyControl(boolean),
HostSettings.setSystemVolumeRange(int)
```

Field Summary

static int	COMPRESSION_HEAVY Constant representing heavy audio level compression.
static int	COMPRESSION_LIGHT Constant representing light audio level compression.
static int	COMPRESSION_MEDIUM Constant representing medium audio level compression.
static int	COMPRESSION_NONE Constant representing no audio compression.
static int	ENCODING_AC3 Constant representing AC3 digital audio encoding.
static int	ENCODING_DISPLAY Constant representing a platform-selected digital audio encoding format.
static int	ENCODING_NONE Constant representing lack of digital audio output.
static int	ENCODING_PCM Constant representing pulse-code modulation (PCM) digital audio encoding.

Field Summary

static int	STEREO_MODE_MONO Constant representing single channel (monaural or mono) audio.
static int	STEREO_MODE_STEREO Constant representing two-channel stereo audio.
static int	STEREO_MODE_SURROUND Constant representing multi-channel stereo surround audio.

Constructor Summary

protected	AudioOutputPort() An application cannot construct an instance of this class directly.
-----------	---

Method Summary

int	getCompression() Get the current audio level compression of the audio device.
java.util.Enumeration	getConnectedVideoOutputPorts() Get the set of VideoOutputPorts whose audio is controlled by this AudioOutputPort instance.
float	getDB() Get the current gain set for this AudioOutputPort in decibels.
int	getEncoding() Get the current encoding format for digital audio output for this audio device.
float	getLevel() Get the current gain set for this AudioOutputPort as a value between 0.0 and 1.0.
float	getMaxDB() Get the maximum gain in decibels for this AudioOutputPort.
float	getMinDB() Get the minimum gain in decibels for this AudioOutputPort.
float	getOptimalLevel() Get the gain level that is optimal for stereo playback.
int	getStereoMode() Get the current stereo mode of the audio device.
int[]	getSupportedCompressions() Get the set of compression levels supported by this audio device.
int[]	getSupportedEncodings() Get the set of encoding formats supported by this audio device.
int[]	getSupportedStereoModes() Get the set of stereo modes supported by this audio device.
boolean	isLoopThru() Get the current loop-thru setting of the audio device.

Method Summary

boolean	isMuted() Get the mute state of the audio signal associated with this host.
void	setCompression (int compression) Set the compression level of the audio device.
float	setDB (float db) Set the gain in decibels for this AudioOutputPort.
void	setEncoding (int encoding) Set the desired encoding format for digital audio output for this audio device.
float	setLevel (float level) Set the gain using a floating point scale with values between 0.0 and 1.0. 0.0 is silence; 1.0 is the loudest useful level that this AudioOutputPort supports.
void	setLoopThru (boolean loopthru) Set the loop-thru setting for the audio device.
void	setMuted (boolean mute) Mute or unmute the signal associated with this AudioOutputPort.
void	setStereoMode (int mode) Set the stereo mode of the audio device.

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

STEREO_MODE_MONO

public static final int **STEREO_MODE_MONO**
Constant representing single channel (monaural or mono) audio.

See Also:

Constant Field Values

STEREO_MODE_STEREO

public static final int **STEREO_MODE_STEREO**
Constant representing two-channel stereo audio.

See Also:

Constant Field Values

STEREO_MODE_SURROUND

public static final int **STEREO_MODE_SURROUND**
Constant representing multi-channel stereo surround audio.

See Also:

Constant Field Values

ENCODING_NONE

```
public static final int ENCODING_NONE
    Constant representing lack of digital audio output.
See Also:
    Constant Field Values
```

ENCODING_DISPLAY

```
public static final int ENCODING_DISPLAY
    Constant representing a platform-selected digital audio encoding format.
See Also:
    Constant Field Values
```

ENCODING_PCM

```
public static final int ENCODING_PCM
    Constant representing pulse-code modulation (PCM) digital audio encoding.
See Also:
    Constant Field Values
```

ENCODING_AC3

```
public static final int ENCODING_AC3
    Constant representing AC3 digital audio encoding.
See Also:
    Constant Field Values
```

COMPRESSION_NONE

```
public static final int COMPRESSION_NONE
    Constant representing no audio compression.
See Also:
    Constant Field Values
```

COMPRESSION_LIGHT

```
public static final int COMPRESSION_LIGHT
    Constant representing light audio level compression.
See Also:
    Constant Field Values
```

COMPRESSION_MEDIUM

```
public static final int COMPRESSION_MEDIUM
    Constant representing medium audio level compression.
See Also:
    Constant Field Values
```

COMPRESSION_HEAVY

```
public static final int COMPRESSION_HEAVY
    Constant representing heavy audio level compression.
See Also:
    Constant Field Values
```

Constructor Detail

AudioOutputPort

protected **AudioOutputPort**()

An application cannot construct an instance of this class directly.

Method Detail

getStereoMode

public int **getStereoMode**()

Get the current stereo mode of the audio device.

Returns:

The current stereo mode. SHALL be one of STEREO_MODE_MONO, STEREO_MODE_STEREO or STEREO_MODE_SURROUND.

See Also:

setStereoMode(int), getSupportedStereoModes()

setStereoMode

public void **setStereoMode**(int mode)

throws FeatureNotSupportedException

Set the stereo mode of the audio device.

Parameters:

mode - The desired stereo mode.

Throws:

java.lang.IllegalArgumentException - if mode is not one of STEREO_MODE_MONO, STEREO_MODE_STEREO or STEREO_MODE_SURROUND

FeatureNotSupportedException - if the given setting is not supported

See Also:

getStereoMode(), getSupportedStereoModes()

getSupportedStereoModes

public int[] **getSupportedStereoModes**()

Get the set of stereo modes supported by this audio device. The returned values SHALL NOT produce a FeatureNotSupportedException when provided to setStereoMode(int).

Returns:

a non-null array containing the set of supported compression level settings

See Also:

setStereoMode(int), getStereoMode()

getCompression

public int **getCompression**()

Get the current audio level compression of the audio device.

Returns:

The current compression level. SHALL be one of COMPRESSION_NONE, COMPRESSION_LIGHT, COMPRESSION_MEDIUM or COMPRESSION_HEAVY

See Also:

setCompression(int), getSupportedCompressions()

setCompression

