OpenCable™ Application Platform Specifications
OCAP Extensions

OCAP Home Networking Extension

OC-SP-OCAP-HNEXT-I11-130530

ISSUED

DISCLAIMER

This document is published by Cable Television Laboratories, Inc. ("CableLabs®").

CableLabs reserves the right to revise this document for any reason including, but not limited to, changes in laws, regulations, or standards promulgated by various agencies; technological advances; or changes in equipment design, manufacturing techniques, or operating procedures described, or referred to, herein. CableLabs makes no representation or warranty, express or implied, with respect to the completeness, accuracy, or utility of the document or any information or opinion contained in the report. Any use or reliance on the information or opinion is at the risk of the user, and CableLabs shall not be liable for any damage or injury incurred by any person arising out of the completeness, accuracy, or utility of any information or opinion contained in the document.

This document is not to be construed to suggest that any affiliated company modify or change any of its products or procedures, nor does this document represent a commitment by CableLabs or any cable member to purchase any product whether or not it meets the described characteristics. Nothing contained herein shall be construed to confer any license or right to any intellectual property, whether or not the use of any information herein necessarily utilizes such intellectual property. This document is not to be construed as an endorsement of any product or company or as the adoption or promulgation of any guidelines, standards, or recommendations.

Document Status Sheet

Document Control Number:	OC-SP-OCAP-HNEXT-I11-130530
Document Title:	OCAP Home Networking Extension
Revision History:	I01 – Released 5/19/05 I02 – Released 12/20/07 I03 – Released 4/18/08 I04 – Released 12/17/09 I05 – Released 6/3/10 I06 – Released 2/24/11 I07 – Released 5/12/11 I08 – Released 1/12/12 I09 – Released 5/31/12 I10 – Released 4/18/13 I11 – Released 5/30/13
Date:	May 30, 2013
Status:	~~Work in Progress~~ ~~Draft~~ **Issued** ~~Closed~~
Distribution Restrictions:	~~Author Only~~ ~~CL/Member~~ ~~CL/ 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

Contents

Figures

Tables

This page left blank intentionally.

1 SCOPE

This document defines a minimal profile specification for a Home Networking software environment for digital cable receivers, with or without local storage. This platform is a modular extension to the OpenCable Application Platform [OCAP]. The OCAP Specification and the OCAP Home Networking Extension 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 and hardware firms.

The OCAP Home Networking Extension is based on [OCAP] and includes that specification in its entirety.

1.1 OCAP Home Networking Extension Purpose

OCAP Home Networking Extension specification describes an application interface that includes all required Application Program Interfaces (APIs), content and data formats, and protocols up to the application level. Applications developed to the OCAP Home Networking Extension (HN) Specification will be executed on OpenCable-compliant Host devices. The OCAP HN Specification allows cable operators to deploy their applications and services on all OpenCable-compliant host devices connected to their networks.

The OCAP Home Networking 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 OCAP Home Networking is to enable competing implementations of the OCAP Home Networking platform by CE manufacturers.

1.2 OCAP Home Networking Extension Requirements

The OCAP Home Networking platform has been designed to meet specific requirements that are not commonly applied to other Home Networking environments. Some of these requirements are related to content protection obligations that cable operators face, to the fact that broadcast event descriptions might be maintained by applications, and to the fact that the platform supports DVR applications deployed by different service providers that have no *a priori* knowledge of each other and, therefore, may compete for resources.

1.3 OCAP Home Networking Environment

The OCAP Home Networking extensions have been designed to provide OCAP home networking applications access to digital entertainment content available on devices connected to a cable service subscriber's home network. This includes content stored on the home networking-capable OpenCable Host device hosting the applications, as well as on other home networking-capable OpenCable Hosts and non-OpenCable Host devices connected to the home network. The OCAP Home Networking APIs assume that the Host and at least some of the other devices connected to the home network implement a set of protocols supporting the following functions:

- Advertisement and discovery of connected devices
- Advertisement and discovery of digital entertainment content stored on devices
- Communication of information about a device's capability to serve and/or receive digital entertainment content
- Communication of information necessary to establish the transfer of digital entertainment content across the home network
- Communication of playback control, file operations, and other commands needed to support the user's enjoyment of digital entertainment content

If these protocols are not implemented by two or more devices connected to the home network, features supported by the OCAP Home Networking extensions are not useable. Specification of the protocols supporting the functions described above is out of scope for the OCAP Home Networking Extension specification.

Figure 1–1 illustrates a possible home network environment in which the OpenCable Host, implementing OCAP Home Network Extensions, can share digital entertainment content with other devices connected to the home network.

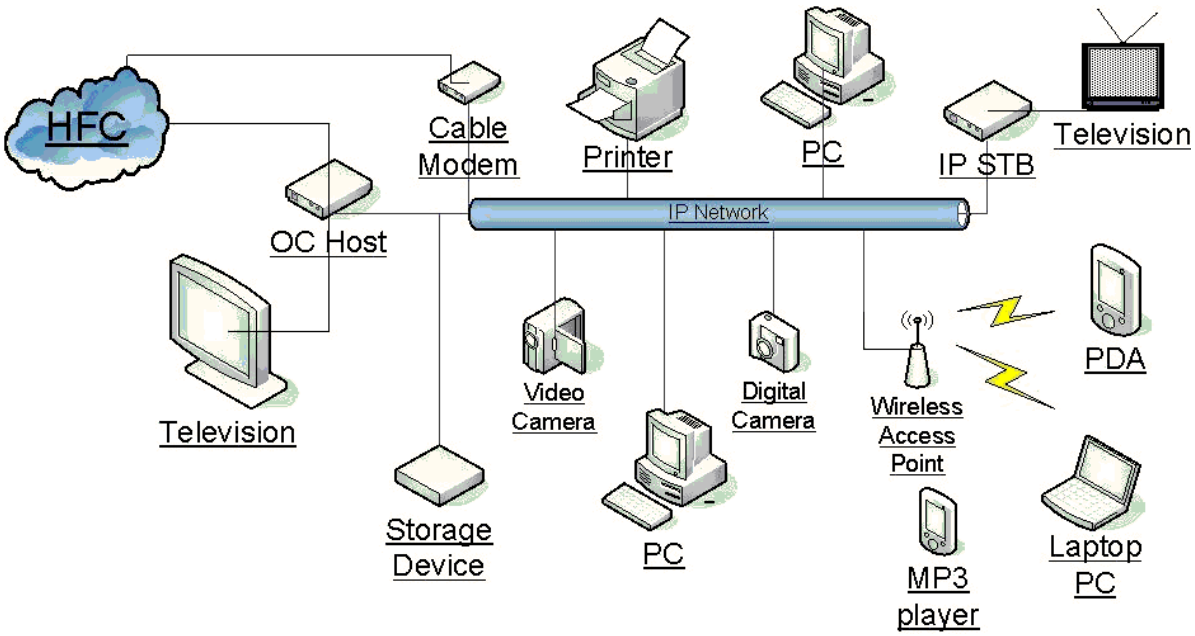

Figure 1–1 - Digital Home Entertainment System

1.4 OCAP Home Networking Application Areas (informative)

The information in this section is informative to the OCAP Home Networking Extensions Specification.

This section identifies the applications and services that could be made available to the viewer when using an OCAP Home Networking-compliant terminal. The descriptions of the applications are intended to demonstrate the scope of services required from OCAP.

1.4.1 Connected Device and Content Discovery

A critical application enabled by this platform is the discovery of connected devices and/or their audio or video content. Discovery of content available on the home network and its presentation to the user is the first step in setting up a transfer of content from a source (server) device to a rendering device, over a home network.

1.4.2 Presenting Content

Presentation of discovered multimedia content to the user offers ample opportunity for application innovation. How the content listing is organized, how it is displayed, how an item is selected, options offered for filtering and managing it, and presentation of metadata are some of the ways an application can differentiate itself and provide value for the end user.

1.4.3 Remote Recording

On set-top boxes that implement both home networking and DVR functionality, requests for scheduling recordings may originate from devices on the home network. Applications may receive, prioritize, and ultimately resolve these network requests to recordings on the local device.

1.4.4 Network Recording Requests

On a user's home network, there may be multiple devices that support DVR functionality. Applications may allow users to schedule DVR recordings on these devices from a single set-top box. Applications may also allow users to browse the existing schedule of recordings across all devices on the home network.

2 REFERENCES

This section provides the normative and informative references used to create this specification.

2.1 Normative References

Note: Information contained in these normative references is required for all implementations. Notwithstanding, intellectual property rights may be required to use or implement these normative references.

All references are subject to revision, and parties to agreement based on this specification are encouraged to investigate the possibility of applying the most recent editions of the documents listed below.

References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific:

- For a specific reference, subsequent revisions do not apply.
- For a non-specific, non-Bundle reference, the latest version applies.
- For non-specific CableLabs references that are part of the [OC-BUNDLE], the versions mandated in a particular Bundle apply.

References	Edition	Description
[OC-BUNDLE]		OC-SP-BUNDLE, OpenCable Bundle Requirements. See Section 2.3.1 to acquire this specification.
[HOST]		OpenCable Host Device 2.1 Core Functional Requirements, OC-SP-HOST2.1, Cable Television Laboratories, Inc. Referenced in [OC-BUNDLE].
[OCAP]		OpenCable Application Platform Specification (OCAP), OC-SP-OCAP, Cable Television Laboratories, Inc. Referenced in [OC-BUNDLE].
[UPnP CDS]	June 25, 2002	UPnP ContentDirectory v3, ContentDirectory:3 Service Template Version 1.01, UPnP Forum, September 30, 2008.
[HN-HOST]		OpenCable Host Home Networking Extension 2.0, OC-SP-HOST-HN2.0, Cable Television Laboratories, Inc. Referenced in [OC-BUNDLE].

2.2 Informative References

None.

2.3 Reference Acquisition

2.3.1 OpenCable Bundle Requirements

The OpenCable Bundle Requirements specification [OC-BUNDLE] indicates the set of CableLabs specifications required for the implementation of the OpenCable Bundle. The version number of [OC-BUNDLE] corresponds to the release number of the OpenCable Bundle that it describes. One or more versions of [OC-BUNDLE] reference this specification. Current and past versions of [OC-BUNDLE] may be obtained from CableLabs at http://www.cablelabs.com/opencable/specifications.

2.3.2 Other References

- Cable Television Laboratories, Inc., 858 Coal Creek Circle, Louisville, CO 80027;
  Phone +1-303-661-9100; Fax +1-303-661-9199; http://www.cablelabs.com
- UPnP Forum, www.upnp.org

3 GLOSSARY

The following definitions are used in this specification:

Connected Device — A physical device that is currently connected to a communication network and is compliant with protocols specified or referred to in this specification, enabling it to be discoverable by OCAP Home Networking applications through OCAP Home Networking APIs.

Content — Streaming media, file-based media, applications, and data, including metadata, that are discoverable by OCAP applications through OCAP Home Networking APIs.

Content Resource Properties — Information about content such as file name, file size, date created, and metadata.

Content Usage Rules — As defined in CHILA and DFAST

Device Capability — The ability of a given device to perform a specific function, such as server, data store, or renderer.

Device Property — A specific attribute of a given device.

Device Type — As defined in the XML Device Schema acquired via the discovery process or as defined in the CableLabs OpenCable Host specification.

DHCP — Dynamic Host Configuration Protocol. DHCP is an Internet standard for assigning IP addresses dynamically to IP hosts.

Discovery — The process by which OCAP HN applications find, directly or by proxy, devices and content that are exposed to OCAP applications.

HNHost — A cable receiver that is compliant with the hardware profile defined by the [HN-HOST] specification.

Home Network — A communication system over which Connected Devices may be discovered and content shared.

Legacy Device — A set-top box with embedded security.

OCAP Home Network — An Internet Protocol (IP)-based communication network in a cable television service subscriber's home that allows the discovery and sharing of content among devices capable of storing, receiving or transferring data within the confines of the subscriber's local area network (LAN) itself. The OCAP Home Network does not support or allow the transfer of copy-protected content beyond the physical boundaries of the subscriber's home LAN.

The OCAP Home Network extensions build upon and references underlying protocols and other capabilities defined (or to be defined) by the OpenCable specifications and/or other CableLabs specifications.

The physical network media are irrelevant to specification of the OCAP Home Networking extensions. Although this specification is largely independent of the underlying network protocols, references to those protocols may appear in portions of this specification.

OCAP Home Network Application	An OCAP application, as defined in [OCAP], designed to use OCAP Home Networking APIs to provide the cable subscriber value by enabling the sharing of digital entertainment content stored on Connected Devices, including but not limited to providing the following services: discovering Connected Devices, discovering content stored on Connected Devices, organizing and presenting to a user information about content stored on Connected Devices, and directing content on one Connected Device to be transferred to and rendered on another Connected Device.
OCAP Home Network Implementation	A hardware and software environment that conforms to the OCAP Home Networking Extensions specification.
OCAP Home Network Platform	The hardware, software, policies, and requirements definitions embodied in the OCAP HN and related specifications.
OpenCable Bundle	The OpenCable Bundle defines a set of specifications required to build a specific version of an OpenCable device. See [OC-BUNDLE].
Rendering	Display of coded content (audio, video, or still image).

4 ACRONYMS

The following acronyms are used in this specification:

API	Application Programming Interface
CCI	Copy Control Information
CHILA	CableLabs CableCARD-Host Interface License Agreement
DFAST	Dynamic Feedback Arrangement Scrambling Technique
DLNA	Digital Living Network Alliance
DVR	Digital Video Recorder
HN	Home Network or Home Networking
HNIMP	OCAP Home Network Implementation
ISO	International Organization for Standardization
IETF	Internet Engineering Task Force
LAN	Local Area Network
OCAP	OpenCable Application Platform
PRF	Permission Request File
PVR	Personal Video Recorder
RFC	Request for Comments
UPnP	Universal Plug and Play
URI	Uniform Resource Identifier

5 CONVENTIONS

The following conventions are used in this manual:

- The following font type is used to indicate code examples, names of properties, and other information that MUST be entered exactly as-is: `code example font`
- **Boldfaced** text is used as emphasis.

5.1 Specification Language

The following words are used throughout this document to define the significance of particular requirements:

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 valid reasons may exist 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, or the adjective OPTIONAL, 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.

Other text is descriptive or explanatory.

5.2 Organization

This document uses [OCAP] as its base. Where applicable, OCAP Home Networking sections reference the corresponding section within [OCAP].

The `org.ocap.hn` API package is defined in Annex A.

The `org.ocap.hn.content` API package is defined in Annex B.

The `org.ocap.hn.content.navigation` API package is defined in Annex C.

The `org.ocap.hn.profiles.upnp` API package is defined in Annex D.

The `org.ocap.hn.service` API package is defined in Annex E.

The `org.ocap.hn.recording` API package is defined in Annex F.

The `org.ocap.hn.security` API package is defined in Annex G.

The `org.ocap.hn.upnp` API package is defined in Annex H.

The `org.ocap.hn.resource` API package is defined in Annex I.

The `org.ocap.hn.ruihsrc` API package is defined in Annex J.

The `org.ocap.hn.transformation` API package is defined in Annex K.

6 OCAP HOME NETWORKING

This chapter describes the OCAP Home Networking (HN) platform. The OCAP HN APIs are defined and described in more detail in Annex A, Annex B, Annex C, Annex D, Annex E, Annex F, Annex G, Annex H, Annex I, Annex J, and Annex K.

6.1 Introduction

This specification fully defines a Home Networking extension to [OCAP]. Implementers of this specification SHALL also fully implement [OCAP]. [OCAP] and this Home Networking extension platform are related in such a way that [OCAP] implementations have no build-time or run-time dependencies on the Home Networking extensions, while OCAP Home Network Implementations depend on the full implementation of [OCAP].

An OpenCable Host that implements the OCAP Home Networking extension MAY implement the OCAP DVR extension as well.

OCAP Home Networking extensions rely upon support in the OCAP Home Network Implementation and in Connected Devices of a set of compatible, IP-based device and Content Discovery LAN communication protocols to expose information about the Connected Devices, their capabilities, and the digital entertainment Content they store, if any. If the hardware and software platform that implements the OCAP Home Networking APIs and Connected Devices do not implement these protocols, then information about the devices, their capabilities, and stored Content is not conveyed to the OCAP Home Network Application through the implementation and the OCAP Home Networking APIs. Connected Devices that implement device and Content Discovery LAN communication protocols compliant with protocols supported by the OCAP Home Network implementation, are referred to as protocol-compliant devices. Definition of the IP-based device and Content Discovery LAN communication protocols is outside the scope of this OCAP Home Networking Extension specification.

IP-based device and Content Discovery LAN communication protocols are specified in the OpenCable Host 2.0 Home Networking Extension specification.

It is assumed that Content stored on the OpenCable Host, using OCAP DVR functionality, could be exposed to protocol compliant devices sharing the IP-based communication network, and Content exposed by protocol compliant devices sharing the IP-based communication network, could be moved or copied to DVR Content storage resources on the OpenCable Host device.

This specification defines a platform to enable applications to discover devices on a Home Network, gain listings of Content resources on Connected Devices, and perform certain operations on Content resources, such as Rendering, copying, and deleting. These functions are considered basic capabilities of a home networking platform. Advanced functions, such as device-specific operations, are not supported by this platform. This chapter describes the platform in a textual format, and places normative requirements on implementations. Annexes to this specification define the platform APIs and contain detailed API descriptions and normative statements.

The OCAP Home Networking APIs assume that the OpenCable Host platform, in which they are implemented, provides an interface to an Internet Protocol (IP)-based communication network in a cable service subscriber's premises.

6.2 Device Discovery

OCAP Home Network Applications are provided with the ability to detect the presence of Connected Devices on the OCAP Home Network, discover the properties and capabilities of those devices, and detect changes in the status of

those devices. Device information acquired by the OCAP Home Network Implementation is exposed to OCAP applications, in particular to OCAP Home Network Applications, through the OCAP Home Networking APIs.

6.2.1 Device Discovery Process

The process used by the OCAP Home Network Implementation for discovering devices connected on the home network includes the following activities:

a) Maintaining a list of Connected Devices.

b) Maintaining metadata for each Connected Device, such that its type, sub-devices, capabilities, properties, and supported functionalities may be exposed to applications.

c) Generating application events when devices are added or removed from the list of Connected Devices.

6.3 Content Discovery and Listing

OCAP Home Networking APIs provide OCAP Home Network Applications with the ability to detect the presence of Content resources accessible on devices connected on the OCAP Home Network. Content information acquired by the OCAP Home Network Implementation is exposed to OCAP applications through the OCAP Home Networking APIs.

6.3.1 Content Discovery and Listing Process

The process for discovering Content on Connected Devices on the Home Network includes the following activities:

a) Generating, on application request, a list of Content resources available on a Connected Device.

b) Generating, on application request, a list of Content resources available on all Connected Devices.

c) Generating, on application request, a list containing a subset of Content resources available on a Connected Device, where the subset is defined by application specified search criteria.

d) Generating, on application request, metadata associated with Content resource.

e) Generating application events when Content resources are added, removed, or modified from a list of Content resources.

6.3.2 Content Searching

The ability to perform detailed searches for Content resources accessible on devices connected on the OCAP Home Network is exposed to OCAP applications through the OCAP Home Networking APIs. The search criteria used by an application to perform such searches SHALL follow the format defined by [UPnP CDS] section 2.5.5.1: Search Criteria String Syntax. Usage of the search string SHALL follow the usage specified by [UPnP CDS] section 2.5.5.2: Search Criteria String Semantics and Examples.

6.3.3 Content Metadata Caching

The home networking implementation SHALL NOT implicitly update metadata associated with any content entry that has been retrieved from the network due to an application request unless directed to do so by an application request. Accessing and inspection of metadata associated with a content entry SHALL NOT cause network activity. Navigation using a content entry (i.e., invocation of methods ContentContainer.getEntry/getEntries or

ContentEntry.getParentEntry) SHALL NOT cause network activity. The HNIMP SHALL cross-link parent-child relationships between the entries returned as a result of the same search where relationships exist.

6.4 Recording Operations

OCAP Home Networking applications are provided with the ability to perform a number of operations effecting scheduled future recordings and recorded Content items stored on devices connected to the OCAP Home Network, including recording schedule management operations, and Rendering.

6.4.1 Remote Recording

Remote Recording is the scheduling and management of DVR recordings on a Connected Device. The OCAP home networking APIs allow privileged applications to request that recordings hosted on a Connected Device be scheduled, deleted, prioritized, disabled, or rescheduled. It is up to the implementation of the Connected Device to determine whether and how such requests are honored.

6.4.2 Network Recording Request Management

Privileged applications may register themselves to be notified of requests for recording operations from Connected Devices on the home network. The home networking implementation SHALL notify registered monitor applications of recording operation requests originating from the OCAP Home Network. Behavior of the OCAP Home Networking implementation in response to network requests for recording operations when no request handler has been registered by any monitor application is implementation-dependent.

Included in the responsibilities of a registered recording request handler are:

a) Handling network requests to schedule recordings on the local device.

b) Handling network requests to modify or reschedule existing recordings on the local device.

c) Handling network requests to stop or cancel existing recordings on the local device.

d) Handling network requests to delete content associated with existing recordings on the local device.

e) Handling network requests to delete metadata associated with existing recordings on the local device.

f) Handling network requests to prioritize scarce resource allocation among existing recordings on the local device.

6.5 Permissions

The Java security model includes the means to restrict access to an API and to applications that have been requested and granted specific permissions. The OCAP Home Network Extension defines HomeNetPermission, a home networking specific permission used to control access to certain home networking APIs.

In addition to the HomeNetPermission class, access to content on the home network is restricted through file access permissions as defined by org.ocap.storage.ExtendedFileAccessPermissions. Access permissions can be retrieved using the ContentEntry.getExtendedFileAccessPermissions() method. Access permissions MAY be signaled through the home networking content discovery protocol. In the event that metadata associated with a ContentEntry does not include access permissions, the OCAP Home Network Implementation SHALL associate an ExtendedFileAccessPermission granting full access with the ContentEntry.

6.5.1 Unsigned Applications

Unsigned applications SHALL NOT be granted access to a home network via the HN API in the org.ocap.hn name space.

6.5.2 Signed Applications

Signed applications MAY perform home networking actions, via the OCAP Home Networking API, in the org.ocap.hn name space. Some of the method calls in the OCAP Home Networking API require org.ocap.hn.HomeNetPermission, which MUST be provided by the applications permission request file. See Section 7.1.

6.5.3 Monitor Applications

Monitor Application permissions are detailed in Section 7.2.

6.6 API Support Property

OCAP Hosts that support the OCAP HN extension SHALL indicate this support with the system properties as defined in section 13.3.12.2 of [OCAP] per Table 6–1.

Table 6–1 - API Support Property

Property	Description	Value	Application Access
ocap.api.option.hn	System property indicating that OCAP HN extension is supported by the Host device	"3.0"	signed and unsigned

6.7 OCAP Home Networking API

The OCAP Home Network Platform extends the OCAP-J API defined in [OCAP]. The additional packages, classes, and interfaces are listed here. For a complete definition and description of the APIs, see Annex A through Annex E. An implementation of the OCAP Home Network Platform SHALL include all of the packages, classes, and interfaces defined in [OCAP], as well as the following packages, classes, and interfaces.

6.7.1 Additions to OCAP Packages

The OCAP Home Network Platform adds no Java interfaces and classes to packages defined in [OCAP].

6.7.2 OCAP Home Networking Packages

The following packages are defined by the OCAP Home Network Platform:

org.ocap.hn

org.ocap.hn.content

org.ocap.hn.content.navigation

org.ocap.hn.profiles.upnp

org.ocap.hn.recording

org.ocap.hn.resource

org.ocap.hn.ruihsrc

org.ocap.hn.security

org.ocap.hn.service

org.ocap.hn.transformation

org.ocap.hn.upnp.client

org.ocap.hn.upnp.common

org.ocap.hn.upnp.server

These packages are described in Annex A through Annex K.

6.7.3 OCAP Home Networking UPnP Support

The OCAP Home Networking packages defined in Annex A through Annex G and Annex I through Annex K SHALL be implemented through the UPnP package defined in Annex H.

6.8 OCAP Application LAN Interface Communication

There are certain scenarios that require an HNHost to obtain a dynamic IP address lease, using DHCP, from a LAN-side device in order to send and receive traffic over the Internet using a home router/gateway. In addition, an OCAP HN application may require IP access to both the LAN and WAN interfaces concurrently. In such cases, a normal IP routing table containing a single default gateway address will not allow proper forwarding of IP packets to devices attached to the LAN interface. To accommodate these use cases, the HNHost is required to implement:

a) binding of source IP addresses to network sockets and

b) creation of multiple routing tables and the addition of specific routes and routing rules to facilitate policy-based routing as described in Section 5.9 of [HN-HOST].

The steps an application should follow for communication over the LAN interface are:

- Using the method `org.ocap.hn.NetworkInterface.getInetAddresses()`, obtain the list of IP addresses assigned to the LAN interface.
- Choose the IPv4 address belonging to the subnet designated for the LAN network.
- With this IP address and a chosen port number, create a `SocketAddress` using the method `java.net.InetSocketAddress()`.
- Create a `Socket`.
- Bind this `Socket` to the `SocketAddress` using the method `java.net.Socket.bind()`.

The binding process will ensure that all outgoing packets associated with this socket carry the chosen IP address in the source address field. The IP source address will be used to navigate to the proper routing table and forwarding of packets to the LAN interface.

7 SECURITY

The OCAP Home Network Platform may not be permitted to share all digital entertainment Content that is stored on the OpenCable Host device or other Connected Devices. Content can only be shared according to CCI bits, as defined in [HOST]. OCAP Home Networking API methods, which cause Content to be streamed or moved to a remote device, SHALL fail if the request violates copy protection signaling, or other security requirements imposed by OpenCable security specifications. These security aspects are transparent to applications and the OCAP Home Networking API does not provide methods for application access and control of them.

7.1 Home Network Permission

In addition to device authorizations, this specification defines the org.ocap.hn.HomeNetPermission class. This class can be used to grant permissions for various OCAP Home Networking API methods. See the Annexes containing APIs for method details where SecurityException can be thrown. For granting purposes, the HomeNetPermission is added to the permission request file (PRF) of an application. Referring to the PRF DTD definition given in [OCAP], Section 14.2.2.1.1, this specification modifies the permissionrequestfile ELEMENT and adds the ocap:homenetpermission as follows:

```
<!ELEMENT permissionrequestfile
(file?, capermission?, applifecyclecontrol?, returnchannel?, tuning?,
servicesel?, userpreferences?, network?, dripfeed?, persistentfilecredential*,
ocap:monitorapplication*, ocap:servicetypepermission*, ocap:homenetpermission*)>
```

The ocap:homenetpermission ELEMENT SHALL be added after the ocap:servicetypepermission in the PRF as follows:

```
<!ELEMENT ocap:homenetpermission EMPTY>
<!ATTLIST ocap:homenetpermission
type (contentmanagement | contentlisting | recording | recordinghandler)
value (true | false) "false"
>
```

7.2 Monitor Application Permission

Host devices that implement the OCAP HN Extension SHALL support the `MonitorAppPermission("handler.homenetwork")` permission name. Annex Q of [OCAP] is extended as follows:

The table within the description of MonitorAppPermission is extended to include the following rows:

Permission Name	What the Permission Allows	Description
handler.homenetwork	Provides access to network passwords and authorization methods.	This permission allows the caller to get and set network passwords as well as register as the authorization handler.

In addition, Section 14.2.2.1.1 in [OCAP] is extended as follows: the enumerated token value type of the `name` attribute of the `ocap:monitorapplication` element type defined by the DTD of the PRF SHALL be considered to contain the "handler.homenetwork" values.

8 REGISTRY OF CONSTANTS

This section contains OCAP-specific constants for the home network profile. All constants are public final. These OCAP-specific JAVA constants are set in the following packages:

org.ocap.hn.ContentServerEvent		
public static final int	CONTENT_ADDED	0
public static final int	CONTENT_CHANGED	2
public static final int	CONTENT_REMOVED	1

org.ocap.hn.Device		
public static final java.lang.String	CAP_RECORDING_SUPPORTED	"RecordingSupported"
public static final java.lang.String	CAP_REMOTE_STORAGE_SUPPORTED	"RemoteStorageSupported"
public static final java.lang.String	CAP_STREAMING_SUPPORTED	"StreamingSupported"
public static final java.lang.String	CAP_TUNER_SUPPORTED	"TunerSupported"
public static final java.lang.String	PROP_DEVICE_TYPE	"deviceType"
public static final java.lang.String	PROP_DEVICE_VERSION	"deviceVersion"
public static final java.lang.String	PROP_FRIENDLY_NAME	"friendlyName"
public static final java.lang.String	PROP_LOCATION	"location"
public static final java.lang.String	PROP_MANUFACTURER	"manufacturer"
public static final java.lang.String	PROP_MANUFACTURER_URL	"manufacturerURL"
public static final java.lang.String	PROP_MIDDLEWARE_PROFILE	"middlewareProfile"
public static final java.lang.String	PROP_MIDDLEWARE_VERSION	"middlewareVersion"
public static final java.lang.String	PROP_MODEL_DESCRIPTION	"modelDescription"
public static final java.lang.String	PROP_MODEL_NAME	"modelName"
public static final java.lang.String	PROP_MODEL_NUMBER	"modelNumber"
public static final java.lang.String	PROP_MODEL_URL	"modelURL"

org.ocap.hn.Device		
public static final java.lang.String	PROP_PRESENTATION_URL	"presentationURL"
public static final java.lang.String	PROP_SERIAL_NUMBER	"serialNumber"
public static final java.lang.String	PROP_UDN	"UDN"
public static final java.lang.String	PROP_UPC	"UPC"
public static final java.lang.String	TYPE_BINARY_LIGHT	"BinaryLight"
public static final java.lang.String	TYPE_DIMMABLE_LIGHT	"DimmableLight"
public static final java.lang.String	TYPE_HVAC_SYSTEM	"HVAC_System"
public static final java.lang.String	TYPE_HVAC_ZONE_THERMOSTAT	"HVAC_ZoneThermostat"
public static final java.lang.String	TYPE_INTERNET_GATEWAY_DEVICE	"InternetGatewayDevice"
public static final java.lang.String	TYPE_LAN_DEVICE	"LANDevice"
public static final java.lang.String	TYPE_MEDIA_RENDERER	"MediaRenderer"
public static final java.lang.String	TYPE_MEDIA_SERVER	"MediaServer"
public static final java.lang.String	TYPE_OCAP_HOST	"OCAP_Host"
public static final java.lang.String	TYPE_OCAP_TERMINAL	"OCAP_Terminal"
public static final java.lang.String	TYPE_PRINTER	"printer"
public static final java.lang.String	TYPE_REMOTE_UI_CLIENT_DEVICE	"RemoteUIClientDevice"
public static final java.lang.String	TYPE_REMOTE_UI_SERVER_DEVICE	"RemoteUIServerDevice"
public static final java.lang.String	TYPE_SCANNER	"Scanner"
public static final java.lang.String	TYPE_WAN_CONNECTION_DEVICE	"WANConnectionDevice"
public static final java.lang.String	TYPE_WAN_DEVICE	"WANDevice"
public static final java.lang.String	TYPE_WLAN_ACCESS_POINT_DEVICE	"WLANAccessPointDevice"

org.ocap.hn.DeviceEvent		
public static final int	DEVICE_ADDED	100

org.ocap.hn.DeviceEvent		
public static final int	DEVICE_REMOVED	101
public static final int	DEVICE_UPDATED	102
public static final int	STATE_CHANGE	201

org.ocap.hn.NetActionEvent		
public static final int	ACTION_CANCELED	1
public static final int	ACTION_COMPLETED	0
public static final int	ACTION_FAILED	2
public static final int	ACTION_IN_PROGRESS	4
public static final int	ACTION_STATUS_NOT_AVAILABLE	3

org.ocap.hn.NetModule		
public static final java.lang.String	CONTENT_LIST	"ContentList"
public static final java.lang.String	CONTENT_MANAGER	"ContentManager"
public static final java.lang.String	CONTENT_RECORDER	"ContentRecorder"
public static final java.lang.String	CONTENT_RENDERER	"ContentRenderer"
public static final java.lang.String	CONTENT_SERVER	"ContentServer"
public static final java.lang.String	PROP_CONTROL_URL	"ControlURL"
public static final java.lang.String	PROP_DESCRIPTION_URL	"DescriptionURL"
public static final java.lang.String	PROP_EventSub_URL	"EventSubURL"
public static final java.lang.String	PROP_NETMODULE_ID	"NetModuleId"
public static final java.lang.String	PROP_NETMODULE_TYPE	"NetModuleType"

org.ocap.hn.NetModuleEvent		
public static final int	MODULE_ADDED	100
public static final int	MODULE_BUSY	103
public static final int	MODULE_REMOVED	101
public static final int	MODULE_UPDATED	102
public static final int	STATE_CHANGE	201

org.ocap.hn.NetworkInterface		
public static final int	MOCA	1
public static final int	UNKNOWN	0
public static final int	WIRED_ETHERNET	2
public static final int	WIRELESS_ETHERNET	3

org.ocap.hn.content.ContentContainer		
public static final java.lang.String	ALBUM_CONTAINER	"object.container.album"

org.ocap.hn.content.ContentContainer		
public static final java.lang.String	ALBUM_CONTAINER_MUSIC	"object.container.album .musicAlbum"
public static final java.lang.String	ALBUM_CONTAINER_PHOTO	"object.container.album .photoAlbum"
public static final java.lang.String	CHANNEL_GROUP_CONTAINER	"object.container. channelGroup"
public static final java.lang.String	CONTAINER	"object.container"
public static final java.lang.String	GENRE_CONTAINER	"object.container.genre"
public static final java.lang.String	GENRE_CONTAINER_MOVIE	"object.container.genre .movieGenre"
public static final java.lang.String	GENRE_CONTAINER_MUSIC	"object.container.genre .musicGenre"
public static final java.lang.String	PERSON_CONTAINER	"object.container.person"
public static final java.lang.String	PERSON_CONTAINER_MUSIC _ARTIST	"object.container.person .musicArtist"
public static final java.lang.String	PLAYLIST_CONTAINER	"object.container .playlistContainer"
public static final java.lang.String	STORAGE_FOLDER_CONTAINER	"object.container .storageFolder"
public static final java.lang.String	STORAGE_SYSTEM_CONTAINER	"object.container .storageSystem"
public static final java.lang.String	STORAGE_VOLUME_CONTAINER	"object.container .storageVolume"

org.ocap.hn.content.ContentItem		
public static final java.lang.String	AUDIO_ITEM	"object.item.audioItem"
public static final java.lang.String	AUDIO_ITEM_BOOK	"object.item.audioItem .audioBook"
public static final java.lang.String	AUDIO_ITEM_BROADCAST	"object.item.audioItem .audioBroadcast"
public static final java.lang.String	AUDIO_ITEM_TRACK	"object.item.audioItem .musicTrack"
public static final java.lang.String	IMAGE_ITEM	"object.item.imageItem"
public static final java.lang.String	IMAGE_ITEM_PHOTO	"object.item.imageItem.photo"
public static final java.lang.String	ITEM	"object.item"
public static final java.lang.String	VIDEO_ITEM	"object.item.videoItem"
public static final java.lang.String	VIDEO_ITEM_BROADCAST	"object.item.videoItem .videoBroadcast"

org.ocap.hn.content.ContentItem		
public static final java.lang.String	VIDEO_ITEM_BROADCAST_VOD	"object.item.videoItem.videoBroadcast.vod"
public static final java.lang.String	VIDEO_ITEM_MOVIE	"object.item.videoItem.movie"
public static final java.lang.String	VIDEO_ITEM_MUSIC_CLIP	"object.item.videoItem.musicVideoClip"
public static final java.lang.String	VIDEO_ITEM_VPOP	"object.item.videoItem.vpop"

org.ocap.hn.content. ContentProfile		
public static final java.lang.String	DASH_MPD	"DASH_MPD"

org.ocap.hn.content.ContentResource		
public static final java.lang.String	UNKNOWN_MIME_TYPE	"unknown"

org.ocap.hn.content.DatabaseException		
public static final int	FIELD_IS_EMPTY	3
public static final int	FIELD_IS_WRONG_FORMAT	4
public static final int	FIELD_NAME_DOES_NOT_EXIST	1
public static final int	GENERAL_ERROR	9
public static final int	INVALID_PARAMETER_SPECIFIED	6
public static final int	QUERY_IS_INVALID	5
public static final int	REMOTE_QUERY_IS_INVALID	8
public static final int	UNABLE_TO_LOCATE_SERVICE	7

org.ocap.hn.content.MetadataIdentifiers		
public static final java.lang.String	PROPRIETARY_DATA	"ocap:proprietaryData"

org.ocap.hn.content.ProtectionType		
public static final java.lang.String	DECE_DRM	"DECE-DRM"
public static final java.lang.String	DTCP_IP	"DTCP-IP"

org.ocap.hn.content. StreamingActivityListener		
public static final int	ACTIVITY_END_ACCESS_WITHDRAWN	4
public static final int	ACTIVITY_END_NETWORK_TIMEOUT	11
public static final int	ACTIVITY_END_OTHER	255
public static final int	ACTIVITY_END_RESOURCE_REMOVED	3
public static final int	ACTIVITY_END_SERVICE_VANISHED	1
public static final int	ACTIVITY_END_USER_STOP	5
public static final int	CONTENT_TYPE_ALL_RESOURCES	0

org.ocap.hn.content. StreamingActivityListener		
public static final int	CONTENT_TYPE_LIVE_RESOURCES	1
public static final int	CONTENT_TYPE_RECORDED_RESOURCES	2

org.ocap.hn.content.navigation.DatabaseQuery		
public static final int	CONTAINS	6
public static final int	EQUALS	1
public static final int	EXISTS	8
public static final int	GREATER_THAN	2
public static final int	GREATER_THAN_OR_EQUALS	4
public static final int	LESS_THAN	3
public static final int	LESS_THAN_OR_EQUALS	5
public static final int	NOT_EQUALS	7

org.ocap.hn.profiles.upnp.UPnPConstants		
public static final java.lang.String	ACTOR	"upnp:actor"
public static final java.lang.String	ACTOR_AT_ROLE	"upnp:actor@role"
public static final java.lang.String	ALBUM	"upnp:album"
public static final java.lang.String	ALBUM_ART	"upnp:albumArtURI"
public static final java.lang.String	ARTIST	"upnp:artist"
public static final java.lang.String	ARTIST_AT_ROLE	"upnp:artist@role"
public static final java.lang.String	ARTIST_DISCOGRAPHY	"upnp:artistDiscographyURI"
public static final java.lang.String	AUTHOR	"upnp:author"
public static final java.lang.String	AUTHOR_AT_ROLE	"upnp:author@role"
public static final java.lang.String	CHANNEL_NAME	"upnp:channelName"
public static final java.lang.String	CHANNEL_NUMBER	"upnp:channelNr"
public static final java.lang.String	COMMENTS	"upnp:userAnnotation"
public static final java.lang.String	CONTRIBUTOR	"dc:contributor"
public static final java.lang.String	CREATION_DATE	"dc:date"

org.ocap.hn.profiles.upnp.UPnPConstants		
public static final java.lang.String	CREATOR	"dc:creator"
public static final java.lang.String	DESCRIPTION	"dc:description"
public static final java.lang.String	DIRECTOR	"upnp:director"
public static final java.lang.String	DVD_REGION_CODE	"upnp:DVDRegionCode"
public static final java.lang.String	GENRE	"upnp:genre"
public static final java.lang.String	ICON_REF	"upnp:icon"
public static final java.lang.String	ID	"id"
public static final java.lang.String	LANGUAGE	"dc:language"
public static final java.lang.String	LONG_DESCRIPTION	"upnp:longDescription"
public static final java.lang.String	LYRICS_REF	"upnp:lyricsURI"
public static final java.lang.String	MEDIA_ID	"upnp:toc"
public static final java.lang.String	PARENT_ID	"parentID"
public static final java.lang.String	PLAYLIST	"upnp:playlist"
public static final java.lang.String	PRODUCER	"upnp:producer"
public static final java.lang.String	PROP_STORAGE_FREE	"upnp:storageFree"
public static final java.lang.String	PROP_STORAGE_TOTAL	"upnp:storageTotal"
public static final java.lang.String	PUBLISHER	"dc:publisher"
public static final java.lang.String	RADIO_BAND	"upnp:radioBand"
public static final java.lang.String	RADIO_CALL_SIGN	"upnp:radioCallSign"
public static final java.lang.String	RADIO_STATION_ID	"upnp:radioStationID"
public static final java.lang.String	RATING	"upnp:rating"
public static final java.lang.String	REGION	"upnp:region"
public static final java.lang.String	RELATION	"dc:relation"

org.ocap.hn.profiles.upnp.UPnPConstants		
public static final java.lang.String	RIGHTS	"dc:rights"
public static final java.lang.String	SCHEDULED_END_TIME	"upnp:scheduledEndTime"
public static final java.lang.String	SCHEDULED_START_TIME	"upnp:scheduledStartTime"
public static final java.lang.String	STORAGE_MEDIUM	"upnp:storageMedium"
public static final java.lang.String	TITLE	"dc:title"
public static final java.lang.String	TRACK_NUMBER	"upnp:originalTrackNumber"

org.ocap.hn.recording.NetRecordingEntry		
public static final java.lang.String	PROP_CDS_REFERENCE	"ocap:cdsReference"
public static final java.lang.String	PROP_RCI_LIST	"ocap:RCIList"
public static final java.lang.String	PROP_SCHEDULED_CDS_ENTRY_ID	"ocap:scheduledCDSEntryID"

org.ocap.hn.recording.RecordingContentItem		
public static final java.lang.String	PROP_ACCESS_PERMISSIONS	"ocap:accessPermissions"
public static final java.lang.String	PROP_APP_ID	"ocap:appID"
public static final java.lang.String	PROP_CONTENT_URI	"ocap:contentURI"
public static final java.lang.String	PROP_DESTINATION	"ocap:destination"
public static final java.lang.String	PROP_DURATION	"ocap:scheduledDuration"
public static final java.lang.String	PROP_EXPIRATION_PERIOD	"ocap:expirationPeriod"
public static final java.lang.String	PROP_MEDIA_FIRST_TIME	"ocap:mediaFirstTime"
public static final java.lang.String	PROP_MSO_CONTENT	"ocap:msoContentIndicator"
public static final java.lang.String	PROP_NET_RECORDING_ENTRY	"ocap:netRecordingEntry"
public static final java.lang.String	PROP_ORGANIZATION	"ocap:organization"
public static final java.lang.String	PROP_PRESENTATION_POINT	"ocap:mediaPresentationPoint"

org.ocap.hn.recording.RecordingContentItem		
public static final java.lang.String	PROP_PRIORITY_FLAG	"ocap:priorityFlag"
public static final java.lang.String	PROP_RECORDING_STATE	"ocap:taskState"
public static final java.lang.String	PROP_RETENTION_PRIORITY	"ocap:retentionPriority"
public static final java.lang.String	PROP_SOURCE_ID	"ocap:scheduledChannelID"
public static final java.lang.String	PROP_SOURCE_ID_TYPE	"ocap:scheduledChannelIDType"
public static final java.lang.String	PROP_SPACE_REQUIRED	"ocap:spaceRequired"
public static final java.lang.String	PROP_START_TIME	"ocap:scheduledStartDateTime"

ocap.hn.transformation.TransformationListener		
public static final int	REASON_CONTENTITEM_DELETED	2
public static final int	REASON_NONMATCHING_INPUT_PROFILE	3
public static final int	REASON_RESOURCE_UNAVAILABLE	1
public static final int	REASON_UNKNOWN	0

org.ocap.hn.upnp.common.UPnPGeneralErrorResponse		
public static final int	NETWORK_ERROR	2
public static final int	NETWORK_TIMEOUT	1

Annex A Home Networking API

Package org.ocap.hn

Interface Summary	
ContentServerListener	Listener interface for classes which are interested in changes to a ContentServerNetModule.
ContentServerNetModule	Class representing a NetModule which serves content.
Device	The `Device` interface represents a home network device that supports homenetwork NetModules.
DeviceEventListener	DeviceEvent callback interface.
NetActionHandler	This interface represents a handler passed to asynchronous methods.
NetActionRequest	All asynchronous actions in the Home networking API return an NetActionRequest.
NetList	A list comprising of home network elements such as Device or NetModule.
NetModule	NetModule is an abstraction of functionality that is provided by a `Device`.
NetModuleEventListener	NetModuleEvent callback interface.

Class Summary	
ContentServerEvent	Event which will be sent to registered ContentServerListeners when ContentEntrys have been added, changed or removed.
DeviceEvent	Represents a Device Event.
HomeNetPermission	The HomeNetPermission class represents permission to execute privileged home networking operations only signed applications MAY be granted.
NetActionEvent	This class represents an event generated in response to an action.
NetManager	The NetManager is a singleton class that registers all the Devices and NetModules within a home network.
NetModuleEvent	Entity for NetModule Event.
NetworkInterface	This class represents a home network interface including MoCA, wired ethernet, and wireless ethernet.
PropertyFilter	The filter for (key,value) pair filtering mechanism.

Exception Summary	
NotAuthorizedException	Exception indicating that the application has no permission to perform certain action.

org.ocap.hn

Class ContentServerEvent

```
java.lang.Object
  └java.util.EventObject
      └org.ocap.hn.ContentServerEvent
```

All Implemented Interfaces:

java.io.Serializable

```
public class ContentServerEvent
extends java.util.EventObject
```

Event which will be sent to registered ContentServerListeners when ContentEntrys have been added, changed or removed.

See Also:

Serialized Form

Field Summary

static int	**CONTENT_ADDED** Event ID indicating that content got added to a ContentServerNetModule.
static int	**CONTENT_CHANGED** Event ID indicating that metadata associated with content has been updated.
static int	**CONTENT_REMOVED** Event ID indicating that content got removed from a ContentServerNetModule

Fields inherited from class java.util.EventObject
source

Constructor Summary

ContentServerEvent(java.lang.Object source, java.lang.String[] content, int evt)

Creates a new ContentServerEvent with the given source object, the ContentItem involved and an event ID indicating whether the content got added or removed.

Method Summary

java.lang.String[]	**getContent**() Returns the IDs associated with the ContentEntrys involved in this event.
ContentServerNetModule	**getContentServerNetModule**() Returns the ContentServerNetModule.
int	**getEventID**() Gets the event ID for this event.

Methods inherited from class java.util.EventObject
`getSource, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Field Detail

CONTENT_ADDED

```
public static final int CONTENT_ADDED
```

Event ID indicating that content got added to a ContentServerNetModule. This event SHALL NOT guarantee that any content items or associated metadata have been communicated to the local device. Applications should utilize the content browsing and searching APIs to retrieve any added content items.

See Also:

`ContentServerNetModule`, Constant Field Values

CONTENT_REMOVED

```
public static final int CONTENT_REMOVED
```

Event ID indicating that content got removed from a ContentServerNetModule

See Also:

Constant Field Values

CONTENT_CHANGED

```
public static final int CONTENT_CHANGED
```

Event ID indicating that metadata associated with content has been updated. This event SHALL NOT guarantee that and changes to content items or associated metadata have been communicated to the local device. Applications should utilize the content browsing and searching APIs to retrieve any updated metadata.

See Also:

`ContentServerNetModule`, Constant Field Values

Constructor Detail

ContentServerEvent

```
public ContentServerEvent(java.lang.Object source,
                          java.lang.String[] content,
                          int evt)
```

Creates a new ContentServerEvent with the given source object, the ContentItem involved and an event ID indicating whether the content got added or removed.

Parameters:

`source` - The source of this event. This must be a ContentServerNetModule.

`content` - the IDs of the ContentEntrys involved.

`evt` - the Event ID, either CONTENT_ADDED,CONTENT_REMOVED or CONTENT_CHANGED.

Method Detail

getContent

`public java.lang.String[] getContent()`

Returns the IDs associated with the ContentEntrys involved in this event.

Returns:

the string IDs of the entries involved.

getContentServerNetModule

`public ContentServerNetModule getContentServerNetModule()`

Returns the ContentServerNetModule. This is the source object of the event.

Returns:

the ContentServerNetModule containing the ContentItem that was added/removed/changed

getEventID

`public int getEventID()`

Gets the event ID for this event. Valid values are CONTENT_ADDED, CONTENT_CHANGED and CONTENT_REMOVED

Returns:

the ID for this event

org.ocap.hn

Interface ContentServerListener

All Superinterfaces:

java.util.EventListener

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

Listener interface for classes which are interested in changes to a ContentServerNetModule.

Method Summary

void	**contentUpdated**(ContentServerEvent evt) Called when a ContentEntry has been added, changed or removed from the ContentServerNetModule

Method Detail

contentUpdated

```
void contentUpdated(ContentServerEvent evt)
```

Called when a ContentEntry has been added, changed or removed from the ContentServerNetModule

Parameters:

evt - the ContentServerEvent

org.ocap.hn

Interface ContentServerNetModule

All Superinterfaces:
NetModule

```
public interface ContentServerNetModule
extends NetModule
```

Class representing a NetModule which serves content.

NetModules which implement this interface SHALL have a NetModule.PROP_NETMODULE_TYPE property value of NetModule.CONTENT_SERVER.

Field Summary

Fields inherited from interface org.ocap.hn.NetModule
CONTENT_LIST, CONTENT_MANAGER, CONTENT_RECORDER, CONTENT_RENDERER, CONTENT_SERVER, PROP_CONTROL_URL, PROP_DESCRIPTION_URL, PROP_EventSub_URL, PROP_NETMODULE_ID, PROP_NETMODULE_TYPE

Method Summary

void	**addContentServerListener**(ContentServerListener listener) Adds a ContentServerListener to this ContentContainer.
void	**addStreamingActivityListener**(StreamingActivityListener listener, int contentTypes) Adds an StreamingActivityListener to this content server.
void	**removeContentServerListener**(ContentServerListener listener) Removes the specified ContentServerListener.
void	**removeStreamingActivityListener**(StreamingActivityListener listener) Removes the specified StreamingActivityListener for all contentItem types specified in addStreamingActivityListener
NetActionRequest	**requestBrowseEntries**(java.lang.String startingEntryID, java.lang.String propertyFilter, boolean browseChildren, int startingIndex, int requestedCount, java.lang.String sortCriteria, NetActionHandler handler) Requests a browse of this ContentServer which results in the creation of a ContentList.
NetActionRequest	**requestRootContainer**(NetActionHandler handler) returns the root ContentContainer for this ContentServerNetModule.
NetActionRequest	**requestSearchCapabilities**(NetActionHandler handler) Returns the list of property keys which applications can search against on this ContentServer using the requestSearchEntries(java.lang.String, java.lang.String, int, int, java.lang.String, java.lang.String, org.ocap.hn.NetActionHandler) method.

Method Summary

NetActionRequest	**requestSearchEntries**(java.lang.String parentID, java.lang.String propertyFilter, int startingIndex, int requestedCount, java.lang.String searchCriteria, java.lang.String sortCriteria, NetActionHandler handler) Requests a search of this ContentServer which results in the creation of a ContentList.
void	**setServiceResolutionHandler**(ServiceResolutionHandler handler) Adds a ServiceResolutionHandler to this ContentServerNetModule.

Methods inherited from interface org.ocap.hn.NetModule
addNetModuleEventListener, getDevice, getKeys, getNetModuleId, getNetModuleType, getProperty, isLocal, removeNetModuleEventListener

Method Detail

requestRootContainer

NetActionRequest **requestRootContainer**(NetActionHandler handler)

returns the root ContentContainer for this ContentServerNetModule. This is an asynchronous method. The caller gets informed via NetActionHandler.notify(NetActionEvent) of the process. On success, an NetActionEvent is created where the NetActionEvent.getResponse() method will return a ContentContainer object representing the root container for this ContentServerNetModule.

Parameters:

handler - NetActionHandler which gets informed once this asynchronous request completes

Returns:

NetActionRequest See NetActionRequest

Throws:

java.lang.SecurityException - if the caller does not have HomeNetPermission("contentlisting")

requestSearchCapabilities

NetActionRequest **requestSearchCapabilities**(NetActionHandler handler)

Returns the list of property keys which applications can search against on this ContentServer using the requestSearchEntries(java.lang.String, java.lang.String, int, int, java.lang.String, java.lang.String, org.ocap.hn.NetActionHandler) method. This is an asynchronous method. The caller gets informed via NetActionHandler.notify(NetActionEvent) of the process. On success an NetActionEvent is created where the NetActionEvent.getResponse() method will return an array of String objects containing the valid property keys. A return of an array with zero length indicates that this server supports no searching functionality. A return containing "*" indicates that any key associated with any content entry on this server may be used.

Parameters:

handler - NetActionHandler which gets informed once this asynchronous request completes

Returns:

NetActionRequest See NetActionRequest

requestBrowseEntries

NetActionRequest **requestBrowseEntries**(java.lang.String startingEntryID,

```
                                    java.lang.String propertyFilter,
                                    boolean browseChildren,
                                    int startingIndex,
                                    int requestedCount,
                                    java.lang.String sortCriteria,
                                    NetActionHandler handler)
```

Requests a browse of this ContentServer which results in the creation of a `ContentList`.

`ContentEntry` objects hosted on the remote server will be browsed starting at the ContentEntry specified. The propertyFilter parameter of this method SHALL contain a comma separated list of properties indicating which metadata fields should be returned in the ContentEntry objects contained in the resulting ContentList. A filter value of "*" indicates all available metadata be returned. The sortCriteria parameter of this method is a string containing the properties and sort modifiers to be used to sort the resulting ContentList. The format of the string containing the sort criteria shall follow the format defined in UPnP Content Directory Service 3.0 specification section 2.3.16: A_ARG_TYPE_SortCriteria.

This is an asynchronous method. The caller gets informed via `NetActionHandler.notify(NetActionEvent)` of the process. On success an `NetActionEvent` is created where the `NetActionEvent.getResponse()` method will return a `ContentList` containing the search results. If no matches are found, this value SHALL be a ContentList with zero entries. A return from `NetActionEvent.getActionStatus()` of `NetActionEvent.ACTION_COMPLETED` SHALL indicate that a valid `ContentList` will be returned from `NetActionEvent.getResponse()`.

Parameters:
`startingEntryID` - the ID of the ContentEntry on the server to start the browse from. A value of "0" SHALL indicate the root container on this server.
`propertyFilter` - the set of property values to return from this browse operation
`browseChildren` - if set to true, this operation will browse all of the direct children of the startingEntryID parameter. If false, this operation will return a content list containing the entry identified by startingEntryID only.
`startingIndex` - starting zero-based offset to enumerate children under the container specified by `parent`.
`requestedCount` - requested number of entries under the `ContentContainer` specified by `parent`. Setting this parameter to `0` indicates request all entries.
`sortCriteria` - properties and sort modifiers to be used to sort the resulting ContentList
`handler` - NetActionHandler which gets informed once the results `ContentList` is created or an error occurs. calling `getResponse()` on handler will return a `ContentList` containing the requested entries, or if the call was unsuccessful will return an error message supplied by the server.
Returns:
NetActionRequest See `NetActionRequest`.
Throws:
`java.lang.IllegalArgumentException` - if the startingEntryID is not available on this ContentServerNetModule, or if the handler parameter is null.
`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentlisting")

requestSearchEntries

```
NetActionRequest requestSearchEntries(java.lang.String parentID,
                                      java.lang.String propertyFilter,
                                      int startingIndex,
                                      int requestedCount,
                                      java.lang.String searchCriteria,
                                      java.lang.String sortCriteria,
                                      NetActionHandler handler)
```

Requests a search of this ContentServer which results in the creation of a `ContentList`.

`ContentEntry` objects hosted on the remote server will be searched for using the specified search criteria. The format of the string containing the search criteria SHALL follow the format defined by the UPnP Content Directory Service 3.0 specification section 2.3.13.1: Search Criteria String Syntax. The propertyFilter parameter of this method SHALL contain a comma separated list of properties indicating which metadata fields should be returned in the ContentEntry objects contained in the resulting ContentList. A filter value of "*" indicates all available metadata be returned. The sortCriteria parameter of this method is a string containing the properties and sort modifiers to be used to sort the resulting ContentList. The format of the string containing the sort criteria shall follow the format defined in UPnP Content Directory Service 3.0 specification section 2.3.16: A_ARG_TYPE_SortCriteria.

This is an asynchronous method. The caller gets informed via `NetActionHandler.notify(NetActionEvent)` of the process. On success an `NetActionEvent` is created where the `NetActionEvent.getResponse()` method will return a `ContentList` containing the search results. If no matches are found, this value SHALL be a ContentList with zero entries. A return from `NetActionEvent.getActionStatus()` of `NetActionEvent.ACTION_COMPLETED` SHALL indicate that a valid `ContentList` will be returned from `NetActionEvent.getResponse()`.

Parameters:
`parentID` - the ID of the ContentContainer on the server to start the search from. A value of "0" SHALL indicate the root container on this server.
`propertyFilter` - the set of property values to return from this browse operation
`startingIndex` - starting zero-based offset to enumerate children under the container specified by `parent`.
`requestedCount` - requested number of entries under the `ContentContainer` specified by `parent`. Setting this parameter to `0` indicates request all entries.
`searchCriteria` - contains the criteria string to search for. If this parameter is null, the implementation SHALL consider all entries in the parent container as matching the search criteria.
`sortCriteria` - properties and sort modifiers to be used to sort the resulting ContentList
`handler` - NetActionHandler which gets informed once the results `ContentList` is created or an error occurs. calling `getResponse()` on handler will return a `ContentList` containing the requested entries, or if the call was unsuccessful will return an error message supplied by the server.
Returns:
NetActionRequest See `NetActionRequest`.
Throws:
`java.lang.IllegalArgumentException` - if the startingEntryID is not available on this ContentServerNetModule, or if the handler parameter is null.
`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentlisting")

addContentServerListener

`void` **`addContentServerListener`**`(ContentServerListener listener)`

Adds a ContentServerListener to this ContentContainer. This ContentServerListener will be notified of additions, removals, or changes to any objects contained within this server
Parameters:
`listener` - the Listener that will receive ContentServerEvents.

removeContentServerListener

`void` **`removeContentServerListener`**`(ContentServerListener listener)`

Removes the specified ContentServerListener.
Parameters:
`listener` - the Listener to remove

setServiceResolutionHandler

`void` **`setServiceResolutionHandler`**`(ServiceResolutionHandler handler)`

Adds a ServiceResolutionHandler to this ContentServerNetModule. This ServiceResolutionHandler will be called when the implementation needs tuning information for a ChannelContentItem (e.g. a switched channel).

If an SPI service provider is already registered for the “ocap://” scheme this method throws an exception. If an SPI service provider (e.g. `SelectionProvider`) is subsequently registered, it SHALL have precedence over the registered `ServiceResolutionHandler`

If a handler is already set when this method is called, it is replaced by the new handler. If the handler parameter is null, the current handler is removed.

Parameters:

`handler` - The handler that will be called to get tuning parameters.

Throws:

`java.lang.UnsupportedOperationException` - if the isLocal method returns false.

`java.lang.UnsupportedOperationException` - if an SPI service provider is already registered for the "ocap://" scheme.

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentmanagement")

addStreamingActivityListener

`void` **`addStreamingActivityListener`**`(StreamingActivityListener listener, int contentTypes)`

Adds an `StreamingActivityListener` to this content server. The `StreamingActivityListener` will be notified of streaming being started, changed or ended.

Parameters:

`listener` - the `StreamingActivityListener` that will receive notification of streaming being started, changed or ended.

`contentTypes` - the contentItem types `StreamingActivityListener` is interested in. Defined in `StreamingActivityListener` 0 for all content with streamable resources 1 for ChannelContentItem and virtual tuners only 2 for RecordingContentItem only

Throws:

`java.lang.UnsupportedOperationException` - if the isLocal method returns false.

`java.lang.IllegalArgumentException` - if contentTypes is not one of the types defined in `StreamingActivityListener`.

removeStreamingActivityListener

`void` **`removeStreamingActivityListener`**`(StreamingActivityListener listener)`

Removes the specified `StreamingActivityListener` for all contentItem types specified in `addStreamingActivityListener`

Parameters:

`listener` - the `StreamingActivityListener` to remove.

org.ocap.hn

Interface Device

```
public interface Device
```

The `Device` interface represents a home network device that supports home network NetModules. A Device is a hierarchical structure with root device being the physical appliance, such as an OCAP_Terminal or an OCAP_HOST. The valid device types for an OCAP root device are OCAP_HOST and OCAP_Terminal. A root device may contain a number of sub-devices, such as a MediaServer or a MediaRenderer. Each sub-device may support one or more NetModule(s) whereas each NetModule only represents one sub-device. A NetModule is some functional unit in the device and examples of NetModules are ContentList, ContentManager, etc. A device may also have certain capabilities and properties associated with it. An application can retrieve these capabilities and properties by using property filters.

Field Summary

`static java.lang.String`	**`CAP_RECORDING_SUPPORTED`** A constant indicating MSO content recording capability.
`static java.lang.String`	**`CAP_REMOTE_STORAGE_SUPPORTED`** A constant indicating remote storage capability.
`static java.lang.String`	**`CAP_STREAMING_SUPPORTED`** A constant indicating streaming capability of the device.
`static java.lang.String`	**`CAP_TUNER_SUPPORTED`** A constant indicating if the device has a tuner.
`static java.lang.String`	**`PROP_DEVICE_TYPE`** A constant indicates device property: device type
`static java.lang.String`	**`PROP_DEVICE_VERSION`** A constant representing a device version number
`static java.lang.String`	**`PROP_FRIENDLY_NAME`** A constant for a friendly name of the device.
`static java.lang.String`	**`PROP_LOCATION`** A constant indicates device property: location of the device.
`static java.lang.String`	**`PROP_MANUFACTURER`** A constant indicating the manufacturer of this device.
`static java.lang.String`	**`PROP_MANUFACTURER_URL`** A constant providing URL to the manufacturer's web site.
`static java.lang.String`	**`PROP_MIDDLEWARE_PROFILE`** A constant indicates device property: middleware profile.
`static java.lang.String`	**`PROP_MIDDLEWARE_VERSION`** A constant indicates device property: middleware version.
`static java.lang.String`	**`PROP_MODEL_DESCRIPTION`** A constant providing description of the device.
`static java.lang.String`	**`PROP_MODEL_NAME`** A constant indicates device property: model name.

Field Summary

static java.lang.String	**PROP_MODEL_NUMBER** A constant indicates device property: model number.
static java.lang.String	**PROP_MODEL_URL** A constant indicates device property: model URL.
static java.lang.String	**PROP_PRESENTATION_URL** A constant indicates device property: presentation URL.
static java.lang.String	**PROP_SERIAL_NUMBER** A constant indicates device property: serial number.
static java.lang.String	**PROP_UDN** A constant indicates device property: unique device name.
static java.lang.String	**PROP_UPC** A constant indicates device property: universal product code.
static java.lang.String	**TYPE_BINARY_LIGHT** A constant indicates device type: Binary Light (on/off).
static java.lang.String	**TYPE_DIMMABLE_LIGHT** A constant indicates device type: Dimmable Light (light intensity control).
static java.lang.String	**TYPE_HVAC_SYSTEM** A constant indicates device type: Heater-Vent-Air Conditioning System.
static java.lang.String	**TYPE_HVAC_ZONE_THERMOSTAT** A constant indicates device type: Heater-Vent-Air Conditioning Thermostat.
static java.lang.String	**TYPE_INTERNET_GATEWAY_DEVICE** A constant indicates device type: Internet gateway device.
static java.lang.String	**TYPE_LAN_DEVICE** A constant indicates device type: LAN device.
static java.lang.String	**TYPE_MEDIA_RENDERER** A constant indicates device type: Media Renderer.
static java.lang.String	**TYPE_MEDIA_SERVER** A constant indicates device type: Media Server.
static java.lang.String	**TYPE_OCAP_HOST** A constant indicates device type: OCAP Host.
static java.lang.String	**TYPE_OCAP_TERMINAL** A constant indicates device type: OCAP terminal.
static java.lang.String	**TYPE_PRINTER** A constant indicates device type: Printer.
static java.lang.String	**TYPE_REMOTE_UI_CLIENT_DEVICE** A constant indicates device type: Remote UI Client Device, Allows for basic operations on a Remote UI client including: user interface connection management, optionally user interface availability management and optionally basic user interaction.
static java.lang.String	**TYPE_REMOTE_UI_SERVER_DEVICE** A constant indicates device type: Remote UI Server Device.

Field Summary

static java.lang.String	**TYPE_SCANNER** A constant indicates device type: Scanner.
static java.lang.String	**TYPE_WAN_CONNECTION_DEVICE** A constant indicates device type: WAN connection device.
static java.lang.String	**TYPE_WAN_DEVICE** A constant indicates device type: WAN device.
static java.lang.String	**TYPE_WLAN_ACCESS_POINT_DEVICE** A constant indicates device type: WAN access point device.

Method Summary

void	**addDeviceEventListener**(DeviceEventListener listener) Adds a DeviceEventListener instance to this Device.
java.util.Enumeration	**getCapabilities**() Returns capabilities of this device in Enumeration.
java.net.InetAddress	**getInetAddress**() Returns the IP address for this device.
java.util.Enumeration	**getKeys**() Returns all property keys supported by this device in Enumeration.
java.lang.String	**getName**() Returns the name of this device.
NetModule	**getNetModule**(java.lang.String moduleId) Returns the NetModule by module id.
NetList	**getNetModuleList**() Returns the list of NetModules supported by this device.
Device	**getParentDevice**() Returns the parent of this device.
java.lang.String	**getProperty**(java.lang.String key) Returns property of this device specified by a key.
NetList	**getSubDevices**() Returns a list of sub devices hosted by this device.
java.lang.String	**getType**() Returns the type of this device, for example, MediaRenderer, MediaServer, etc.
java.lang.String	**getVersion**() Returns the version number associated with this Device's device type.
boolean	**isLocal**() Returns true when this is the local device.
void	**removeDeviceEventListener**(DeviceEventListener listener) Removes a DeviceEventListener instance from this Device.
void	**setFriendlyName**(java.lang.String value) Sets the value of the PROP_FRIENDLY_NAME property.

Field Detail

CAP_STREAMING_SUPPORTED

`static final java.lang.String` **`CAP_STREAMING_SUPPORTED`**

A constant indicating streaming capability of the device.
See Also:
Constant Field Values

CAP_TUNER_SUPPORTED

`static final java.lang.String` **`CAP_TUNER_SUPPORTED`**

A constant indicating if the device has a tuner.
See Also:
Constant Field Values

CAP_REMOTE_STORAGE_SUPPORTED

`static final java.lang.String` **`CAP_REMOTE_STORAGE_SUPPORTED`**

A constant indicating remote storage capability.
See Also:
Constant Field Values

CAP_RECORDING_SUPPORTED

`static final java.lang.String` **`CAP_RECORDING_SUPPORTED`**

A constant indicating MSO content recording capability.
See Also:
Constant Field Values

PROP_FRIENDLY_NAME

`static final java.lang.String` **`PROP_FRIENDLY_NAME`**

A constant for a friendly name of the device.
See Also:
Constant Field Values

PROP_MANUFACTURER

`static final java.lang.String` **`PROP_MANUFACTURER`**

A constant indicating the manufacturer of this device.
See Also:
Constant Field Values

PROP_MANUFACTURER_URL

`static final java.lang.String` **`PROP_MANUFACTURER_URL`**

A constant providing URL to the manufacturer's web site.
See Also:
Constant Field Values

PROP_MODEL_DESCRIPTION

`static final java.lang.String` **`PROP_MODEL_DESCRIPTION`**

A constant providing description of the device.
See Also:
Constant Field Values

PROP_MODEL_NAME

`static final java.lang.String` **`PROP_MODEL_NAME`**
A constant indicates device property: model name.
See Also:
Constant Field Values

PROP_MODEL_NUMBER

`static final java.lang.String` **`PROP_MODEL_NUMBER`**
A constant indicates device property: model number.
See Also:
Constant Field Values

PROP_MODEL_URL

`static final java.lang.String` **`PROP_MODEL_URL`**
A constant indicates device property: model URL.
See Also:
Constant Field Values

PROP_SERIAL_NUMBER

`static final java.lang.String` **`PROP_SERIAL_NUMBER`**
A constant indicates device property: serial number.
See Also:
Constant Field Values

PROP_UDN

`static final java.lang.String` **`PROP_UDN`**
A constant indicates device property: unique device name.
See Also:
Constant Field Values

PROP_UPC

`static final java.lang.String` **`PROP_UPC`**
A constant indicates device property: universal product code.
See Also:
Constant Field Values

PROP_PRESENTATION_URL

`static final java.lang.String` **`PROP_PRESENTATION_URL`**
A constant indicates device property: presentation URL.
See Also:
Constant Field Values

PROP_LOCATION

`static final java.lang.String` **`PROP_LOCATION`**

A constant indicates device property: location of the device.

See Also:

Constant Field Values

PROP_MIDDLEWARE_PROFILE

`static final java.lang.String` **`PROP_MIDDLEWARE_PROFILE`**

A constant indicates device property: middleware profile.

See Also:

Constant Field Values

PROP_MIDDLEWARE_VERSION

`static final java.lang.String` **`PROP_MIDDLEWARE_VERSION`**

A constant indicates device property: middleware version.

See Also:

Constant Field Values

PROP_DEVICE_TYPE

`static final java.lang.String` **`PROP_DEVICE_TYPE`**

A constant indicates device property: device type

See Also:

Constant Field Values

PROP_DEVICE_VERSION

`static final java.lang.String` **`PROP_DEVICE_VERSION`**

A constant representing a device version number

See Also:

Constant Field Values

TYPE_HVAC_SYSTEM

`static final java.lang.String` **`TYPE_HVAC_SYSTEM`**

A constant indicates device type: Heater-Vent-Air Conditioning System.

See Also:

Constant Field Values

TYPE_HVAC_ZONE_THERMOSTAT

`static final java.lang.String` **`TYPE_HVAC_ZONE_THERMOSTAT`**

A constant indicates device type: Heater-Vent-Air Conditioning Thermostat.

See Also:

Constant Field Values

TYPE_INTERNET_GATEWAY_DEVICE

`static final java.lang.String` **`TYPE_INTERNET_GATEWAY_DEVICE`**

A constant indicates device type: Internet gateway device.

See Also:

Constant Field Values

TYPE_LAN_DEVICE

`static final java.lang.String` **`TYPE_LAN_DEVICE`**

A constant indicates device type: LAN device.

See Also:

Constant Field Values

TYPE_WAN_CONNECTION_DEVICE

`static final java.lang.String` **`TYPE_WAN_CONNECTION_DEVICE`**

A constant indicates device type: WAN connection device.

See Also:

Constant Field Values

TYPE_WAN_DEVICE

`static final java.lang.String` **`TYPE_WAN_DEVICE`**

A constant indicates device type: WAN device.

See Also:

Constant Field Values

TYPE_BINARY_LIGHT

`static final java.lang.String` **`TYPE_BINARY_LIGHT`**

A constant indicates device type: Binary Light (on/off).

See Also:

Constant Field Values

TYPE_DIMMABLE_LIGHT

`static final java.lang.String` **`TYPE_DIMMABLE_LIGHT`**

A constant indicates device type: Dimmable Light (light intensity control).

See Also:

Constant Field Values

TYPE_MEDIA_SERVER

`static final java.lang.String` **`TYPE_MEDIA_SERVER`**

A constant indicates device type: Media Server.

See Also:

Constant Field Values

TYPE_MEDIA_RENDERER

`static final java.lang.String` **`TYPE_MEDIA_RENDERER`**

A constant indicates device type: Media Renderer.

See Also:

Constant Field Values

TYPE_PRINTER

`static final java.lang.String` **`TYPE_PRINTER`**

A constant indicates device type: Printer.

See Also:

Constant Field Values

TYPE_REMOTE_UI_CLIENT_DEVICE

`static final java.lang.String` **`TYPE_REMOTE_UI_CLIENT_DEVICE`**

A constant indicates device type: Remote UI Client Device, Allows for basic operations on a Remote UI client including: user interface connection management, optionally user interface availability management and optionally basic user interaction.

See Also:
Constant Field Values

TYPE_REMOTE_UI_SERVER_DEVICE

`static final java.lang.String` **`TYPE_REMOTE_UI_SERVER_DEVICE`**

A constant indicates device type: Remote UI Server Device.

See Also:
`TYPE_REMOTE_UI_CLIENT_DEVICE`, Constant Field Values

TYPE_SCANNER

`static final java.lang.String` **`TYPE_SCANNER`**

A constant indicates device type: Scanner.

See Also:
Constant Field Values

TYPE_WLAN_ACCESS_POINT_DEVICE

`static final java.lang.String` **`TYPE_WLAN_ACCESS_POINT_DEVICE`**

A constant indicates device type: WAN access point device.

See Also:
Constant Field Values

TYPE_OCAP_HOST

`static final java.lang.String` **`TYPE_OCAP_HOST`**

A constant indicates device type: OCAP Host.

See Also:
Constant Field Values

TYPE_OCAP_TERMINAL

`static final java.lang.String` **`TYPE_OCAP_TERMINAL`**

A constant indicates device type: OCAP terminal.

See Also:
Constant Field Values

Method Detail

getCapabilities

`java.util.Enumeration` **`getCapabilities`**`()`

Returns capabilities of this device in `Enumeration`. Capabilities are defined in `Device`.

Returns:
An enumeration of String objects representing capabilities of this device.

getName

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

Returns the name of this device. The naming rule is proprietary. For example,"LivingRoom:OCAP_HOST1".

Returns:

name of this device

getProperty

`java.lang.String` **`getProperty`**`(java.lang.String key)`

Returns property of this device specified by a key. Minimum supported keys are defined in `Device`, like PROP_MANUFACTURER, PROP_MODEL_NUMBER, etc.

Parameters:

`key` - key of the property

Returns:

property value specified by the key

getKeys

`java.util.Enumeration` **`getKeys`**`()`

Returns all property keys supported by this device in `Enumeration`. Keys returned may include standardized keys (as documented with constants in this interface), as well as additional keys supported by this device.

Returns:

An enumeration of String objects representing all property keys supported by this device

getNetModuleList

`NetList` **`getNetModuleList`**`()`

Returns the list of NetModules supported by this device.

Returns:

`NetList` supported by this device

getNetModule

`NetModule` **`getNetModule`**`(java.lang.String moduleId)`

Returns the NetModule by module id. Module id is unique within a device.

Parameters:

`moduleId` - unique id of a NetModule

Returns:

NetModule by id, if specified NetModule is not supported by this device, then null is returned.

getSubDevices

`NetList` **`getSubDevices`**`()`

Returns a list of sub devices hosted by this device.

Returns:

list of sub-devices.

getParentDevice

`Device` **`getParentDevice`**`()`

Returns the parent of this device.

Returns:
the parent device, or null if this device has no parent.

getType

`java.lang.String` **`getType`**`()`

Returns the type of this device, for example, MediaRenderer, MediaServer, etc. All OCAP-HN device types are defined in `Device`.

Returns:
type of this device

getVersion

`java.lang.String` **`getVersion`**`()`

Returns the version number associated with this Device's device type.

Returns:
a String representing the version of this Device's device type

isLocal

`boolean` **`isLocal`**`()`

Returns true when this is the local device.

Returns:
true if this is the local device

addDeviceEventListener

`void` **`addDeviceEventListener`**`(DeviceEventListener listener)`

Adds a DeviceEventListener instance to this Device. If the listener passed in is already registered with this Device, this method does nothing.

Parameters:
`listener` - a DeviceEventListener instance to be notified of DeviceEvents.

removeDeviceEventListener

`void` **`removeDeviceEventListener`**`(DeviceEventListener listener)`

Removes a DeviceEventListener instance from this Device. If the specified instance is not registered with this Device, this method does nothing.

Parameters:
`listener` - a DeviceEventListener instance to be removed from this Device.

getInetAddress

`java.net.InetAddress` **`getInetAddress`**`()`

Returns the IP address for this device.

Returns:
an InetAddress representing this device's IP address

setFriendlyName

`void` **`setFriendlyName`**`(java.lang.String value)`

Sets the value of the PROP_FRIENDLY_NAME property. When network applications make use of the NetManager.getDevice method, operators are advised to provide an application that uses this method to set a device friendly name to a home network unique value.

Parameters:
`value` - The value to set the property to.
Throws:
`java.lang.IllegalArgumentException` - if the parameter violates the format specified by protocol mapping.
`java.lang.UnsupportedOperationException` - if the `Device` is not local; see the `isLocal` method.
`java.lang.SecurityException` - if the calling application has not been granted HomeNetPermission("contentmanagement").

org.ocap.hn

Class DeviceEvent

```
java.lang.Object
  └java.util.EventObject
      └org.ocap.hn.DeviceEvent
```

All Implemented Interfaces:

java.io.Serializable

```
public class DeviceEvent
extends java.util.EventObject
```

Represents a Device Event. There are two types of Device events: one that is generated by the NetManager when a Device is added or removed from the home network. Application may register as a listener to NetManager to receive such events. The other DeviceEvent is generated by the Device itself when its internal state changes. Application should register as a listener with a particular Device for such events. In both scenarios, the Device that was the source of the event is returned.

See Also:

Serialized Form

Field Summary

static int	**DEVICE_ADDED** A constant indicating new device is registered to home network.
static int	**DEVICE_REMOVED** A constant indicating a device is removed from home network.
static int	**DEVICE_UPDATED** A constant indicating a device is updated from home network.
static int	**STATE_CHANGE** A constant indicating a device's internal state has changed.

Fields inherited from class java.util.EventObject
source

Constructor Summary

DeviceEvent(int type, java.lang.Object source) Constructs a DeviceEvent by specifying type and source.

Method Summary

java.lang.Object	**getSource**() Returns device event source, which is always a Device.
int	**getType**() Returns device event type, as defined in DeviceEvent.

Methods inherited from class java.util.EventObject
`toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Field Detail

DEVICE_ADDED

`public static final int` **`DEVICE_ADDED`**

A constant indicating new device is registered to home network.

See Also:

Constant Field Values

DEVICE_REMOVED

`public static final int` **`DEVICE_REMOVED`**

A constant indicating a device is removed from home network.

See Also:

Constant Field Values

DEVICE_UPDATED

`public static final int` **`DEVICE_UPDATED`**

A constant indicating a device is updated from home network.

See Also:

Constant Field Values

STATE_CHANGE

`public static final int` **`STATE_CHANGE`**

A constant indicating a device's internal state has changed.

See Also:

Constant Field Values

Constructor Detail

DeviceEvent

```
public DeviceEvent(int type,
                   java.lang.Object source)
```

Constructs a DeviceEvent by specifying type and source.

Parameters:

`type` - Device change type, allowed type are defined in `DeviceEvent`

`source` - Device where the change happens.

Method Detail

getType

`public int` **`getType`**`()`

Returns device event type, as defined in `DeviceEvent`.

Returns:
device event type

getSource

`public java.lang.Object` **`getSource`**`()`

Returns device event source, which is always a Device.

Overrides:
`getSource` in class `java.util.EventObject`

Returns:
device event source

org.ocap.hn

Interface DeviceEventListener

All Superinterfaces:
java.util.EventListener

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

DeviceEvent callback interface. When a Device is registered or removed from NetManager, or if the internal status of a Device changes, then system will notify all registered listeners.

Method Summary

void	**notify**(DeviceEvent event) Callback function for Device events.