```
public void setCompression(int compression)
    throws FeatureNotSupportedException
```

Set the compression level of the audio device. Compression reduces the dynamic range of an audio signal by reducing the gain when the signal level is higher than a given threshold.

The compression levels correspond to implementation-specific ratios of input level to output gain. The following table describes the compression levels.

Level	Description
COMPRESSION_NONE	Always a 1:1 ratio of input level to output gain. No compression is applied to the output signal
COMPRESSION_LIGHT	The lightest level of compression. Example ratio would be 2:1, where a 2dB change in input level is required to effect a 1dB change in output, above the threshold.
COMPRESSION_MEDIUM	An equal or higher level of compression than COMPRESSION_LIGHT
COMPRESSION_HEAVY	The highest level of compression. This may be a high enough level so as to be considered <i>limiting</i> .

Other attributes of audio level compression (e.g., threshold, attack, and release) are implementation-specific and not exposed by this API.

Parameters:

`compression` - The desired compression level.

Throws:

`java.lang.IllegalArgumentException` - if `compression` is not one of `COMPRESSION_NONE`, `COMPRESSION_LIGHT`, `COMPRESSION_MEDIUM` or `COMPRESSION_HEAVY`

`FeatureNotSupportedException` - if the given setting is not supported

See Also:

`getCompression()`, `getSupportedCompressions()`

getSupportedCompressions

```
public int[] getSupportedCompressions()
```

Get the set of compression levels supported by this audio device. The returned values SHALL NOT produce a `FeatureNotSupportedException` when provided to `setCompression(int)`.

Returns:

a non-null array containing the set of supported compression level settings

See Also:

`setCompression(int)`, `getCompression()`

getEncoding

```
public int getEncoding()
```

Get the current encoding format for digital audio output for this audio device.

This method will never return `ENCODING_DISPLAY`. Instead, the current platform-selected encoding will be returned. The platform selects such an encoding based upon the device connected to this audio output port.

Returns:

The current encoding format. SHALL be one of `ENCODING_NONE`, `ENCODING_PCM` or `ENCODING_AC3`

See Also:

```
setEncoding(int), getSupportedEncodings()
```

setEncoding

```
public void setEncoding(int encoding)
    throws FeatureNotSupportedException
```

Set the desired encoding format for digital audio output for this audio device.

This method MAY be used to control the desired encoding format for digital audio output for this audio device. The following table describes the supported values for the *encoding* parameter:

Value	Description
ENCODING_PCM	Pulse code modulation digital audio encoding.
ENCODING_AC3	AC-3 digital audio encoding.
ENCODING_DISPLAY	No explicit format is specified, instead the Host device will select the preferred output format.

Parameters:

encoding - The desired digital encoding format.

Throws:

java.lang.IllegalArgumentException - if *encoding* is not one of ENCODING_DISPLAY, ENCODING_PCM or ENCODING_AC3

FeatureNotSupportedException - if the given setting is not supported

See Also:

getEncoding(), getSupportedEncodings()

getSupportedEncodings

```
public int[] getSupportedEncodings()
```

Get the set of encoding formats supported by this audio device. The returned values SHALL NOT produce a FeatureNotSupportedException when provided to setEncoding(int).

Returns:

a non-null array containing the set of supported digital encoding formats

See Also:

setEncoding(int), getEncoding()

getDB

```
public float getDB()
```

Get the current gain set for this AudioOutputPort in decibels.

Returns:

The gain in dB.

See Also:

setDB(float)

setDB

```
public float setDB(float db)
```

Set the gain in decibels for this AudioOutputPort. Setting the gain to 0.0 (the default) implies that the audio signal is neither amplified nor attenuated. Positive values amplify the audio signal and negative values attenuate the signal.

Parameters:

db - The new gain in dB.

Returns:

The gain that was actually set.

See Also:

GainControl, getDB(), getMaxDB(), getMinDB(), setLevel(float)

getMaxDB

```
public float getMaxDB()
```

Get the maximum gain in decibels for this AudioOutputPort. Calling setDB(float) with values greater than those returned by this API will have no effect.

Returns:

The maximum gain in decibels.

getMinDB

```
public float getMinDB()
```

Get the minimum gain in decibels for this AudioOutputPort. Calling setDB(float) with values less than those returned by this API will have no effect.

Returns:

The minimum gain in decibels.

getLevel

```
public float getLevel()
```

Get the current gain set for this AudioOutputPort as a value between 0.0 and 1.0.

Returns:

The gain in the level scale (0.0-1.0).

See Also:

setLevel(float)

setLevel

```
public float setLevel(float level)
```

Set the gain using a floating point scale with values between 0.0 and 1.0. 0.0 is silence; 1.0 is the loudest useful level that this AudioOutputPort supports.

Parameters:

level - The new gain value specified in the level scale.

Returns:

The level that was actually set.

See Also:

GainControl, getLevel(), getDB(), setDB(float)

getOptimalLevel

```
public float getOptimalLevel()
```

Get the gain level that is optimal for stereo playback. Selection of this gain level (using setLevel(float)) will reduce audio distortion on televisions that contain stereo decoders.

Where a fixed volume level is desired, it is recommended that the *optimal* level returned by this method be used.

Returns:

The optimal volume level.

See Also:

setLevel(float)

isMuted