Method Detail

notify

```
void notify(DeviceEvent event)
```

Callback function for Device events.

Parameters:

`event` - Device event

org.ocap.hn

Class HomeNetPermission

```
java.lang.Object
  └java.security.Permission
      └java.security.BasicPermission
          └org.ocap.hn.HomeNetPermission
```

All Implemented Interfaces:

java.io.Serializable, java.security.Guard

```
public final class HomeNetPermission
extends java.security.BasicPermission
```

The HomeNetPermission class represents permission to execute privileged home networking operations only signed applications MAY be granted.

A HomeNetPermission consists of a permission name, representing a single privileged operation. The name given in the constructor may end in "*" to represent all permissions beginning with the given string, such as `"*"` to allow all HomeNetPermission operations.

The following table lists all HomeNetPermission permission names.

Permission Name	What the Permission Allows	Description
contentmanagement	Provides management of local or remote content	Applications with this permission can copy, move, delete content as well as allocate and delete logical volumes on a local network device.
contentlisting	Provides listing of content on remote devices	Applications with this permission can discover and query lists of content stored on or streamable from remote devices.
recording	Provides recording operations on remote devices	Applications with this permission can request that recordings be scheduled, prioritized, and deleted on remote devices.
recordinghandler	Provides recording request handler functionality on the local device	Applications with this permission can manage network recording requests for the local device.

Other permissions may be added as necessary.

See Also:

Serialized Form

Constructor Summary

HomeNetPermission(java.lang.String name)

Constructor for the HomeNetPermission

Method Summary

Methods inherited from class java.security.BasicPermission
equals, getActions, hashCode, implies, newPermissionCollection

Methods inherited from class java.security.Permission
checkGuard, getName, toString

Methods inherited from class java.lang.Object
clone, finalize, getClass, notify, notifyAll, wait, wait, wait

Constructor Detail

HomeNetPermission

`public HomeNetPermission(java.lang.String name)`

Constructor for the HomeNetPermission

Parameters:

`name` - The name of this permission (see table in class description).

org.ocap.hn

Class NetActionEvent

```
java.lang.Object
  └java.util.EventObject
      └org.ocap.hn.NetActionEvent
```

All Implemented Interfaces:

java.io.Serializable

```
public class NetActionEvent
extends java.util.EventObject
```

This class represents an event generated in response to an action. NetActionEvent instances can only be created by the implementation.

See Also:

Serialized Form

Field Summary

static int	**ACTION_CANCELED** ACTION_CANCELED is returned by getActionStatus() when the action has been canceled.
static int	**ACTION_COMPLETED** Action status for a completed action
static int	**ACTION_FAILED** ACTION_FAILED is returned by getActionStatus() when the action has failed.
static int	**ACTION_IN_PROGRESS** ACTION_IN_PROGRESS is returned by getActionStatus() when the action is currently on going.
static int	**ACTION_STATUS_NOT_AVAILABLE** ACTION_STATUS_NOT_AVAILABLE is returned by getActionStatus() when the transaction has completed successfully or failed sometime before this method was called and the implementation is no longer maintaining a status for it.

Fields inherited from class java.util.EventObject
source

Constructor Summary

protected	**NetActionEvent**(java.lang.Object request, java.lang.Object response, int error, int status) Two argument constructor.

Method Summary

NetActionRequest	**getActionRequest**() Returns the ActionRequest which identifies the action instance.
int	**getActionStatus**() Returns the status of the requested action.
int	**getError**() Gets the error value when getActionStatus returns NetActionEvent.ACTION_FAILED.
java.lang.Object	**getResponse**() Returns the response of the Action.

Methods inherited from class java.util.EventObject
getSource, toString

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Field Detail

ACTION_COMPLETED

public static final int **ACTION_COMPLETED**

Action status for a completed action

See Also:

getActionStatus(), Constant Field Values

ACTION_CANCELED

public static final int **ACTION_CANCELED**

ACTION_CANCELED is returned by getActionStatus() when the action has been canceled.

See Also:

getActionStatus(), Constant Field Values

ACTION_FAILED

public static final int **ACTION_FAILED**

ACTION_FAILED is returned by getActionStatus() when the action has failed.

See Also:

getActionStatus(), Constant Field Values

ACTION_STATUS_NOT_AVAILABLE

public static final int **ACTION_STATUS_NOT_AVAILABLE**

ACTION_STATUS_NOT_AVAILABLE is returned by getActionStatus() when the transaction has completed successfully or failed sometime before this method was called and the implementation is no longer maintaining a status for it.

See Also:

Constant Field Values

ACTION_IN_PROGRESS

```
public static final int ACTION_IN_PROGRESS
```

`ACTION_IN_PROGRESS` is returned by `getActionStatus()` when the action is currently on going.

See Also:

`getActionStatus()`, Constant Field Values

Constructor Detail

NetActionEvent

```
protected NetActionEvent(java.lang.Object request,
                         java.lang.Object response,
                         int error,
                         int status)
```

Two argument constructor.

Parameters:

`request` - - NetActionRequest that instigated the response.

`response` - - An object representing the response to the action and which is specific to the action.

`error` - - error code for this event if action failed

`status` - - status of the associated net action

Method Detail

getResponse

```
public java.lang.Object getResponse()
```

Returns the response of the Action. Object is dependent on the Action.

Returns:

The response to an asynchronous action.

getActionRequest

```
public NetActionRequest getActionRequest()
```

Returns the ActionRequest which identifies the action instance.

Returns:

the ActionRequest

getActionStatus

```
public int getActionStatus()
```

Returns the status of the requested action.

Returns:

the status of the action; for possible return values see ACTION_* constants in this class.

getError

```
public int getError()
```

Gets the error value when getActionStatus returns `NetActionEvent.ACTION_FAILED`. If the action is not in error this method SHALL return -1. Error code values are dependent on the underlying network protocol error code values.

Returns:

The error value; -1 if no error.

org.ocap.hn

Interface NetActionHandler

```
public interface NetActionHandler
```

This interface represents a handler passed to asynchronous methods.

Method Summary

void	**notify**(NetActionEvent event) Notifies the application of an action event.

Method Detail

notify

```
void notify(NetActionEvent event)
```

Notifies the application of an action event. This method is called by the implementation when a response to an action or a failure for the action is detected.

org.ocap.hn

Interface NetActionRequest

```
public interface NetActionRequest
```

All asynchronous actions in the Home networking API return an NetActionRequest. The NetActionRequest can be used a) to cancel any pending action or b) to identify which Action got completed.

See Also:
`NetActionHandler`, `NetActionEvent`

Method Summary

`boolean`	**`cancel`**`()` Cancels the Action associated with this ActionRequest.
`int`	**`getActionStatus`**`()` Gets the current status of the requested action.
`int`	**`getError`**`()` Gets the error value when getActionStatus returns `NetActionEvent.ACTION_FAILED`.
`float`	**`getProgress`**`()` Gets the progress of the action in percent (0.0 - 1.0).

Method Detail

cancel

`boolean` **`cancel`**`()`

Cancels the Action associated with this ActionRequest. Returns false if the action can't be canceled.

Returns:
false if action can't be canceled, otherwise returns true.

getProgress

`float` **`getProgress`**`()`

Gets the progress of the action in percent (0.0 - 1.0). If the progress of an action can't be determined, -1.0 shall be returned.

Returns:
the progress of the action (0.0 - 1.0) or -1.0 if the progress can't be determined.

getActionStatus

`int` **`getActionStatus`**`()`

Gets the current status of the requested action.

Returns:
the current action status; see ACTION_* constants in `NetActionEvent` for possible return values.

getError

`int` **`getError`**`()`

Gets the error value when getActionStatus returns `NetActionEvent.ACTION_FAILED`. The error code returned will be equivalent to the error code returned by `NetActionEvent.getError()` for the NetActionEvent associated with the completion of this action request. If the action is not in error or has not completed, this method SHALL return -1.

Returns:

The error value; -1 if no error,

org.ocap.hn

Interface NetList

```
public interface NetList
```

A list comprising of home network elements such as Device or NetModule. The application may retrieve such a list from `NetManager`, `getNetModules` * or `getDevices`. The application may refine the list by applying a `PropertyFilter`.

Method Summary

boolean	**contains**(java.lang.Object element) Indicates whether an element is included in this NetList.
NetList	**filterElement**(PropertyFilter filter) Applies a new `PropertyFilter` to this element list and returns a new list.
java.lang.Object	**getElement**(int index) Returns the element indexed by a number.
java.util.Enumeration	**getElements**() Returns all elements in this NetList in `Enumeration`.
int	**indexOf**(java.lang.Object element) Returns the index of an element in this element list.
int	**size**() Returns the size of this list.

Method Detail

contains

```
boolean contains(java.lang.Object element)
```

Indicates whether an element is included in this NetList.

Parameters:
`element` - the element to check whether it is in the list

Returns:
true if the element is in the list; otherwise false.

getElement

```
java.lang.Object getElement(int index)
```

Returns the element indexed by a number.

Parameters:
`index` - specified index of the element

Returns:
element indexed by the number

getElements

```
java.util.Enumeration getElements()
```

Returns all elements in this NetList in `Enumeration`. In Homenetwork, NetList can be used to retrieve a list of Devices or a list of NetModules. In either case, a corresponding type of object is returned.
Returns:
An enumeration of Device or NetModule elements

filterElement

`NetList` **`filterElement`**`(PropertyFilter filter)`

Applies a new `PropertyFilter` to this element list and returns a new list.
Parameters:
`filter` - new filter
Returns:
new element list generated by new filter
See Also:
`PropertyFilter`

indexOf

`int` **`indexOf`**`(java.lang.Object element)`

Returns the index of an element in this element list.
Parameters:
`element` - to be checked
Returns:
index of an element in this list. If there is no such element in this list, returns -1.

size

`int` **`size`**`()`

Returns the size of this list.
Returns:
size of the element list

org.ocap.hn

Class NetManager

```
java.lang.Object
  └org.ocap.hn.NetManager

public abstract class NetManager
extends java.lang.Object
```

The NetManager is a singleton class that registers all the Devices and NetModules within a home network. It maintains an implementation dependent database of devices and NetModules.

The NetManager may be used to retrieve list of `NetModule` and `Device` in the network. The application can filter the list by specifying a name or by applying filtering rules. For example, "modelNumber = h6315, location = LivingRoom". Application can monitor availability of NetModules by registering as a listener to NetManager instance.

Constructor Summary

NetManager()

Method Summary

abstract void	**addDeviceEventListener**(DeviceEventListener listener) Adds a Device event listener to NetManager.
abstract void	**addNetModuleEventListener**(NetModuleEventListener listener) Adds a NetModule event listener to NetManager.
abstract Device	**getDevice**(java.lang.String name) Returns device by name, for example, "BallRoom:DVD_PLAYER1".
abstract NetList	**getDeviceList**(PropertyFilter filter) Returns devices that match all properties set by a given filter.
abstract NetList	**getDeviceList**(java.lang.String name) Returns all devices that match the specified device name.
static NetManager	**getInstance**() Returns the singleton NetManager.
abstract NetModule	**getNetModule**(java.lang.String deviceName, java.lang.String moduleID) Returns NetModule by device and module ID.
abstract NetList	**getNetModuleList**(PropertyFilter filter) Returns NetModules that match all properties set by a given filter.
abstract NetList	**getNetModuleList**(java.lang.String deviceName, java.lang.String moduleID) Returns all NetModules that match the specified device name and module identifier.

Method Summary

abstract void	**removeDeviceEventListener**(DeviceEventListener listener) Removes a Device event listener from NetManager.
abstract void	**removeNetModuleEventListener**(NetModuleEventListener listener) Removes a NetModule event listener from NetManager.
abstract void	**updateDeviceList**() Requests that the NetManager proactively refresh its local database of connected devices.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

b

Constructor Detail

NetManager

public **NetManager**()

Method Detail

getInstance

public static NetManager **getInstance**()

Returns the singleton NetManager. This is the entry point for home network. If the calling application is unsigned, this method SHALL return null.

Returns:

Singleton instance of NetManager or null if the calling application is unsigned.

getNetModuleList

public abstract NetList **getNetModuleList**(PropertyFilter filter)

Returns NetModules that match all properties set by a given filter. Passing a null filter will return a NetList with all known NetModules.

Parameters:

`filter` - Filter to select out NetModules from all available NetModules

Returns:

List of NetModules satisfying filter

getNetModuleList

public abstract NetList **getNetModuleList**(java.lang.String deviceName, java.lang.String moduleID)

Returns all NetModules that match the specified device name and module identifier. Passing a null or empty device name with a non-null module identifier will result in a `NetList` containing all `NetModule`s whose module ids match the non-null module identifier. Passing a null or empty module identifier with a non-null device name will result in a `NetList` containing all NetModules whose devices match the non-null device name. Passing a null or empty module identifier and null or empty device name will return a `NetList` containing all known `NetModule`s.

Parameters:
`deviceName` - name of the device hosting the module to retrieve
`moduleID` - module identifier
Returns:
List of NetModules satisfying device name and module identifier

getNetModule

```
public abstract NetModule getNetModule(java.lang.String deviceName,
                                       java.lang.String moduleID)
```

Returns NetModule by device and module ID. If multiple devices have the same device name and share the same module identifier, then the value returned by this method is implementation specific.
Parameters:
`deviceName` - name of the device hosting the module to retrieve
`moduleID` - module identifier
Returns:
NetModule with the specified identifier

getDeviceList

```
public abstract NetList getDeviceList(PropertyFilter filter)
```

Returns devices that match all properties set by a given filter. All known devices and sub-devices are passed through the given filter. Passing a null filter will return a `NetList` with all known devices and sub-devices.
Parameters:
`filter` - Filter to select out devices from all connected devices
Returns:
List of devices satisfying filter

getDeviceList

```
public abstract NetList getDeviceList(java.lang.String name)
```

Returns all devices that match the specified device name. Passing a null or empty device name will result in an empty `NetList`.
Parameters:
`name` - Device name.
Returns:
List of devices satisfying device name

getDevice

```
public abstract Device getDevice(java.lang.String name)
```

Returns device by name, for example, "BallRoom:DVD_PLAYER1". If multiple devices have the same name, then the value returned by this method is implementation specific.
Parameters:
`name` - Device name
Returns:
Device matching the specified name

addNetModuleEventListener

```
public abstract void
addNetModuleEventListener(NetModuleEventListener listener)
```

Adds a NetModule event listener to NetManager. Listener will receive a NetModuleEvent when a new NetModule is registered or an old NetModule is removed from home network. If listener is already registered, no action is performed.

Parameters:

`listener` - Listener which listens to NetModule change events on home network

See Also:

`removeNetModuleEventListener`

removeNetModuleEventListener

```
public abstract void
removeNetModuleEventListener(NetModuleEventListener listener)
```

Removes a NetModule event listener from NetManager. If the listener is not registered yet, no action is performed.

Parameters:

`listener` - Listener which listens to NetModule change events on home network

See Also:

`addNetModuleEventListener`

addDeviceEventListener

```
public abstract void addDeviceEventListener(DeviceEventListener listener)
```

Adds a Device event listener to NetManager. Listener will receive a DeviceEvent when a new Device is registered, an existing Device is removed from home network, or a Device's internal state has changed. If the listener passed in is already registered, no action is performed. When a device listener is registered, the implementation SHALL NOT generate DEVICE_ADDED events for devices previously discovered by the implementation.

Parameters:

`listener` - Listener which listens to Device change events on the home network

See Also:

`removeDeviceEventListener`

removeDeviceEventListener

```
public abstract void removeDeviceEventListener(DeviceEventListener listener)
```

Removes a Device event listener from NetManager. If the listener is not registered yet, no action is performed.

Parameters:

`listener` - Listener which listens to Device change events on home network

See Also:

`addDeviceEventListener`

updateDeviceList

```
public abstract void updateDeviceList()
```

Requests that the NetManager proactively refresh its local database of connected devices. This operation will be performed asynchronously. Any listeners registered with the NetManager changes to connected Devices or NetModules will be notified of any changes discovered during this process.

org.ocap.hn

Interface NetModule

All Known Subinterfaces:

ContentServerNetModule, NetRecordingRequestManager, RecordingNetModule

```
public interface NetModule
```

NetModule is an abstraction of functionality that is provided by a `Device`. It is a group of related actions. A NetModule is always associated with a home network `Device`. Application may monitor a NetModule's status by subscribing as a listener to this NetModule.

Field Summary

`static java.lang.String`	**`CONTENT_LIST`** A constant indicating content listing NetModule.
`static java.lang.String`	**`CONTENT_MANAGER`** A constant indicating content manager NetModule.
`static java.lang.String`	**`CONTENT_RECORDER`** A constant indicating content recording NetModule.
`static java.lang.String`	**`CONTENT_RENDERER`** A constant indicating content renderer NetModule.
`static java.lang.String`	**`CONTENT_SERVER`** A constant indicating content server NetModule.
`static java.lang.String`	**`PROP_CONTROL_URL`** A constant providing URL for NetModule control.
`static java.lang.String`	**`PROP_DESCRIPTION_URL`** A constant providing URL for NetModule description.
`static java.lang.String`	**`PROP_EventSub_URL`** A constant providing URL for NetModule eventing.
`static java.lang.String`	**`PROP_NETMODULE_ID`** A constant indicating NetModuleID.
`static java.lang.String`	**`PROP_NETMODULE_TYPE`** A constant providing this NetModule's type.

Method Summary

`void`	**`addNetModuleEventListener`**`(NetModuleEventListener listener)` Adds a NetModuleEventListener instance to this NetModule.
`Device`	**`getDevice`**`()` Returns the device that provides this NetModule.
`java.util.Enumeration`	**`getKeys`**`()` Returns the property keys supported by this NetModule.
`java.lang.String`	**`getNetModuleId`**`()` Returns the id of this NetModule, which is unique within the device.

Method Summary

java.lang.String	**getNetModuleType**() Returns the type of this NetModule.
java.lang.String	**getProperty**(java.lang.String key) Returns the property value for specified key.
boolean	**isLocal**() Returns true if this NetModule is hosted on the local device.
void	**removeNetModuleEventListener**(NetModuleEventListener listener) Removes a NetModuleEventListener instance from this NetModule.

Field Detail

CONTENT_LIST

static final java.lang.String **CONTENT_LIST**

A constant indicating content listing NetModule.

See Also:

Constant Field Values

CONTENT_MANAGER

static final java.lang.String **CONTENT_MANAGER**

A constant indicating content manager NetModule.

See Also:

Constant Field Values

CONTENT_RENDERER

static final java.lang.String **CONTENT_RENDERER**

A constant indicating content renderer NetModule.

See Also:

Constant Field Values

CONTENT_SERVER

static final java.lang.String **CONTENT_SERVER**

A constant indicating content server NetModule.

See Also:

Constant Field Values

CONTENT_RECORDER

static final java.lang.String **CONTENT_RECORDER**

A constant indicating content recording NetModule.

See Also:

Constant Field Values

PROP_NETMODULE_ID

static final java.lang.String **PROP_NETMODULE_ID**

A constant indicating NetModuleID.
See Also:
Constant Field Values

PROP_DESCRIPTION_URL

`static final java.lang.String` **`PROP_DESCRIPTION_URL`**
A constant providing URL for NetModule description.
See Also:
Constant Field Values

PROP_CONTROL_URL

`static final java.lang.String` **`PROP_CONTROL_URL`**
A constant providing URL for NetModule control.
See Also:
Constant Field Values

PROP_EventSub_URL

`static final java.lang.String` **`PROP_EventSub_URL`**
A constant providing URL for NetModule eventing.
See Also:
Constant Field Values

PROP_NETMODULE_TYPE

`static final java.lang.String` **`PROP_NETMODULE_TYPE`**
A constant providing this NetModule's type.
See Also:
Constant Field Values

Method Detail

getDevice

`Device` **`getDevice`**`()`
Returns the device that provides this NetModule.
Returns:
device that offers this NetModule

getKeys

`java.util.Enumeration` **`getKeys`**`()`
Returns the property keys supported by this NetModule.
Returns:
An enumeration of String object representing property keys for this NetModule

getProperty

`java.lang.String` **`getProperty`**`(java.lang.String key)`
Returns the property value for specified key.
Parameters:
`key` - specified property key

Returns:
property value for specified key

getNetModuleType

`java.lang.String` **`getNetModuleType`**`()`

Returns the type of this NetModule. The allowed types are defined as constant field in `NetModule`, for example, CONTENT_MANAGER, CONTENT_LIST.

Returns:
type of this NetModule

getNetModuleId

`java.lang.String` **`getNetModuleId`**`()`

Returns the id of this NetModule, which is unique within the device. An example could be, ContentListing1.

Returns:
id of this NetModule

addNetModuleEventListener

`void` **`addNetModuleEventListener`**`(NetModuleEventListener listener)`

Adds a NetModuleEventListener instance to this NetModule. If the listener passed in is already registered with this NetModule, this method does nothing.

Parameters:
`listener` - a NetModuleEventListener instance to be notified of NetModuleEvents.

removeNetModuleEventListener

`void` **`removeNetModuleEventListener`**`(NetModuleEventListener listener)`

Removes a NetModuleEventListener instance from this NetModule. If the specified instance is not registered with this NetModule, this method does nothing.

Parameters:
`listener` - a NetModuleEventListener instance to be removed from this NetModule.

isLocal

`boolean` **`isLocal`**`()`

Returns true if this NetModule is hosted on the local device.

Returns:
true if this NetModule is hosted on the local device, false otherwise.

org.ocap.hn

Class NetModuleEvent

```
java.lang.Object
  └java.util.EventObject
      └org.ocap.hn.NetModuleEvent
```

All Implemented Interfaces:

java.io.Serializable

```
public class NetModuleEvent
extends java.util.EventObject
```

Entity for NetModule Event. There are two types of NetModule events: one that is generated by the NetManager when a NetModule is added or removed from the home network. Application may register as a listener to NetManager to receive such events. The other NetModuleEvent is generated by the NetModule itself when its internal state changes. Application should register as a listener with a particular NetModule for such events. In both scenarios, the NetModule that was the source of the event is returned.

See Also:

Serialized Form

Field Summary

static int	**MODULE_ADDED** A constant indicating new module is registered to home network.
static int	**MODULE_BUSY** A constant indicating a module is busy and cannot respond to request now.
static int	**MODULE_REMOVED** A constant indicating a module is removed from home network.
static int	**MODULE_UPDATED** A constant indicating a module is updated from home network.
static int	**STATE_CHANGE** A constant indicating a module's internal status changed.

Fields inherited from class java.util.EventObject
source

Constructor Summary

NetModuleEvent(int type, java.lang.Object source) Constructs a NetModuleEvent by specifying type and source.

Method Summary

java.lang.Object	**getSource**() Returns module event source, which is always a NetModule.

Method Summary

int	**getType**() Returns module event type, as defined in NetModuleEvent.

Methods inherited from class java.util.EventObject
toString

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Field Detail

MODULE_ADDED

public static final int **MODULE_ADDED**

A constant indicating new module is registered to home network.

See Also:

Constant Field Values

MODULE_REMOVED

public static final int **MODULE_REMOVED**

A constant indicating a module is removed from home network.

See Also:

Constant Field Values

MODULE_UPDATED

public static final int **MODULE_UPDATED**

A constant indicating a module is updated from home network.

See Also:

Constant Field Values

MODULE_BUSY

public static final int **MODULE_BUSY**

A constant indicating a module is busy and cannot respond to request now.

See Also:

Constant Field Values

STATE_CHANGE

public static final int **STATE_CHANGE**

A constant indicating a module's internal status changed.

See Also:

Constant Field Values

Constructor Detail

NetModuleEvent

```
public NetModuleEvent(int type,
                      java.lang.Object source)
```

Constructs a NetModuleEvent by specifying type and source.

Parameters:

`type` - NetModule change type, allowed type are defined in `NetModuleEvent`

`source` - NetModule where the change happens.

Method Detail

getType

```
public int getType()
```

Returns module event type, as defined in `NetModuleEvent`.

Returns:

module event type

getSource

```
public java.lang.Object getSource()
```

Returns module event source, which is always a NetModule.

Overrides:

`getSource` in class `java.util.EventObject`

Returns:

module event source

org.ocap.hn

Interface NetModuleEventListener

All Superinterfaces:

java.util.EventListener

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

NetModuleEvent callback interface. When a NetModule is registered or removed from NetManager, or if the internal status of a NetModule changes, then system will notify all registered listeners.

Method Summary

void	**notify**(NetModuleEvent event) Callback function for NetModule event.

Method Detail

notify

```
void notify(NetModuleEvent event)
```

Callback function for NetModule event. Callee will be notified when NetModule event happens

Parameters:

`event` - NetModule event

org.ocap.hn

Class NetworkInterface

```
java.lang.Object
  └org.ocap.hn.NetworkInterface
```

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

This class represents a home network interface including MoCA, wired ethernet, and wireless ethernet. Reverse channel interfaces are not represented by objects of this class. For each wired ethernet, wireless ethernet, MoCA interface, or interface that is not a reverse channel interface the HNIMP SHALL create an instance of this class.

Field Summary

static int	**MOCA** Network interface type for hard-wired and MoCA based.
static int	**UNKNOWN** Unknown network type.
static int	**WIRED_ETHERNET** Network interface type for hard-wired and ethernet based.
static int	**WIRELESS_ETHERNET** Network interface type for wireless and ethernet based.

Constructor Summary

protected	**NetworkInterface**() Protected constructor.

Method Summary

java.lang.String	**getDisplayName**() Gets a humanly readable name for this interface, e.g.
java.net.InetAddress	**getInetAddress**() Gets the InetAddress of this interface.
java.net.InetAddress[]	**getInetAddresses**() Gets an array of InetAddress containing all of the IP addresses configured for this NetworkInterface.
java.lang.String	**getMacAddress**() Gets the MAC address of this interface.
static NetworkInterface[]	**getNetworkInterfaces**() Gets an array of NetworkInterface instances that represent all of the network interfaces supported by the device.
int	**getType**() Gets the type of this network interface.

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Field Detail

UNKNOWN

`public static final int` **`UNKNOWN`**

Unknown network type.

See Also:

Constant Field Values

MOCA

`public static final int` **`MOCA`**

Network interface type for hard-wired and MoCA based.

See Also:

Constant Field Values

WIRED_ETHERNET

`public static final int` **`WIRED_ETHERNET`**

Network interface type for hard-wired and ethernet based.

See Also:

Constant Field Values

WIRELESS_ETHERNET

`public static final int` **`WIRELESS_ETHERNET`**

Network interface type for wireless and ethernet based.

See Also:

Constant Field Values

Constructor Detail

NetworkInterface

`protected` **`NetworkInterface`**`()`

Protected constructor.

Method Detail

getNetworkInterfaces

`public static NetworkInterface[]` **`getNetworkInterfaces`**`()`

Gets an array of `NetworkInterface` instances that represent all of the network interfaces supported by the device.

Returns:

An array of NetworkInterface instances.

getType

`public int **getType**()`

Gets the type of this network interface. Possibilities include UNKNOWN, MOCA, WIRED_ETHERNET, WIRELESS_ETHERNET.

Returns:

The type of this interface.

getDisplayName

`public java.lang.String **getDisplayName**()`

Gets a humanly readable name for this interface, e.g. "ie0".

Returns:

The display name of this interface.

getInetAddress

`public java.net.InetAddress **getInetAddress**()`

Gets the `InetAddress` of this interface. Returns one of the `InetAddress` instances in the array returned by the `getInetAddresses` method. If the array contains multiple `InetAddress` instances, unless specified elsewhere, the determination of which `InetAddress` to return is implementation specific.

Returns:

The `InetAddress` of this interface.

See Also:

`NetAuthorizationHandler2.notifyActivityStart, NetAuthorizationHandler2.notifyAction, HttpRequestResolutionHandler.resolveHttpRequest`

getInetAddresses

`public java.net.InetAddress[] **getInetAddresses**()`

Gets an array of `InetAddress` containing all of the IP addresses configured for this `NetworkInterface`.

Returns:

The array of `InetAddress` for this interface.

getMacAddress

`public java.lang.String **getMacAddress**()`

Gets the MAC address of this interface.

Returns:

The MAC address of this interface.

org.ocap.hn

Class NotAuthorizedException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └org.ocap.hn.NotAuthorizedException
```

All Implemented Interfaces:

java.io.Serializable

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

Exception indicating that the application has no permission to perform certain action.

See Also:

Serialized Form

Constructor Summary

`NotAuthorizedException``()` Constructs a NotAuthorizedException object.
`NotAuthorizedException``(java.lang.String reason)` Constructs a NotAuthorizedException object with a reason.

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

NotAuthorizedException

`public` **`NotAuthorizedException`**`()`

Constructs a NotAuthorizedException object.

NotAuthorizedException

`public` **`NotAuthorizedException`**`(java.lang.String reason)`

Constructs a NotAuthorizedException object with a reason.

Parameters:
`reason` - reason for this exception

org.ocap.hn

Class PropertyFilter

```
java.lang.Object
  └org.ocap.hn.PropertyFilter
```

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

The filter for (key,value) pair filtering mechanism. If a device or a NetModule has same value on all of the specified keys, it is regarded as a match.

Constructor Summary

PropertyFilter(java.util.Properties prop) Constructs a PropertyFilter object.

Method Summary

boolean	**accept**(java.lang.Object element) Checks whether an element is accepted by this filter, the element must be either NetModule or Device.
void	**addProperty**(java.lang.String key, java.lang.String value) Adds a (key,value) pair to the filter.
boolean	**contains**(java.lang.String key) Checks whether a key is in the list.
void	**removeKey**(java.lang.String key) Remove a key from the filter, if the key is not in the property list, no action is taken.
void	**removeKeys**(java.lang.String[] keys) Remove keys from the filter, if a key is not in the property list, it is disregarded; while others are processed as normal.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

PropertyFilter

```
public PropertyFilter(java.util.Properties prop)
```

Constructs a PropertyFilter object.

Parameters:

prop - Initial properties for this Property filter

Method Detail

addProperty

```
public void addProperty(java.lang.String key,
                        java.lang.String value)
```

Adds a (key,value) pair to the filter. If the key is already in the list, no action is taken.

Parameters:

`key` - New key which will be used for filtering.

`value` - Value for the new key.

contains

```
public boolean contains(java.lang.String key)
```

Checks whether a key is in the list.

Parameters:

`key` - Key to be checked against.

Returns:

True if key is in the list; otherwise returns false.

accept

```
public boolean accept(java.lang.Object element)
```

Checks whether an element is accepted by this filter, the element must be either `NetModule` or `Device`. If a NetModule/Device's properties share the same value as all properties from this filter, it is accepted and true is returned; otherwise, false is returned.

Parameters:

`element` - Element to be checked against.

Returns:

True if the element is accepted by the PropertyFilter, otherwise returns false.

removeKey

```
public void removeKey(java.lang.String key)
```

Remove a key from the filter, if the key is not in the property list, no action is taken.

Parameters:

`key` - Key to be removed from list.

removeKeys

```
public void removeKeys(java.lang.String[] keys)
```

Remove keys from the filter, if a key is not in the property list, it is disregarded; while others are processed as normal.

Parameters:

`keys` - Keys to be removed from the list.

Annex B Content API

Package org.ocap.hn.content

Provides representation of the content and content containers, and means of identifying content type.

Interface Summary

AudioResource	Interface implemented by subclasses of ContentResource to identify that a content contains audio.
ChannelContentItem	This interface represents a video or audio broadcast channel object.
ContentContainer	This class represents a container that contains one or more content entries.
ContentEntry	This interface represents a basic content entry.
ContentFormat	This interface represents a content format.
ContentItem	This class represents a piece of content.
ContentProfile	Interface defining constants that represent content profile identifiers to be used in conjunction with the `ContentFormat` interface.
ContentResource	Abstract class representing a media stream/file.
IOStatus	This interface represents the ability to detect whether any asset represented by an object or its children is in use on the home network and hence the object should not be deleted.
OutputVideoContentFormat	This interface provides additional parameters for transforming video content.
ProtectionType	Interface defining constants that represent supported output protection types to be used in conjunction with the `ContentFormat` interface.
StreamableContentResource	Abstract class representing content that can be streamed, e.g., MPEG file.
StreamingActivityListener	This interface represents a listener for notification of streaming being started, changed or ended
VideoResource	ContentResource to identify that a content item contains video/still image material.

Class Summary

ContentEntryFactory	This factory can be used to create `ContentEntry` instances.
MetadataIdentifiers	This abstract class represents access to standardized metadata identifiers.
MetadataNode	A collection of metadata entries, each of which is a key/value pair where the key identifies a property and the value is the property value, and a collection of supporting namespace declarations.

Exception Summary

DatabaseException	Exception that is thrown when a database error occurs

org.ocap.hn.content

Interface AudioResource

All Superinterfaces:
ContentResource

```
public interface AudioResource
extends ContentResource
```

Interface implemented by subclasses of ContentResource to identify that a content contains audio.

Field Summary

Fields inherited from interface org.ocap.hn.content.ContentResource
UNKNOWN_MIME_TYPE

Method Summary

int	**getBitsPerSample**() Returns the number of bits per sample or -1 if not known.
java.lang.String[]	**getLanguages**() Returns an array of languages associated with this audio content or a zero length array if not known.
int	**getNumberOfChannels**() Returns the number of audio channels, for example, 1 for mono, 2 for stereo, 6 for DTS 5.1 and 7 for DTS 6.1
int	**getSampleFrequency**() Returns the sample frequency in Hz of this audio content or -1 if not known.

Methods inherited from interface org.ocap.hn.content.ContentResource
delete, getContentFormat, getContentItem, getContentSize, getCreationDate, getExtendedFileAccessPermissions, getLocator, getNetwork, getProtocol, getResourceProperty, isRenderable

Method Detail

getSampleFrequency

int **getSampleFrequency**()

Returns the sample frequency in Hz of this audio content or -1 if not known.

Returns:
the sample frequency of the content of -1 if not known.

getNumberOfChannels

int **getNumberOfChannels**()

Returns the number of audio channels, for example, 1 for mono, 2 for stereo, 6 for DTS 5.1 and 7 for DTS 6.1

Returns:

the sample frequency of the content of -1 if not known.

getBitsPerSample

`int` **`getBitsPerSample`**`()`

Returns the number of bits per sample or -1 if not known.

Returns:

the number of bits per sample or -1 if not known.

getLanguages

`java.lang.String[]` **`getLanguages`**`()`

Returns an array of languages associated with this audio content or a zero length array if not known.

Returns:

the languages associated with this audio.

org.ocap.hn.content

Interface ChannelContentItem

All Superinterfaces:
ContentEntry, ContentItem

```
public interface ChannelContentItem
extends ContentItem
```

This interface represents a video or audio broadcast channel object.

Field Summary

Fields inherited from interface org.ocap.hn.content.ContentItem
AUDIO_ITEM, AUDIO_ITEM_BOOK, AUDIO_ITEM_BROADCAST, AUDIO_ITEM_TRACK, IMAGE_ITEM, IMAGE_ITEM_PHOTO, ITEM, VIDEO_ITEM, VIDEO_ITEM_BROADCAST, VIDEO_ITEM_MOVIE, VIDEO_ITEM_MUSIC_CLIP

Method Summary

OcapLocator	**getChannelLocator**() Gets the locator for this ChannelContentItem set in createChannelContentItem.
java.lang.String	**getChannelName**() Gets The channel name for this ChannelContentItem
java.lang.String	**getChannelNumber**() Gets The channel number for this ChannelContentItem
java.lang.String	**getChannelTitle**() Gets The title for this ChannelContentItem, or null if the title is unknown.
java.lang.String	**getChannelType**() Gets The channel type for this ChannelContentItem
ExtendedFileAccessPermissions	**getExtendedFileAccessPermissions**() Gets the extended file access permissions for this ChannelContentItem.
OcapLocator	**getTuningLocator**() Gets the frequency based tuning locator used for service resolution.
boolean	**setTuningLocator**(OcapLocator locator) Sets the tuning locator for this ChannelContentItem that the implementation can use for tuning a broadcast channel.

Methods inherited from interface org.ocap.hn.content.ContentItem
containsResource, deleteEntry, getContentClass, getItemService, getRenderableResources, getResource, getResourceCount, getResourceIndex, getResources, getTitle, hasAudio, hasStillImage, hasVideo, isRenderable

Methods inherited from interface org.ocap.hn.content.ContentEntry
`getContentSize, getCreationDate, getEntryParent, getID, getParentID, getRootMetadataNode, getServer, isLocal`

Method Detail

getChannelType

`java.lang.String` **`getChannelType`**`()`

Gets The channel type for this ChannelContentItem

Returns:

the String channel type for this item, or null if unknown.

getChannelNumber

`java.lang.String` **`getChannelNumber`**`()`

Gets The channel number for this ChannelContentItem

Returns:

The String channel number for this item, or null if unknown.

getChannelName

`java.lang.String` **`getChannelName`**`()`

Gets The channel name for this ChannelContentItem

Returns:

the String channel name for this item, or null if unknown.

getChannelTitle

`java.lang.String` **`getChannelTitle`**`()`

Gets The title for this ChannelContentItem, or null if the title is unknown.

Returns:

the String title for this item, or null if unknown.

getChannelLocator

`OcapLocator` **`getChannelLocator`**`()`

Gets the locator for this ChannelContentItem set in createChannelContentItem.

Returns:

The locator for this ChannelContentItem, returns null if the isLocal method returns false.

getExtendedFileAccessPermissions

`ExtendedFileAccessPermissions` **`getExtendedFileAccessPermissions`**`()`

Gets the extended file access permissions for this ChannelContentItem.

Specified by:

`getExtendedFileAccessPermissions` in interface `ContentEntry`

Returns:

The extended file access permissions.

getTuningLocator

`OcapLocator` **`getTuningLocator`**`()`

Gets the frequency based tuning locator used for service resolution.

Returns:

The frequency based tuning locator if previously resolved, null otherwise.

setTuningLocator

```
boolean setTuningLocator(OcapLocator locator)
                         throws javax.tv.locator.InvalidLocatorException
```

Sets the tuning locator for this ChannelContentItem that the implementation can use for tuning a broadcast channel. Returns false if the #isLocal method returns false.

An application may call this method to update a channel's tuning parameters (for example, when an SDV channel's program number or frequency changes). Upon a successful update of the channel's tuning parameters the implementation SHALL be responsible for updating any active streaming sessions to the new tuning parameters. When a JavaTV Service represents this ChannelContentItem the implementation SHALL modify the transport dependent locator of the Service to match the locator parameter.

Setting the channel tuning locator to a null locator makes the channel item no longer tunable (for example, when the application ends an SDV session). Setting the tuning locator to null while streaming ends streaming, removes the locator, and `StreamingActivityListener.notifyStreamingEnded(org.ocap.hn.content.ContentItem, int, int)` is called with an activityID of `ACTIVITY_END_SERVICE_VANISHED`.

Parameters:

`locator` - A frequency-based locator for the channel, or null to remove the current locator.

Returns:

true when the locator parameter is set correctly, otherwise, returns false.

Throws:

`javax.tv.locator.InvalidLocatorException` - if the locator parameter is not frequency based and is not null.

`java.lang.SecurityException` - if the calling application is not granted write permission by the permissions returned from the getExtendedFileAccessPermissions method.

org.ocap.hn.content

Interface ContentContainer

All Superinterfaces:
ContentEntry

```
public interface ContentContainer
extends ContentEntry
```

This class represents a container that contains one or more content entries. Can contain children containers.

Field Summary

static java.lang.String	**ALBUM_CONTAINER** Represents the base album container.
static java.lang.String	**ALBUM_CONTAINER_MUSIC** Represents a music album container.
static java.lang.String	**ALBUM_CONTAINER_PHOTO** Represents a photo album container.
static java.lang.String	**CHANNEL_GROUP_CONTAINER** Represents the (extended tuner) channel group container class.
static java.lang.String	**CONTAINER** Represents the base container class.
static java.lang.String	**GENRE_CONTAINER** Represents an unordered collection of 'objects' that "belong" to the genre.
static java.lang.String	**GENRE_CONTAINER_MOVIE** Represents a movie genre container.
static java.lang.String	**GENRE_CONTAINER_MUSIC** Represents a music genre container.
static java.lang.String	**PERSON_CONTAINER** Represents an unordered collection of 'objects' that "belong" to the people.
static java.lang.String	**PERSON_CONTAINER_MUSIC_ARTIST** Represents a music artist person container.
static java.lang.String	**PLAYLIST_CONTAINER** Represents a collection of objects.
static java.lang.String	**STORAGE_FOLDER_CONTAINER** Represents all, or a partition of some physical storage unit of a single type.
static java.lang.String	**STORAGE_SYSTEM_CONTAINER** Represents a potentially heterogeneous collection of storage media.
static java.lang.String	**STORAGE_VOLUME_CONTAINER** Represents all, or a partition of, some physical storage unit of a single type.

Method Summary

boolean	**addContentEntries**(ContentEntry[] entries) Adds ContentEntry objects to this ContentContainer.
boolean	**addContentEntry**(ContentEntry entry) Adds a ContentEntry to this ContentContainer.
boolean	**contains**(ContentEntry entry) Checks whether the given ContentEntry is in this ContentContainer in local cache only.
ContentContainer	**createChannelGroupContainer**(java.lang.String name, ExtendedFileAccessPermissions permissions) Creates a new channel group ContentContainer as a child of this ContentContainer, when the host device is capable of supporting tuner requests from the home network.
boolean	**createContentContainer**(java.lang.String name, ExtendedFileAccessPermissions permissions) Creates a new ContentContainer as a child of this ContentContainer.
boolean	**createContentItem**(java.io.File content, java.lang.String name, ExtendedFileAccessPermissions permissions) Creates a new ContentItem representing a local file as a child of this ContentContainer.
boolean	**delete**() Deletes this ContentContainer if and only if it is empty.
boolean	**deleteContents**() Deletes all the ContentEntry objects in this container except for ContentContainer entries.
boolean	**deleteEntry**() Deletes this ContentContainer and all of the ContentEntry objects in this container.
boolean	**deleteRecursive**(boolean recursive) If the recursive parameter is true, this method behaves in a manner equivalent to deleteEntry().
int	**getComponentCount**() Gets the number of ContentEntry objects in this ContentContainer.
java.lang.String	**getContainerClass**() Returns the container class of this container.
long	**getContentSize**() Gets the size of the ContentContainer and all its content including all its contained ContentContainer objects.
java.util.Date	**getCreationDate**() Returns the creation date of this ContentContainer.
java.util.Enumeration	**getEntries**() Gets an Enumeration over all entries in this ContentContainer, from local cache only; does not cause network activity.

Method Summary

ContentList	**getEntries**(ContentDatabaseFilter filter, boolean traverse) Returns a ContentList which contains the filtered ContentItems of this ContentContainer.
ContentEntry	**getEntry**(int n) Returns the n^{th} ContentEntry in this container, from local cache only; does not cause network activity.
ContentEntry	**getEntry**(java.lang.String ID) Returns the ContentEntry associated with the given ID in this container, or null if no entry is found.
ExtendedFileAccessPermissions	**getExtendedFileAccessPermissions**() Gets the ExtendedFileAccessPermissions of this ContentContainer.
int	**getIndex**(ContentEntry n) Gets the index of the specified ContentEntry, from local cache only; does not cause network activity.
java.lang.String	**getName**() Gets the name of this ContentContainer.
boolean	**isEmpty**() Returns an empty indication.
boolean	**removeContentEntries**(ContentEntry[] entries) Removes ContentEntry objects from this ContentContainer.
boolean	**removeContentEntry**(ContentEntry entry) Removes a ContentEntry from this ContentContainer.
ContentEntry[]	**toArray**() Returns an array of all `ContentEntry` in this `ContentContainers` including other `ContentContainers`.

Methods inherited from interface org.ocap.hn.content.ContentEntry
getEntryParent, getID, getParentID, getRootMetadataNode, getServer, isLocal

Field Detail

CONTAINER

`static final java.lang.String` **`CONTAINER`**

Represents the base container class.

See Also:

Constant Field Values

ALBUM_CONTAINER

`static final java.lang.String` **`ALBUM_CONTAINER`**

Represents the base album container.

See Also:

Constant Field Values

ALBUM_CONTAINER_PHOTO

`static final java.lang.String` **`ALBUM_CONTAINER_PHOTO`**

Represents a photo album container. In addition to being an ALBUM_CONTAINER container may be a photo album.

See Also:
Constant Field Values

ALBUM_CONTAINER_MUSIC

`static final java.lang.String` **`ALBUM_CONTAINER_MUSIC`**

Represents a music album container. In addition to being an ALBUM_CONTAINER container may be a music album.

See Also:
Constant Field Values

GENRE_CONTAINER

`static final java.lang.String` **`GENRE_CONTAINER`**

Represents an unordered collection of 'objects' that "belong" to the genre.

See Also:
Constant Field Values

GENRE_CONTAINER_MUSIC

`static final java.lang.String` **`GENRE_CONTAINER_MUSIC`**

Represents a music genre container. In addition to being a GENRE_CONTAINER a container may be a music genre container

See Also:
Constant Field Values

GENRE_CONTAINER_MOVIE

`static final java.lang.String` **`GENRE_CONTAINER_MOVIE`**

Represents a movie genre container. In addition to being a GENRE_CONTAINER a container may be a movie genre container

See Also:
Constant Field Values

PLAYLIST_CONTAINER

`static final java.lang.String` **`PLAYLIST_CONTAINER`**

Represents a collection of objects.

See Also:
Constant Field Values

PERSON_CONTAINER

`static final java.lang.String` **`PERSON_CONTAINER`**

Represents an unordered collection of 'objects' that "belong" to the people.

See Also:
Constant Field Values

PERSON_CONTAINER_MUSIC_ARTIST

`static final java.lang.String` **`PERSON_CONTAINER_MUSIC_ARTIST`**

Represents a music artist person container. In addition to being a PERSON_CONTAINER a container may be a music artist.

See Also:
Constant Field Values

STORAGE_SYSTEM_CONTAINER

`static final java.lang.String` **`STORAGE_SYSTEM_CONTAINER`**

Represents a potentially heterogeneous collection of storage media.

See Also:
Constant Field Values

STORAGE_VOLUME_CONTAINER

`static final java.lang.String` **`STORAGE_VOLUME_CONTAINER`**

Represents all, or a partition of, some physical storage unit of a single type.

See Also:
Constant Field Values

STORAGE_FOLDER_CONTAINER

`static final java.lang.String` **`STORAGE_FOLDER_CONTAINER`**

Represents all, or a partition of some physical storage unit of a single type.

See Also:
Constant Field Values

CHANNEL_GROUP_CONTAINER

`static final java.lang.String` **`CHANNEL_GROUP_CONTAINER`**

Represents the (extended tuner) channel group container class.

See Also:
Constant Field Values

Method Detail

getContainerClass

`java.lang.String` **`getContainerClass`**`()`

Returns the container class of this container.

Returns:
The content class of this item.

See Also:
`ALBUM_CONTAINER`, `ALBUM_CONTAINER_MUSIC`, `ALBUM_CONTAINER_PHOTO`, `GENRE_CONTAINER`, `GENRE_CONTAINER_MUSIC`, `GENRE_CONTAINER_MOVIE`, `PLAYLIST_CONTAINER`, `PERSON_CONTAINER`, `PERSON_CONTAINER_MUSIC_ARTIST`, `STORAGE_SYSTEM_CONTAINER`, `STORAGE_VOLUME_CONTAINER`, `STORAGE_FOLDER_CONTAINER`, `CHANNEL_GROUP_CONTAINER`

toArray

`ContentEntry[]` **`toArray`**`()`

Returns an array of all `ContentEntry` in this `ContentContainers` including other `ContentContainers`. Returns ContentEntry objects stored in local cache only; does not cause network activity.

Returns:

array containing all entries of this ContentContainers

contains

`boolean` **`contains`**`(ContentEntry entry)`

Checks whether the given `ContentEntry` is in this ContentContainer in local cache only.

Parameters:

`entry` - To search for in this ContentEntry.

Returns:

true if the ContentEntry is contained in this container, otherwise returns false.

getEntry

`ContentEntry` **`getEntry`**`(java.lang.String ID)`

Returns the ContentEntry associated with the given ID in this container, or null if no entry is found. This method SHALL recursively search this container and any sub-containers. This method searches local cache only; does not cause network activity.

Parameters:

`ID` - String ID of the ContentEntry to return

Returns:

the associated ContentEntry.

See Also:

`ContentEntry.getID()`

getEntry

`ContentEntry` **`getEntry`**`(int n)`

Returns the n^{th} ContentEntry in this container, from local cache only; does not cause network activity.

Parameters:

`n` - Index of the entry to get.

Returns:

the n^{th} ContentEntry.

Throws:

`java.lang.ArrayIndexOutOfBoundsException` - if the n^{th} value does not exist.

getEntries

`java.util.Enumeration` **`getEntries`**`()`

Gets an Enumeration over all entries in this ContentContainer, from local cache only; does not cause network activity.

Returns:

Enumeration over all entries in this ContentContainers, or null if there are no entries.

getIndex

`int` **`getIndex`**`(ContentEntry n)`

Gets the index of the specified ContentEntry, from local cache only; does not cause network activity.

Parameters:

`n` - The index of the ContentEntry to search for.

Returns:

The index of the ContentEntry or -1 if it doesn't exist in this container.

createContentItem

```
boolean createContentItem(java.io.File content,
                          java.lang.String name,
                          ExtendedFileAccessPermissions permissions)
```

Creates a new ContentItem representing a local file as a child of this ContentContainer. If this ContentContainer #isLocal method returns false this method will return false. The resulting ContentItem will contain a single ContentResource containing the content parameter passed to this method.

Parameters:

`content` - The file containing the content to be represented

`name` - The name of the new ContentItem.

`permissions` - Access permissions of the new ContentContainer.

Returns:

True if a new ContentItem has been created, otherwise return false; specifically, returns false if this container is a channel group container.

Throws:

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentmanagement"), or if the caller does not have write permission on this container.

createContentContainer

```
boolean createContentContainer(java.lang.String name,
                               ExtendedFileAccessPermissions permissions)
```

Creates a new ContentContainer as a child of this ContentContainer. If this ContentContainer #isLocal method returns false this method will return false. This ContentContainer may not contain ChannelContentItem entries. Can be used to create a directory structure.

Parameters:

`name` - The name of the new ContentContainer.

`permissions` - Access permissions of the new ContentContainer.

Returns:

true if a new ContentContainer has been created, otherwise returns false.

Throws:

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentmanagement"), or if the caller does not have write permission on this container.

createChannelGroupContainer

```
ContentContainer createChannelGroupContainer(java.lang.String name,

ExtendedFileAccessPermissions permissions)
```

Creates a new channel group `ContentContainer` as a child of this ContentContainer, when the host device is capable of supporting tuner requests from the home network. This channel group only contains `ChannelContentItem` instances representing broadcast channels that can be tuned by the host device. If this ContentContainer #isLocal method returns false, this method will return null. If the ContentServerNetModule that contains this ContentContainer is not prepared to support tuners, this method will return null.

Parameters:

`name` - The name of the new ContentContainer.

`permissions` - Access permissions of the new ContentContainer.

Returns:

ContentContainer if a new ContentContainer has been created, otherwise returns null.

Throws:

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentmanagement"), or if the caller does not have write permission on this container.

getEntries

```
ContentList getEntries(ContentDatabaseFilter filter,
                       boolean traverse)
```

Returns a ContentList which contains the filtered ContentItems of this ContentContainer. If the traverse parameter is true the ContentItems of all its children ContentContainers is included. The list returned is filtered by the filter parameter. If the filter is null all items are returned.

Parameters:

`filter` - A ContentDatabaseFilter to filter the ContentItems. If the filter is null all entries are returned

`traverse` - If true entries in the sub-containers are returned, otherwise only entries in this ContentContainer are returned.

Returns:

a ContentList filtered by the `ContentDatabaseFilter`

getName

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

Gets the name of this ContentContainer.

Returns:

The name of this ContentContainer.

See Also:

`ContentEntry.getID()`

delete

```
boolean delete()
                throws java.io.IOException
```

Deletes this ContentContainer if and only if it is empty. This method removes the content container from its parent. This method returns false if this is a root container. This method deletes a local ContentContainer only. If the #isLocal method returns false an exception is thrown.

Returns:

True if this ContentContainer was deleted, otherwise returns false.

Throws:

`java.lang.SecurityException` - if the application is denied to perform the action

`java.io.IOException` - if this ContentContainer is not local.

deleteContents

```
boolean deleteContents()
                       throws java.io.IOException
```

Deletes all the ContentEntry objects in this container except for ContentContainer entries. This method deletes local ContentEntry instances only. If the #isLocal method returns false, an exception is thrown.

Returns:

true if all of the ContentEntry objects required to be deleted are deleted, otherwise returns false (e.g. ContentContainer entries are not required to be deleted).

Throws:

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentmanagement"), or if the caller does not have write permission on this container or any entries contained in this container (except for ContentContainer entries).

`java.io.IOException` - if this ContentContainer is not local.

deleteEntry

```
boolean deleteEntry()
                    throws java.io.IOException
```

Deletes this ContentContainer and all of the ContentEntry objects in this container. Calls the `ContentEntry.deleteEntry()` method on each ContentEntry in a recursive manner. This method deletes local ContentEntry instances only. If the #isLocal method returns false, an exception is thrown. If a SecurityException is thrown due to insufficient write access permissions on any entry contained within this ContentContainer, this method MAY delete a partial subset of the entries contained within.

Specified by:

`deleteEntry` in interface `ContentEntry`

Returns:

true if this ContentContainer and all of the ContentEntry objects in this container were deleted, otherwise returns false.

Throws:

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentmanagement"), or if the caller does not have write permission on this container or any entries contained in this container.

`java.io.IOException` - if this ContentContainer is not local.

deleteRecursive

```
boolean deleteRecursive(boolean recursive)
                        throws java.io.IOException
```

If the recursive parameter is true, this method behaves in a manner equivalent to `deleteEntry()`. If the recursive parameter is false, this method behaves in a manner equivalent to `deleteContents()`. This method deletes local ContentEntry instances only. If the #isLocal method returns false, an exception is thrown. If a SecurityException is thrown due to insufficient write access permissions on any entry contained within this ContentContainer, this method MAY delete a partial subset of the entries contained within.

Parameters:

`recursive` - if true all entries and their entries are to be deleted.

Returns:

True if all ContentEntry objects that are required to be deleted are deleted, otherwise returns false.

Throws:

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentmanagement"), or if the caller does not have write permission on this container or any entries contained in this container.

`java.io.IOException` - if this ContentContainer is not local.

See Also:

`deleteContents()`, `delete()`

addContentEntry

```
boolean addContentEntry(ContentEntry entry)
```

Adds a ContentEntry to this ContentContainer. Can only add local ContentEntry objects to local ContentContainer. If this entry is already has a parent ContentContainer, it will be removed from that container.

Parameters:

`entry` - the content entry to be added to this container

Returns:

True if the entry was added. Returns false if the isLocal method of this ContentContainer or the parameter ContentEntry returns false. Returns false if this ContentContainer is a channel group container, and the entry is not a ChannelContentItem.

Throws:

`java.lang.IllegalStateException` - if this ContentContainer does not have a parentID property, i.e., this ContentContainer is not added to the CDS.
`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentmanagement"), or if the caller does not have write permission on this container.

addContentEntries

`boolean` **`addContentEntries`**`(ContentEntry[] entries)`

Adds ContentEntry objects to this ContentContainer. Can only add local ContentEntry objects to local ContentContainer. If any entry passed to this method already has a parent ContentContainer, it will be removed from that container.

Parameters:
`entries` - the content entries to be added to this container
Returns:
True if the entries were added. Returns false if the isLocal method of this ContentContainer or the parameter ContentEntry returns false. Returns false if this container is a channel group container and ANY entry in the input argument is NOT a ChannelContentItem.
Throws:
`java.lang.IllegalStateException` - if this ContentContainer does not have a parentID property, i.e., this ContentContainer is not added to the CDS.
`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentmanagement"), or if the caller does not have write permission on this container.

removeContentEntry

`boolean` **`removeContentEntry`**`(ContentEntry entry)`

Removes a ContentEntry from this ContentContainer. Can only remove local ContentEntry objects from local ContentContainers. When the ContentEntry parameter is a ContentContainer, all of its ContentEntry objects are removed from the parameter. For entries that are ContentContainer objects, a possible implementation is a recursive traversal where these objects are removed in a bottom-up fashion by calling this method on each one.

Parameters:
`entry` - the content entry to be removed from this container
Returns:
True if the entry was removed. Returns false if the isLocal method of this ContentContainer or the parameter ContentEntry is not contained in this container.
Throws:
`java.lang.IllegalArgumentException` - if the ContentEntry parameter is a NetRecordingEntry which contains one or more RecordingContentItems.
`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentmanagement"), or if the caller does not have write permission on this container.

removeContentEntries

`boolean` **`removeContentEntries`**`(ContentEntry[] entries)`

Removes ContentEntry objects from this ContentContainer. Can only remove local ContentEntry objects from local ContentContainer. If any ContentEntry is not contained within this container, this method will return false and no entries will be removed. When the ContentEntry parameter is a ContentContainer, all of its ContentEntry objects are removed from the parameter. For entries that are ContentContainer objects, a possible implementation is a recursive traversal where these objects are removed in a bottom-up fashion by calling removeContentEntry method on each one.

Parameters:

`entries` - the content entries to be removed from this container

Returns:

True if the entries were removed. Returns false if the isLocal method of this ContentContainer or if any of the ContentEntry objects are not contained in this container.

Throws:

`java.lang.IllegalArgumentException` - if the parameter includes a NetRecordingEntry which contains one or more RecordingContentItems.

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentmanagement"), or if the caller does not have write permission on this container.

getContentSize

`long` **`getContentSize`**`()`

Gets the size of the ContentContainer and all its content including all its contained ContentContainer objects. Note that the size may have changed during the call to this method.

Specified by:

`getContentSize` in interface `ContentEntry`

Returns:

The content size in bytes or -1 if the size is indeterminate.

getCreationDate

`java.util.Date` **`getCreationDate`**`()`

Returns the creation date of this ContentContainer.

Specified by:

`getCreationDate` in interface `ContentEntry`

Returns:

The Date the content was created or null if the creation date is indeterminate.

getExtendedFileAccessPermissions

`ExtendedFileAccessPermissions` **`getExtendedFileAccessPermissions`**`()`

Gets the ExtendedFileAccessPermissions of this ContentContainer.

Specified by:

`getExtendedFileAccessPermissions` in interface `ContentEntry`

Returns:

The ExtendedFileAccessPermission.

getComponentCount

`int` **`getComponentCount`**`()`

Gets the number of ContentEntry objects in this ContentContainer. Does not include component count of entries within ContentContainer objects contained in this ContentContainer.

Returns:

Number of entries.

isEmpty

`boolean` **`isEmpty`**`()`

Returns an empty indication.

Returns:

True if this ContentContainer does not contain any ContentEntry objects, otherwise returns false.

org.ocap.hn.content

Interface ContentEntry

All Known Subinterfaces:

ContentContainer, ContentItem, NetRecordingEntry, RecordingContentItem

```
public interface ContentEntry
```

This interface represents a basic content entry. Each ContentEntry instance can only be contained in one ContentContainer and the implementation SHALL create a new ContentEntry for equal entries placed in multiple ContentContainer instances.

Method Summary

boolean	**deleteEntry**() Deletes this ContentEntry.
long	**getContentSize**() Gets the size of the content associated with this ContentEntry.
java.util.Date	**getCreationDate**() Gets the creation date of the content associated with this ContentEntry.
ContentContainer	**getEntryParent**() Returns the `ContentContainer` this ContentEntry belongs to.
ExtendedFileAccessPermissions	**getExtendedFileAccessPermissions**() Gets the file permissions of this ContentEntry, or null if unknown.
java.lang.String	**getID**() Returns the ID of this ContentEntry.
java.lang.String	**getParentID**() Returns the ID of `ContentContainer` this ContentEntry belongs to.
MetadataNode	**getRootMetadataNode**() Gets the metadata for this ContentEntry.
ContentServerNetModule	**getServer**() Gets the server where this ContentEntry is located.
boolean	**isLocal**() Returns true if this content entry is on the local device, false if it is hosted by another device on the network.

Method Detail

getID

```
java.lang.String getID()
```

Returns the ID of this ContentEntry. The format of this string ID is implementation and protocol mapping dependent.

Returns:
The ID of content entry.

getServer

`ContentServerNetModule` **`getServer`**`()`

Gets the server where this ContentEntry is located.
Returns:
The server housing this container.

deleteEntry

`boolean` **`deleteEntry`**`()`
`throws java.io.IOException`

Deletes this ContentEntry. This is a local delete only. If the #isLocal method returns false, this method SHALL throw an exception. This method does not delete any content associated with this content entry.
Returns:
True if the ContentEntry was deleted, otherwise returns false.
Throws:
`java.lang.SecurityException` - if the calling application does not have write ExtendedFileAccessPermission for this entry.
`java.io.IOException` - if the entry is not local.

getEntryParent

`ContentContainer` **`getEntryParent`**`()`
`throws java.io.IOException`

Returns the `ContentContainer` this ContentEntry belongs to. This method SHALL return null if this ContentEntry represents a root container. If it is determined that this ContentEntry has a parent container, but the implementation does not have sufficient local cached information to construct the ContentContainer, this method SHALL throw an IOException.
Returns:
The parent ContentContainer.
Throws:
`java.io.IOException` - if the implementation does not have sufficient local cached information to construct the parent ContentContainer

getParentID

`java.lang.String` **`getParentID`**`()`

Returns the ID of `ContentContainer` this ContentEntry belongs to. This method SHALL return "-1" if this ContentEntry represents a root container. This method SHALL return null if the parent ID is unknown.
Returns:
the ID of this entry's parent container
See Also:
`getID()`, `getEntryParent()`

getContentSize

`long` **`getContentSize`**`()`

Gets the size of the content associated with this ContentEntry.
Returns:
The content size in bytes or -1 if unknown.

getCreationDate

`java.util.Date` **`getCreationDate`**`()`

Gets the creation date of the content associated with this ContentEntry.

Returns:

The Date the content was created or null if unknown.

getExtendedFileAccessPermissions

`ExtendedFileAccessPermissions` **`getExtendedFileAccessPermissions`**`()`

Gets the file permissions of this ContentEntry, or null if unknown.

Returns:

The extended file access permissions of this ContentEntry or null if unknown.

getRootMetadataNode

`MetadataNode` **`getRootMetadataNode`**`()`

Gets the metadata for this ContentEntry.

Returns:

Root MetadataNode.

isLocal

`boolean` **`isLocal`**`()`

Returns true if this content entry is on the local device, false if it is hosted by another device on the network.

Returns:

true if the content is local, false otherwise

org.ocap.hn.content

Class ContentEntryFactory

```
java.lang.Object
  └org.ocap.hn.content.ContentEntryFactory
```

```
public abstract class ContentEntryFactory
extends java.lang.Object
```

This factory can be used to create `ContentEntry` instances. There are specialty methods for application convenience when creating channel content items.

Constructor Summary

protected	**ContentEntryFactory**() Singleton behavior.

Method Summary

abstract ChannelContentItem	**createChannelContentItem**(java.lang.String channelType, java.lang.String channelTitle, java.lang.String channelName, java.lang.String channelNumber, OcapLocator channelLocator, ExtendedFileAccessPermissions permissions) Creates a new `ChannelContentItem` representing a broadcast channel.
static ContentEntryFactory	**getInstance**() Gets an instance of the factory.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

ContentEntryFactory

```
protected ContentEntryFactory()
```

Singleton behavior.

Method Detail

getInstance

```
public static ContentEntryFactory getInstance()
```

Gets an instance of the factory.

Returns:
A content entry factory instance.

createChannelContentItem

```
public abstract ChannelContentItem
createChannelContentItem(java.lang.String channelType,

java.lang.String channelTitle,

java.lang.String channelName,

java.lang.String channelNumber,

OcapLocator channelLocator,

ExtendedFileAccessPermissions permissions)
```

Creates a new `ChannelContentItem` representing a broadcast channel. A `ChannelContentItem` can only be added to a container created by `createChannelGroupContainer`.
This content item is not automatically added to a parent container. Applications can publish multiple channels in a single method call by creating an array of `ChannelContentItem` instances and passing it to the `addContentEntries` of a container created by `createChannelGroupContainer` in CDS.
At the point that the created ChannelContentItem is requested by a DMC, the implementation SHALL determine if the channelLocator is transport-dependent (e.g. a frequency-based Locator) and use the Locator to acquire the Service for streaming. If the channelLocator is a transport-independent (e.g. a sourceID-based) Locator and can be resolved via JavaTV, the implementation SHALL use JavaTV (e.g. SIManager.getService) to acquire the Service for streaming. If the transport-independent Locator cannot be resolved via JavaTV, and the Locator returned from getTuningLocator() is null, the implementation SHALL invoke the ServiceResolutionHandler.resolveChannelItem method. If resolveChannelItem returns true, the implementation SHALL use this Locator returned from getTuningLocator() to acquire the Service for streaming.

Parameters:

`channelType` - The type of broadcast channel, can be one of `VIDEO_ITEM_BROADCAST`, `VIDEO_ITEM_BROADCAST_VOD`, or `AUDIO_ITEM_BROADCAST` defined in { @link ContentItem }.

`channelTitle` - The title of the new ChannelContentItem.

`channelName` - The name of the new ChannelContentItem.

`channelNumber` - String representing type and channel number, where type must be "Analog" or "Digital", number is (major channel number) for analog channels, and number is (major channel and minor channel) for digital channels, in the format "type, number [, number]". For example: "Analog,12", or "Digital,15,2". May also be null, when application is not providing this information.

`channelLocator` - An `OcapLocator` for the channel

`permissions` - Access permissions of the new `ChannelContentItem` for local server applications only.

Returns:
true if a new ChannelContentItem has been created, otherwise return false.

Throws:

`java.lang.IllegalArgumentException` - if channelType is not AUDIO_ITEM_BROADCAST, `VIDEO_ITEM_BROADCAST_VOD`, or VIDEO_ITEM_BROADCAST, or if channelNumber format is invalid.

`java.lang.NullPointerException` - if channelName, channelTitle, or channelLocator arguments passed to this method are null.

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("contentmanagement").

org.ocap.hn.content

Interface ContentFormat

All Known Subinterfaces:

OutputVideoContentFormat

```
public interface ContentFormat
```

This interface represents a content format. Instances of this interface represent the specific formats in which content is either available or can be transformed into.

Method Summary

java.lang.String	**getContentProfile**() This method returns an identifier representing the media format of the content.
java.lang.String	**getProtectionType**() Returns the protection type of the content.

Method Detail

getContentProfile

```
java.lang.String getContentProfile()
```

This method returns an identifier representing the media format of the content. See [OC-BUNDLE] and `ContentProfile` for a list of valid identifiers. See `ContentProfile`

Returns:

The media format of the content.

getProtectionType

```
java.lang.String getProtectionType()
```

Returns the protection type of the content. This method returns the `ProtectionType` registered as an approved output with CableLabs. See `ProtectionType`

Returns:

The protection type. Returns empty string if the content is not protected.

org.ocap.hn.content

Interface ContentItem

All Superinterfaces:
ContentEntry
All Known Subinterfaces:
ChannelContentItem, RecordingContentItem

```
public interface ContentItem
extends ContentEntry
```

This class represents a piece of content. This can be audio, video or still image content. It is not directly linked to any file. This is done via the ContentResources. A ContentItem can have multiple ContentResources.

Field Summary

static java.lang.String	**AUDIO_ITEM** Represents the base audio content.
static java.lang.String	**AUDIO_ITEM_BOOK** In addition to being an AUDIO_ITEM content MAY be an audio book.
static java.lang.String	**AUDIO_ITEM_BROADCAST** In addition to being an AUDIO_ITEM content MAY be broadcast on a radio station.
static java.lang.String	**AUDIO_ITEM_TRACK** In addition to being an AUDIO_ITEM content MAY be a track such as a song.
static java.lang.String	**IMAGE_ITEM** Base image item.
static java.lang.String	**IMAGE_ITEM_PHOTO** In addition to being an IMAGE_ITEM content MAY be a photo.
static java.lang.String	**ITEM** Represents the base content item.
static java.lang.String	**VIDEO_ITEM** Represents the base video item.
static java.lang.String	**VIDEO_ITEM_BROADCAST** In addition to being a VIDEO_ITEM content MAY be a video broadcast.
static java.lang.String	**VIDEO_ITEM_BROADCAST_VOD** In addition to being a VIDEO_ITEM_BROADCAST content MAY be VOD content.
static java.lang.String	**VIDEO_ITEM_MOVIE** In addition to being a VIDEO_ITEM content MAY be a movie.
static java.lang.String	**VIDEO_ITEM_MUSIC_CLIP** In addition to being a VIDEO_ITEM content MAY be a music video clip, e.g., music video.

Field Summary

static java.lang.String	**VIDEO_ITEM_VPOP** In addition to being a VIDEO_ITEM content MAY be a VPOP item; the item represents content binary that is presenting to video output ports, e.g., HDMI, component.

Method Summary

boolean	**containsResource**(ContentResource entry) Checks whether the given ContentResource is part of this ContentItem.
boolean	**deleteEntry**() Deletes this ContentItem.
java.lang.String	**getContentClass**() Returns the content class of this content item.
javax.tv.service.Service	**getItemService**() If this ContentItem is presentable as a JavaTV Service than this method returns a javax.tv.service.Service, or derivative of a Service, e.g., RecordedService, which can be used to play this ContentItem.
ContentResource[]	**getRenderableResources**() Gets an array copy of renderable ContentResources which are part of this ContentItem.
ContentResource	**getResource**(int n) Returns the nth ContentResource of this ContentItem.
int	**getResourceCount**() Returns the number of ContentResources which are associated with this ContentItem.
int	**getResourceIndex**(ContentResource r) Returns the index of the specified ContentResource or -1 if the ContentResource does not exist in this ContentItem.
ContentResource[]	**getResources**() Gets an array copy of ContentResources which are part of this ContentItem.
java.lang.String	**getTitle**() Gets the title for this ContentItem, or null if the title is unknown.
boolean	**hasAudio**() Returns a boolean indicating if this content has audio.
boolean	**hasStillImage**() Returns a boolean indicating if the ContentItem has a still image.
boolean	**hasVideo**() Returns a boolean indicating if the ContentItem has video associated with it.
boolean	**isRenderable**() Checks whether the local device has the capabilities to render this content item.

Methods inherited from interface org.ocap.hn.content.ContentEntry
`getContentSize, getCreationDate, getEntryParent, getExtendedFileAccessPermissions, getID, getParentID, getRootMetadataNode, getServer, isLocal`

Field Detail

ITEM

`static final java.lang.String` **`ITEM`**

Represents the base content item.

See Also:

Constant Field Values

AUDIO_ITEM

`static final java.lang.String` **`AUDIO_ITEM`**

Represents the base audio content.

See Also:

Constant Field Values

AUDIO_ITEM_TRACK

`static final java.lang.String` **`AUDIO_ITEM_TRACK`**

In addition to being an AUDIO_ITEM content MAY be a track such as a song.

See Also:

Constant Field Values

AUDIO_ITEM_BROADCAST

`static final java.lang.String` **`AUDIO_ITEM_BROADCAST`**

In addition to being an AUDIO_ITEM content MAY be broadcast on a radio station.

See Also:

Constant Field Values

AUDIO_ITEM_BOOK

`static final java.lang.String` **`AUDIO_ITEM_BOOK`**

In addition to being an AUDIO_ITEM content MAY be an audio book.

See Also:

Constant Field Values

VIDEO_ITEM

`static final java.lang.String` **`VIDEO_ITEM`**

Represents the base video item.

See Also:

Constant Field Values

VIDEO_ITEM_MOVIE

`static final java.lang.String` **`VIDEO_ITEM_MOVIE`**

In addition to being a VIDEO_ITEM content MAY be a movie.

See Also:

Constant Field Values

VIDEO_ITEM_BROADCAST

`static final java.lang.String` **`VIDEO_ITEM_BROADCAST`**

In addition to being a VIDEO_ITEM content MAY be a video broadcast.

See Also:

Constant Field Values

VIDEO_ITEM_BROADCAST_VOD

`static final java.lang.String` **`VIDEO_ITEM_BROADCAST_VOD`**

In addition to being a VIDEO_ITEM_BROADCAST content MAY be VOD content. For Broadcast VOD content the implementation SHOULD NOT support random access or trick modes.

See Also:

Constant Field Values

VIDEO_ITEM_MUSIC_CLIP

`static final java.lang.String` **`VIDEO_ITEM_MUSIC_CLIP`**

In addition to being a VIDEO_ITEM content MAY be a music video clip, e.g. music video.

See Also:

Constant Field Values

VIDEO_ITEM_VPOP

`static final java.lang.String` **`VIDEO_ITEM_VPOP`**

In addition to being a VIDEO_ITEM content MAY be a VPOP item; the item represents content binary that is presenting to video output ports, e.g. HDMI, component.

See Also:

Constant Field Values

IMAGE_ITEM

`static final java.lang.String` **`IMAGE_ITEM`**

Base image item.

See Also:

Constant Field Values

IMAGE_ITEM_PHOTO

`static final java.lang.String` **`IMAGE_ITEM_PHOTO`**

In addition to being an IMAGE_ITEM content MAY be a photo.

See Also:

Constant Field Values

Method Detail

hasAudio

`boolean` **`hasAudio`**`()`

Returns a boolean indicating if this content has audio.

Returns:

True if the content type has audio, otherwise returns false.

hasVideo

`boolean` **`hasVideo`**`()`

Returns a boolean indicating if the ContentItem has video associated with it.

Returns:

True if the ContentItem contains video, otherwise returns false.

hasStillImage

`boolean` **`hasStillImage`**`()`

Returns a boolean indicating if the ContentItem has a still image.

Returns:

True if the ContentItem has a still image, otherwise returns false.

getItemService

`javax.tv.service.Service` **`getItemService`**`()`

If this ContentItem is presentable as a JavaTV Service than this method returns a javax.tv.service.Service, or derivative of a Service, e.g. RecordedService, which can be used to play this ContentItem. If the ContentItem is not local the returned Service SHALL be a RemoteService. If the content associated with item has been deleted or is no longer accessible, this method SHALL return null.

Returns:

A JavaTV service if this content is presentable as a Service, null otherwise.

getContentClass

`java.lang.String` **`getContentClass`**`()`

Returns the content class of this content item.

Returns:

The content class of this item.

See Also:

`AUDIO_ITEM`, `AUDIO_ITEM_BOOK`, `AUDIO_ITEM_BROADCAST`, `AUDIO_ITEM_TRACK`, `IMAGE_ITEM`, `VIDEO_ITEM`, `VIDEO_ITEM_BROADCAST`, `VIDEO_ITEM_BROADCAST_VOD`, `VIDEO_ITEM_MOVIE`, `VIDEO_ITEM_MUSIC_CLIP`, `VIDEO_ITEM_VPOP`, `IMAGE_ITEM_PHOTO`

getTitle

`java.lang.String` **`getTitle`**`()`

Gets the title for this ContentItem, or null if the title is unknown.

Returns:

the String title for this item, or null if unknown.

deleteEntry

`boolean` **`deleteEntry`**`()`
`throws java.io.IOException`

Deletes this ContentItem. Calls the `ContentResource.delete()` method on each ContentResource contained in this ContentItem. Deletes a local ContentItem only. If the #isLocal method returns false an exception is thrown.

Note: this overrides the definition of `ContentEntry.deleteEntry()`. If an application calls the ContentEntry.deleteEntry method on an object that is an instance of ContentItem, the implementation SHALL delete the ContentItem as defined by this method.

Specified by:

`deleteEntry` in interface `ContentEntry`

Returns:

True if the ContentItem was deleted, otherwise returns false.

Throws:

`java.lang.SecurityException` - if the application does not have write ExtendedFileAccessPermission.

`java.io.IOException` - if the ContentItem is not local.

getResourceCount

`int` **`getResourceCount`**`()`

Returns the number of `ContentResource`s which are associated with this ContentItem.

Returns:

number of ContentResources.

getResource

`ContentResource` **`getResource`**`(int n)`

Returns the n^{th} `ContentResource` of this ContentItem.

Parameters:

`n` - the index of the ContentResource

Returns:

the n^{th}

Throws:

`java.lang.ArrayIndexOutOfBoundsException` - if the n^{th} value does not exist.

getResourceIndex

`int` **`getResourceIndex`**`(ContentResource r)`

Returns the index of the specified ContentResource or -1 if the ContentResource does not exist in this ContentItem.

Parameters:

`r` - The ContentResource to check for.

Returns:

The index of the ContentResource or -1 if it doesn't exist in this ContentItem.

containsResource

`boolean` **`containsResource`**`(ContentResource entry)`

Checks whether the given `ContentResource` is part of this ContentItem.

Parameters:

`entry` - The ContentResource to check for.

Returns:

True if the ContentResource is part of this ContentItem, otherwise returns false.

getResources

`ContentResource[]` **`getResources`**`()`

Gets an array copy of ContentResources which are part of this ContentItem.

Returns:

Array of ContentResources.

getRenderableResources

`ContentResource[]` **`getRenderableResources`**`()`

Gets an array copy of renderable ContentResources which are part of this ContentItem.

Returns:

Array of ContentResources contained in this ContentItem for which ContentResource.isRenderable() returns true.

isRenderable

`boolean` **`isRenderable`**`()`

Checks whether the local device has the capabilities to render this content item. This includes the ability to negotiate media protocol with the host device, the ability of the local device to render this content item's media format, and sufficient access permissions for the calling application. This method will return true if any of the ContentResources contained in this ContentItem are renderable. This call does not consider immediate availability of resources required for presentation of this content.

Returns:

true if this content is renderable on the local device, false otherwise.

org.ocap.hn.content

Interface ContentProfile

`public interface` **`ContentProfile`**

Interface defining constants that represent content profile identifiers to be used in conjunction with the `ContentFormat` interface. See [OC-BUNDLE] for a list of content profile identifiers.

Field Summary

`static java.lang.String`	**`DASH_MPD`** MPEG-DASH Content.

Field Detail

DASH_MPD

`static final java.lang.String` **`DASH_MPD`**

MPEG-DASH Content.

See Also:

Constant Field Values

org.ocap.hn.content

Interface ContentResource

All Known Subinterfaces:

AudioResource, StreamableContentResource, VideoResource

```
public interface ContentResource
```

Abstract class representing a media stream/file. Subclasses of this class can implement `AudioResource` and/or `VideoResource` depending on whether the content represents audio and/or video.

Field Summary

`static java.lang.String`	**`UNKNOWN_MIME_TYPE`** Constant for an unknown MIME type.

Method Summary

`boolean`	**`delete`**`()` Deletes the binary representation of this `ContentResource` and the `ContentResource` is removed from any containing `ContentEntry`.
`java.lang.String`	**`getContentFormat`**`()` Returns the content format.
`ContentItem`	**`getContentItem`**`()` Gets the ContentItem this resource belongs to.
`long`	**`getContentSize`**`()` Gets the size of the content in bytes or -1 if not known.
`java.util.Date`	**`getCreationDate`**`()` Gets the creation date of the content or NULL if not known.
`ExtendedFileAccessPermissions`	**`getExtendedFileAccessPermissions`**`()` Returns the file permissions of a ContentResource.
`javax.tv.locator.Locator`	**`getLocator`**`()` Gets an OcapLocator to the content associated with this ContentResource if the content can be located with that Locator type, otherwise returns an implementation specific Locator to the content.
`java.lang.String`	**`getNetwork`**`()` Returns the network on which the content is available.
`java.lang.String`	**`getProtocol`**`()` Returns the protocol which can be used to retrieve the content.
`java.lang.Object`	**`getResourceProperty`**`(java.lang.String key)` Returns properties of the resource.
`ContentFormat[]`	**`getTransformedContentFormats`**`()` Returns a list of transformed `ContentFormat` that are available.

Method Summary

boolean	**isRenderable**() Checks whether the local device has the capabilities to render this content resource.

Field Detail

UNKNOWN_MIME_TYPE

```
static final java.lang.String UNKNOWN_MIME_TYPE
```

Constant for an unknown MIME type.

See Also:

Constant Field Values

Method Detail

delete

```
boolean delete()
                throws java.io.IOException
```

Deletes the binary representation of this `ContentResource` and the `ContentResource` is removed from any containing `ContentEntry`. The `ContentResource` is not valid anymore after this call. This method deletes a local `ContentResource` only. If the #isLocal method on the containing `ContentItem` returns false an exception is thrown. Does not delete the content associated with this `ContentResource`; see #getLocator.

Returns:

True if this ContentResource was deleted, otherwise returns false.

Throws:

`java.lang.SecurityException` - if the application does not have a write ExtendedFileAccessPermission.

`java.io.IOException` - if the associated `ContentItem` is not local.

getContentItem

```
ContentItem getContentItem()
```

Gets the ContentItem this resource belongs to.

Returns:

The ContentItem parent of this resource or null if this ContentResource is an independent ContentEntry.

getContentSize

```
long getContentSize()
```

Gets the size of the content in bytes or -1 if not known.

Returns:

the content size in bytes

getCreationDate

```
java.util.Date getCreationDate()
```

Gets the creation date of the content or NULL if not known.

Returns:

The Date the content was created.

getLocator

`javax.tv.locator.Locator` **`getLocator`**`()`

Gets an OcapLocator to the content associated with this ContentResource if the content can be located with that Locator type, otherwise returns an implementation specific Locator to the content.

Returns:
Locator to the content associated with this entry.

getExtendedFileAccessPermissions

`ExtendedFileAccessPermissions` **`getExtendedFileAccessPermissions`**`()`

Returns the file permissions of a ContentResource.

Returns:
the extended file access permissions of the ContentEntry.

getProtocol

`java.lang.String` **`getProtocol`**`()`

Returns the protocol which can be used to retrieve the content. The returned String can be a wild card "*". Possible protocols are

- "http-get"
- "rtsp-rtp-udp"
- "internal"
- "iec61883"
- Registered ICANN domain name of vendor

Returns:
String representation of the protocol

getNetwork

`java.lang.String` **`getNetwork`**`()`

Returns the network on which the content is available. The returned String can be a wild card "*".

<protocol>	**<network>**
"http-get"	"*"
"rtsp-rtp-udp"	"*"
"internal"	IP address of the device hosting the Connection manager
"iec61883"	GUID of the 1394 bus Isochronous Resource Manager
ICANN domain	Vendor-defined, may be "*"

Returns:
String describing the network on which the resource is available.

getContentFormat

`java.lang.String` **`getContentFormat`**`()`

Returns the content format. The returned String can be a wild card "*".

<protocol>	**<network>**
"http-get"	MIME-type

"rtsp-rtp-udp"	Name of RTP payload type
"internal"	Vendor-defined, may be "*"
"iec61883"	Name standardised by IEC61883
ICANN domain	Vendor-defined, may be "*"

Returns:
String describing the content format.

getResourceProperty

`java.lang.Object` **`getResourceProperty`**`(java.lang.String key)`

Returns properties of the resource. There is a set of defined properties which can be accessed via the methods in `AudioResource` and `VideoResource`. This method allows for custom or new properties.

Parameters:
`key` - The key of the property.

Returns:
The value of the property, or null if the key parameter does not match any property.

isRenderable

`boolean` **`isRenderable`**`()`

Checks whether the local device has the capabilities to render this content resource. This includes the ability to negotiate media protocol with the host device, the ability of the local device to render this content item's media format, and sufficient access permissions for the calling application. This call does not consider immediate availability of resources required for presentation of this content.

Returns:
true if this content is renderable on the local device, false otherwise.

getTransformedContentFormats

`ContentFormat[]` **`getTransformedContentFormats`**`()`
`throws java.io.IOException`

Returns a list of transformed `ContentFormat` that are available. There may be multiple `ContentFormat` instances returned that correspond to a single `ContentResource` in case of HTTP Adaptive content. If the #isLocal method on the containing `ContentItem` returns false an exception is thrown.

Returns:
Array of ContentFormat.

Throws:
`java.io.IOException` - if the associated ContentItem is not local.

org.ocap.hn.content

Class DatabaseException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └org.ocap.hn.content.DatabaseException
```

All Implemented Interfaces:

java.io.Serializable

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

Exception that is thrown when a database error occurs

See Also:

Serialized Form

Field Summary

static int	**FIELD_IS_EMPTY**
static int	**FIELD_IS_WRONG_FORMAT**
static int	**FIELD_NAME_DOES_NOT_EXIST**
static int	**GENERAL_ERROR**
static int	**INVALID_PARAMETER_SPECIFIED**
static int	**QUERY_IS_INVALID**
static int	**REMOTE_QUERY_IS_INVALID**
static int	**UNABLE_TO_LOCATE_SERVICE**

Constructor Summary

DatabaseException(java.lang.String sMessage, int iNumber)

Method Summary

int	**getExceptionNumber**() Returns the unique exception code related to the database.