```
public boolean isMuted()
```

Get the mute state of the audio signal associated with this host.

Returns:

The current mute state: true if muted and false otherwise.

See Also:

setMuted(boolean)

setMuted

```
public void setMuted(boolean mute)
```

Mute or unmute the signal associated with this `AudioOutputPort`. Redundant invocations of this method are ignored. The mute state does not effect the gain (as represented by `getLevel()` or `getDB()`).

Parameters:

mute - The new mute state: true mutes the signal and false unmutes the signal.

See Also:

isMuted()

isLoopThru

```
public boolean isLoopThru()
```

Get the current loop-thru setting of the audio device.

Returns:

The current loop-thru state: true if loop-thru is enabled and false otherwise.

See Also:

setLoopThru(boolean)

setLoopThru

```
public void setLoopThru(boolean loopthru)  
    throws FeatureNotSupportedException
```

Set the loop-thru setting for the audio device.

Audio loop-thru refers to a mechanism that allows audio from other devices (e.g., DVD player or VCR) to connect to a home theater or TV through this Host device when this `AudioOutputPort` is otherwise not in use. When loop-thru is enabled, audio inputs associated with this `AudioOutputPort` will be routed through this `AudioOutputPort`.

Parameters:

loopthru - The new loop-thru state: (true enables loop-thru and false disables loop-thru.

Throws:

`FeatureNotSupportedException` - if the requested loop-thru setting can not be achieved by the device

See Also:

isLoopThru()

getConnectedVideoOutputPorts

```
public java.util.Enumeration getConnectedVideoOutputPorts()
```

Get the set of `VideoOutputPorts` whose audio is controlled by this `AudioOutputPort` instance.

Returns:

The set of controlled `VideoOutputPorts` as an `Enumeration`.

org.ocap.hardware.device Class DynamicVideoOutputConfiguration

```
java.lang.Object
└─ org.ocap.hardware.device.DynamicVideoOutputConfiguration
All Implemented Interfaces:
    VideoOutputConfiguration
```

```
public class DynamicVideoOutputConfiguration
extends java.lang.Object
implements VideoOutputConfiguration
```

An instance of this class may be used to represent a dynamic selection of video output configurations based upon specified input video. An instance of `DynamicVideoOutputConfiguration` would mainly be used to allow for the `HScreen` resolution and the video output port resolution to closely match the resolution of the input video, generally with the intention of letting the display monitor connected to the output port manage aspect ratio conversions.

Such configurations are only valid for the current `main` video output port for a given `HScreen`. If a video output port is not the current *main* output port or ceases to be the *main*, then this configuration setting SHALL be effectively ignored and output resolution settings SHALL revert to an implementation-specific configuration.

Application of the input-to-output video resolution mapping described by an instance of `DynamicVideoOutputConfiguration` MAY result in configuration changes for the component `HScreenDevices` of the relevant `HScreen` as if the output resolution were selected via a `static` configuration.

Field Summary

static VideoResolution	INPUT_HD Constant representing any High Definition input video.
static VideoResolution	INPUT_SD Constant representing any Standard Definition input video.

Constructor Summary

DynamicVideoOutputConfiguration() Construct a new instance of <code>DynamicVideoOutputConfiguration</code> .
--

Method Summary

void	addOutputResolution (VideoResolution inputResolution, FixedVideoOutputConfiguration outputResolution) Add a desired input video resolution to output video resolution mapping.
java.util.Enumeration	getInputResolutions () Get the set of input video resolutions for which a mapping to output configuration is specified.
java.lang.String	getName () Returns "Dynamic".
FixedVideoOutputConfiguration	getOutputResolution (VideoResolution inputResolution) Get the output configuration that should be applied for the given input resolution if this configuration were successfully set on a video output port.

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

INPUT_SD

public static final VideoResolution **INPUT_SD**
Constant representing any Standard Definition input video.

INPUT_HD

public static final VideoResolution **INPUT_HD**
Constant representing any High Definition input video.

Constructor Detail

DynamicVideoOutputConfiguration

public **DynamicVideoOutputConfiguration** ()
Construct a new instance of DynamicVideoOutputConfiguration. The newly created DynamicVideoOutputConfiguration contains no mappings from input video to output video configuration.

Method Detail

getName

public java.lang.String **getName** ()
Returns "Dynamic".
Specified by:

getName in interface VideoOutputConfiguration

Returns:
"Dynamic"

See Also:
VideoOutputConfiguration.getName()

addOutputResolution

```
public void addOutputResolution(VideoResolution inputResolution,
FixedVideoOutputConfiguration outputResolution)
```

Add a desired input video resolution to output video resolution mapping. If this configuration is applied successfully, the video output port would use the given output resolution whenever the main video input resolution matched the given input resolution.

The desired video output resolution is specified as an instance of FixedVideoOutputConfiguration. Valid configurations are those returned by VideoOutputSettings.getSupportedConfigurations() for a given video output port instance. This class does not guard against addition of an invalid video resolution configuration. Instead, the instance of DynamicVideoOutputConfiguration would be rejected by the video output port.

For a given input resolution, wildcard values may be specified for some attributes. The following table documents accepted wildcard (or "don't care") values.

Attribute	Wildcard value
VideoResolution.getPixelResolution()	null
VideoResolution.getAspectRatio()	VideoFormatControl.ASPECT_RATIO_UNKNOWN
VideoResolution.getRate()	<= 0.0F
VideoResolution.getScanMode()	VideoResolution.SCANMODE_UNKNOWN

Parameters:

inputResolution - The given input video resolution. May be an application-created instance of VideoResolution; or one of INPUT_SD or INPUT_HD may be specified to indicate a wildcard value covering all SD or HD resolutions.

outputResolution - The desired output configuration that video of the given input resolution should be mapped.

See Also:

getOutputResolution(org.ocap.hardware.device.VideoResolution)

getOutputResolution