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

Field Detail

FIELD_NAME_DOES_NOT_EXIST

```
public static final int FIELD_NAME_DOES_NOT_EXIST
```

See Also:

Constant Field Values

FIELD_IS_EMPTY

```
public static final int FIELD_IS_EMPTY
```

See Also:

Constant Field Values

FIELD_IS_WRONG_FORMAT

```
public static final int FIELD_IS_WRONG_FORMAT
```

See Also:

Constant Field Values

QUERY_IS_INVALID

```
public static final int QUERY_IS_INVALID
```

See Also:

Constant Field Values

INVALID_PARAMETER_SPECIFIED

```
public static final int INVALID_PARAMETER_SPECIFIED
```

See Also:

Constant Field Values

UNABLE_TO_LOCATE_SERVICE

```
public static final int UNABLE_TO_LOCATE_SERVICE
```

See Also:

Constant Field Values

REMOTE_QUERY_IS_INVALID

```
public static final int REMOTE_QUERY_IS_INVALID
```

See Also:

Constant Field Values

GENERAL_ERROR

```
public static final int GENERAL_ERROR
```

See Also:

Constant Field Values

Constructor Detail

DatabaseException

```
public DatabaseException(java.lang.String sMessage,
                         int iNumber)
```

Method Detail

getExceptionNumber

```
public int getExceptionNumber()
```

Returns the unique exception code related to the database.

Returns:

the integer value with an exception code

org.ocap.hn.content

Interface IOStatus

```
public interface IOStatus
```

This interface represents the ability to detect whether any asset represented by an object or its children is in use on the home network and hence the object should not be deleted.

ContentResource and ContentEntry objects representing local assets on a server SHALL implement IOStatus.

Method Summary

boolean	**isInUse**() Returns an indication of whether any asset within this object is in use on the home network.

Method Detail

isInUse

```
boolean isInUse()
```

Returns an indication of whether any asset within this object is in use on the home network. "In Use" is indicated if there is an active network transport protocol session (for example HTTP, RTSP) to the asset.

For objects which logically contain other objects, recursively iterates through all logical children of this object. For ContentContainer objects, recurses through all ContentEntry objects they contain. For NetRecordingEntry objects, iterates through all RecordingContentItem objects they contain.

Returns:
True if there is an active network transport protocol session to any asset that this ContentResource, ContentEntry, or any children of the ContentEntry contain, otherwise false.

org.ocap.hn.content

Class MetadataIdentifiers

```
java.lang.Object
  └org.ocap.hn.content.MetadataIdentifiers
```

```
public abstract class MetadataIdentifiers
extends java.lang.Object
```

This abstract class represents access to standardized metadata identifiers. Each identifier, e.g. "title", can be used to search for corresponding metadata in a ContentList. The set of identifiers returned by the #getIdentifiers method SHALL contain the PROPRIETARY_DATA identifier and MAY contain identifiers defined in other OCAP HN profiles, e.g. UPnP.

Field Summary

static java.lang.String	**PROPRIETARY_DATA** This identifies proprietary data.

Constructor Summary

MetadataIdentifiers()

Method Summary

static boolean	**contains**(java.lang.String identifier) Indicates if the parameter identifier is contained within the set of supported identifiers.
static java.lang.String[]	**getIdentifiers**() Gets all metadata identifiers for all HN profiles supported by this Host device.
static int	**getNumberOfIdentifiers**() Gets the number of identifiers in the set of supported identifiers returned by the #getIdentifiers method.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

PROPRIETARY_DATA

```
public static final java.lang.String PROPRIETARY_DATA
```

This identifies proprietary data. The Object returned when using this as a Metadata identifier is defined by the application creating the metadata.

The value of this field is an OCAP defined string `"ocap:proprietaryData"`. If the proprietary data is an array of bytes the data should be transported as a base 64 String.

See Also:
Constant Field Values

Constructor Detail

MetadataIdentifiers

`public MetadataIdentifiers()`

Method Detail

getIdentifiers

`public static java.lang.String[] getIdentifiers()`

Gets all metadata identifiers for all HN profiles supported by this Host device.

Returns:
Array of Metadata identifiers.

getNumberOfIdentifiers

`public static int getNumberOfIdentifiers()`

Gets the number of identifiers in the set of supported identifiers returned by the #getIdentifiers method.

Returns:
Number of supported metadata identifiers.

contains

`public static boolean contains(java.lang.String identifier)`

Indicates if the parameter identifier is contained within the set of supported identifiers.

Parameters:
`identifier` - Name of the identifier to search for.

Returns:
True if the identifier is supported, otherwise returns false.

org.ocap.hn.content

Class MetadataNode

```
java.lang.Object
  └org.ocap.hn.content.MetadataNode
```

```
public abstract class MetadataNode
extends java.lang.Object
```

A collection of metadata entries, each of which is a key/value pair where the key identifies a property and the value is the property value, and a collection of supporting namespace declarations. Each property may have an `ExtendedFileAccessPermissions` defining its accessibility. Property values (also called metadata) can be `MetadataNode`s; thus treelike metadata structures are supported.

One example is given below:

```
 RootNode
 |
 --- TITLE    - String("Best Movie Ever")
 |
 --- CREW     - MetadataNode
 |                  |
 |                  --- MAIN_ACTOR  - String("Joe Sixpack")
 |                  |
 |                  --- MAIN_ACTOR2 - String("Doris Dosenkohl")
 |
 --- SYNOPSIS - String("Don't know - I fell asleep after 5 seconds")
```

It is possible to get metadata from nested `MetadataNode`s directly by concatenating the different keys using # to separate them. In the above example, getMetadata("CREW#MAIN_ACTOR") would return "Joe Sixpack".

The `MetadataNode` represents the current snapshot of metadata associated with a network entity as cached on the local device. This may or may not reflect an accurate or complete view of the metadata that exists on the network. It is the responsibility of the application to explicitly update metadata using the home network APIs (e.g., `ContentServerNetModule.requestSearchEntries(String, String, int, int, String, String, org.ocap.hn.NetActionHandler)`).

Constructor Summary

protected	**MetadataNode**() Constructor; only to be invoked by subclass constructors.

Method Summary

abstract void	**addMetadata**(java.lang.String key, java.lang.Object value) Adds a new metadata entry, modifies an existing metadata entry, or removes an existing metadata entry.

Method Summary

abstract void	**addMetadata**(java.lang.String key, java.lang.Object value, ExtendedFileAccessPermissions efap) Adds a new metadata entry, modifies an existing metadata entry, or removes an existing metadata entry, with specification of ExtendedFileAccessPermissions.
abstract void	**addNameSpace**(java.lang.String namespace, java.lang.String URI) Adds a namespace declaration to this MetadataNode.
static MetadataNode	**createMetadataNode**() Creates a MetadataNode.
static MetadataNode	**createMetadataNode**(java.lang.String key) **Deprecated.** *Replaced by createMetadataNode().* *Creates a MetadataNode.*
abstract ExtendedFileAccessPermissions	**getExtendedFileAccessPermissions** (java.lang.String key) Gets the ExtendedFileAccessPermissions for the property identified by the specified key.
abstract java.lang.String	**getKey**() Gets the key string, which can be utilized to retrieve this MetadataNode from its parent.
abstract java.lang.String[]	**getKeys**() Gets the keys for all top-level metadata in this MetadataNode.
abstract java.util.Enumeration	**getMetadata**() Gets an Enumeration of all top-level metadata in this MetadataNode.
abstract java.lang.Object	**getMetadata**(java.lang.String key) Gets a reference to the value associated with the specified key, if any.
abstract MetadataNode	**getParentNode**() Gets the parent MetadataNode of this MetadataNode.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

MetadataNode

protected **MetadataNode**()

Constructor; only to be invoked by subclass constructors.

Method Detail

createMetadataNode

`public static MetadataNode createMetadataNode()`

Creates a `MetadataNode`.

Returns:
The newly created `MetadataNode`.

createMetadataNode

`public static MetadataNode createMetadataNode(java.lang.String key)`

Deprecated. *Replaced by `createMetadataNode()`. Creates a `MetadataNode`.*

Parameters:
`key` - Unused. (In particular, it SHALL NOT be used to set the key string retrievable by the `getKey` method.)

Returns:
The newly created `MetadataNode`.

getMetadata

`public abstract java.lang.Object getMetadata(java.lang.String key)`

Gets a reference to the value associated with the specified key, if any.

In order to query metadata in a hierarchy, query strings like IDENT1#IDENT2#IDENT3 are possible, where IDENT1 maps to a `MetadataNode` in this `MetadataNode`, IDENT2 a `MetadataNode` in the IDENT1 `MetadataNode`, and IDENT3 an `Object` in the IDENT2 `MetadataNode`.

This method only returns locally cached metadata. This method SHALL NOT cause network activity.

Parameters:
`key` - The key.

Returns:
A reference to the value associated with the key, or null if there is none.

Throws:
`java.lang.SecurityException` - if the calling application does not have extended file access permission to read from the property identified by the key argument.

addMetadata

```
public abstract void addMetadata(java.lang.String key,
                                 java.lang.Object value)
```

Adds a new metadata entry, modifies an existing metadata entry, or removes an existing metadata entry.

If the key argument does not contain an "@" character, the value argument must be either null or a reference to an instance of one of the following classes:

- `org.dvb.application.AppID`
- `org.ocap.storage.ExtendedFileAccessPermissions`
- a class extending `org.ocap.hn.content.MetadataNode` and not implementing `java.io.Serializable`
- a nonarray class implementing `java.io.Serializable`

- `T[]`, where `T` is one of the above

If the key argument contains an "@" character but does not begin with an "@" character, the value argument must be either null or a reference to an instance of one of the following classes:

- `java.lang.String`
- `java.lang.String[]`

If the key argument begins with an "@" character, the value argument must be either null or a reference to an instance of the following class:

- `java.lang.String`

If the value argument is null or a reference to a zero-length array, any existing value associated with the key argument SHALL be removed. Otherwise, if a value is already associated with the key argument, the value argument's referent SHALL replace that value; if not, the value argument's referent SHALL be associated with the key argument.

Keys MAY contain a namespace prefix as part of their definition. Namespace prefixes SHALL appear in keys as a colon separated prefix in the key (e.g. *upnp:class*). Vendor-specific namespace prefixes MUST be declared in this `MetadataNode` using the `addNameSpace` method prior to usage in this method, or be a valid namespace prefix in any containing parent `MetadataNode`.

Each property added by this method, whether new or pre-existing, SHALL be given an `ExtendedFileAccessPermissions` that matches that of the closest containing `ContentEntry`, if there is one.

Parameters:
`key` - The key, e.g. "TITLE". When the value argument is a reference to an instance of a class extending `MetadataNode`, the implementation SHALL set that `MetadataNode`'s key string retrievable by the `getKey` method to this key.
`value` - Null or a reference to the value to associate with the key.
Throws:
`java.lang.IllegalArgumentException` - if the key argument contains an undeclared namespace prefix or a "#" character, or if the value argument is neither null nor a reference to an instance of an acceptable class, or if the value argument is a reference to an instance of a class that implements `java.io.Serializable` but that instance cannot be successfully serialized.
`java.lang.SecurityException` - if the property identified by the key argument does not exist and the calling application does not have sufficient file access permissions to write to the `ContentEntry` containing this `MetadataNode`, or if the property identified by the key argument does exist and the calling application does not have sufficient file access permissions to write to the property.
See Also:
`addNameSpace(String namespace, String URI)`

addMetadata

```
public abstract void addMetadata(java.lang.String key,
                                 java.lang.Object value,
                                 ExtendedFileAccessPermissions efap)
```

Adds a new metadata entry, modifies an existing metadata entry, or removes an existing metadata entry, with specification of `ExtendedFileAccessPermissions`.

If the key argument does not contain an "@" character, the value argument must be either null or a reference to an instance of one of the following classes:

- `org.dvb.application.AppID`

- `org.ocap.storage.ExtendedFileAccessPermissions`
- a class extending `org.ocap.hn.content.MetadataNode` and not implementing `java.io.Serializable`
- a nonarray class implementing `java.io.Serializable`
- `T[]`, where `T` is one of the above

If the key argument contains an "@" character but does not begin with an "@" character, the value argument must be either null or a reference to an instance of one of the following classes:

- `java.lang.String`
- `java.lang.String[]`

If the key argument begins with an "@" character, the value argument must be either null or a reference to an instance of the following class:

- `java.lang.String`

If the value argument is null or a reference to a zero-length array, any existing value associated with the key argument SHALL be removed. Otherwise, if a value is already associated with the key argument, the value argument's referent SHALL replace that value; if not, the value argument's referent SHALL be associated with the key argument.

Keys MAY contain a namespace prefix as part of their definition. Namespace prefixes SHALL appear in keys as a colon separated prefix in the key (e.g. *upnp:class*). Vendor-specific namespace prefixes MUST be declared in this `MetadataNode` using the `addNameSpace` method prior to usage in this method, or be a valid namespace prefix in any containing parent `MetadataNode`.

Each property added by this method, whether new or pre-existing, SHALL be given an `ExtendedFileAccessPermissions` that matches the efap argument.

Parameters:
`key` - The key, e.g. "TITLE". When the value argument is a reference to an instance of a class extending `MetadataNode`, the implementation SHALL set that `MetadataNode`'s key string retrievable by the `getKey` method to this key.
`value` - Null or a reference to the value to associate with the key.
`efap` - The `ExtendedFileAccessPermissions` for the property added or modified by this method.
Throws:
`java.lang.IllegalArgumentException` - if the key argument contains an undeclared namespace prefix or a "#" character, or if the value argument is neither null nor a reference to an instance of an acceptable class, or if the value argument is a reference to an instance of a class that implements `java.io.Serializable` but that instance cannot be successfully serialized, or if the efap argument is null.
`java.lang.SecurityException` - if the property identified by the key argument does not exist and the calling application does not have sufficient file access permissions to write to the `ContentEntry` containing this `MetadataNode`, or if the property identified by the key argument does exist and the calling application does not have sufficient file access permissions to write to the property.
See Also:
`addNameSpace(String namespace, String URI)`

addNameSpace

```
public abstract void addNameSpace(java.lang.String namespace,
```

java.lang.String URI)

Adds a namespace declaration to this MetadataNode. Once a namespace is declared, its prefix SHALL be valid as a metadata key qualifier for this MetadataNode and any of its descendants.

Parameters:

namespace - The namespace prefix (e.g. "av").

URI - The namespace name (e.g. "urn:schemas-upnp-org:av:av").

getMetadata

public abstract java.util.Enumeration **getMetadata**()

Gets an Enumeration of all top-level metadata in this MetadataNode.

This method only returns an Enumeration of locally cached metadata. This method SHALL NOT cause network activity.

Returns:

An Enumeration of all top-level metadata in this MetadataNode.

getParentNode

public abstract MetadataNode **getParentNode**()

Gets the parent MetadataNode of this MetadataNode.

Returns:

The parent MetadataNode of this MetadataNode, or null if there is none.

getKey

public abstract java.lang.String **getKey**()

Gets the key string, which can be utilized to retrieve this MetadataNode from its parent. If this MetadataNode has no parent, this method SHALL return null.

Returns:

The key string associated with this MetadataNode.

getKeys

public abstract java.lang.String[] **getKeys**()

Gets the keys for all top-level metadata in this MetadataNode.

This method only returns the keys for locally cached metadata. This method SHALL NOT cause network activity.

Returns:

An array of Strings that are the keys for all top-level metadata in this MetadataNode.

getExtendedFileAccessPermissions

public abstract ExtendedFileAccessPermissions
getExtendedFileAccessPermissions(java.lang.String key)

Gets the ExtendedFileAccessPermissions for the property identified by the specified key.

In order to query metadata in a hierarchy, query strings like IDENT1#IDENT2#IDENT3 are possible, where IDENT1 maps to a MetadataNode in this MetadataNode, IDENT2 a MetadataNode in the IDENT1 MetadataNode, and IDENT3 an Object in the IDENT2 MetadataNode.

Parameters:

key - The key identifying the property to get the ExtendedFileAccessPermissions for.

Returns:

The `ExtendedFileAccessPermissions` of the property identified by the key, or null if there is no such property or if the property has no `ExtendedFileAccessPermissions`.

org.ocap.hn.content

Interface OutputVideoContentFormat

All Superinterfaces:
ContentFormat

```
public interface OutputVideoContentFormat
extends ContentFormat
```

This interface provides additional parameters for transforming video content.

Method Summary

int	**getBitRate**() Returns the bit rate in bits per second (bps) of the output content.
int	**getHorizontalResolution**() Returns the content maximum horizontal resolution in pixels.
int	**getVerticalResolution**() Returns the content maximum vertical resolution in pixels.
boolean	**isProgressive**() Returns an indication of progressive or interlaced.

Methods inherited from interface org.ocap.hn.content.ContentFormat
getContentProfile, getProtectionType

Method Detail

getVerticalResolution

```
int getVerticalResolution()
```

Returns the content maximum vertical resolution in pixels.

Returns:
The vertical content resolution.

getHorizontalResolution

```
int getHorizontalResolution()
```

Returns the content maximum horizontal resolution in pixels.

Returns:
The content horizontal resolution.

getBitRate

```
int getBitRate()
```

Returns the bit rate in bits per second (bps) of the output content.

Returns:
The bitrate of the content in bits per second

isProgressive

`boolean` **`isProgressive`**`()`

Returns an indication of progressive or interlaced.

Returns:

True if progressive, false if interlaced.

org.ocap.hn.content

Interface ProtectionType

```
public interface ProtectionType
```

Interface defining constants that represent supported output protection types to be used in conjunction with the `ContentFormat` interface.

Field Summary

`static java.lang.String`	**`DECE_DRM`** DECE DRM type.
`static java.lang.String`	**`DTCP_IP`** DTCP-IP protection type as defined in [OC-BUNDLE].

Field Detail

DTCP_IP

```
static final java.lang.String DTCP_IP
```

DTCP-IP protection type as defined in [OC-BUNDLE].

See Also:
Constant Field Values

DECE_DRM

```
static final java.lang.String DECE_DRM
```

DECE DRM type. Used with any DRM that is part of DECE as defined in [OC-BUNDLE].

See Also:
Constant Field Values

org.ocap.hn.content

Interface StreamableContentResource

All Superinterfaces:

ContentResource

```
public interface StreamableContentResource
extends ContentResource
```

Abstract class representing content that can be streamed, e.g., MPEG file.

Field Summary

Fields inherited from interface org.ocap.hn.content.ContentResource
UNKNOWN_MIME_TYPE

Method Summary

long	**getBitrate**() Gets the Bitrate this content is encoded with or -1 if not known.
javax.media.Time	**getDuration**() Gets the duration of this content.

Methods inherited from interface org.ocap.hn.content.ContentResource
delete, getContentFormat, getContentItem, getContentSize, getCreationDate, getExtendedFileAccessPermissions, getLocator, getNetwork, getProtocol, getResourceProperty, isRenderable

Method Detail

getDuration

javax.media.Time **getDuration**()

Gets the duration of this content.

Returns:

the duration of the content or null if the duration is not known.

getBitrate

long **getBitrate**()

Gets the Bitrate this content is encoded with or -1 if not known.

Returns:

the bit rate of the content in bytes per second or -1 if the bit rate is not known.

org.ocap.hn.content

Interface StreamingActivityListener

All Superinterfaces:
java.util.EventListener

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

This interface represents a listener for notification of streaming being started, changed or ended

Field Summary

static int	**ACTIVITY_END_ACCESS_WITHDRAWN**
static int	**ACTIVITY_END_NETWORK_TIMEOUT**
static int	**ACTIVITY_END_OTHER**
static int	**ACTIVITY_END_RESOURCE_REMOVED**
static int	**ACTIVITY_END_SERVICE_VANISHED** Reason for activity ended.
static int	**ACTIVITY_END_USER_STOP**
static int	**CONTENT_TYPE_ALL_RESOURCES** contentTypes.
static int	**CONTENT_TYPE_LIVE_RESOURCES**
static int	**CONTENT_TYPE_RECORDED_RESOURCES**

Method Summary

void	**notifyStreamingChange**(ContentItem contentItem, int activityID, java.lang.String URI, NetworkInterface tuner) Notifies the StreamingActivityListener when streaming parameter on this activity such as tuning parameter changes
void	**notifyStreamingEnded**(ContentItem contentItem, int activityID, int reasonOfEnd) Notifies the StreamingActivityListener when content streaming to the home network in response to a request for ContentItem streaming has ended.
void	**notifyStreamingStarted**(ContentItem contentItem, int activityID, java.lang.String URI, NetworkInterface tuner) Notifies the StreamingActivityListener when content begins streaming the content to the home network in response to a request for ContentItem streaming.

Field Detail

ACTIVITY_END_SERVICE_VANISHED

`static final int` **`ACTIVITY_END_SERVICE_VANISHED`**

Reason for activity ended.

See Also:

Constant Field Values

ACTIVITY_END_RESOURCE_REMOVED

`static final int` **`ACTIVITY_END_RESOURCE_REMOVED`**

See Also:

Constant Field Values

ACTIVITY_END_ACCESS_WITHDRAWN

`static final int` **`ACTIVITY_END_ACCESS_WITHDRAWN`**

See Also:

Constant Field Values

ACTIVITY_END_USER_STOP

`static final int` **`ACTIVITY_END_USER_STOP`**

See Also:

Constant Field Values

ACTIVITY_END_NETWORK_TIMEOUT

`static final int` **`ACTIVITY_END_NETWORK_TIMEOUT`**

See Also:

Constant Field Values

ACTIVITY_END_OTHER

`static final int` **`ACTIVITY_END_OTHER`**

See Also:

Constant Field Values

CONTENT_TYPE_ALL_RESOURCES

`static final int` **`CONTENT_TYPE_ALL_RESOURCES`**

contentTypes.

See Also:

Constant Field Values

CONTENT_TYPE_LIVE_RESOURCES

`static final int` **`CONTENT_TYPE_LIVE_RESOURCES`**

See Also:

Constant Field Values

CONTENT_TYPE_RECORDED_RESOURCES

`static final int` **`CONTENT_TYPE_RECORDED_RESOURCES`**

See Also:

Constant Field Values

Method Detail

notifyStreamingStarted

```
void notifyStreamingStarted(ContentItem contentItem,
                            int activityID,
                            java.lang.String URI,
                            NetworkInterface tuner)
```

Notifies the `StreamingActivityListener` when content begins streaming the content to the home network in response to a request for `ContentItem` streaming.

Parameters:

`contentItem` - The `ContentItem` requested.

`activityID` - A unique value assigned by the implementation for this streaming activity.

`URI` - the URI of the content that's been requested.

`tuner` - The `NetworkInterface` representing the tuner the content is being streamed from if used. The value will be null if no tuner is used.

notifyStreamingChange

```
void notifyStreamingChange(ContentItem contentItem,
                           int activityID,
                           java.lang.String URI,
                           NetworkInterface tuner)
```

Notifies the `StreamingActivityListener` when streaming parameter on this activity such as tuning parameter changes

Parameters:

`contentItem` - The `ContentItem` associated with this activity

`activityID` - A unique value assigned by the implementation for this streaming activity.

`URI` - the URI of the content that's been changed.

`tuner` - The `NetworkInterface` representing the tuner the content is being streamed from if used. The value will be null if no tuner is used.

notifyStreamingEnded

```
void notifyStreamingEnded(ContentItem contentItem,
                          int activityID,
                          int reasonOfEnd)
```

Notifies the `StreamingActivityListener` when content streaming to the home network in response to a request for `ContentItem` streaming has ended.

Parameters:

`contentItem` - The `ContentItem` requested.

`activityID` - A unique value assigned by the implementation for this streaming activity.

`reasonOfEnd` - Unique value defined in this class for the end of this streaming activity

org.ocap.hn.content

Interface VideoResource

All Superinterfaces:
ContentResource

```
public interface VideoResource
extends ContentResource
```

ContentResource to identify that a content item contains video/still image material.

Field Summary

Fields inherited from interface org.ocap.hn.content.ContentResource
UNKNOWN_MIME_TYPE

Method Summary

int	**getColorDepth**() Returns the color depth (in bits) of the video/still image.
java.awt.Dimension	**getResolution**() Returns the resolution of the video/still image.

Methods inherited from interface org.ocap.hn.content.ContentResource
delete, getContentFormat, getContentItem, getContentSize, getCreationDate, getExtendedFileAccessPermissions, getLocator, getNetwork, getProtocol, getResourceProperty, isRenderable

Method Detail

getResolution

java.awt.Dimension **getResolution**()

Returns the resolution of the video/still image.

Returns:
the resolution of the video/still image or null if the resolution is not known.

getColorDepth

int **getColorDepth**()

Returns the color depth (in bits) of the video/still image.

Returns:
the color depth (in bits) of the video/still image or -1 if the color depth is not known.

Annex C Content Navigation API

Package org.ocap.hn.content.navigation

Interface Summary	
ContentList	This interface represents a list of filtered ContentEntry objects.

Class Summary	
ContentDatabaseFilter	This class represent a filtering criteria to be applied while creating a ContentList.
DatabaseQuery	DatabaseQuery class.
DeviceFilter	This class represents a filtering criteria based on a particular device.

org.ocap.hn.content.navigation

Class ContentDatabaseFilter

```
java.lang.Object
  └ org.ocap.hn.content.navigation.ContentDatabaseFilter
```

Direct Known Subclasses:

DatabaseQuery, DeviceFilter

```
public abstract class ContentDatabaseFilter
extends java.lang.Object
```

This class represent a filtering criteria to be applied while creating a ContentList.

Constructor Summary

protected	**ContentDatabaseFilter**()

Method Summary

abstract boolean	**accept**(ContentEntry entry) This method is called for every ContentEntry in the database/list this filter is applied to.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

ContentDatabaseFilter

```
protected ContentDatabaseFilter()
```

Method Detail

accept

```
public abstract boolean accept(ContentEntry entry)
```

This method is called for every ContentEntry in the database/list this filter is applied to. Implementations should return true if the specified ContentItem should be in the filtered list. If the ContentItem should not be listed in the new list, false should be returned.

Parameters:

`entry` - the ContentEntry to filter

Returns:

true if the ContentEntry should be in the filtered ContentList, false otherwise.

org.ocap.hn.content.navigation

Interface ContentList

All Superinterfaces:
java.util.Enumeration

```
public interface ContentList
extends java.util.Enumeration
```

This interface represents a list of filtered ContentEntry objects. A ContentList may contain a complete or partial subset of entries resulting from an application requested filter, browse or search.

Method Summary

ContentList	**filterContentList**(ContentDatabaseFilter filter) Filters the ContentList.
ContentEntry	**find**(java.lang.String[] keys, java.lang.Object[] values) Finds the first ContentEntry which matches the search.
ContentEntry	**find**(java.lang.String key, java.lang.Object value) Finds the first ContentEntry which identifier for the key 'key' equals the given object obj.
ContentList	**findAll**(java.lang.String[] keys, java.lang.Object[] values) Finds all ContentEntry objects which match the search.
java.lang.String	**getSortOrder**() Gets the sort order set by the #setSortOrder method.
void	**setSortOrder**(java.lang.String sortOrder) Sets the metadata sort order of the items in this list based on metadata key identifiers using signed property values.
int	**size**() Gets the number of ContentEntry objects in this ContentList.
int	**totalMatches**() Gets to total number of ContentEntry matches in the filter, browse or search operation that generated this ContentList.

Methods inherited from interface java.util.Enumeration
hasMoreElements, nextElement

Method Detail

size

```
int size()
```

Gets the number of ContentEntry objects in this ContentList.

Returns:
Number of entries in this list. Returns 0 if the list is empty.

totalMatches

`int totalMatches()`

Gets to total number of ContentEntry matches in the filter, browse or search operation that generated this ContentList. This value SHALL be greater than or equal to the value returned from the size() method of this ContentList. This value SHALL be greater than the value returned from the size() method of this ContentList if the *requestedCount* parameter of the originating content entry request was less than the total entry matches of the requesting operation. See `ContentServerNetModule`.

Returns:

the total number of ContentEntry matches from the originating content entry request

setSortOrder

`void setSortOrder(java.lang.String sortOrder)`

Sets the metadata sort order of the items in this list based on metadata key identifiers using signed property values. The sortOrder parameter of this method is a string containing the properties and sort modifiers to be used to sort the resulting ContentList. The format of the string containing the sort criteria shall follow the format defined in UPnP Content Directory Service 3.0 specification section 2.3.16: A_ARG_TYPE_SortCriteria.

Parameters:

`sortOrder` - a String representing the sortOrder for this ContentList.

getSortOrder

`java.lang.String getSortOrder()`

Gets the sort order set by the #setSortOrder method.

Returns:

The array of sort keys, or null if the setPreferredSortOrder method has not been called for this list.

find

```
ContentEntry find(java.lang.String key,
                  java.lang.Object value)
```

Finds the first `ContentEntry` which identifier for the key '`key`' equals the given object `obj`. For instance, if key == "Title" then obj represents the title, e.g. "Best movie ever" and this method will return the first ContentEntry in the list than contains a match for the (key, value) pair.

Parameters:

`key` - The identifier key.

`value` - The object to compare to

Returns:

The first matched ContentEntry, or null if no match found.

find

```
ContentEntry find(java.lang.String[] keys,
                  java.lang.Object[] values)
```

Finds the first ContentEntry which matches the search. The keys and values parameters are parallel arrays. For example, if keys[0] == "TITLE" and values[0] == "Best movie ever", the implementation SHALL match the first ContentEntry in the list where the metadata contains that (key, value) pair, as well as matches any other entries in the parameter arrays.

Parameters:

`keys` - Array of identifier keys.

`values` - Array of values.

Returns:
The first matching ContentEntry found, or null if no match. If the parameter arrays are not the same length this method returns null.

findAll

```
ContentList findAll(java.lang.String[] keys,
                    java.lang.Object[] values)
```

Finds all ContentEntry objects which match the search. Same as the #find(String[], Object[]) method except all matches are returned instead of just the first match.

Parameters:
`keys` - Array of identifier keys.
`values` - Array of values.

Returns:
A ContentList containing all matches, or null if no matches were found.

filterContentList

```
ContentList filterContentList(ContentDatabaseFilter filter)
                              throws DatabaseException
```

Filters the ContentList. The returned ContentList is a new ContentList only containing ContentItems on which ContentDatabaseFilter.accept returned true.

Parameters:
`filter` - the ContentDatabaseFilter

Returns:
newly created ContentList containing only the filtered ContentItems.

Throws:
`DatabaseException;` - see DatabaseException for exception reasons.
`DatabaseException`

org.ocap.hn.content.navigation

Class DatabaseQuery

```
java.lang.Object
  └org.ocap.hn.content.navigation.ContentDatabaseFilter
      └org.ocap.hn.content.navigation.DatabaseQuery
```

```
public abstract class DatabaseQuery
extends ContentDatabaseFilter
```

DatabaseQuery class. This class represents a query of the metadata database. Methods are provided so that complex AND, OR, NOT expressions can be created.

Note that this object is immutable; the `and()`, `or()` and `negate()` methods do not affect this object but return references to a new query.

For example:

```
 DatabaseQuery qa = DatabaseQuery.newInstance("Genre",
DatabaseQuery.EQUALS, "3.4.11");
 DatabaseQuery qb = DatabaseQuery.newInstance ("SpokenLanguage",
DatabaseQuery.EQUALS,"fr-CA");
 DatabaseQuery qc = qb.and(qa.negate());
```

So the final statement has no effect on the state of `qa` or `qb` objects, which still represent non-negated, root predicates.

Field Summary

static int	**CONTAINS** Operator to check for a String being contained within the field The comparison is case insensitive, non whole word.
static int	**EQUALS** Used to specify a test for equality.
static int	**EXISTS** Operator to check to see if a field exists
static int	**GREATER_THAN** Operator to specify greater than.
static int	**GREATER_THAN_OR_EQUALS** Operator to specify greater than or equals.
static int	**LESS_THAN** Operator to specify less than.
static int	**LESS_THAN_OR_EQUALS** Operator to specify less than or equals.
static int	**NOT_EQUALS** Operator to check for in-equality.

Constructor Summary

DatabaseQuery()

Method Summary

abstract DatabaseQuery	**and**(DatabaseQuery query) Create a new DatabaseQuery based upon the logical AND of the predicates represented by this query and the argument query.
abstract DatabaseQuery	**and**(DatabaseQuery query, java.lang.String contextNode) Create a new DatabaseQuery object based upon the logical AND of the predicates represented by this query and the argument query.
abstract DatabaseQuery	**negate**() Create a new DatabaseQuery, which is the logical NOT of this query.
static DatabaseQuery	**newInstance**(java.lang.String fieldName, int comparison, java.lang.String value) Make a new DatabaseQuery for a specific value in a specific field.
abstract DatabaseQuery	**or**(DatabaseQuery query) Create a new DatabaseQuery based upon the logical OR of the predicates represented by this query and the argument query.

Methods inherited from class org.ocap.hn.content.navigation.ContentDatabaseFilter
accept

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

EQUALS

public static final int **EQUALS**

Used to specify a test for equality. For numbers and dates, an exact match is made. For strings, a case-insensitive comparison is made

See Also:

Constant Field Values

GREATER_THAN

public static final int **GREATER_THAN**

Operator to specify greater than. For numbers the test is "a>b". For Date fields, the test is "a is after b". For strings, the comparison is based on case-insensitive dictionary ordering (i.e. which value occurs first when sorted in alphabetical order), e.g. "dog" > "cat" == true, "Chimp" > "dog" == false

See Also:

Constant Field Values

LESS_THAN

`public static final int LESS_THAN`

Operator to specify less than . For numbers the test is "a<b". For Date fields, the test is "a is before b". For strings, the comparison is based on case-insensitive dictionary ordering (i.e. which one occurs first when sorted in alphabetical order).

See Also:
Constant Field Values

GREATER_THAN_OR_EQUALS

`public static final int GREATER_THAN_OR_EQUALS`

Operator to specify greater than or equals. See rules for GREATER_THAN and EQUALS for how non-numeric fields are compared.

See Also:
Constant Field Values

LESS_THAN_OR_EQUALS

`public static final int LESS_THAN_OR_EQUALS`

Operator to specify less than or equals. See rules for LESS_THAN and EQUALS for how non-numeric fields are compared.

See Also:
Constant Field Values

CONTAINS

`public static final int CONTAINS`

Operator to check for a String being contained within the field The comparison is case insensitive, non whole word.