```
public FixedVideoOutputConfiguration
getOutputResolution(VideoResolution inputResolution)
```

Get the output configuration that should be applied for the given input resolution if this configuration were successfully set on a video output port.

Parameters:

inputResolution - The given input video resolution. May be an application-created instance of VideoResolution; or one of INPUT_SD or INPUT_HD may be specified to indicate a wildcard value covering all SD or HD resolutions.

Returns:

The output video configuration mapped to by the given input resolution or null if no mapping is defined.

See Also:

```
addOutputResolution(org.ocap.hardware.device.VideoResolution,  
org.ocap.hardware.device.FixedVideoOutputConfiguration)
```

getInputResolutions

```
public java.util.Enumeration getInputResolutions()
```

Get the set of input video resolutions for which a mapping to output configuration is specified.

Returns:

A non-null Enumeration of VideoResolution instances.

org.ocap.hardware.device Class FeatureNotSupportedException

```
java.lang.Object
├─ java.lang.Throwable
│   └─ java.lang.Exception
│       └─ org.ocap.hardware.device.FeatureNotSupportedException
```

All Implemented Interfaces:

```
java.io.Serializable
```

```
public class FeatureNotSupportedException
extends java.lang.Exception
```

Thrown when an application attempts to query/set/get a feature not supported on the device.

See Also:

Serialized Form

Constructor Summary

FeatureNotSupportedException()

Creates an FeatureNotSupportedException object.

FeatureNotSupportedException(java.lang.String message)

Creates an FeatureNotSupportedException object with a specified reason string.

Method Summary

Methods inherited from class java.lang.Throwable

fillInStackTrace, getLocalizedMessage, getMessage, printStackTrace, printStackTrace, printStackTrace, toString

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

FeatureNotSupportedException

```
public FeatureNotSupportedException()
```

Creates an FeatureNotSupportedException object. See the class description for details of constructor parameters and default values.

FeatureNotSupportedException

```
public FeatureNotSupportedException(java.lang.String message)
```

Creates an FeatureNotSupportedException object with a specified reason string.

Parameters:

message - the reason why the exception was raised

org.ocap.hardware.device Interface FixedVideoOutputConfiguration

All Superinterfaces:

VideoOutputConfiguration

```
public interface FixedVideoOutputConfiguration
extends VideoOutputConfiguration
```

An instance of this class represents a video output configuration defined by a fixed set of attributes.

When such a configuration is successfully applied to a `VideoOutputPort`, the output resolution SHALL be suitably adjusted. This MAY in turn adjust the aspect ratio and resolution of the associated `HScreen` and its component `HScreenDevices`. Any such changes will be announced via the dispatch of `HScreenConfigurationEvents`.

See Also:

```
VideoOutputSettings.getSupportedConfigurations(),
VideoOutputSettings.setOutputConfiguration(org.ocap.hardware.device.VideoOutputConfiguration)
```

Method Summary

VideoResolution	getVideoResolution() Get the fixed video resolution represented by this VideoOutputConfiguration.
-----------------	---

Methods inherited from interface org.ocap.hardware.device.VideoOutputConfiguration

getName

Method Detail

getVideoResolution

```
VideoResolution getVideoResolution()
```

Get the fixed video resolution represented by this VideoOutputConfiguration.

Returns:

the fixed video resolution represented by this configuration

org.ocap.hardware.device Interface HostSettings

```
public interface HostSettings
```

System-level extensions to OCAP Host. On Host devices implementing this specification, the instance of Host returned by Host.getInstance() SHALL also implement this interface.

Field Summary

static int	RANGE_NARROW Constant representing a <i>narrow</i> range of volume control.
static int	RANGE_NORMAL Constant representing a <i>normal</i> range of volume control.
static int	RANGE_WIDE Constant representing a <i>wide</i> range of volume control.

Method Summary

java.util.Enumeration	getAudioOutputs() Get the set of individually controllable audio outputs for this host.
VideoOutputPort	getMainVideoOutputPort(HScreen screen) Get the <i>main</i> video output port for the given HScreen.
void	resetAllDefaults() Reset all Host device settings to their factory default values.
void	setMainVideoOutputPort(HScreen screen, VideoOutputPort port) Set the <i>main</i> video output port for the given HScreen.
void	setPowerMode(int mode) Transition the power mode of the system to the given mode.
void	setSystemMuteKeyControl(boolean enable) Enable or disable system handling of the mute key.
void	setSystemVolumeKeyControl(boolean enable) Enable or disable system handling of volume keys.
void	setSystemVolumeRange(int range) Set the range of volume level controlled by the system volume keys.

Field Detail

RANGE_NARROW

```
static final int RANGE_NARROW
```

Constant representing a *narrow* range of volume control.
See Also:

Constant Field Values

RANGE_NORMAL

static final int **RANGE_NORMAL**
Constant representing a *normal* range of volume control.
See Also:
Constant Field Values

RANGE_WIDE

static final int **RANGE_WIDE**
Constant representing a *wide* range of volume control.
See Also:
Constant Field Values

Method Detail

setPowerMode

void **setPowerMode**(int mode)
Transition the power mode of the system to the given mode.

If the power mode is already in the target mode, this method SHALL do nothing. Setting host power mode to low-power SHALL not disrupt any ongoing recording. In devices where a separate power mode is maintained for standby recordings, setting the power mode to low-power SHALL transition to standby-recording power mode when a recording is in progress.

A change of power mode SHALL be communicated to installed
PowerModeChangeListeners.

Parameters:

mode - The new power mode for the system.

Throws:

java.lang.IllegalArgumentException - if *mode* is not one of Host.FULL_POWER or Host.LOW_POWER

java.lang.SecurityException - if the caller does not have
MonitorAppPermission("deviceController" or "powerMode")

getAudioOutputs

java.util.Enumeration **getAudioOutputs**()
Get the set of individually controllable audio outputs for this host.

Returns:

The set of AudioOutputPorts as an Enumeration.

Throws:

java.lang.SecurityException - if the caller does not have
MonitorAppPermission("deviceController")

setSystemVolumeKeyControl

void **setSystemVolumeKeyControl**(boolean enable)

Enable or disable system handling of volume keys. This method may be used by applications to disable the OCAP default behavior of system volume level being handled by the OCAP device based upon volume keys (i.e., `VK_VOLUME_UP` and `VK_VOLUME_DOWN`). OCAP-defined behavior SHALL be the default if this method is never called.

If the system volume control is disabled, the device SHALL not process volume key events internally. In other words, the volume keys SHALL no longer be considered *system* keys when system volume control is disabled. While disabled it is the responsibility of applications to manage the master volume of the device via the `AudioOutputPort` API.

Parameters:

`enable` - Enable or disable system handling of volume keys. If `true` is specified, then system volume SHALL be handled by the OCAP device (as is the default). If `false` is specified, then the system volume SHALL NOT be managed directly by the OCAP device based upon volume keys.

Throws:

`java.lang.SecurityException` - If the caller does not have `MonitorAppPermission("deviceController")`

setSystemMuteKeyControl

```
void setSystemMuteKeyControl(boolean enable)
```

Enable or disable system handling of the mute key. This method may be used by applications to disable the OCAP default behavior of system volume muting being handled by the OCAP device based upon the mute key (i.e., `VK_MUTE`). OCAP-defined behavior SHALL be the default if this method is never called.

If the system volume mute control is disabled, the device SHALL not process the volume mute events internally. In other words, the mute key SHALL no longer be considered a *system* key when system volume mute control is disabled. While disabled it is the responsibility of applications to manage the master volume mute state of the device via the `AudioOutputPort` API.

Parameters:

`enable` - Enable or disable system handling of the volume mute key. If `true` is specified, then system volume mute SHALL be handled by the OCAP device (as is the default). If `false` is specified, then the system volume mute SHALL NOT be managed directly by the OCAP device based upon the mute key.

Throws:

`java.lang.SecurityException` - If the caller does not have `MonitorAppPermission("deviceController")`

setSystemVolumeRange

```
void setSystemVolumeRange(int range)
```

Set the range of volume level controlled by the system volume keys.

This method may be used by applications to control the range in dB levels controlled by the system volume keys. If system volume control is disabled (via `setSystemVolumeKeyControl(boolean)`) then this setting SHALL have no effect. The following table describes the range values.

Range	Description
<code>RANGE_NARROW</code>	A very limited range of volume level is controllable via system volume keys.
<code>RANGE_NORMAL</code>	A limited range of volume level is controllable via system volume keys.

Range	Description
RANGE_WIDE	The full volume level is controllable via system volume keys.

Parameters:

range - The desired control range. One of RANGE_NARROW, RANGE_NORMAL, or RANGE_WIDE

Throws:

java.lang.IllegalArgumentException - if an invalid range value is specified

java.lang.SecurityException - if the caller does not have
MonitorAppPermission("deviceController")

getMainVideoOutputPort

VideoOutputPort **getMainVideoOutputPort**(HScreen screen)

Get the *main* video output port for the given HScreen.

Returns:

The instance of VideoOutputPort that represents the *main* video output port for the given screen

setMainVideoOutputPort

```
void setMainVideoOutputPort(HScreen screen,
                             VideoOutputPort port)
    throws FeatureNotSupportedException
```

Set the *main* video output port for the given HScreen.

Changing this setting MAY affect the configuration of HScreenDevices of the given HScreen to maintain consistent display aspect ratios as described in the body of this specification.

Parameters:

screen - The specified HScreen for which to set the *main* video output port.

port - The desired main VideoOutputPort.

Throws:

java.lang.SecurityException - if the caller does not have

MonitorAppPermission("deviceController")

FeatureNotSupportedException - if the given setting is not supported

resetAllDefaults

```
void resetAllDefaults()
```

Reset all Host device settings to their factory default values.

Calling this method SHALL result in the Host restoring all configurable scalar settings to their default values, regardless of storage location. This includes both settings that persist and do not persist across Host device reboots.

This method SHALL affect the following:

- All settings defined by this specification. Including audio and video as well as those defined by this class.
- All Host settings. Including RF bypass and AC outlet settings.
- Closed-captioning settings controllable via the ClosedCaptioningAttribute.
- User preferences controllable via the UserPreferenceManager.

Further, any subsequent operations that would be affected by this change in settings SHALL be affected as if the corresponding API were invoked directly.

Throws:

`java.lang.SecurityException` - if the caller does not have `MonitorAppPermission("deviceController")`

org.ocap.hardware.device
Class OCSound

```
java.lang.Object
├─ org.havi.ui.HSound
│   └─ org.ocap.hardware.device.OCSound
```

```
public class OCSound
extends HSound
```

Extends the HAVi `HSound` class, adding additional configuration options. Instances of this class provide control over audio gain level, muting, and output ports.

See Also:

`AudioOutputPort`

Constructor Summary

OCSound()	Creates an OCSound object.
------------------	----------------------------

Method Summary

void	addAudioOutput (AudioOutputPort au) Add an AudioOutputPort to the set of audio output ports where this clip will be played.
AudioOutputPort[]	getAudioOutputs () Get the audio output ports on which this audio clip would be played.
float	getLevel () Get the current gain set for this OCSound as a value between 0.0 and 1.0.
boolean	isMuted () Get the mute state of the audio signal associated with this audio clip.
void	removeAudioOutput (AudioOutputPort au) Remove an AudioOutputPort from the set of audio output ports where this clip will be played.
float	setLevel (float level) Set the gain using a floating point scale with values between 0.0 and 1.0. 0.0 is silence; 1.0 is the loudest level for associated audio output ports.
void	setMuted (boolean mute) Mute or unmute the signal associated with this OCSound.

Methods inherited from class org.havi.ui.HSound

`dispose`, `load`, `load`, `loop`, `play`, `set`, `stop`

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail**OCSound**