See Also:
Constant Field Values

NOT_EQUALS

`public static final int NOT_EQUALS`

Operator to check for in-equality. NOT_EQUALS follows the same equality checking rules as EQUALS

See Also:
Constant Field Values

EXISTS

`public static final int EXISTS`

Operator to check to see if a field exists

See Also:
Constant Field Values

Constructor Detail

DatabaseQuery

`public DatabaseQuery()`

Method Detail

newInstance

```
public static DatabaseQuery newInstance(java.lang.String fieldName,
                                        int comparison,
                                        java.lang.String value)
                                 throws DatabaseException
```

Make a new DatabaseQuery for a specific value in a specific field. This is how the root predicates of a query is formed. For example:

```
 DatabaseQuery q1 = DatabaseQuery.newInstance("Title",
DatabaseQuery.CONTAINS, "Foxes");
 DatabaseQuery q2 = DatabaseQuery.newInstance("Genre",
DatabaseQuery.CONTAINS, "3.4.11");
 DatabaseQuery the_query = q1.and(q2);
```

Parameters:
`fieldName` - The name of the field to compare. This field must be the name of known field ID, i.e. FieldID.getInstance().hasKey(fieldName) must return true.
`comparison` - The type of comparison to make (CONTAINS, EXISTS, LESS_THAN, etc.)
`value` - The value to check. For numeric fields, the value must be a string that contains a number. For date fields, the value must be a date in an ISO 8601 compliant format (e.g. of the form "2001-01-05T14:30:00Z" or "2001-01-05T15:30:00+01:00"). For fields that contain an item from a classification scheme (e.g. the "Genre" field), the ID of the controlled term must be used (e.g. "3.4.11"). For comparison==DatabaseQuery.EXISTS, this parameter is not used (it is safe to pass null in this case).
Throws:
`DatabaseException` - if the specified fieldName is unknown, if the comparison is unknown or value is invalid.

and

```
public abstract DatabaseQuery and(DatabaseQuery query)
                           throws DatabaseException
```

Create a new DatabaseQuery based upon the logical AND of the predicates represented by this query and the argument query.
Parameters:
`query` - The second predicate for the AND
Throws:
`DatabaseException` - if the AND of these two queries is known to be invalid

and

```
public abstract DatabaseQuery and(DatabaseQuery query,
                                  java.lang.String contextNode)
                           throws DatabaseException
```

Create a new DatabaseQuery object based upon the logical AND of the predicates represented by this query and the argument query. The resulting predicate will only match within the scope of the specified node. For example:

```
 DatabaseQuery a =
DatabaseQuery.newInstance("Title",DatabaseQuery.CONTAINS,"grave");
 DatabaseQuery b =
DatabaseQuery.newInstance("TitleLanguage",DatabaseQuery.EQUALS,"en");
 DatabaseQuery query = a.and(b,"Title");
```

will cause a search for a title that contains the name "grave" and has an English language title, within the same title node. Without the context node, a valid match would be found for:

```
Dilemme
Grave Serious
Dilemma
```

because there is a title with lang="en" and a title which contains "grave".

Parameters:

`query` - The second predicate for the AND

`contextNode` - The node within which all of the AND must be satisfied. This node must be the name of known field ID, (i.e. FieldID.getInstance().hasKey(fieldName) must return true) and must be a node that is at least the highest level element represented by the two predicates (i.e. the context node is not a child node of either predicate).

Throws:

`DatabaseException` - if the AND of these two queries is known to be invalid, or the contextNode is invalid

or

```
public abstract DatabaseQuery or(DatabaseQuery query)
                          throws DatabaseException
```

Create a new DatabaseQuery based upon the logical OR of the predicates represented by this query and the argument query.

Parameters:

`query` - The second predicate for the OR

Throws:

`DatabaseException` - if the OR of these two queries is known to be invalid

negate

```
public abstract DatabaseQuery negate()
```

Create a new DatabaseQuery, which is the logical NOT of this query.

Returns:

a new DatabaseQuery object that is the logical negative of this object.

org.ocap.hn.content.navigation
Class DeviceFilter

```
java.lang.Object
  └org.ocap.hn.content.navigation.ContentDatabaseFilter
      └org.ocap.hn.content.navigation.DeviceFilter
```

```
public class DeviceFilter
extends ContentDatabaseFilter
```

This class represents a filtering criteria based on a particular device. Applications may use this filter to create a ContentList with content entries from a particular device.

Constructor Summary
DeviceFilter(Device device) Creates a new DeviceFilter.

Method Summary	
boolean	**accept**(ContentEntry entry) Inherited from ContentDatabaseFilter.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

DeviceFilter

`public DeviceFilter(Device device)`

Creates a new DeviceFilter. Only ContentItems which are hosted by the specified Device will pass this filter.

Parameters:

`device` - the device to filter on

Method Detail

accept

`public boolean accept(ContentEntry entry)`

Inherited from ContentDatabaseFilter. This function SHALL return true if the entry passed in is from the device specified in the constructor of this class.

Specified by:

`accept` in class `ContentDatabaseFilter`

Parameters:

`entry` - The entry to test for acceptance

Returns:

`true` if the entry passed in is from the associated device, `false` otherwise.

Annex D UPnP Profiles API

Package org.ocap.hn.profiles.upnp

Interface Summary	
UPnPConstants	This interface contains constants that are specific to UPnP and used in conjunction with the `org.ocap.hn.content.MetadataNode` interface.

org.ocap.hn.profiles.upnp

Interface UPnPConstants

```
public interface UPnPConstants
```

This interface contains constants that are specific to UPnP and used in conjunction with the `org.ocap.hn.content.MetadataNode` interface.

Field Summary

`static java.lang.String`	**`ACTOR`** Name of an actor appearing in a video item.
`static java.lang.String`	**`ACTOR_AT_ROLE`** Role of an actor in the work.
`static java.lang.String`	**`ALBUM`** This identifies a `ALBUM` this piece of content belongs to.
`static java.lang.String`	**`ALBUM_ART`** Reference to album art.
`static java.lang.String`	**`ARTIST`** Name of an artist.
`static java.lang.String`	**`ARTIST_AT_ROLE`** Role of an artist in the work.
`static java.lang.String`	**`ARTIST_DISCOGRAPHY`** Reference to artist discography.
`static java.lang.String`	**`AUTHOR`** Name of an author.
`static java.lang.String`	**`AUTHOR_AT_ROLE`** Role of an author in the work (e.g. lyrics, music).
`static java.lang.String`	**`CHANNEL_NAME`** Used for identification of channels themselves, or information associated with a piece of recorded content.
`static java.lang.String`	**`CHANNEL_NUMBER`** Used for identification of tuner channels themselves or information associated with a piece of recorded content.
`static java.lang.String`	**`COMMENTS`** General-purpose tag where a user can annotate an object with some user-specific information.
`static java.lang.String`	**`CONTRIBUTOR`** Name of a contributor.
`static java.lang.String`	**`CREATION_DATE`** This identifies the `CREATION_DATE` of a piece of content.
`static java.lang.String`	**`CREATOR`** This identifies the `CREATOR` of a piece of content.

Field Summary

static java.lang.String	**DESCRIPTION** A brief description of the content item.
static java.lang.String	**DIRECTOR** Name of a director.
static java.lang.String	**DVD_REGION_CODE** DVD region code.
static java.lang.String	**GENRE** Name of the genre to which an object belongs.
static java.lang.String	**ICON_REF** Reference to an icon which can be used to represent the content.
static java.lang.String	**ID** An identifier for the object.
static java.lang.String	**LANGUAGE** Language as defined by RFC 3066, e.g. "en-US".
static java.lang.String	**LONG_DESCRIPTION** A long description of the content item.
static java.lang.String	**LYRICS_REF** Reference to lyrics of a track or album.
static java.lang.String	**MEDIA_ID** Unique identifier of an audio CD (e.g. freedb or cddb id).
static java.lang.String	**PARENT_ID** An identifier for the parent of this object.
static java.lang.String	**PLAYLIST** Name of a playlist this object belongs to.
static java.lang.String	**PRODUCER** Name of a producer.
static java.lang.String	**PROP_STORAGE_FREE** Property indicating current storage space available on a storage container.
static java.lang.String	**PROP_STORAGE_TOTAL** Property indicating total storage on a storage container.
static java.lang.String	**PUBLISHER** Name of a publisher.
static java.lang.String	**RADIO_BAND** Radio station frequency band.
static java.lang.String	**RADIO_CALL_SIGN** Radio station call sign, e.g. "KSJO".
static java.lang.String	**RADIO_STATION_ID** Some identification, e.g. "107.7", broadcast frequency of the radio station.
static java.lang.String	**RATING** Rating of the object's resource, for 'parental control' filtering purposes, such as "R", "PG-13", "X".

Field Summary

`static java.lang.String`	**`REGION`** Some identification of the region, associated with the 'source' of the object, e.g. "US", "Latin America", "Seattle".
`static java.lang.String`	**`RELATION`** Reference to related resources.
`static java.lang.String`	**`RIGHTS`** Element Description: Information about rights held in and over the resource.
`static java.lang.String`	**`SCHEDULED_END_TIME`** End time of a scheduled program.
`static java.lang.String`	**`SCHEDULED_START_TIME`** Start time of a scheduled program.
`static java.lang.String`	**`STORAGE_MEDIUM`** Indicates the type of storage medium used for the content.
`static java.lang.String`	**`TITLE`** The identifier for the title of an item.
`static java.lang.String`	**`TRACK_NUMBER`** Original track number on a CD or other medium.

Field Detail

ID

`static final java.lang.String` **`ID`**

An identifier for the object. The value of each object id property must be unique with respect to the server hosting this content.

The value is `didl-lite:`(object)`@"id"`

See Also:
Constant Field Values

TITLE

`static final java.lang.String` **`TITLE`**

The identifier for the title of an item. This could be the title of a song, a recording, a photo etc. This identifier is valid for all kinds of content.
Queries for `TITLE` should always return a String.

The value of this key is "dc:title".

See Also:
Constant Field Values

CREATOR

`static final java.lang.String` **`CREATOR`**

This identifies the `CREATOR` of a piece of content. In the case of e.g., MP3s, this maps to the 'Artist' ID3 tag, In case of a recording/live broadcast, this is the Broadcaster e.g., BBC1.
Queries for `CREATOR` should always return a String.

The value of this key is "dc:creator".

See Also:
Constant Field Values

ARTIST

`static final java.lang.String` **`ARTIST`**

Name of an artist.

The value of this field is `"upnp:artist"`.

See Also:
Constant Field Values

ARTIST_AT_ROLE

`static final java.lang.String` **`ARTIST_AT_ROLE`**

Role of an artist in the work.

The value of this field is `"upnp:artist@role"`

See Also:
Constant Field Values

ACTOR

`static final java.lang.String` **`ACTOR`**

Name of an actor appearing in a video item.

The value of this field is `"upnp:actor"`.

See Also:
Constant Field Values

ACTOR_AT_ROLE

`static final java.lang.String` **`ACTOR_AT_ROLE`**

Role of an actor in the work.

The value of this field is `"upnp:actor@role"`

getMetadata returns a String.
See Also:
`MetadataNode.getMetadata(String)`, Constant Field Values

AUTHOR

`static final java.lang.String` **`AUTHOR`**

Name of an author.

The value of this field is `"upnp:author"`.

getMetadata() will return an array of Strings.
See Also:
`MetadataNode.getMetadata(String)`, Constant Field Values

AUTHOR_AT_ROLE

`static final java.lang.String` **`AUTHOR_AT_ROLE`**

Role of an author in the work (e.g. lyrics, music).

The value of this field is `"upnp:author@role"`

getMetadata returns a String.
See Also:
`MetadataNode.getMetadata(String)`, Constant Field Values

PRODUCER

`static final java.lang.String` **`PRODUCER`**

Name of a producer.

The value of this field is `"upnp:producer"`.

getMetadata() will return an array of Strings.
See Also:
`MetadataNode.getMetadata(String)`, Constant Field Values

DIRECTOR

`static final java.lang.String` **`DIRECTOR`**

Name of a director.

The value of this field is `"upnp:director"`.

getMetadata() will return an array of Strings.
See Also:
`MetadataNode.getMetadata(String)`, Constant Field Values

PUBLISHER

`static final java.lang.String` **`PUBLISHER`**

Name of a publisher.

The value of this field is `"dc:publisher"`.

getMetadata() will return an array of Strings.
See Also:
`MetadataNode.getMetadata(String)`, Constant Field Values

CONTRIBUTOR

`static final java.lang.String` **`CONTRIBUTOR`**

Name of a contributor. It is recommended that CONTRIBUTOR includes the name of the primary content creator (see Dublin Core 'creator' property)

The value of this field is `"dc:contributor"`.

getMetadata() will return an array of Strings.
See Also:
`MetadataNode.getMetadata(String)`, Constant Field Values

GENRE

`static final java.lang.String` **`GENRE`**

Name of the genre to which an object belongs. Can be more than one.

The value of this field is `"upnp:genre"`.

getMetadata() will return an array of Strings.
See Also:
`MetadataNode.getMetadata(String)`, Constant Field Values

ALBUM

`static final java.lang.String` **`ALBUM`**

This identifies a `ALBUM` this piece of content belongs to. For example, in MP3 files this maps to the 'Album' ID3 tag, In case of a recording/live broadcast this could be the series to which it belongs (e.g., Buffy).

The value of this field is `"upnp:album"`

getMetadata() will return a String.
See Also:
`MetadataNode.getMetadata(String)`, Constant Field Values

PLAYLIST

`static final java.lang.String` **`PLAYLIST`**

Name of a playlist this object belongs to. Can be more than one.

The value of this field is `"upnp:playlist"`.

getMetadata() will return an array of Strings.
See Also:
`MetadataNode.getMetadata(String)`, Constant Field Values

ALBUM_ART

`static final java.lang.String` **`ALBUM_ART`**

Reference to album art. Can be more than one.

The value of this field is `"upnp:albumArtURI"`.

Values must be properly escaped URIs as described in [RFC 2396].
getMetadata() will return an array of Strings.
See Also:
`MetadataNode.getMetadata(String)`, Constant Field Values

ARTIST_DISCOGRAPHY

`static final java.lang.String` **`ARTIST_DISCOGRAPHY`**

Reference to artist discography.

The value of this field is `"upnp:artistDiscographyURI"`.

Values must be properly escaped URIs as described in [RFC 2396].
getMetadata() will return a String.
See Also:
`MetadataNode.getMetadata(String)`, Constant Field Values

LYRICS_REF

`static final java.lang.String` **`LYRICS_REF`**

Reference to lyrics of a track or album.

The value of this field is `"upnp:lyricsURI"`.

Values must be properly escaped URIs as described in [RFC 2396].
getMetadata() will return an array of Strings.
See Also:
`MetadataNode.getMetadata(String)`, Constant Field Values

RELATION

`static final java.lang.String` **`RELATION`**

Reference to related resources.

The value of this field is `"dc:relation.`

Values must be properly escaped URIs as described in [RFC 2396].
getMetadata() will return an array of Strings.
See Also:
MetadataNode.getMetadata(String), Constant Field Values

STORAGE_MEDIUM

`static final java.lang.String` **`STORAGE_MEDIUM`**

Indicates the type of storage medium used for the content. Potentially useful for user-interface purposes. Allowed values are defined by UPnP and include:

- "UNKNOWN"
- "DV"
- "MINI-DV"
- "VHS"
- "W-VHS"
- "S-VHS"
- "D-VHS"
- "VHSC"
- "VIDEO8"
- "HI8"
- "CD-ROM"
- "CD-DA"
- "CD-R"
- "CD-RW"
- "VIDEO-CD"

- "SACD"
- "MD-AUDIO"
- "MD-PICTURE"
- "DVD-ROM"
- "DVD-VIDEO"
- "DVD-R"
- "DVD+RW"
- "DVD-RW"
- "DVD-RAM"
- "DVD-AUDIO"
- "DAT"
- "LD"
- "HDD"
- "SD"
- "PC-CARD"
- "MMC"
- "CF"
- "BD"
- "MS"

The value of this field is "upnp:storageMedium".

See Also:
Constant Field Values

DESCRIPTION

static final java.lang.String **DESCRIPTION**

A brief description of the content item.

The value of this field is "dc:description".

See Also:
MetadataNode.getMetadata(String), Constant Field Values

LONG_DESCRIPTION

static final java.lang.String **LONG_DESCRIPTION**

A long description of the content item.

The value of this field is "upnp:longDescription".

See Also:
MetadataNode.getMetadata(String), Constant Field Values

ICON_REF

static final java.lang.String **ICON_REF**
Reference to an icon which can be used to represent the content.

The value of this field is "upnp:icon".

Values must be properly escaped URIs as described in [RFC 2396].
See Also:
MetadataNode.getMetadata(String), Constant Field Values

REGION

static final java.lang.String **REGION**
Some identification of the region, associated with the 'source' of the object, e.g. "US", "Latin America", "Seattle".

The value of this field is "upnp:region"

See Also:
MetadataNode.getMetadata(String), Constant Field Values

RATING

static final java.lang.String **RATING**
Rating of the object's resource, for 'parental control' filtering purposes, such as "R", "PG-13", "X".

The value of this field is "upnp:rating"

See Also:
MetadataNode.getMetadata(String), Constant Field Values

RIGHTS

static final java.lang.String **RIGHTS**
Element Description: Information about rights held in and over the resource. Typically a Rights element will contain a rights management statement for the resource, or reference a service providing such information. Rights information often encompasses Intellectual Property Rights (IPR), Copyright, and various Property Rights. If the rights element is absent, no assumptions can be made about the status of these and other rights with respect to the resource. Guidelines for content creation: The Rights element may be used for either a textual statement or a URL pointing to a rights statement, or a combination, when a brief statement and a more lengthy one are available. Examples:
Rights="Access limited to members"
Rights="http://cs-tr.cs.cornell.edu/Dienst/Repository/2.0/Terms"

The value of this field is "dc:rights"

getMetadata() returns an array of Strings.
See Also:
MetadataNode.getMetadata(String), Constant Field Values

CREATION_DATE

static final java.lang.String **CREATION_DATE**
This identifies the CREATION_DATE of a piece of content. In the case of e.g., MP3's this maps to the 'Year' ID3 tag, In case of a recording/live

broadcast this is when the content was created. For Images this is the date the photo was made.
Queries for CREATION_DATE should always return a java.util.Date. Only the year of the Date might actually be valid (e.g., for MP3s).

The value of this field is "dc:date"

See Also:
MetadataNode.getMetadata(String), Constant Field Values

LANGUAGE

static final java.lang.String **LANGUAGE**
Language as defined by RFC 3066, e.g. "en-US".

The value of this field is "dc:language"

getMetadata() will return an array of Strings.
See Also:
MetadataNode.getMetadata(String), Constant Field Values

RADIO_CALL_SIGN

static final java.lang.String **RADIO_CALL_SIGN**
Radio station call sign, e.g. "KSJO".

The value of this field is "upnp:radioCallSign"

See Also:
MetadataNode.getMetadata(String), Constant Field Values

RADIO_STATION_ID

static final java.lang.String **RADIO_STATION_ID**
Some identification, e.g. "107.7", broadcast frequency of the radio station.

The value of this field is "upnp:radioStationID"

See Also:
MetadataNode.getMetadata(String), Constant Field Values

RADIO_BAND

static final java.lang.String **RADIO_BAND**
Radio station frequency band. Recommended values are "AM", "FM", "Shortwave", "Internet", "Satellite". Vendor's may extend this.

The value of this field is "upnp:radioBand"

See Also:
MetadataNode.getMetadata(String), Constant Field Values

CHANNEL_NUMBER

static final java.lang.String **CHANNEL_NUMBER**
Used for identification of tuner channels themselves or information associated with a piece of recorded content.

The value of this field is "upnp:channelNr"

getMetadata() returns an Integer.
See Also:
MetadataNode.getMetadata(String), Constant Field Values

CHANNEL_NAME

static final java.lang.String **CHANNEL_NAME**
Used for identification of channels themselves, or information associated with a piece of recorded content.

The value of this field is "upnp:channelName"

See Also:
MetadataNode.getMetadata(String), Constant Field Values

SCHEDULED_START_TIME

static final java.lang.String **SCHEDULED_START_TIME**
Start time of a scheduled program.

The value of this field is "upnp:scheduledStartTime"

getMetadata() returns java.util.Date.
See Also:
MetadataNode.getMetadata(String), Constant Field Values

SCHEDULED_END_TIME

static final java.lang.String **SCHEDULED_END_TIME**
End time of a scheduled program.

The value of this field is "upnp:scheduledEndTime"

getMetadata() returns java.util.Date.
See Also:
MetadataNode.getMetadata(String), Constant Field Values

DVD_REGION_CODE

static final java.lang.String **DVD_REGION_CODE**
DVD region code.

The value of this field is "upnp:DVDRegionCode"

getMetadata() returns an Integer.
See Also:
MetadataNode.getMetadata(String), Constant Field Values

TRACK_NUMBER

static final java.lang.String **TRACK_NUMBER**
Original track number on a CD or other medium.

The value of this field is "upnp:originalTrackNumber"

getMetadata() returns an Integer.
See Also:
MetadataNode.getMetadata(String), Constant Field Values

MEDIA_ID

static final java.lang.String **MEDIA_ID**
Unique identifier of an audio CD (e.g. freedb or cddb id).

The value of this field is "upnp:toc"

See Also:
MetadataNode.getMetadata(String), Constant Field Values

COMMENTS

static final java.lang.String **COMMENTS**
General-purpose tag where a user can annotate an object with some user-specific information.

The value of this field is "upnp:userAnnotation"

getMetadata() will return an array of Strings.
See Also:
MetadataNode.getMetadata(String), Constant Field Values

PROP_STORAGE_TOTAL

static final java.lang.String **PROP_STORAGE_TOTAL**
Property indicating total storage on a storage container.

The value of this field is "upnp:storageTotal"

getMetadata() will return a Long.
See Also:
MetadataNode.getMetadata(String), Constant Field Values

PROP_STORAGE_FREE

static final java.lang.String **PROP_STORAGE_FREE**
Property indicating current storage space available on a storage container.

The value of this field is "upnp:storageFree"

getMetadata() will return a Long.
See Also:
MetadataNode.getMetadata(String), Constant Field Values

PARENT_ID

static final java.lang.String **PARENT_ID**
An identifier for the parent of this object.

The value is didl-lite:(object)@"parentID"

See Also:
Constant Field Values

Annex E Service API

Package org.ocap.hn.service

A service which is hosted or provided by another device on the home network.

Interface Summary	
HttpRequestResolutionHandler	This interface provides a handler that can be registered with an implementation to provide a mapping service between an incoming HTTP GET or HEAD request from a client and a content binary served by the OC-DMS.
RemoteService	A RemoteService is a service which is hosted or provided by another device on the home network.
ServiceResolutionHandler	This interface provides a handler that can be registered with an implementation to provide a Locator for an otherwise non-tunable ChannelContentItem, such as an SDV channel.

Class Summary	
MediaServerManager	This class provides access to the configuration of the content server.

org.ocap.hn.service

Interface HttpRequestResolutionHandler

```
public interface HttpRequestResolutionHandler
```

This interface provides a handler that can be registered with an implementation to provide a mapping service between an incoming HTTP GET or HEAD request from a client and a content binary served by the OC-DMS.

If a `HttpRequestResolutionHandler` is registered then the implementation SHALL invoke the handler and use mapping provided by the handler. If the handler is not registered or fails to provide a mapping, the implementation attempts to map the request URI itself.

Method Summary

java.net.URL	**resolveHttpRequest**(java.net.InetAddress inetAddress, java.net.URL url, java.lang.String[] request, NetworkInterface networkInterface) Resolves the incoming HTTP request to a URL that identifies a content binary

Method Detail

resolveHttpRequest

```
java.net.URL resolveHttpRequest(java.net.InetAddress inetAddress,
                                java.net.URL url,
                                java.lang.String[] request,
                                NetworkInterface networkInterface)
```

Resolves the incoming HTTP request to a URL that identifies a content binary

Parameters:

`inetAddress` - IP address the transaction was sent from.

`url` - The `URL` requested by the transaction.

`request` - The HTTP message request; i.e., the request line and subsequent message headers.

`networkInterface` - The `NetworkInterface` the request came on. The `getInetAddress` method of this parameter SHALL return the `InetAddress` on which the `request` was received.

Returns:

URL of the content binary if a match is found, null otherwise.

org.ocap.hn.service

Class MediaServerManager

```
java.lang.Object
  └org.ocap.hn.service.MediaServerManager
```

```
public abstract class MediaServerManager
extends java.lang.Object
```

This class provides access to the configuration of the content server. It permits privileged applications to register a handler that maps incoming HTTP requests for content streaming to a content binary.

Constructor Summary

protected	`MediaServerManager()` Protected constructor; not for application use.

Method Summary

abstract int	`getHttpMediaPortNumber()` Gets the port number used in the URL of audio and video content items that are streamed over HTTP.
static MediaServerManager	`getInstance()` Get the MediaServerManager instance
abstract void	`setHttpRequestResolutionHandler (HttpRequestResolutionHandler hrrh)` Registers a HTTP Request resolution handler.

Methods inherited from class java.lang.Object

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

Constructor Detail

MediaServerManager

`protected MediaServerManager()`

Protected constructor; not for application use.

Method Detail

getInstance

`public static MediaServerManager getInstance()`

Get the MediaServerManager instance

setHttpRequestResolutionHandler

```
public abstract void
setHttpRequestResolutionHandler(HttpRequestResolutionHandler hrrh)
```

Registers a HTTP Request resolution handler. If a handler is already registered, this method SHALL replace it. If the `hrrh` parameter is null, any previously registered handler is removed.

Parameters:

`hrrh` - The HttpRequestResolutionHandler to register.

Throws:

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

getHttpMediaPortNumber

```
public abstract int getHttpMediaPortNumber()
```

Gets the port number used in the URL of audio and video content items that are streamed over HTTP.

Returns:

The port number on which the content server is listening for HTTP streaming requests.

org.ocap.hn.service

Interface RemoteService

All Superinterfaces:

javax.tv.service.Service

```
public interface RemoteService
extends javax.tv.service.Service
```

A RemoteService is a service which is hosted or provided by another device on the home network.

Method Summary

ContentItem	**getContentItem**() Returns the ContentItem associated with this remote service

Methods inherited from interface javax.tv.service.Service
equals, getLocator, getName, getServiceType, hashCode, hasMultipleInstances, retrieveDetails

Method Detail

getContentItem

`ContentItem` **`getContentItem`**`()`

Returns the ContentItem associated with this remote service

Returns:

The `ContentItem` associated with this service.

org.ocap.hn.service

Interface ServiceResolutionHandler

```
public interface ServiceResolutionHandler
```

This interface provides a handler that can be registered with an implementation to provide a Locator for an otherwise non-tunable ChannelContentItem, such as an SDV channel. If a `ServiceResolutionHandler` is not registered then the implementation fails any attempts to tune to such channels.

This interface also provides a handler that can be used by the implementation to notify the application that tuning attempts with the application provided locator for a broadcast channel item have failed.

Note: This interface is intended to be used by applications which do not provide a DVB SPI SelectionProvider for the "ocap" scheme.

Method Summary

`boolean`	**`notifyTuneFailed`**`(ChannelContentItem channel)` Notifies the application of a tuning failure for a remote streaming request for a ChannelContentItem.
`boolean`	**`resolveChannelItem`**`(ChannelContentItem channel)` Requests that the application provide tuning parameters for the `ChannelContentItem`.

Method Detail

notifyTuneFailed

`boolean` **`notifyTuneFailed`**`(ChannelContentItem channel)`

Notifies the application of a tuning failure for a remote streaming request for a ChannelContentItem.

Parameters:
`channel` - A ChannelContentItem

Returns:
If the return value is true, the implementation SHALL retry tuning using the current value of the ChannelContentItem's tuning locator; if the return value is false or the tuning locator is null, the implementation SHALL fail the tuning request.

resolveChannelItem

`boolean` **`resolveChannelItem`**`(ChannelContentItem channel)`

Requests that the application provide tuning parameters for the `ChannelContentItem`. When the application is able to resolve the item to a tunable channel, the application calls the `ChannelContentItem.setTuningLocator` method, and this method returns true.

Parameters:
`channel` - A ChannelContentItem

Returns:
true if application resolved the channel item and updated the ChannelContentItem locator, false otherwise

Annex F Recording API

Package org.ocap.hn.recording

Interface Summary	
NetRecordingEntry	This ContentEntry represents a series recording that has been scheduled on the home network.
NetRecordingRequestHandler	A class implementing this interface processes recording requests received from devices on the home network.
NetRecordingRequestManager	This interface represents a local RecordingNetModule.
RecordingContentItem	This ContentItem represents a recording that has been scheduled on the home network.
RecordingNetModule	An interface representing a NetModule which provides DVR functionality.

Class Summary	
NetRecordingSpec	This class represents a network recording specification.

org.ocap.hn.recording

Interface NetRecordingEntry

All Superinterfaces:
ContentEntry

```
public interface NetRecordingEntry
extends ContentEntry
```

This ContentEntry represents a series recording that has been scheduled on the home network.

Field Summary

`static java.lang.String`	**`PROP_CDS_REFERENCE`** Key constant for retrieving the CDS reference of this recording entry from this entry's metadata.
`static java.lang.String`	**`PROP_RCI_LIST`** Key constant for retrieving the RCI list of this recording entry from this entry's metadata.
`static java.lang.String`	**`PROP_SCHEDULED_CDS_ENTRY_ID`** Key constant for retrieving the scheduled CDS entry ID of this recording entry from this entry's metadata.

Method Summary

`void`	**`addRecordingContentItem`**`(RecordingContentItem item)` Adds a local RecordingContentItem to this recording object
`boolean`	**`deleteEntry`**`()` Deletes this NetRecordingEntry if and only if it contains no RecordingContentItems.
`java.lang.String[]`	**`getRecordingContentItemIDs`**`()` Retrieves ObjectIDs of the individual RecordingContentItems that make up this series recording.
`RecordingContentItem[]`	**`getRecordingContentItems`**`()` Retrieves the local individual RecordingContentItems that make up this series recording.
`void`	**`removeRecordingContentItem`**`(RecordingContentItem item)` Removes a local RecordingContentItem from this recording object.

Methods inherited from interface org.ocap.hn.content.ContentEntry
`getContentSize, getCreationDate, getEntryParent, getExtendedFileAccessPermissions, getID, getParentID, getRootMetadataNode, getServer, isLocal`

Field Detail

PROP_CDS_REFERENCE

`static final java.lang.String` **`PROP_CDS_REFERENCE`**

Key constant for retrieving the CDS reference of this recording entry from this entry's metadata. Values returned for this key will be represented as a String.

See Also:

Constant Field Values

PROP_RCI_LIST

`static final java.lang.String` **`PROP_RCI_LIST`**

Key constant for retrieving the RCI list of this recording entry from this entry's metadata. Values returned for this key will be represented as a String.

See Also:

Constant Field Values

PROP_SCHEDULED_CDS_ENTRY_ID

`static final java.lang.String` **`PROP_SCHEDULED_CDS_ENTRY_ID`**

Key constant for retrieving the scheduled CDS entry ID of this recording entry from this entry's metadata. Values returned for this key will be represented as a String.

See Also:

Constant Field Values

Method Detail

deleteEntry

`boolean` **`deleteEntry`**`()`
`throws java.io.IOException`

Deletes this NetRecordingEntry if and only if it contains no RecordingContentItems. Deletes a local NetRecordingEntry only. If the #isLocal method returns false an exception is thrown.

Specified by:

`deleteEntry` in interface `ContentEntry`

Returns:

True if this NetRecordingEntry was deleted, otherwise returns false.

Throws:

`java.lang.SecurityException` - if the application does not have write ExtendedFileAccessPermission.

`java.io.IOException` - if the NetRecordingEntry is not local.

getRecordingContentItems

`RecordingContentItem[]` **`getRecordingContentItems`**`()`
`throws java.io.IOException`

Retrieves the local individual RecordingContentItems that make up this series recording.

Returns:

the RecordingContentItems in this series

Throws:

`java.io.IOException` - if this isLocal() method of this object does not return true

addRecordingContentItem

```
void addRecordingContentItem(RecordingContentItem item)
                             throws java.io.IOException
```

Adds a local RecordingContentItem to this recording object

Parameters:

`item` - The recording content item to add to this series

Throws:

`java.io.IOException` - if this isLocal() method of this object does not return true

`java.lang.IllegalStateException` - if this recording object is not associated with a UPnP AV Scheduled Recording Service Object (RecordSchedule)

`java.lang.IllegalArgumentException` - if the RecordingContentItem parameter has the associated UPnP AV Scheduled Recording Service Object (RecordTask)

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("recordinghandler")

removeRecordingContentItem

```
void removeRecordingContentItem(RecordingContentItem item)
                                throws java.io.IOException
```

Removes a local RecordingContentItem from this recording object. If the RecordingContentItem passed into this method is not contained in this NetRecordingObject, this method has no effect.

Parameters:

`item` - The recording content item to remove from this series

Throws:

`java.io.IOException` - if this isLocal() method of this object does not return true

`java.lang.IllegalArgumentException` - if the RecordingContentItem parameter has the associated UPnP AV Scheduled Recording Service Object (RecordTask)

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("recordinghandler")

getRecordingContentItemIDs

```
java.lang.String[] getRecordingContentItemIDs()
```

Retrieves ObjectIDs of the individual RecordingContentItems that make up this series recording.

Returns:

the ObjectIDs of the RecordingContentItems in this series

org.ocap.hn.recording

Interface NetRecordingRequestHandler

```
public interface NetRecordingRequestHandler
```

A class implementing this interface processes recording requests received from devices on the home network.

An application which has a class implementing this interface may set an instance of it in a local RecordingNetModule. It is up to the application to interpret metadata associated with NetRecordingSpecs and RecordingContentItems delivered in the callback methods, and translate these requests into local DVR recordings.

Method Summary

boolean	**notifyDelete**(java.net.InetAddress address, ContentEntry recording) Notifies this NetRecordingRequestHandler that a device on the home network has requested that metadata associated with a recording be deleted.
boolean	**notifyDeleteService**(java.net.InetAddress address, ContentEntry recording) Notifies this NetRecordingRequestHandler that a device on the home network has requested that content associated with a recorded service be deleted.
boolean	**notifyDisable**(java.net.InetAddress address, ContentEntry recording) Notifies this NetRecordingRequestHandler that a device on the home network has requested that a recording be disabled.
boolean	**notifyPrioritization**(java.net.InetAddress address, NetRecordingEntry[] recordings) Notifies this NetRecordingRequestHandler that a device on the home network has requested that a group of recordings be re-prioritized.
boolean	**notifyPrioritization**(java.net.InetAddress address, RecordingContentItem[] recordings) Notifies this NetRecordingRequestHandler that a device on the home network has requested that a group of individual recordings be re-prioritized.
boolean	**notifyReschedule**(java.net.InetAddress address, ContentEntry recording, NetRecordingEntry spec) Notifies this NetRecordingRequestHandler that a device on the home network has requested that a recording be rescheduled.
boolean	**notifySchedule**(java.net.InetAddress address, NetRecordingEntry spec) Notifies this NetRecordingRequestHandler that a device on the home network has requested that a recording be scheduled.

Method Detail

notifySchedule

```
boolean notifySchedule(java.net.InetAddress address,
                       NetRecordingEntry spec)
```

Notifies this NetRecordingRequestHandler that a device on the home network has requested that a recording be scheduled. Handler applications MAY inspect any metadata associated with the NetRecordingEntry passed with the method invocation, and translate such metadata in one or more local

DVR recordings. Applications SHOULD associate such recordings with the NetRecordingEntry by adding the recordings to the entry using the *NetRecordingEntry.addRecordingContentItem()* method.

Parameters:

`address` - IP address of the device on the home network which has issues this request

`spec` - the NetRecordingEntry which describes the requested recording

Returns:

true if the schedule request can be successfully processed, or false if the request will not be processed.

See Also:

`NetRecordingEntry.addRecordingContentItem(RecordingContentItem)`

notifyReschedule

```
boolean notifyReschedule(java.net.InetAddress address,
                         ContentEntry recording,
                         NetRecordingEntry spec)
```

Notifies this NetRecordingRequestHandler that a device on the home network has requested that a recording be rescheduled. Handler applications MAY inspect any metadata contained within the NetRecordingEntry passed into this method, and utilize such metadata to reschedule the local DVR recording represented by the given ContentEntry. This ContentEntry may represent an individual recording as a RecordingContentItem, or may represent a collection of recordings contained within a NetRecordingEntry object.

Parameters:

`address` - the IP address of the device on the home network which has issues this request

`recording` - the RecordingContentItem or NetRecordingEntry to be rescheduled

`spec` - the NetRecordingEntry object containing the metadata to be used to reschedule the recording

Returns:

true if the reschedule request can be successfully processed, or false if the request will not be processed.

notifyDisable

```
boolean notifyDisable(java.net.InetAddress address,
                      ContentEntry recording)
```

Notifies this NetRecordingRequestHandler that a device on the home network has requested that a recording be disabled. If the recording is in progress, this is a request to stop the recording. If the recording is pending, this is a request to cancel the recording. Applications MAY cancel or stop the given recording in response to this request.

Parameters:

`address` - the IP address of the device on the home network which has issues this request

`recording` - the RecordingContentItem or RecordingNetEntry to be disabled

Returns:

true if the disable request can be successfully processed, or false if the request will not be processed.

notifyDelete

```
boolean notifyDelete(java.net.InetAddress address,
                     ContentEntry recording)
```

Notifies this NetRecordingRequestHandler that a device on the home network has requested that metadata associated with a recording be deleted. Applications MAY delete the given recording in response to this request.

Parameters:

`address` - the IP address of the device on the home network which has issues this request

`recording` - the RecordingContentItem or NetRecordingEntry to be deleted

Returns:

true if the delete request can be successfully processed, or false if the request will not be processed.

notifyDeleteService

```
boolean notifyDeleteService(java.net.InetAddress address,
                            ContentEntry recording)
```

Notifies this NetRecordingRequestHandler that a device on the home network has requested that content associated with a recorded service be deleted. Applications MAY delete the content associated with the given recording in response to this request.

Parameters:

`address` - the IP address of the device on the home network which has issues this request

`recording` - requested the RecordingContentItem or NetRecordingEntry

Returns:

true if the delete request can be successfully processed, or false if the request will not be processed.

notifyPrioritization

```
boolean notifyPrioritization(java.net.InetAddress address,
                             NetRecordingEntry[] recordings)
```

Notifies this NetRecordingRequestHandler that a device on the home network has requested that a group of recordings be re-prioritized. The requested prioritization is represented by the ordering of the NetRecordingEntry objects in the given array, with the highest priority at index 0 of the array. Applications MAY prioritize some or all of the local DVR recordings contained within the NetRecordingEntry array.

Parameters:

`address` - the IP address of the device on the home network which has issues this request

`recordings` - the NetRecordingEntrys to be prioritized

Returns:

true if the prioritization request can be successfully processed, or false if the request will not be processed.

notifyPrioritization

```
boolean notifyPrioritization(java.net.InetAddress address,
                             RecordingContentItem[] recordings)
```

Notifies this NetRecordingRequestHandler that a device on the home network has requested that a group of individual recordings be re-prioritized. The requested prioritization is represented by the ordering of the RecordingContentItem objects in the given array, with the highest priority at index 0 of the array. Applications MAY prioritize the local DVR recordings contained within the RecordingContentItem array.

Parameters:

`address` - IP address of the device on the home network which has issued this request.

`recordings` - The recording content items associated with the recordings to be prioritized.