```
public OCSound()
```

Creates an OCSound object. The following defaults apply upon construction:

Attribute	Method	Default
Level	getLevel()	1.0
Mute	isMuted()	false
Outputs	getAudioOutputs()	the default audio output for the application constructing the OCSound instance

Method Detail**setLevel**

```
public float setLevel(float level)
```

Set the gain using a floating point scale with values between 0.0 and 1.0. 0.0 is silence; 1.0 is the loudest level for associated audio output ports.

Parameters:

level - The new gain value specified in the level scale.

Returns:

The level that was actually set.

See Also:

getLevel(), AudioOutputPort.setLevel(float), AudioOutputPort.getLevel()

getLevel

```
public float getLevel()
```

Get the current gain set for this OCSound as a value between 0.0 and 1.0.

Returns:

The gain in the level scale (0.0-1.0).

See Also:

setLevel(float)

isMuted

```
public boolean isMuted()
```

Get the mute state of the audio signal associated with this audio clip.

Returns:

The current mute state: true if muted and false otherwise.

See Also:

setMuted(boolean)

setMuted

```
public void setMuted(boolean mute)
```

Mute or unmute the signal associated with this `OCSound`. Redundant invocations of this method are ignored. The mute state does not effect the gain (as represented by `getLevel()`).

Parameters:

`mute` - The new mute state: `true` mutes the signal and `false` unmutes the signal.

See Also:

`isMuted()`

getAudioOutputs

```
public AudioOutputPort[] getAudioOutputs()
```

Get the audio output ports on which this audio clip would be played. By default, audio-clips will be played on the default audio output port for the application that created this `OCSound`. Unless `AudioOutputPorts` have been removed by calling `removeAudioOutput`, this method **SHALL** return the at least the default `AudioOutputPort` for the application. Unless `AudioOutputPorts` have been added by calling `addAudioOutput`, this method **SHALL** return at most the default `AudioOutputPort` for the application.

Returns:

The set of target `AudioOutputPorts` as an array. If all ports have been removed, then an empty array **SHALL** be returned.

See Also:

`addAudioOutput(org.ocap.hardware.device.AudioOutputPort)`,
`removeAudioOutput(org.ocap.hardware.device.AudioOutputPort)`

addAudioOutput

```
public void addAudioOutput(AudioOutputPort au)
```

Add an `AudioOutputPort` to the set of audio output ports where this clip will be played.

Redundant additions **SHALL** have no effect.

Parameters:

`au` - The `AudioOutputPort` to add.

removeAudioOutput

```
public void removeAudioOutput(AudioOutputPort au)
```

Remove an `AudioOutputPort` from the set of audio output ports where this clip will be played.

Attempting to remove an `AudioOutputPort` that is not currently in the set of audio output ports for this `OCSound` **SHALL** have no effect.

Parameters:

`au` - The `AudioOutputPort` to remove.

org.ocap.hardware.device**Interface VideoOutputConfiguration****All Known Subinterfaces:**

FixedVideoOutputConfiguration

All Known Implementing Classes:

DynamicVideoOutputConfiguration

```
public interface VideoOutputConfiguration
```

Describes a VideoOutputConfiguration supported by a VideoOutputPort.

See Also:

VideoOutputSettings

Method Summary

java.lang.String	getName() Get the String representation of this VideoOutputConfiguration, suitable for display to the user.
------------------	---

Method Detail**getName**

```
java.lang.String getName()
```

Get the String representation of this VideoOutputConfiguration, suitable for display to the user.

Returns:

String representation of this object

org.ocap.hardware.device Interface VideoOutputPortListener

All Superinterfaces:

java.util.EventListener

```
public interface VideoOutputPortListener
extends java.util.EventListener
```

The callback interface to be implemented by application classes that wish to receive notification of changes to the status of a VideoOutputPort.

Method Summary

void	configurationChanged (VideoOutputPort source, VideoOutputConfiguration oldConfig, VideoOutputConfiguration newConfig) Method to be invoked when the configuration of a VideoOutputPort changes.
void	connectionStatusChanged (VideoOutputPort source, boolean status) Method to be invoked when a display device is connected or disconnected.
void	enabledStatusChanged (VideoOutputPort source, boolean status) Method to be invoked when the VideoOutputPort is enabled or disabled.

Method Detail

connectionStatusChanged

```
void connectionStatusChanged(VideoOutputPort source,
                             boolean status)
```

Method to be invoked when a display device is connected or disconnected.

Note that this method will not be invoked where such information cannot be known by the Host device.

Parameters:

source - the VideoOutputPort whose status has changed

status - true when a display device is connected; false when a display device is disconnected

See Also:

VideoOutputSettings.isDisplayConnected()

enabledStatusChanged

```
void enabledStatusChanged(VideoOutputPort source,
                          boolean status)
```

Method to be invoked when the VideoOutputPort is enabled or disabled.

Parameters:

source - the VideoOutputPort whose status has changed

status - true when the video output port is enabled; false when the video output port is disabled

See Also:

VideoOutputPort.enable(), VideoOutputPort.disable(),

VideoOutputPort.status()

configurationChanged

```
void configurationChanged(VideoOutputPort source,  
                           VideoOutputConfiguration oldConfig,  
                           VideoOutputConfiguration newConfig)
```

Method to be invoked when the configuration of a VideoOutputPort changes.

Parameters:

source - the VideoOutputPort whose status has changed

oldConfig - the previous configuration

newConfig - the new configuration

org.ocap.hardware.device
Interface VideoOutputSettings