Returns:

True if the prioritization request can be successfully processed, or false if the request will not be processed.

org.ocap.hn.recording

Interface NetRecordingRequestManager

All Superinterfaces:

NetModule, RecordingNetModule

```
public interface NetRecordingRequestManager
extends RecordingNetModule
```

This interface represents a local RecordingNetModule. An instance of RecordingNetModule for which the isLocal() method returns true will also be an instance of NetRecordingRequestManager.

Field Summary

Fields inherited from interface org.ocap.hn.NetModule
CONTENT_LIST, CONTENT_MANAGER, CONTENT_RECORDER, CONTENT_RENDERER, CONTENT_SERVER, PROP_CONTROL_URL, PROP_DESCRIPTION_URL, PROP_EventSub_URL, PROP_NETMODULE_ID, PROP_NETMODULE_TYPE

Method Summary

NetRecordingEntry	**createNetRecordingEntry**() This method creates a local entry which represents a network visible collection of recording items.
void	**setNetRecordingRequestHandler**(NetRecordingRequestHandler handler) This method sets the specified NetRecordingRequestHandler that processes requests to schedule recordings from remote devices.

Methods inherited from interface org.ocap.hn.recording.RecordingNetModule
requestDelete, requestDeleteService, requestDisable, requestPrioritize, requestPrioritize, requestReschedule, requestSchedule

Methods inherited from interface org.ocap.hn.NetModule
addNetModuleEventListener, getDevice, getKeys, getNetModuleId, getNetModuleType, getProperty, isLocal, removeNetModuleEventListener

Method Detail

createNetRecordingEntry

```
NetRecordingEntry createNetRecordingEntry()
                                          throws java.io.IOException
```

This method creates a local entry which represents a network visible collection of recording items.

Throws:

`java.io.IOException` - if the isLocal() method of this object does not return true

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("recordinghandler")

setNetRecordingRequestHandler

`void` **`setNetRecordingRequestHandler`**`(NetRecordingRequestHandler handler)`

This method sets the specified NetRecordingRequestHandler that processes requests to schedule recordings from remote devices. Only one instance of NetRecordingRequestHandler can be set on a given RecordingNetModule at a time. A NetRecordingRequestHandler can only be set on an instance of RecordingNetModule that is local to the device.

Parameters:

`handler` - the handler to be set for this RecordingNetModule. If null is specified, the currently set handler will be removed, and no application notification will occur for recording requests.

Throws:

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("recordinghandler")

org.ocap.hn.recording

Class NetRecordingSpec

```
java.lang.Object
  └org.ocap.hn.recording.NetRecordingSpec
```

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

This class represents a network recording specification. NetRecordingSpec object may be used to request recordings be scheduled on remote devices. Metadata contained within this object can be used to schedule or modify recordings on the home network.

Constructor Summary

NetRecordingSpec() Default constructor for the NetRecordingSpec.
NetRecordingSpec(MetadataNode metadata) Metadata constructor for the NetRecordingSpec.

Method Summary

MetadataNode	**getMetadata**() Retrieves the root metadata node for this recording spec.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

NetRecordingSpec

public **NetRecordingSpec**()

Default constructor for the NetRecordingSpec. Upon creation, NetRecordingSpecs will contain a single empty metadata node.

NetRecordingSpec

public **NetRecordingSpec**(MetadataNode metadata)

Metadata constructor for the NetRecordingSpec. Applications can use this form of the constructor to specify the root metadata node at time of construction.

Parameters:

metadata - root metadata node for the NetRecordingSpec

Method Detail

getMetadata

`public MetadataNode **getMetadata**()`

Retrieves the root metadata node for this recording spec. Metadata added to this recording spec will be utilized to identify recording requests on remote devices.

Returns:

The root MetadataNode for this NetRecordingSpec

org.ocap.hn.recording

Interface RecordingContentItem

All Superinterfaces:
ContentEntry, ContentItem

```
public interface RecordingContentItem
extends ContentItem
```

This ContentItem represents a recording that has been scheduled on the home network. This interface represents a DVR recording which can be published to the home network. On devices which support both the OCAP Home Networking API and the OCAP DVR API, objects implementing `org.ocap.dvr.OcapRecordingRequest` will also implement this interface. When a RecordingRequest is deleted, implementations SHALL call the RecordingContentItem.deleteEntry method in the same object.

Field Summary

`static java.lang.String`	**PROP_ACCESS_PERMISSIONS** Key constant for retrieving the file access permissions of this recording item from this item's metadata.
`static java.lang.String`	**PROP_APP_ID** Key constant for retrieving the application ID of this recording item from this item's metadata.
`static java.lang.String`	**PROP_CONTENT_URI** Key constant for retrieving the location of content associated with this recording item from this item's metadata.
`static java.lang.String`	**PROP_DESTINATION** Key constant for retrieving the destination of this recording item from this item's metadata.
`static java.lang.String`	**PROP_DURATION** Key constant for retrieving the duration in milliseconds of this recording item from this item's metadata.
`static java.lang.String`	**PROP_EXPIRATION_PERIOD** Key constant for retrieving the expiration period for this recording item from this item's metadata.
`static java.lang.String`	**PROP_MEDIA_FIRST_TIME** Key constant for retrieving the media first time for this recording item from this item's metadata.
`static java.lang.String`	**PROP_MSO_CONTENT** Key constant for retrieving the MSO content indicator for this recording item from this item's metadata.
`static java.lang.String`	**PROP_NET_RECORDING_ENTRY** Key constant for retrieving the ID of any NetRecordingEntry containing this RecordingContentItem.
`static java.lang.String`	**PROP_ORGANIZATION** Key constant for retrieving the organization of this recording item from this item's metadata.

Field Summary

static java.lang.String	**PROP_PRESENTATION_POINT** Key constant for retrieving the presentation point for this recording item from this item's metadata.
static java.lang.String	**PROP_PRIORITY_FLAG** Key constant for retrieving the priority flag of this recording item from this item's metadata.
static java.lang.String	**PROP_RECORDING_STATE** Key constant for retrieving the state of this recording item from this item's metadata.
static java.lang.String	**PROP_RETENTION_PRIORITY** Key constant for retrieving the retention priority of this recording item from this item's metadata.
static java.lang.String	**PROP_SOURCE_ID** Key constant for retrieving the source ID of this recording item from this item's metadata.
static java.lang.String	**PROP_SOURCE_ID_TYPE** Key constant for retrieving the source ID type of this recording item from this item's metadata.
static java.lang.String	**PROP_SPACE_REQUIRED** Key constant for retrieving the estimated space required for this recording item from this item's metadata.
static java.lang.String	**PROP_START_TIME** Key constant for retrieving the start time of this recording item from this item's metadata.

Fields inherited from interface org.ocap.hn.content.ContentItem
AUDIO_ITEM, AUDIO_ITEM_BOOK, AUDIO_ITEM_BROADCAST, AUDIO_ITEM_TRACK, IMAGE_ITEM, IMAGE_ITEM_PHOTO, ITEM, VIDEO_ITEM, VIDEO_ITEM_BROADCAST, VIDEO_ITEM_MOVIE, VIDEO_ITEM_MUSIC_CLIP

Method Summary

boolean	**deleteEntry**() Deletes this RecordingContentItem, but does not remove the physical recording.
NetRecordingEntry	**getRecordingEntry**() Returns the NetRecordingEntry which contains this recording content item if the NetRecordingEntry is available.
java.lang.String	**getRecordingEntryID**() Returns the ObjectID of the NetRecordingEntry which contains this recording content item.
NetActionRequest	**requestConflictingRecordings**(NetActionHandler handler) Requests a list of recordings whose usage of resources conflict with this recording content item.

Method Summary

NetActionRequest	**requestSetMediaTime**(javax.media.Time time, NetActionHandler handler) Requests that the presentation point of this recording be updated.

Methods inherited from interface org.ocap.hn.content.ContentItem
containsResource, getContentClass, getItemService, getRenderableResources, getResource, getResourceCount, getResourceIndex, getResources, getTitle, hasAudio, hasStillImage, hasVideo, isRenderable

Methods inherited from interface org.ocap.hn.content.ContentEntry
getContentSize, getCreationDate, getEntryParent, getExtendedFileAccessPermissions, getID, getParentID, getRootMetadataNode, getServer, isLocal

Field Detail

PROP_RECORDING_STATE

`static final java.lang.String` **`PROP_RECORDING_STATE`**

Key constant for retrieving the state of this recording item from this item's metadata. Values returned for this key will be represented as an Integer.

See Also:
Constant Field Values

PROP_START_TIME

`static final java.lang.String` **`PROP_START_TIME`**

Key constant for retrieving the start time of this recording item from this item's metadata. Values returned for this key will be represented as a java.util.Date.

See Also:
Constant Field Values

PROP_DURATION

`static final java.lang.String` **`PROP_DURATION`**

Key constant for retrieving the duration in milliseconds of this recording item from this item's metadata. Values returned for this key will be represented as an Integer.

See Also:
Constant Field Values

PROP_SOURCE_ID

`static final java.lang.String` **`PROP_SOURCE_ID`**

Key constant for retrieving the source ID of this recording item from this item's metadata. Values returned for this key will be represented as a String.

See Also:
Constant Field Values

PROP_SOURCE_ID_TYPE

`static final java.lang.String` **`PROP_SOURCE_ID_TYPE`**

Key constant for retrieving the source ID type of this recording item from this item's metadata. Values returned for this key will be represented as a String.

See Also:
Constant Field Values

PROP_DESTINATION

`static final java.lang.String` **`PROP_DESTINATION`**

Key constant for retrieving the destination of this recording item from this item's metadata. Values returned for this key will be represented as a String.

See Also:
Constant Field Values

PROP_PRIORITY_FLAG

`static final java.lang.String` **`PROP_PRIORITY_FLAG`**

Key constant for retrieving the priority flag of this recording item from this item's metadata. Values returned for this key will be represented as an Integer.

See Also:
Constant Field Values

PROP_RETENTION_PRIORITY

`static final java.lang.String` **`PROP_RETENTION_PRIORITY`**

Key constant for retrieving the retention priority of this recording item from this item's metadata. Values returned for this key will be represented as an Integer.

See Also:
Constant Field Values

PROP_ACCESS_PERMISSIONS

`static final java.lang.String` **`PROP_ACCESS_PERMISSIONS`**

Key constant for retrieving the file access permissions of this recording item from this item's metadata. Values returned for this key will be represented as an org.ocap.storage.ExtendedFileAccessPermissions.

See Also:
Constant Field Values

PROP_ORGANIZATION

`static final java.lang.String` **`PROP_ORGANIZATION`**

Key constant for retrieving the organization of this recording item from this item's metadata. Values returned for this key will be represented as a String.

See Also:
Constant Field Values

PROP_APP_ID

`static final java.lang.String` **`PROP_APP_ID`**

Key constant for retrieving the application ID of this recording item from this item's metadata. Values returned for this key will be represented as an org.dvb.application.AppID.

See Also:
Constant Field Values

PROP_SPACE_REQUIRED

`static final java.lang.String` **`PROP_SPACE_REQUIRED`**

Key constant for retrieving the estimated space required for this recording item from this item's metadata. Values returned for this key will be represented as a Long.

See Also:
Constant Field Values

PROP_CONTENT_URI

`static final java.lang.String` **`PROP_CONTENT_URI`**

Key constant for retrieving the location of content associated with this recording item from this item's metadata. Values returned for this key will be represented as a String.

See Also:
Constant Field Values

PROP_MEDIA_FIRST_TIME

`static final java.lang.String` **`PROP_MEDIA_FIRST_TIME`**

Key constant for retrieving the media first time for this recording item from this item's metadata. Values returned for this key will be represented as a Long.

See Also:
Constant Field Values

PROP_PRESENTATION_POINT

`static final java.lang.String` **`PROP_PRESENTATION_POINT`**

Key constant for retrieving the presentation point for this recording item from this item's metadata. Values returned for this key will be represented as a Long.

See Also:
Constant Field Values

PROP_EXPIRATION_PERIOD

`static final java.lang.String` **`PROP_EXPIRATION_PERIOD`**

Key constant for retrieving the expiration period for this recording item from this item's metadata. Values returned for this key will be represented as an Long.

See Also:
Constant Field Values

PROP_MSO_CONTENT

`static final java.lang.String` **`PROP_MSO_CONTENT`**

Key constant for retrieving the MSO content indicator for this recording item from this item's metadata. Values returned for this key will be represented as an Boolean.

See Also:
Constant Field Values

PROP_NET_RECORDING_ENTRY

`static final java.lang.String` **`PROP_NET_RECORDING_ENTRY`**

Key constant for retrieving the ID of any NetRecordingEntry containing this RecordingContentItem. Values returned for this key will be represented as a String.

See Also:
Constant Field Values

Method Detail

deleteEntry

```
boolean deleteEntry()
                    throws java.io.IOException
```

Deletes this RecordingContentItem, but does not remove the physical recording. Deletes a local RecordingContentItem only. If the #isLocal method returns false an exception is thrown.

Note: this overrides the definition of `ContentItem.deleteEntry()`. If an application calls the ContentEntry.deleteEntry method on an object that is an instance of RecordingContentItem, the implementation SHALL delete the RecordingContentItem as defined by this method.

Specified by:
`deleteEntry` in interface `ContentEntry`
Specified by:
`deleteEntry` in interface `ContentItem`
Returns:
True if this RecordingContentItem was deleted, otherwise returns false.
Throws:
`java.lang.SecurityException` - if the application does not have write ExtendedFileAccessPermission.
`java.io.IOException` - if the RecordingContentItem is not local.

requestSetMediaTime

```
NetActionRequest requestSetMediaTime(javax.media.Time time,
                                     NetActionHandler handler)
```

Requests that the presentation point of this recording be updated.
Parameters:
`time` - The presentation point of this recording.
`handler` - The NetActionHandler which gets informed once this request completes.
Returns:
NetActionRequest which can be used to monitor asynchronous action progress

requestConflictingRecordings

```
NetActionRequest requestConflictingRecordings(NetActionHandler handler)
```

Requests a list of recordings whose usage of resources conflict with this recording content item. The resulting list of recordings SHALL be returned as an array of RecordingContentItem objects from the NetActionEvent.getResponse() method of the resulting NetActionEvent.
Parameters:
`handler` - The NetActionHandler implementation to receive the asynchronous response to this request
Returns:
NetActionRequest which can be used to monitor asynchronous action progress

getRecordingEntry

```
NetRecordingEntry getRecordingEntry()
```

Returns the NetRecordingEntry which contains this recording content item if the NetRecordingEntry is available.

Returns:
null if this RecordingContentItem is not added to any NetRecordingEntry or if the NetRecordingEntry containing this RecordingContentItem is not available. Otherwise the NetRecordingEntry containing this RecordingContentItem

getRecordingEntryID

`java.lang.String` **`getRecordingEntryID`**`()`

Returns the ObjectID of the NetRecordingEntry which contains this recording content item. The ObjectID can be obtained from ocap:netRecordingEntry property of this recording content item.

Returns:
null if this RecordingContentItem does not contain ocap:netRecordingEntry property. Otherwise, the value contained in ocap:netRecordingEntry property of this RecordingContentItem.

org.ocap.hn.recording

Interface RecordingNetModule

All Superinterfaces:
NetModule
All Known Subinterfaces:
NetRecordingRequestManager

```
public interface RecordingNetModule
extends NetModule
```

An interface representing a NetModule which provides DVR functionality.

NetModules which implement this interface SHALL have a NetModule.PROP_NETMODULE_TYPE property value of NetModule.CONTENT_RECORDER.

Field Summary

Fields inherited from interface org.ocap.hn.NetModule
CONTENT_LIST, CONTENT_MANAGER, CONTENT_RECORDER, CONTENT_RENDERER, CONTENT_SERVER, PROP_CONTROL_URL, PROP_DESCRIPTION_URL, PROP_EventSub_URL, PROP_NETMODULE_ID, PROP_NETMODULE_TYPE

Method Summary

NetActionRequest	**requestDelete**(ContentEntry recording, NetActionHandler handler) Requests that metadata associated with a scheduled recording be deleted from storage.
NetActionRequest	**requestDeleteService**(ContentEntry recording, NetActionHandler handler) Requests that content associated with a scheduled recording be deleted from storage.
NetActionRequest	**requestDisable**(ContentEntry recording, NetActionHandler handler) Requests that an in progress recording be disabled on this network recording device.
NetActionRequest	**requestPrioritize**(NetRecordingEntry[] recordings, NetActionHandler handler) Requests that a group of scheduled recording request be prioritized on this network recording device, where each recording request may represent one or more individual recordings on the remote device.
NetActionRequest	**requestPrioritize**(RecordingContentItem[] recordings, NetActionHandler handler) Requests that a group of scheduled individual recordings be prioritized on this network recording device.

Method Summary

NetActionRequest	**requestReschedule**(ContentEntry recording, NetRecordingSpec recordingSpec, NetActionHandler handler) Requests that a recording be rescheduled on this network recording device.
NetActionRequest	**requestSchedule**(NetRecordingSpec recordingSpec, NetActionHandler handler) Requests that a recording be scheduled on this network recording device.

Methods inherited from interface org.ocap.hn.NetModule
addNetModuleEventListener, getDevice, getKeys, getNetModuleId, getNetModuleType, getProperty, isLocal, removeNetModuleEventListener

Method Detail

requestSchedule

```
NetActionRequest requestSchedule(NetRecordingSpec recordingSpec,
                                 NetActionHandler handler)
```

Requests that a recording be scheduled on this network recording device. metadata added to the NetRecordingSpec prior to calling this method will be utilized by the remote device in identifying the recording or recordings to be scheduled. Upon completion of this operation, a NetActionEvent SHALL be delivered to the given handler indicating success or failure. Upon success, values returned by calls to `NetActionEvent.getResponse()` SHALL contain a NetRecordingEntry representing the newly created recording.

Parameters:

`recordingSpec` - a recording spec containing the metadata used to identify the recordings to be scheduled.

`handler` - The NetActionHandler which gets informed once this request completes.

Returns:

NetActionRequest to inform calling application of results.

Throws:

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("recording")

`java.lang.IllegalArgumentException` - if recordingSpec has an empty MetadataNode, or if MetadataNode which is associated with recordingSpec does not contain the necessary metadata entry such as scheduledChannelID, scheduledStartDateTime, scheduledDuration

requestReschedule

```
NetActionRequest requestReschedule(ContentEntry recording,
                                   NetRecordingSpec recordingSpec,
                                   NetActionHandler handler)
```

Requests that a recording be rescheduled on this network recording device. Metadata added to the NetRecordingSpec prior to calling this method will be utilized by the remote device in identifying changes the recording or recordings to be rescheduled. Upon completion of this operation, a NetActionEvent SHALL be delivered to the given handler indicating success or failure.

Parameters:

`recording` - the previously scheduled RecordingContentItem or NetRecordingEntry to be rescheduled.

`recordingSpec` - a recording spec containing the metadata used to identify the changes to recordings to be rescheduled.

`handler` - The NetActionHandler which gets informed once this request completes.

Returns:

NetActionRequest to inform calling application of results.

Throws:

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("recording")

`java.lang.IllegalArgumentException` - if the recording parameter is neither the NetRecordingEntry with upnp:srsRecordScheduleID metadata entry nor the RecordingContentItem with upnp:srsRecordTaskID metadata entry in its own MetadataNode, or if recordingSpec has an empty MetadataNode, or if MetadataNode which is associated with recordingSpec does not contain the necessary metadata entry such as scheduledChannelID, scheduledStartDateTime, scheduledDuration

requestDisable

```
NetActionRequest requestDisable(ContentEntry recording,
                                NetActionHandler handler)
```

Requests that an in progress recording be disabled on this network recording device. If the recording is in progress, this method requests that the recording be stopped. If the recording is pending, this method requests that the recording be canceled. Upon completion of this operation, a NetActionEvent SHALL be delivered to the given handler indicating success or failure.

Parameters:

`recording` - a RecordingContentItem or NetRecordingEntry that identifies the recording(s) to be canceled.

`handler` - The NetActionHandler which gets informed once this request completes.

Returns:

NetActionRequest to inform calling application of results.

Throws:

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("recording")

`java.lang.IllegalArgumentException` - if the recording parameter is neither the NetRecordingEntry with upnp:srsRecordScheduleID metadata entry nor the RecordingContentItem with upnp:srsRecordTaskID metadata entry in its own MetadataNode

requestDeleteService

```
NetActionRequest requestDeleteService(ContentEntry recording,
                                      NetActionHandler handler)
```

Requests that content associated with a scheduled recording be deleted from storage. Upon completion of this operation, a NetActionEvent SHALL be delivered to the given handler indicating success or failure.

Parameters:

`recording` - a RecordingContentItem or NetRecordingEntry that identifies the recording(s) to be deleted.

`handler` - The NetActionHandler which gets informed once this request completes.

Returns:

NetActionRequest to inform calling application of results.

Throws:

`java.lang.SecurityException` - if the caller does not have HomeNetPermission("recording")

requestDelete

```
NetActionRequest requestDelete(ContentEntry recording,
                               NetActionHandler handler)
```

Requests that metadata associated with a scheduled recording be deleted from storage. Upon completion of this operation, a NetActionEvent SHALL be delivered to the given handler indicating success or failure.

Parameters:

`recording` - a recording that identifies the recording to be deleted.

`handler` - The NetActionHandler which gets informed once this request completes.

Returns:

NetActionRequest to inform calling application of results.

Throws:
`java.lang.SecurityException` - if the caller does not have HomeNetPermission("recording")

requestPrioritize

```
NetActionRequest requestPrioritize(RecordingContentItem[] recordings,
                                   NetActionHandler handler)
```

Requests that a group of scheduled individual recordings be prioritized on this network recording device. Prioritization is determined by the ordering of recordings in the array of RecordingContentItems, with highest priority given to the entry at element 0 in the array. Upon completion of this operation, a NetActionEvent SHALL be delivered to the given handler indicating success or failure.

Parameters:
`recordings` - a prioritized array of RecordingContentItems
`handler` - The NetActionHandler which gets informed once this request completes.

Returns:
NetActionRequest to inform calling application of results.

Throws:
`java.lang.SecurityException` - if the caller does not have HomeNetPermission("recording")

requestPrioritize

```
NetActionRequest requestPrioritize(NetRecordingEntry[] recordings,
                                   NetActionHandler handler)
```

Requests that a group of scheduled recording request be prioritized on this network recording device, where each recording request may represent one or more individual recordings on the remote device. Prioritization is determined by the ordering of recordings in the array of NetRecordingEntries with highest priority given to the entry at element 0 in the array. Upon completion of this operation, a NetActionEvent SHALL be delivered to the given handler indicating success or failure.

Parameters:
`recordings` - a prioritized array of NetRecordingEntries
`handler` - The NetActionHandler which gets informed once this request completes.

Returns:
NetActionRequest to inform calling application of results.

Throws:
`java.lang.SecurityException` - if the caller does not have HomeNetPermission("recording")

Annex G Security API

Package org.ocap.hn.security

Interface Summary	
NetAuthorizationHandler	This interface represents a callback mechanism to a registered network authorization handler.
NetAuthorizationHandler2	This interface represents a callback mechanism to a registered network authorization handler.

Class Summary	
NetSecurityManager	This class provides access to home network security capabilities including password handling.

org.ocap.hn.security

Interface NetAuthorizationHandler

```
public interface NetAuthorizationHandler
```

This interface represents a callback mechanism to a registered network authorization handler.

Method Summary

boolean	**notifyAction**(java.lang.String actionName, java.net.InetAddress inetAddress, java.lang.String macAddress, int activityID) Notifies the authorization handler that an action it registered interest in has been received.
void	**notifyActivityEnd**(int activityID) Notifies the registered authorization handler that an activity has ended.
boolean	**notifyActivityStart**(java.net.InetAddress inetAddress, java.lang.String macAddress, java.net.URL url, int activityID) Notifies the registered authorization handler that an activity to access cable services has been started.

Method Detail

notifyActivityStart

```
boolean notifyActivityStart(java.net.InetAddress inetAddress,
                            java.lang.String macAddress,
                            java.net.URL url,
                            int activityID)
```

Notifies the registered authorization handler that an activity to access cable services has been started. The handler will permit or deny the ability for the activity to continue.

Parameters:

`inetAddress` - IP address the transaction was sent from.

`macAddress` - An empty string; this parameter is not used.

`url` - The `URL` requested by the transaction.

`activityID` - The unique identifier of the activity.

Returns:

true if the activity is accepted; false otherwise.

notifyActivityEnd

```
void notifyActivityEnd(int activityID)
```

Notifies the registered authorization handler that an activity has ended.

Parameters:

`activityID` - Unique identifier assigned to the activity and passed to the `notifyActivityStart` method.

notifyAction

```
boolean notifyAction(java.lang.String actionName,
                     java.net.InetAddress inetAddress,
```

```
                java.lang.String macAddress,
                int activityID)
```

Notifies the authorization handler that an action it registered interest in has been received.

Parameters:

`actionName` - Name of the action received. Will match a name in the `actionNames` parameter previously passed to `NetSecurityManager.setAuthorizationHandler(NetAuthorizationHandler, String[], boolean)`.

`inetAddress` - IP address the transaction was sent from.

`macAddress` - An empty string; this parameter is not used.

`activityID` - The unique identifier of the activity if known. If no activityID is associated with the transaction the implementation SHALL pass a value of -1;

Returns:

True if the activity is accepted, otherwise returns false.

org.ocap.hn.security

Interface NetAuthorizationHandler2

```
public interface NetAuthorizationHandler2
```

This interface represents a callback mechanism to a registered network authorization handler.

Method Summary

boolean	**notifyAction**(java.lang.String actionName, java.net.InetAddress inetAddress, int activityID, java.lang.String[] request, NetworkInterface networkInterface) Notifies the authorization handler that an action it registered interest in has been received.
void	**notifyActivityEnd**(int activityID, int resultCode) Notifies the registered authorization handler that an activity has ended.
boolean	**notifyActivityStart**(java.net.InetAddress inetAddress, java.net.URL url, int activityID, ContentEntry entry, java.lang.String[] request, NetworkInterface networkInterface) Notifies the registered authorization handler that an activity to access cable services has been started.

Method Detail

notifyActivityStart

```
boolean notifyActivityStart(java.net.InetAddress inetAddress,
                            java.net.URL url,
                            int activityID,
                            ContentEntry entry,
                            java.lang.String[] request,
                            NetworkInterface networkInterface)
```

Notifies the registered authorization handler that an activity to access cable services has been started. The handler will permit or deny the ability for the activity to continue.

Parameters:

`inetAddress` - IP address the transaction was sent from.

`url` - The `URL` requested by the transaction.

`activityID` - The unique identifier of the activity.

`entry` - The `ContentEntry` corresponding to the `url` parameter. If no entry can be uniquely matched then null is passed in.

`request` - The HTTP message request; i.e., the request line and subsequent message headers.

`networkInterface` - The `NetworkInterface` the session identified by `activityID` took place on. The `getInetAddress` method of this parameter SHALL return the `InetAddress` on which the `request` was received.

Returns:

true if the activity is accepted; false otherwise.

notifyActivityEnd

```
void notifyActivityEnd(int activityID,
                       int resultCode)
```

Notifies the registered authorization handler that an activity has ended.

Parameters:

`activityID` - Unique identifier assigned to the activity and passed to the `notifyActivityStart` method.

`resultCode` - Value indicating success or failure.

notifyAction

```
boolean notifyAction(java.lang.String actionName,
                     java.net.InetAddress inetAddress,
                     int activityID,
                     java.lang.String[] request,
                     NetworkInterface networkInterface)
```

Notifies the authorization handler that an action it registered interest in has been received.

Parameters:

`actionName` - Name of the action received. Will match a name in the `actionNames` parameter previously passed to `NetSecurityManager.setAuthorizationHandler(NetAuthorizationHandler2, String[], boolean)`.

`inetAddress` - IP address the transaction was sent from.

`activityID` - The unique identifier of the activity if known. If no activityID is associated with the transaction the implementation SHALL pass a value of -1;

`request` - The HTTP message request; i.e., the request line and subsequent message headers.

`networkInterface` - The `NetworkInterface` the session identified by `activityID` took place on. The `getInetAddress` method of this parameter SHALL return the `InetAddress` on which the `request` was received.

Returns:

true if the activity is accepted; false otherwise.

org.ocap.hn.security

Class NetSecurityManager

```
java.lang.Object
  └org.ocap.hn.security.NetSecurityManager
```

```
public abstract class NetSecurityManager
extends java.lang.Object
```

This class provides access to home network security capabilities including password handling. The passwords that can be handled are specific to each home network interface at the link layer. Upper layer (e.g. TLS) and Administrator passwords cannot be accessed using this class. When the network interface type returned by `NetworkInterface.getType()` is MOCA the implementation SHALL associate the getNetworkPassword and setNetworkPassword methods in this interface to the MoCA link layer password used for the network interface. When the network interface type returned by `NetworkInterface.getType()` is WIRELESS_ETHERNET the implementation SHALL associate the getNetworkPassword and setNetworkPassword in this interface to the link layer password, e.g. WEP, used for the network interface.

This class also permits privileged applications to register a handler to authorize home network activity.

See Also:

`NetAuthorizationHandler`, `NetAuthorizationHandler2`

Constructor Summary

protected	**NetSecurityManager**() Protected constructor; not for application use.

Method Summary

void	**disableMocaPrivacy**(NetworkInterface networkInterface) Disables MoCA privacy.
void	**enableMocaPrivacy**(NetworkInterface networkInterface) Enables MoCA privacy.
static NetSecurityManager	**getInstance**() Get the network security manager.
java.lang.String	**getNetworkPassword**(NetworkInterface networkInterface) Gets a network interface password.
boolean	**queryTransaction**(java.lang.String actionName, java.net.InetAddress inetAddress, java.lang.String macAddress, java.net.URL url, int activityID) Queries the implementation to determine if it has sent a transaction matching the parameters.
void	**revokeAuthorization**(int activityID) Revokes a session authorization granted by the authorization handler.

Method Summary

void	**setAuthorizationHandler**(NetAuthorizationHandler nah) Registers an authorization handler.
void	**setAuthorizationHandler** (NetAuthorizationHandler2 nah, java.lang.String[] actionNames, boolean notifyTransportRequests) Registers an authorization handler.
void	**setAuthorizationHandler**(NetAuthorizationHandler nah, java.lang.String[] actionNames, boolean notifyTransportRequests) Registers an authorization handler.
void	**setNetworkPassword** (NetworkInterface networkInterface, java.lang.String password) Sets a network interface password.

Methods inherited from class java.lang.Object

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

Constructor Detail

NetSecurityManager

protected **NetSecurityManager**()

Protected constructor; not for application use.

Method Detail

getInstance

public static NetSecurityManager **getInstance**()

Get the network security manager.

getNetworkPassword

public java.lang.String **getNetworkPassword**(NetworkInterface networkInterface)

Gets a network interface password.

Parameters:

networkInterface - The interface to get the password for.

Returns:

The value of the password requested, or 0 length String if no password is set for the interface. If the interface type is MoCA this method returns a string value equal to the corresponding mocaIfPassword MIB. In this case the password MAY have been set using means other than the setNetworkPassword method.

Throws:

java.lang.UnsupportedOperationException - if a password cannot be retrieved for the network interface.

`java.lang.SecurityException` - if the caller has not been granted MonitorAppPermission("handler.homenetwork").

setNetworkPassword

```
public void setNetworkPassword(NetworkInterface networkInterface,
                               java.lang.String password)
```

Sets a network interface password. If the network interface type is MoCA then privacy must also be enabled for the password to have affect. See the `enableMocaPrivacy` method. If the interface type is MoCA and the parameter is acceptable this method writes the corresponding mocaIfPassword MIB.

Parameters:

`networkInterface` - The home network interface the password is to be set for.

`password` - The value of the password to set.

Throws:

`java.lang.IllegalArgumentException` - if the password format is invalid for the interface type. A password for a MoCA interface that is less than 12 characters or greater than 17 characters or has any non-numerical characters is invalid.

`java.lang.UnsupportedOperationException` - if a password cannot be set for the network interface.

`java.lang.SecurityException` - if the caller has not been granted MonitorAppPermission("handler.homenetwork").

setAuthorizationHandler

```
public void setAuthorizationHandler(NetAuthorizationHandler nah)
```

Registers an authorization handler. If a handler is already registered (whether a NetAuthorizationHandler or NetAuthorizationHandler2) this method SHALL replace it. If the `nah` parameter is null, any previously registered handler is removed.

A call to this method is equivalent to calling `setAuthorizationHandler(nah, actionNames, true)`, where `actionNames` is an empty array.

Parameters:

`nah` - The network authorization handler to register.

Throws:

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

See Also:

`setAuthorizationHandler(NetAuthorizationHandler, String[], boolean)`

setAuthorizationHandler

```
public void setAuthorizationHandler(NetAuthorizationHandler nah,
                                    java.lang.String[] actionNames,
                                    boolean notifyTransportRequests)
```

Registers an authorization handler. If a handler is already registered (whether a NetAuthorizationHandler or NetAuthorizationHandler2) this method SHALL replace it. If the `nah` parameter is null, any previously registered handler is removed.

The `actionNames` parameter permits the caller to specify an array of names indicating the actions that the handler wishes to authorize; an empty array indicates that `NetAuthorizationHandler.notifyAction(java.lang.String, java.net.InetAddress, java.lang.String, int)` will not be called.

The `notifyTransportRequests` parameter permits the caller to control whether `NetAuthorizationHandler.notifyActivityStart(java.net.InetAddress,`

`java.lang.String, java.net.URL, int)` is called for every transport protocol (e.g., HTTP, RTP/RTSP) request in the session or only the initial one.

Parameters:
`nah` - The network authorization handler to register.
`actionNames` - An array of action names the hander is interested in authorizing. The format of the names is out-of-scope for this definition.
`notifyTransportRequests` - If true, `NetAuthorizationHandler.notifyActivityStart(java.net.InetAddress, java.lang.String, java.net.URL, int)` is always called when a transport protocol message is received; if false, `NetAuthorizationHandler.notifyActivityStart(java.net.InetAddress, java.lang.String, java.net.URL, int)` is only called for the first message in a session.
Throws:
`java.lang.SecurityException` - if the caller does not have MonitorAppPermission("handler.homenetwork").
`java.lang.IllegalArgumentException` - if the actionNames parameter contains a name that cannot be matched to a known action.

setAuthorizationHandler

```
public void setAuthorizationHandler(NetAuthorizationHandler2 nah,
                                    java.lang.String[] actionNames,
                                    boolean notifyTransportRequests)
```

Registers an authorization handler. This method takes a `NetAuthorizationHander2` parameter which provides additional information about the activity to the notify methods. If a handler is already registered (whether a NetAuthorizationHandler or NetAuthorizationHandler2) this method SHALL replace it. If the `nah` parameter is null, any previously registered handler is removed.

The `actionNames` parameter permits the caller to specify an array of names indicating the actions that the handler wishes to authorize; an empty array indicates that `NetAuthorizationHandler2.notifyAction(java.lang.String, java.net.InetAddress, int, java.lang.String[], org.ocap.hn.NetworkInterface)` will not be called.

The `notifyTransportRequests` parameter permits the caller to control whether `NetAuthorizationHandler2.notifyActivityStart(java.net.InetAddress, java.net.URL, int, org.ocap.hn.content.ContentEntry, java.lang.String[], org.ocap.hn.NetworkInterface)` is called for every transport protocol (e.g., HTTP, RTP/RTSP) request in the session or only the initial one.

Parameters:
`nah` - The network authorization handler to register.
`actionNames` - An array of action names the hander is interested in authorizing. The format of the names is out-of-scope for this definition.
`notifyTransportRequests` - If true, `NetAuthorizationHandler2.notifyActivityStart(java.net.InetAddress, java.net.URL, int, org.ocap.hn.content.ContentEntry, java.lang.String[], org.ocap.hn.NetworkInterface)` is always called when a transport protocol message is received; if false, `NetAuthorizationHandler2.notifyActivityStart(java.net.InetAddress, java.net.URL, int, org.ocap.hn.content.ContentEntry, java.lang.String[], org.ocap.hn.NetworkInterface)` is only called for the first message in a session.
Throws:
`java.lang.SecurityException` - if the caller does not have MonitorAppPermission("handler.homenetwork").
`java.lang.IllegalArgumentException` - if the actionNames parameter contains a name that cannot be matched to a known action.

revokeAuthorization

```
public void revokeAuthorization(int activityID)
```

Revokes a session authorization granted by the authorization handler.

Parameters:

`activityID` - The activity identifier that was passed to the authorization handler's `notifyActivityStart` method.

Throws:

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

queryTransaction

```
public boolean queryTransaction(java.lang.String actionName,
                                java.net.InetAddress inetAddress,
                                java.lang.String macAddress,
                                java.net.URL url,
                                int activityID)
```

Queries the implementation to determine if it has sent a transaction matching the parameters.

Parameters:

`actionName` - Name of the request type if known. If not known an empty string MAY be used. The format of the name is out-of-scope of this definition.

`inetAddress` - IP address the transaction was sent to.

`macAddress` - MAC address the transaction was sent from if known. Can be empty `String` if not known. The format is EUI-48 with 6 colon separated 2 digit bytes in hexadecimal notation with no leading "0x", e.g. "00:11:22:AA:BB:CC".

`url` - The URL requested by the transaction if known. If not known an empty string may be used.

`activityID` - The activity identifier this device set for the connection. A value of -1 indicates the parameter will not be used for transaction matching purposes.

Returns:

True if activityID and other known parameters can be matched to a transaction sent by the implementation. If activityID match cannot be found, or if activityID match is found but any of the other known parameters do not match the transaction then this method returns false.

Throws:

`java.lang.IllegalArgumentException` - if the MAC address is malformed.

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

enableMocaPrivacy

```
public void enableMocaPrivacy(NetworkInterface networkInterface)
```

Enables MoCA privacy. For MoCA interface types this method enables privacy and writes the corresponding mocaIfPrivacyEnable MIB with a value of 'true'.

Parameters:

`networkInterface` - Interface to enable privacy on.

Throws:

`java.lang.UnsupportedOperationException` - if the parameter interface is not a MoCA interface type.

`java.lang.SecurityException` - if the caller has not been granted MonitorAppPermission("handler.homenetwork").

disableMocaPrivacy

```
public void disableMocaPrivacy(NetworkInterface networkInterface)
```

Disables MoCA privacy. For MoCA interface types this method disables privacy and writes the corresponding mocaIfPrivacyEnable MIB with a value of 'false'.

Parameters:

`networkInterface` - Interface to disable privacy on.

Throws:

`java.lang.UnsupportedOperationException` - if the parameter interface is not a MoCA interface type.

`java.lang.SecurityException` - if the caller has not been granted MonitorAppPermission("handler.homenetwork").

Annex H UPnP Diagnostics API

Provides mechanisms for application access to a Host device Universal Plug and Play (UPnP) implementation. An application can use the `org.ocap.hn.upnp` package in order to directly invoke UPnP functionality. The package contains three sub-packages containing client, server, and common functionality:

- org.ocap.hn.upnp.client
- org.ocap.hn.upnp.server
- org.ocap.hn.upnp.common

There are two entry points (singletons) into the package for client and server functionality:

1. **Client:** The UPnP Control Point class (org.ocap.hn.upnp.client.UPnPControlPoint) provides access to devices and services on a home network.

2. **Server:** The UPnP Device Manager class (org.ocap.hn.upnp.server.UPnPDeviceManager) provides management of devices and services in the local Host device.

Package org.ocap.hn.upnp.client

Provides UPnP client functionality, permitting access to devices and services on a home network.

See:

Description

Interface Summary

UPnPActionResponseHandler	This interface represents a listener for an asynchronous UPnP action response.
UPnPClientDevice	This interface provides the client representation of a UPnP device, associated with a single IP address.
UPnPClientDeviceIcon	This interface is the client representation of a UPnP Device Icon.
UPnPClientDeviceListener	This interface represents a listener to UPnP device availability on a home network.
UPnPClientService	This interface is the client representation of a UPnP service.
UPnPClientStateVariable	This interface is the client representation of a UPnP state variable.
UPnPStateVariableListener	This interface represents a client variable change listener for a `UPnPClientService` object.