```
public interface VideoOutputSettings
```

The `VideoOutputSettings` interface extends the functionality of the video outputs to support configuration of video output resolution. In Host devices supporting this specification, all instances of `VideoOutputPort` SHALL implement this interface.

An application MAY query the set of output configurations supported by a given video output port by calling `getSupportedConfigurations()`. The supported configurations MAY be used to configure the given video output port by calling `setOutputConfiguration(org.ocap.hardware.device.VideoOutputConfiguration)`.

See Also:

`VideoOutputPort`

Method Summary

void	addListener (VideoOutputPortListener l) Add the given listener to monitor this video output port for status changes.
AudioOutputPort	getAudioOutputPort () Get the AudioOutputPort object which can be used to control the audio output port associated with this video output port.
int	getDisplayAspectRatio () Get the aspect ratio of the display connected to this video output port.
java.util.Hashtable	getDisplayAttributes () Get the set of attributes describing the display currently connected to this VideoOutputPort.
VideoOutputConfiguration	getOutputConfiguration () Get the current output configuration for this video output port.
VideoOutputConfiguration[]	getSupportedConfigurations () Get the fixed set of supported output configurations for this video output port.
boolean	isContentProtected () Get the protection status of content on this video output port.
boolean	isDisplayConnected () Get the connection status of this video output port.
boolean	isDynamicConfigurationSupported () Determine if this video output port supports dynamic output configuration based upon input video attributes.
void	removeListener (VideoOutputPortListener l) Remove the given listener from further notification of status changes for this video output port.

The output resolution configuration is applied at method invocation time. Changes to a configuration instance SHALL have no effect on the current configuration unless applied via invocation of this method specifying the given configuration.

Changing this setting on the "main" video output port for an `HScreen` MAY affect the configuration of the `HScreenDevices` of that `HScreen` to maintain consistent display aspect ratios as described in the body of this specification.

Parameters:

`config` - The new output configuration.

Throws:

`java.lang.IllegalArgumentException` - if the output resolution specified does not correspond to one of the supported configurations

`FeatureNotSupportedException` - if the given configuration corresponds to one of the supported resolutions, but cannot be satisfied

`java.lang.SecurityException` - if the caller does not have `MonitorAppPermission("deviceController")`

See Also:

`getOutputConfiguration()`

getAudioOutputPort

`AudioOutputPort` **getAudioOutputPort()**

Get the `AudioOutputPort` object which can be used to control the audio output port associated with this video output port.

Returns:

the `AudioOutputPort` for this video output port

addListener

`void` **addListener**(`VideoOutputPortListener l`)

Add the given listener to monitor this video output port for status changes.

Parameters:

`l` - the listener to add

See Also:

`removeListener(VideoOutputPortListener)`

removeListener

`void` **removeListener**(`VideoOutputPortListener l`)

Remove the given listener from further notification of status changes for this video output port.

Parameters:

`l` - the listener to remove

See Also:

`addListener(VideoOutputPortListener)`

getDisplayAttributes

`java.util.Hashtable` **getDisplayAttributes()**

Get the set of attributes describing the display currently connected to this `VideoOutputPort`. The display attributes information is returned in the form of a `Hashtable`, mapping attribute names (specified as `Strings`) to attribute values (specified as attribute-specific `Objects`).

For example, where EEDID (Enhanced Extended Display Identification Data) retrieval is supported, this information SHALL be accessible as attributes of the connected display. In case of EEDID, the following table describes the attribute mappings that SHALL be supported and used where appropriate.

Attribute name	Attribute value
"Manufacturer Name"	3-character EISA manufacturer ID as a String
"Product Code"	Vendor assigned product code as a Short
"Serial Number"	Vendor assigned serial number as a Integer
"Manufacture Week"	Week of manufacture as a Byte
"Manufacture Year"	Year of manufacture (offset from 1990) as a Byte

Returns:

a Hashtable describing known display attributes; null is returned if `isDisplayConnected()` would return false

isDisplayConnected

boolean `isDisplayConnected()`

Get the connection status of this video output port.

If the Host device is incapable of determining connection status (e.g., for component video), then false SHALL be returned.

Returns:

true if a display is known to be connected; false otherwise

isContentProtected

boolean `isContentProtected()`

Get the protection status of content on this video output port.

This SHALL return true if both this port and the connected display support content protection and content is protected; false SHALL be returned otherwise.

Returns:

whether content is currently protected on this video output port

getDisplayAspectRatio

int `getDisplayAspectRatio()`

Get the aspect ratio of the display connected to this video output port.

Returns:

one of `VideoFormatControl.DAR_4_3`, `VideoFormatControl.DAR_16_9`, or -1 if unknown

org.ocap.hardware.device Class VideoResolution

```
java.lang.Object
└─org.ocap.hardware.device.VideoResolution
```

```
public class VideoResolution
extends java.lang.Object
```

Specifies the attributes of a video stream. Instances of `VideoResolution` may be used to describe attributes of input or output video.

See Also:

```
FixedVideoOutputConfiguration.getVideoResolution(),
DynamicVideoOutputConfiguration.getInputResolutions(),
DynamicVideoOutputConfiguration.addOutputResolution(VideoResolution,
FixedVideoOutputConfiguration)
```

Field Summary

static int	SCANMODE_INTERLACED Constant indicating interlaced line scan mode.
static int	SCANMODE_PROGRESSIVE Constant indicating progressive line scan mode.
static int	SCANMODE_UNKNOWN Constant indicating an unknown or unspecified line scan mode.

Constructor Summary

VideoResolution(java.awt.Dimension rez, int ar, float rate, int scan)
Creates an instance of `VideoResolution` based upon the given attributes.

Method Summary

int	getAspectRatio() Return the aspect ratio of the output video as specified by this object.
java.awt.Dimension	getPixelResolution() Return the pixel resolution of the video.
float	getRate() Return the frame or field rate of the video as specified by this object.
int	getScanMode() Return the video scan mode, as specified by this object.

Methods inherited from class java.lang.Object

```
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString,
wait, wait, wait
```

Field Detail

SCANMODE_UNKNOWN

```
public static final int SCANMODE_UNKNOWN
```

Constant indicating an unknown or unspecified line scan mode.

See Also:
Constant Field Values

SCANMODE_INTERLACED

```
public static final int SCANMODE_INTERLACED
```

Constant indicating interlaced line scan mode.

See Also:
Constant Field Values

SCANMODE_PROGRESSIVE

```
public static final int SCANMODE_PROGRESSIVE
```

Constant indicating progressive line scan mode.

See Also:
Constant Field Values

Constructor Detail

VideoResolution

```
public VideoResolution(java.awt.Dimension rez,
                       int ar,
                       float rate,
                       int scan)
```

Creates an instance of `VideoResolution` based upon the given attributes.

Parameters:

- `rez` - The desired pixel resolution; `null` MAY be specified to indicate a *wildcard* value.
- `ar` - The desired aspect ratio. `VideoFormatControl.ASPECT_RATIO_UNKNOWN` MAY be specified to indicate a *wildcard* value.
- `rate` - The desired field or frame rate. Values less than or equal to 0.0F may be specified to indicate a *wildcard* value.
- `scan` - The desired scan mode. `SCANMODE_UNKNOWN` MAY be specified to indicate a *wildcard* value.

Method Detail

getPixelResolution

```
public java.awt.Dimension getPixelResolution()
```

Return the pixel resolution of the video.

A value of `null` MAY be returned, indicating that the resolution is unknown or unspecified.

Returns:

an instance of `Dimension` specifying the pixel resolution or `null`

getAspectRatio

```
public int getAspectRatio()
```

Return the aspect ratio of the output video as specified by this object.

A value of `ASPECT_RATIO_UNKNOWN` MAY be returned, indicating that the aspect ratio is unknown or unspecified.

Returns:

one of `VideoFormatControl.ASPECT_RATIO_UNKNOWN`,
`VideoFormatControl.ASPECT_RATIO_4_3`,
`VideoFormatControl.ASPECT_RATIO_16_9`, or
`VideoFormatControl.ASPECT_RATIO_2_21_1`.

getRate

```
public float getRate()
```

Return the frame or field rate of the video as specified by this object.

Possible return values are:

- 24000F/1001 (23.976...)
- 24F
- 25F
- 30000F/1001 (29.97...)
- 30F
- 50F
- 60000F/1001 (59.94...)
- 60F

A value of less than or equal to 0.0F may be returned, indicating that the rate is unknown or unspecified.

Return value specifies the field rate if `getScanMode()` returns `SCANMODE_INTERLACED` and the frame rate if `getScanMode()` returns `SCANMODE_PROGRESSIVE`.

Returns:

the frame or field rate of the output video

See Also:

`getScanMode()`

getScanMode

```
public int getScanMode()
```

Return the video scan mode, as specified by this object.

A value of `SCANMODE_UNKNOWN` MAY be returned, indicating that the scan mode is unknown or unspecified.

Returns:

one of `SCANMODE_UNKNOWN`, `SCANMODE_INTERLACED`, or `SCANMODE_PROGRESSIVE`.

org.ocap.hardware.device
Interface VideoZoomPreference

```
public interface VideoZoomPreference
```

An interface that defines the constants that can be used for specifying zoom mode preferences in an HVideoConfigTemplate.

See Also:

VideoFormatControl, HVideoConfigTemplate

Field Summary

static int	<p>ZOOM_MODE</p> <p>A value for use in the preference field of the setPreference, getPreferenceObject and getPreferencePriority methods in the HVideoConfigTemplate that indicates that the HVideoConfiguration implies a specific decoder format conversion as specified in an Integer object.</p>
------------	--

Field Detail

ZOOM_MODE

```
static final int ZOOM_MODE
```

A value for use in the preference field of the setPreference, getPreferenceObject and getPreferencePriority methods in the HVideoConfigTemplate that indicates that the HVideoConfiguration implies a specific decoder format conversion as specified in an Integer object. Valid preference objects for this preference are instances of Integer to be interpreted as a decoder format conversion constant for the profile (e.g., one of the VideoFormatControl DFC_* constants).

The ZOOM_MODE preference SHALL be applied to convert input video with one aspect ratio when presented on a full-screen HVideoDevice with another display aspect ratio when not overridden by an application-directed setting. That is, the ZOOM_MODE preference setting for the HVideoDevice is applied by default for newly created JMF Players or when VideoFormatControl.DFC_PLATFORM is the current decoder format conversion setting for a JMF Player.

Instances of HVideoConfigTemplate generated by the platform and returned to applications (e.g., from HVideoConfiguration.getConfigTemplate()) SHALL have this preference set to a valid platform-supported DFC constant (as an instance of Integer) with REQUIRED priority.

See Also:

Constant Field Values

Appendix I Revision History

The following ECNs were incorporated into version I02 of this specification:

ECN	Date Accepted	Title of EC
OCAP-DS-EXT-N-07.1061-1	7/16/07	Correct ocap:monitorapplication reference; Persistence; Reset host settings; AudioOutput updates
OCAP-DS-EXT-N-07.1149-1	1/22/08	Minor corrections to 5.2.1 and 5.2.3
OCAP-DS-EXT-N-08.1265-1	7/15/08	Replace reference to setOutputResolutionConfig with setOutputConfiguration
OCAP-DS-EXT-N-09.1389-2	9/30/09	Update references to base OCAP and Host specs; remove unchecked exceptions
OCAP-DS-EXT-N-09.1435-1	9/30/09	Make DS (setPowerMode()) backward compatible with OCAP 1.1.1