Class Summary

UPnPControlPoint	This class represents a device control point that can discover devices and services.

Package org.ocap.hn.upnp.client Description

Provides UPnP client functionality, permitting access to devices and services on a home network.

The UPnP Control Point class (`UPnPControlPoint`) provides access to devices discovered by the Host device during a UPnP discovery process as defined by the UPnP Device Architecture specification. As defined by UPnP, a device consists of a root device and 0 or more sub-devices. Each device can contain services such as the Content Directory Service (CDS). Each service can contain actions, e.g. CDS:Search, and state variables, e.g. CDS:SystemUpdateID. The `UPnPControlPoint.getDevices()` method returns a data structure that

represents the UPnP devices, services, etc., discovered in a home network. The data structure returned matches the hierarchy of the discovered UPnP device documents.

In the `org.ocap.hn.upnp.client` package UPnP entities are represented as follows:

- UPnP Device - `UPnPClientDevice`
- UPnP Device Icon - `UPnPClientDeviceIcon`
- UPnP Service - `UPnPClientService`
- UPnP State Variable - `UPnPClientStateVariable`

Objects of these types are immutable in order to represent entities that actually reside in remote Host devices.

Once an application calls the `UPnPControlPoint.getDevices()` method it can peruse the data structure returned, access the root device for a specific home network server and exhibit typical UPnP behaviors such as the following:

- Evaluate device properties, services, and sub-devices.
- Evaluate service properties, actions, and state variables.
- Send an action and get a response.
- Subscribe to state variable events.

In addition to typical UPnP behaviors an application can set itself using as the incoming and/or outgoing message handler using the `UPnPControlPoint`. An incoming message handler can modify incoming messages before Host device evaluation. An outgoing message handler can modify outgoing messages before transmission. This is useful when, for instance, a server device sends non-standard properties in a verbose manner and an application needs to prune for various reasons.

org.ocap.hn.upnp.client

Interface UPnPActionResponseHandler

```
public interface UPnPActionResponseHandler
```

This interface represents a listener for an asynchronous UPnP action response.

Method Summary

void	**notifyUPnPActionResponse**(UPnPResponse response) Notifies the listener when a UPnP action response is received.

Method Detail

notifyUPnPActionResponse

```
void notifyUPnPActionResponse(UPnPResponse response)
```

Notifies the listener when a UPnP action response is received.

Parameters:

`response` - The response to a UPnP action: a `UPnPActionResponse`, a `UPnPErrorResponse`, or a `UPnPGeneralErrorResponse`.

org.ocap.hn.upnp.client

Interface UPnPClientDevice

All Superinterfaces:
UPnPAdvertisedDevice, UPnPDevice

```
public interface UPnPClientDevice
extends UPnPAdvertisedDevice
```

This interface provides the client representation of a UPnP device, associated with a single IP address.

Method Summary

UPnPClientDevice[]	**getEmbeddedDevices**() Gets the embedded devices for this UPnP Device.
UPnPClientDeviceIcon[]	**getIcons**() Gets the icons of this device.
UPnPClientDevice	**getParentDevice**() Returns the parent UPnP Device of this device, if any.
UPnPClientService[]	**getServices**() Gets the services supported by this device.

Methods inherited from interface org.ocap.hn.upnp.common.UPnPAdvertisedDevice
getAdvertisedIcons, getAdvertisedServices, getEmbeddedAdvertisedDevices, getInetAddress, getPresentationURL, getURLBase, getXML

Methods inherited from interface org.ocap.hn.upnp.common.UPnPDevice
getDeviceType, getFriendlyName, getManufacturer, getManufacturerURL, getModelDescription, getModelName, getModelNumber, getModelURL, getSerialNumber, getSpecVersion, getUDN, getUPC, isRootDevice

Method Detail

getParentDevice

`UPnPClientDevice` **`getParentDevice`**`()`

Returns the parent UPnP Device of this device, if any.

Returns:
This device's parent device. Returns null if this device has no parent.

getEmbeddedDevices

`UPnPClientDevice[]` **`getEmbeddedDevices`**`()`

Gets the embedded devices for this UPnP Device.

Returns:
The embedded devices for this device. If this device has no embedded devices, returns a zero length array. Returns only the next level of embedded devices, not recursing through embedded devices for subsequent levels of embedded devices.

getServices

`UPnPClientService[] getServices()`

Gets the services supported by this device. Does not return services held in embedded devices.

Returns:

The services supported by this device. If the serviceList element in the device description is empty, this method returns a zero length array.

getIcons

`UPnPClientDeviceIcon[] getIcons()`

Gets the icons of this device. This returned array is derived from the icon elements within the `iconList` element of a device description. If the iconList element in the device description is empty or not present, returns a zero length array.

Returns:

The icons that the device declares.

org.ocap.hn.upnp.client

Interface UPnPClientDeviceIcon

All Superinterfaces:
UPnPAdvertisedDeviceIcon, UPnPDeviceIcon

```
public interface UPnPClientDeviceIcon
extends UPnPAdvertisedDeviceIcon
```

This interface is the client representation of a UPnP Device Icon.

Method Summary

UPnPClientDevice	**getDevice**() Gets the UPnP device that this icon is associated with.

Methods inherited from interface org.ocap.hn.upnp.common.UPnPAdvertisedDeviceIcon
getURL

Methods inherited from interface org.ocap.hn.upnp.common.UPnPDeviceIcon
getColorDepth, getHeight, getMimeType, getWidth

Method Detail

getDevice

```
UPnPClientDevice getDevice()
```

Gets the UPnP device that this icon is associated with.

Returns:
The device that this icon is associated with.

org.ocap.hn.upnp.client

Interface UPnPClientDeviceListener

All Superinterfaces:
java.util.EventListener

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

This interface represents a listener to UPnP device availability on a home network.

See Also:
`UPnPManagedDeviceListener`

Method Summary

void	**notifyDeviceAdded**(UPnPClientDevice device) Notifies the listener that a UPnP device was added to a home network.
void	**notifyDeviceRemoved**(UPnPClientDevice device) Notifies the listener that a UPnP device was removed from a home network, or did not renew its advertisement prior to expiration of the prior advertisement.

Method Detail

notifyDeviceAdded

```
void notifyDeviceAdded(UPnPClientDevice device)
```

Notifies the listener that a UPnP device was added to a home network.

Parameters:
`device` - The `UPnPDevice` that was added.

notifyDeviceRemoved

```
void notifyDeviceRemoved(UPnPClientDevice device)
```

Notifies the listener that a UPnP device was removed from a home network, or did not renew its advertisement prior to expiration of the prior advertisement.

Parameters:
`device` - The `UPnPDevice` that was removed.

org.ocap.hn.upnp.client

Interface UPnPClientService

All Superinterfaces:
UPnPAdvertisedService, UPnPService

```
public interface UPnPClientService
extends UPnPAdvertisedService
```

This interface is the client representation of a UPnP service.

Method Summary

void	**addStateVariableListener** (UPnPStateVariableListener listener) Adds a state variable listener to this UPnPClientService.
UPnPClientDevice	**getDevice**() Gets the UPnP device that this service is a part of.
UPnPClientStateVariable	**getStateVariable** (java.lang.String stateVariableName) Gets a UPnP state variable from the UPnP description of this service.
UPnPClientStateVariable[]	**getStateVariables**() Gets all of the UPnP state variables supported by this service.
boolean	**getSubscribedStatus**() Gets the subscription status of the service.
void	**postActionInvocation** (UPnPActionInvocation actionInvocation, UPnPActionResponseHandler handler) Posts an action to the network.
void	**removeStateVariableListener** (UPnPStateVariableListener listener) Removes a change listener.
void	**setSubscribedStatus**(boolean subscribed) Attempts to subscribe or unsubscribe the control point to/from this service.

Methods inherited from interface org.ocap.hn.upnp.common.UPnPAdvertisedService
getAdvertisedStateVariable, getAdvertisedStateVariables, getControlURL, getEventSubURL, getSCPDURL, getXML

Methods inherited from interface org.ocap.hn.upnp.common.UPnPService
getAction, getActions, getServiceId, getServiceType, getSpecVersion

Method Detail

getDevice

```
UPnPClientDevice getDevice()
```

Gets the UPnP device that this service is a part of.

Returns:

The device that this service is a part of.

postActionInvocation

```
void postActionInvocation(UPnPActionInvocation actionInvocation,
                          UPnPActionResponseHandler handler)
```

Posts an action to the network. Sends the action from the control point to the device the service is in. The device MAY be on the local host. If no handler is set when this method is called, the response is consumed by the implementation in an implementation-specific fashion.

Parameters:

`actionInvocation` - The action invocation to post.

`handler` - The handler that will be notified when the action response is received. May be null, in which case the action response will be discarded.

Throws:

`java.lang.NullPointerException` - if action is null.

See Also:

`UPnPActionInvocation`

addStateVariableListener

```
void addStateVariableListener(UPnPStateVariableListener listener)
```

Adds a state variable listener to this `UPnPClientService`. If this service has evented state variables, this method will cause the control point to attempt to subscribe to the service if it is not already subscribed. See UPnP Device Architecture specification for UPnP service and state variable subscription.

Adding a listener which is the same instance as a previously added (and not removed) listener has no effect.

Parameters:

`listener` - The listener to add.

See Also:

`setSubscribedStatus(boolean)`

removeStateVariableListener

```
void removeStateVariableListener(UPnPStateVariableListener listener)
```

Removes a change listener.

Parameters:

`listener` - The listener to remove.

setSubscribedStatus

```
void setSubscribedStatus(boolean subscribed)
```

Attempts to subscribe or unsubscribe the control point to/from this service. Changes to subscription status are signaled asynchronously via the `UPnPStateVariableListener` interface.

Parameters:

`subscribed` - True to subscribe to evented state variable updates, false to unsubscribe.

Throws:

`java.lang.UnsupportedOperationException` - if `subscribed` is `true` but the service has no evented state variables.

getSubscribedStatus

`boolean` **`getSubscribedStatus`**`()`

Gets the subscription status of the service. Defaults to subscribed if the service has evented state variables; false otherwise.

Returns:

True if the control point is presently registered to receive UPnP events from the service, false if not.

getStateVariable

`UPnPClientStateVariable` **`getStateVariable`**`(java.lang.String stateVariableName)`

Gets a UPnP state variable from the UPnP description of this service. Supported state variable names are provided by a UPnP device in the name element of each stateVariable element in a device service description.

Parameters:

`stateVariableName` - The name of the state variable to get.

Returns:

The state variable corresponding to the `stateVariableName` parameter.

Throws:

`java.lang.IllegalArgumentException` - if the `stateVariableName` does not match a state variable name in this service.

getStateVariables

`UPnPClientStateVariable[]` **`getStateVariables`**`()`

Gets all of the UPnP state variables supported by this service. UPnP state variable information is taken from the `stateVariable` elements in the UPnP service description.

Returns:

The UPnP state variables supported by this service. If the service has no state variables, returns a zero-length array.

org.ocap.hn.upnp.client

Interface UPnPClientStateVariable

All Superinterfaces:
UPnPAdvertisedStateVariable, UPnPStateVariable

```
public interface UPnPClientStateVariable
extends UPnPAdvertisedStateVariable
```

This interface is the client representation of a UPnP state variable.

Method Summary

java.lang.String	**getEventedValue**() Gets the value of the UPnP state variable corresponding to this UPnPClientStateVariable object.
UPnPClientService	**getService**() Gets the UPnP service that this state variable is a member of.

Methods inherited from interface org.ocap.hn.upnp.common.UPnPStateVariable
getAllowedValues, getDataType, getDefaultValue, getMaximumValue, getMinimumValue, getName, getStepValue, isEvented

Method Detail

getEventedValue

java.lang.String **getEventedValue**()

Gets the value of the UPnP state variable corresponding to this UPnPClientStateVariable object.

Returns:
The most recently received evented value of the state variable.

Throws:
java.lang.UnsupportedOperationException - if the UPnP state variable corresponding to this UPnPStateVariable object is not evented as can be determined by calling the isEvented() method.

getService

UPnPClientService **getService**()

Gets the UPnP service that this state variable is a member of.

Returns:
The service that this state variable is part of.

org.ocap.hn.upnp.client

Class UPnPControlPoint

```
java.lang.Object
  └org.ocap.hn.upnp.client.UPnPControlPoint
```

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

This class represents a device control point that can discover devices and services. It also offers a facility to directly monitor and modify communication between the control point and any devices.

Constructor Summary

protected	**UPnPControlPoint**() Construct the instance.

Method Summary

void	**addDeviceListener**(UPnPClientDeviceListener listener) Adds a listener for device changes.
UPnPClientDevice[]	**getDevices**() Gets a client representation of all UPnP root devices visible to this host.
UPnPClientDevice[]	**getDevicesByServiceType**(java.lang.String type) Gets a client representation of all UPnP devices containing a service of the specified type, visible to this host.
UPnPClientDevice[]	**getDevicesByType**(java.lang.String type) Gets a client representation of all UPnP devices of the specified type visible to this host.
UPnPClientDevice[]	**getDevicesByUDN**(java.lang.String UDN) Gets a client representation of the UPnP devices of the specified UDN visible to this host.
java.net.InetAddress[]	**getInetAddresses**() Gets the InetAddresses that this UPnPControlPoint is associated with.
static UPnPControlPoint	**getInstance**() Obtain the local UPnP device control point.
void	**removeDeviceListener**(UPnPClientDeviceListener listener) Removes a device listener.
void	**search**(int mx) Initiate a UPnP M-SEARCH for UPnP root devices.
void	**setIncomingMessageHandler**(UPnPIncomingMessageHandler inHandler) Sets a message handler for incoming messages (advertisements, evented state variables, action responses, device and service descriptions).

Method Summary

java.net.InetAddress[]	**setInetAddresses**(java.net.InetAddress[] addresses) Sets the InetAddresses that the UPnPControlPoint is associated with.
void	**setOutgoingMessageHandler**(UPnPOutgoingMessageHandler outHandler) Sets a message handler for outgoing messages (action invocations, subscription requests, device and service retrievals).

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

UPnPControlPoint

```
protected UPnPControlPoint()
```

Construct the instance.

Method Detail

getInstance

```
public static UPnPControlPoint getInstance()
```

Obtain the local UPnP device control point.

Returns:

The singleton UPnPControlPoint.

setInetAddresses

```
public java.net.InetAddress[]
setInetAddresses(java.net.InetAddress[] addresses)
                                        throws java.lang.SecurityException
```

Sets the InetAddresses that the UPnPControlPoint is associated with. The control point will only send searches and listen for device advertisements on the most appropriate interface for each of the addresses specified.

The passed array replaces any prior addresses that the control point was associated with. The control point may need to perform searches and update its list of devices in response to this method being invoked.

Note that the control point defaults to all home network interfaces with all their associated IP addresses. A client application would not normally need to invoke this method.

Parameters:

addresses - Array of InetAddress objects representing the IP addresses that the control point is associated with. May be zero length.

Returns:

Array of prior addresses that were associated with the UPnPControlPoint. If there were no prior addresses, returns a zero-length array.

Throws:

java.lang.NullPointerException - if addresses or any of its elements is null.

java.lang.SecurityException - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

getInetAddresses

`public java.net.InetAddress[] getInetAddresses()`

Gets the InetAddresses that this UPnPControlPoint is associated with.

Returns:

Array of InetAddress objects representing the network interfaces that this control point is associated with. If the control point has no associated network interfaces, returns a zero length array.

getDevices

`public UPnPClientDevice[] getDevices()`

Gets a client representation of all UPnP root devices visible to this host. This does not cause a search to take place, but simply returns the currently known devices.

Returns:

The UPnP devices visible to this host. Each element in the array of UPnPClientDevices returned represents one root device found by the local host via UPnP discovery. If no root devices are found, returns a zero-length array.

getDevicesByType

`public UPnPClientDevice[] getDevicesByType(java.lang.String type)`

Gets a client representation of all UPnP devices of the specified type visible to this host. This does not cause a search to take place, but simply returns the currently known devices.

Parameters:

type - The type of devices to return. Of the form urn:schemas-upnp-org:device:deviceType:v where deviceType is replaced with a type specific to the device being requested, and v is a version specifier as defined in UPnP Device Architecture.

Returns:

The UPnP devices visible to this host matching the type specified, of the specified version or lower version number. Each element in the array of UPnPClientDevices returned represents one device found by the local host via UPnP discovery. If no devices matching the type are found, returns a zero-length array.

getDevicesByUDN

`public UPnPClientDevice[] getDevicesByUDN(java.lang.String UDN)`

Gets a client representation of the UPnP devices of the specified UDN visible to this host. This does not cause a search to take place, but simply returns the currently known devices.

Note that normally a UDN is unique and would return a single device. In multi-homed server and control point environments, a single server may be visible over multiple interfaces, resulting in multiple UPnPClientDevice representations.

Parameters:

UDN - The UDN of the devices to return.

Returns:

The UPnP devices visible to this host matching the UDN specified. Each element in the array of UPnPClientDevices returned represents one device found by the local host via UPnP discovery. If no devices matching the UDN are found, returns a zero-length array.

getDevicesByServiceType

`public UPnPClientDevice[] getDevicesByServiceType(java.lang.String type)`

Gets a client representation of all UPnP devices containing a service of the specified type, visible to this host. This does not cause a search to take place, but simply returns the currently known devices.

Parameters:

`type` - The type of service to use in determining which devices to return. Of the form urn:schemas-upnp-org:service:serviceType:v where serviceType is replaced with a type specific to the service being requested, and v is a version specifier as defined in UPnP Device Architecture.

Returns:

The UPnP devices visible to this host containing a service matching the type specified, of the specified version or lower version number. Each element in the array of `UPnPClientDevice`s returned represents one device found by the local host via UPnP discovery. Returns only devices directly containing a service of the matching type, not devices where only their embedded devices contain a service of the matching type. If no devices matching the criteria are found, returns a zero-length array.

search

```
public void search(int mx)
```

Initiate a UPnP M-SEARCH for UPnP root devices. The UPnP stack constantly monitors for device arrival and departure. This method is used to assist with detection of devices which may not renew advertisements correctly and only respond to search requests.

Parameters:

`mx` - The maximum time in seconds for client devices to respond to this search.

addDeviceListener

```
public void addDeviceListener(UPnPClientDeviceListener listener)
```

Adds a listener for device changes. Each `UPnPClientDeviceListener` is notified when a UPnP device is added to or removed from a home network.

Adding a listener which is the same instance as a previously added (and not removed) listener has no effect.

Parameters:

`listener` - The listener to add.

removeDeviceListener

```
public void removeDeviceListener(UPnPClientDeviceListener listener)
```

Removes a device listener.

Parameters:

`listener` - The listener to remove.

setIncomingMessageHandler

```
public void setIncomingMessageHandler(UPnPIncomingMessageHandler inHandler)
                                throws java.lang.SecurityException
```

Sets a message handler for incoming messages (advertisements, evented state variables, action responses, device and service descriptions). Calls to set the message handler replace any prior incoming message handler.

A message handler may be removed by passing null as the inHandler. In the absence of a registered message handler the stack will parse the incoming messages.

If the application-provided handler throws any exceptions during execution, the stack will attempt to process the message with the default (stack-provided) handler.

Parameters:

`inHandler` - The incoming message handler to set.

Throws:

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

setOutgoingMessageHandler

```
public void setOutgoingMessageHandler(UPnPOutgoingMessageHandler outHandler)
                                throws java.lang.SecurityException
```

Sets a message handler for outgoing messages (action invocations, subscription requests, device and service retrievals). Calls to set the message handler replace any prior outgoing message handler.

A message handler may be removed by passing null as the outHandler. In the absence of a registered message handler the stack will process the outgoing messages.

If the application-provided handler throws any exceptions during execution, the stack will attempt to process the message with the default (stack-provided) handler.

Parameters:
`outHandler` - The outgoing message handler to set.
Throws:
`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

org.ocap.hn.upnp.client

Interface UPnPStateVariableListener

All Superinterfaces:
java.util.EventListener

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

This interface represents a client variable change listener for a `UPnPClientService` object.

Method Summary

`void`	**`notifySubscribed`**`(UPnPClientService service)` Notifies the listener that the control point has successfully subscribed to receive state variable eventing from the specified service.
`void`	**`notifyUnsubscribed`**`(UPnPClientService service)` Notifies the listener that the control point has successfully unsubscribed from receiving state variable eventing from the specified service.
`void`	**`notifyValueChanged`**`(UPnPClientStateVariable variable)` Notifies the listener that the value of the UPnP state variable being listened to has changed.

Method Detail

notifyValueChanged

`void` **`notifyValueChanged`**`(UPnPClientStateVariable variable)`

Notifies the listener that the value of the UPnP state variable being listened to has changed.

Parameters:
`variable` - The UPnP state variable that changed.

notifySubscribed

`void` **`notifySubscribed`**`(UPnPClientService service)`

Notifies the listener that the control point has successfully subscribed to receive state variable eventing from the specified service.

Parameters:
`service` - The UPnP service that was subscribed to.

notifyUnsubscribed

`void` **`notifyUnsubscribed`**`(UPnPClientService service)`

Notifies the listener that the control point has successfully unsubscribed from receiving state variable eventing from the specified service.

Parameters:
`service` - The UPnP service that was un-subscribed from.

Package org.ocap.hn.upnp.common

Provides UPnP APIs common to client- and server-side use.

See:

Description

Interface Summary

UPnPAction	This interface represents the description of a UPnP service action, parsed from the UPnP service description XML.
UPnPAdvertisedDevice	This interface represents a UPnP device as it is advertised on a particular network.
UPnPAdvertisedDeviceIcon	This interface represents a UPnP device icon as it is described on a particular network.
UPnPAdvertisedService	This interface represents a UPnP service as it is advertised on a particular network.
UPnPAdvertisedStateVariable	This interface represents a UPnP state variable as it is described on a particular network.
UPnPDevice	This interface is an abstract representation of a UPnP device.
UPnPDeviceIcon	This interfaces is an abstract representation of a UPnP device icon.
UPnPIncomingMessageHandler	This interface represents an incoming message handler that can monitor and modify any incoming messages to the UPnP stack.
UPnPOutgoingMessageHandler	This interface represents an outgoing message handler that can monitor and modify any outgoing messages from the UPnP stack.
UPnPService	This interface is an abstract representation of a UPnP service.
UPnPStateVariable	This interface is an abstract representation of a UPnP state variable.

Class Summary

UPnPActionInvocation	This class represents a UPnP service action invocation, carrying only IN arguments and a reference to the action definition (`UPnPAction`).
UPnPActionResponse	The class represents a response to a successfully completed UPnP action.
UPnPErrorResponse	The class represents a response to an unsuccessfully completed UPnP action, where the server responded with HTTP code 500 (Internal Server Error).
UPnPGeneralErrorResponse	The class represents a response to an unsuccessfully completed UPnP action, where the server either did not respond, or responded with an HTTP error code other than 500 (Internal Server Error).
UPnPMessage	The class represents a UPnP message comprising an HTTP start line, zero or more headers and an optional XML document.
UPnPResponse	This class represents a response to a UPnP action.

Package org.ocap.hn.upnp.common Description

Provides UPnP APIs common to client- and server-side use.

org.ocap.hn.upnp.common

Interface UPnPAction

```
public interface UPnPAction
```

This interface represents the description of a UPnP service action, parsed from the UPnP service description XML. It contains both IN and OUT argument descriptions, but does not carry any values.

Method Summary

java.lang.String[]	**getArgumentNames**() Gets the action argument names from the action description in the UPnP service description.
java.lang.String	**getName**() Gets the name of the action from the action name element in the UPnP service description.
UPnPStateVariable	**getRelatedStateVariable**(java.lang.String name) Gets the UPnP state variable associated with the specified argument name.
UPnPService	**getService**() Gets the UPnP service that this UPnPAction is associated with.
boolean	**isInputArgument**(java.lang.String name) Gets the direction of an argument.
boolean	**isRetval**(java.lang.String name) Determines whether the specified argument is flagged as a return value in the service description.

Method Detail

getName

java.lang.String **getName**()

Gets the name of the action from the action name element in the UPnP service description.

Returns:

name of the action.

getArgumentNames

java.lang.String[] **getArgumentNames**()

Gets the action argument names from the action description in the UPnP service description.

Returns:

The IN and OUT argument names for this action, in the order specified by the UPnP service description defining this action. If the action has no arguments, returns a zero length array.

isInputArgument

boolean **isInputArgument**(java.lang.String name)

Gets the direction of an argument.

Parameters:

name - Name of the argument.
Returns:
True if the argument is an input argument.
Throws:
java.lang.IllegalArgumentException - if the name does not represent a valid argument name for the action.

isRetval

boolean **isRetval**(java.lang.String name)

Determines whether the specified argument is flagged as a return value in the service description.
Parameters:
name - Name of the argument.
Returns:
true if the argument is flagged as a retval.
Throws:
java.lang.IllegalArgumentException - if the name does not represent a valid argument name for the action.

getService

UPnPService **getService**()

Gets the UPnP service that this UPnPAction is associated with. The returned UPnPService object may be cast to a UPnPManagedService by server applications, or to a UPnPClientService by client applications.
Returns:
The UPnP service that this action is associated with.

getRelatedStateVariable

UPnPStateVariable **getRelatedStateVariable**(java.lang.String name)

Gets the UPnP state variable associated with the specified argument name. The returned UPnPStateVariable object may be cast to a UPnPManagedStateVariable by server applications, or to a UPnPClientStateVariable by client applications.
Parameters:
name - Name of the argument.
Returns:
The UPnP state variable associated with the specified argument name.
Throws:
java.lang.IllegalArgumentException - if the name does not represent a valid argument name for the action.

org.ocap.hn.upnp.common

Class UPnPActionInvocation

```
java.lang.Object
  └org.ocap.hn.upnp.common.UPnPActionInvocation
```

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

This class represents a UPnP service action invocation, carrying only IN arguments and a reference to the action definition (`UPnPAction`). It is constructed by a client application and passed to a `UPnPService` via the `postActionInvocation` method in order to invoke an action on a UPnP server.

See Also:

`UPnPClientService.postActionInvocation(UPnPActionInvocation, UPnPActionResponseHandler)`, `UPnPActionHandler.notifyActionReceived(UPnPActionInvocation)`

Constructor Summary

`UPnPActionInvocation``(java.lang.String[] argVals, UPnPAction action)` Constructs a `UPnPActionInvocation` that conforms to the IN argument requirements of its associated `UPnPAction`.	

Method Summary

`UPnPAction`	**`getAction`**`()` Gets the `UPnPAction` that this `UPnPActionInvocation` is associated with.
`java.lang.String[]`	**`getArgumentNames`**`()` Gets the argument names specified by this action invocation, in the order they were specified in the constructor.
`java.lang.String`	**`getArgumentValue`**`(java.lang.String name)` Gets the value of the specified argument.
`java.lang.String`	**`getName`**`()` Gets the name of the action as specified by the action name element in the UPnP service description.

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

UPnPActionInvocation

```
public UPnPActionInvocation(java.lang.String[] argVals,
                            UPnPAction action)
```

Constructs a `UPnPActionInvocation` that conforms to the IN argument requirements of its associated `UPnPAction`.

This constructor ensures that the resulting action invocation provides an argument value, in the proper `dataType` format, for each of the IN arguments of the specified UPnP action.

For objects created through this constructor, `getArgumentNames()` will report, in order, the required IN argument names of the specified `UPnPAction`.

Parameters:
`argVals` - An array of argument values corresponding, in order, to the IN arguments of `action`.
`action` - The UPnP action that this action invocation relates to.
Throws:
`java.lang.IllegalArgumentException` - if `argVals` does not conform to the IN argument requirements of `action`.
`java.lang.NullPointerException` - if `action` is `null`, or `argVals` or any of its array elements is `null`.

Method Detail

getName

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

Gets the name of the action as specified by the action name element in the UPnP service description. Calls `getAction().getName()`.
Returns:
the name of the action.
See Also:
`getAction()`

getArgumentNames

`public java.lang.String[] **getArgumentNames**()`

Gets the argument names specified by this action invocation, in the order they were specified in the constructor.
Returns:
The argument names of this action invocation. If no arguments have been specified, returns a zero-length array.

getArgumentValue

`public java.lang.String **getArgumentValue**(java.lang.String name)`

Gets the value of the specified argument.
Parameters:
`name` - The name of the argument.
Returns:
The value of the argument.
Throws:
`java.lang.IllegalArgumentException` - if `name` does not match one of the argument names specified for this action invocation.
See Also:
`getArgumentNames()`

getAction

`public UPnPAction` **`getAction`**`()`

Gets the `UPnPAction` that this `UPnPActionInvocation` is associated with.

Returns:

The `UPnPAction` that this action invocation is associated with.

org.ocap.hn.upnp.common

Class UPnPActionResponse

```
java.lang.Object
  └org.ocap.hn.upnp.common.UPnPResponse
      └org.ocap.hn.upnp.common.UPnPActionResponse
```

```
public class UPnPActionResponse
extends UPnPResponse
```

The class represents a response to a successfully completed UPnP action. It carries only the OUT arguments from the action. Instances of this class are constructed by the `UPnPActionHandler` on a UPnP server, and are passed to a client in the `UPnPActionResponseHandler`.

See Also:

`UPnPActionHandler.notifyActionReceived(UPnPActionInvocation)`, `UPnPActionResponseHandler.notifyUPnPActionResponse(UPnPResponse)`

Constructor Summary

`UPnPActionResponse``(java.lang.String[] argVals, UPnPActionInvocation actionInvocation)` Constructs a `UPnPActionResponse` that conforms to the OUT argument requirements of its associated `UPnPAction`.

Method Summary

`java.lang.String[]`	**`getArgumentNames`**`()` Gets the output argument names specified by this action response, in the order they were specified in the constructor.
`java.lang.String`	**`getArgumentValue`**`(java.lang.String name)` Gets the value of the specified argument.
`java.lang.String[]`	**`getArgumentValues`**`()` Gets the output argument values specified by this action response, in the order they were specified in the constructor.

Methods inherited from class org.ocap.hn.upnp.common.UPnPResponse
`getActionInvocation, getHTTPResponseCode`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

UPnPActionResponse

```
public UPnPActionResponse(java.lang.String[] argVals,
```

UPnPActionInvocation actionInvocation)

Constructs a UPnPActionResponse that conforms to the OUT argument requirements of its associated UPnPAction.

This constructor ensures that the resulting action response provides an argument value, in the proper dataType format, for each of the OUT arguments of the UPnP action reported by actionInvocation.getAction().

For objects created through this constructor, getArgumentNames() will report, in order, the required OUT argument names of the associated UPnP action.

Parameters:
argVals - An array of argument values corresponding, in order, to the OUT arguments of actionInvocation.getAction().
actionInvocation - The action invocation that this action response relates to.
Throws:
java.lang.IllegalArgumentException - if argVals does not conform to the OUT argument requirements of actionInvocation.getAction().
java.lang.NullPointerException - if action is null, or argVals or any of its array elements is null.

Method Detail

getArgumentNames

public java.lang.String[] **getArgumentNames**()

Gets the output argument names specified by this action response, in the order they were specified in the constructor.
Returns:
The action response output argument names. If the action response has no output arguments, returns a zero-length array.

getArgumentValues

public java.lang.String[] **getArgumentValues**()

Gets the output argument values specified by this action response, in the order they were specified in the constructor.
Returns:
The action response output argument values. If the action response has no output arguments, returns a zero-length array.

getArgumentValue

public java.lang.String **getArgumentValue**(java.lang.String name)

Gets the value of the specified argument.
Parameters:
name - The name of the argument.
Returns:
The value of the argument.
Throws:
java.lang.IllegalArgumentException - if name does not match one of the argument names specified for this action response.
See Also:
getArgumentNames()

org.ocap.hn.upnp.common

Interface UPnPAdvertisedDevice

All Superinterfaces:
UPnPDevice
All Known Subinterfaces:
UPnPClientDevice

```
public interface UPnPAdvertisedDevice
extends UPnPDevice
```

This interface represents a UPnP device as it is advertised on a particular network. It provides the data constituting the device, portions of which depend on the network interface on which it is advertised. Corresponds to the information carried in the UPnP device description document.

Method Summary

`UPnPAdvertisedDeviceIcon[]`	**`getAdvertisedIcons`**`()` Gets the icons of this device.
`UPnPAdvertisedService[]`	**`getAdvertisedServices`**`()` Gets the services supported by this device.
`UPnPAdvertisedDevice[]`	**`getEmbeddedAdvertisedDevices`**`()` Gets the embedded devices for this UPnP Device.
`java.net.InetAddress`	**`getInetAddress`**`()` Returns the IP address from which this device was advertised.
`java.lang.String`	**`getPresentationURL`**`()` Gets the UPnP presentation page URL of this device.
`java.lang.String`	**`getURLBase`**`()` Reports the base URL for all relative URLs of this device.
`org.w3c.dom.Document`	**`getXML`**`()` Gets the device description document in XML.

Methods inherited from interface org.ocap.hn.upnp.common.UPnPDevice
`getDeviceType, getFriendlyName, getManufacturer, getManufacturerURL, getModelDescription, getModelName, getModelNumber, getModelURL, getSerialNumber, getSpecVersion, getUDN, getUPC, isRootDevice`

Method Detail

getEmbeddedAdvertisedDevices

`UPnPAdvertisedDevice[]` **`getEmbeddedAdvertisedDevices`**`()`

Gets the embedded devices for this UPnP Device.

Returns:

The embedded devices for this device. If this device has no embedded devices, returns a zero length array. Returns only the next level of embedded devices, not recursing through embedded devices for subsequent levels of embedded devices.

getAdvertisedIcons

`UPnPAdvertisedDeviceIcon[] getAdvertisedIcons()`

Gets the icons of this device. This returned array is derived from the icon elements within the `iconList` element of a device description. If the iconList element in the device description is empty or not present, returns a zero length array.

Returns:
The icons that the device declares.

getAdvertisedServices

`UPnPAdvertisedService[] getAdvertisedServices()`

Gets the services supported by this device. Does not return services held in embedded devices.

Returns:
The services supported by this device. If the serviceList element in the device description is empty, this method returns a zero length array.

getInetAddress

`java.net.InetAddress getInetAddress()`

Returns the IP address from which this device was advertised.

Returns:
an InetAddress representing this device's IP address.

getPresentationURL

`java.lang.String getPresentationURL()`

Gets the UPnP presentation page URL of this device. This value is taken from the value of the presentationURL element within a device description.

If the presentationURL is empty or not present, returns the empty String.

Returns:
The presentationURL of this device.

getURLBase

`java.lang.String getURLBase()`

Reports the base URL for all relative URLs of this device. This value is obtained from the `URLBase` element within the device description document. If this is an embedded device, the `URLBase` element of the root device is returned.

If the `URLBase` property is not specified in the device description document, this method returns the URL from which the device description may be retrieved.

Returns:
The base URL for all relative URLs of this UPnP Device.

getXML

`org.w3c.dom.Document getXML()`

Gets the device description document in XML. The form of the document is defined by the UPnP Device Architecture specification.

For a root device, returns the document starting with the <?xml> node. For an embedded device, returns the sub-document starting with the <device> node of the embedded device. Returns the complete XML document from the level that is appropriate, including any embedded devices.

Returns:
The device description document.

org.ocap.hn.upnp.common

Interface UPnPAdvertisedDeviceIcon

All Superinterfaces:
UPnPDeviceIcon
All Known Subinterfaces:
UPnPClientDeviceIcon

```
public interface UPnPAdvertisedDeviceIcon
extends UPnPDeviceIcon
```

This interface represents a UPnP device icon as it is described on a particular network. It provides the data constituting a UPnP device icon, portions of which depend on the network interface on which it is advertised. Corresponds to the `icon` entry in the UPnP device description `iconList` element.

Method Summary

`java.lang.String`	**`getURL`**`()` Gets the URL for retrieval of this icon.

Methods inherited from interface org.ocap.hn.upnp.common.UPnPDeviceIcon
`getColorDepth, getHeight, getMimeType, getWidth`

Method Detail

getURL

`java.lang.String` **`getURL`**`()`

Gets the URL for retrieval of this icon.

Returns:
The URL used to retrieve this icon.

org.ocap.hn.upnp.common

Interface UPnPAdvertisedService

All Superinterfaces:
UPnPService
All Known Subinterfaces:
UPnPClientService

```
public interface UPnPAdvertisedService
extends UPnPService
```

This interface represents a UPnP service as it is advertised on a particular network. It provides the data constituting a UPnP service, portions of which depend on the network interface on which it is advertised. Corresponds to the information carried in the UPnP service description document plus service-specific data from the UPnP device description document.

Method Summary

UPnPAdvertisedStateVariable	**getAdvertisedStateVariable**(java.lang.String stateVariableName) Gets a UPnP state variable from the UPnP description of this service.
UPnPAdvertisedStateVariable[]	**getAdvertisedStateVariables**() Gets all of the UPnP state variables supported by this service.
java.lang.String	**getControlURL**() Gets the UPnP controlURL of this service.
java.lang.String	**getEventSubURL**() Gets the UPnP eventSubURL of this service.
java.lang.String	**getSCPDURL**() Gets the UPnP SCPDURL of this service.
org.w3c.dom.Document	**getXML**() Gets the service description document (SCPD document) in XML.

Methods inherited from interface org.ocap.hn.upnp.common.UPnPService
getAction, getActions, getServiceId, getServiceType, getSpecVersion

Method Detail

getControlURL

java.lang.String **getControlURL**()

Gets the UPnP controlURL of this service. This value is taken from the value of the controlURL element within the device description.

Returns:
The URL used by a control point to invoke actions on this service.

getEventSubURL

`java.lang.String` **`getEventSubURL`**`()`

Gets the UPnP eventSubURL of this service. This value is taken from the value of the `eventSubURL` element within a device description. If this service does not have eventing, the value returned is the empty string.

Returns:

The URL used by a control point to subscribe to evented state variables.

getSCPDURL

`java.lang.String` **`getSCPDURL`**`()`

Gets the UPnP SCPDURL of this service. This value is taken from the value of the `SCPDURL` element within a device description.

Returns:

The URL used to retrieve the service description of this service.

getAdvertisedStateVariable

`UPnPAdvertisedStateVariable`
`getAdvertisedStateVariable``(java.lang.String stateVariableName)`

Gets a UPnP state variable from the UPnP description of this service. Supported state variable names are provided by a UPnP device in the name element of each stateVariable element in a device service description.

Parameters:

`stateVariableName` - The name of the state variable to get.

Returns:

The state variable corresponding to the `stateVariableName` parameter.

Throws:

`java.lang.IllegalArgumentException` - if the `stateVariableName` does not match a state variable name in this service.

getAdvertisedStateVariables

`UPnPAdvertisedStateVariable[]` **`getAdvertisedStateVariables`**`()`

Gets all of the UPnP state variables supported by this service. UPnP state variable information is taken from the `stateVariable` elements in the UPnP service description.

Returns:

The UPnP state variables supported by this service. If the service has no state variables, returns a zero-length array.

getXML

`org.w3c.dom.Document` **`getXML`**`()`

Gets the service description document (SCPD document) in XML. The form of the document is defined by the UPnP Device Architecture specification.

Returns:

The service description document.

org.ocap.hn.upnp.common

Interface UPnPAdvertisedStateVariable

All Superinterfaces:
UPnPStateVariable
All Known Subinterfaces:
UPnPClientStateVariable

```
public interface UPnPAdvertisedStateVariable
extends UPnPStateVariable
```

This interface represents a UPnP state variable as it is described on a particular network. Corresponds to the `stateVariable` entry in the UPnP service description `serviceStateTable` element.

Method Summary

Methods inherited from interface org.ocap.hn.upnp.common.UPnPStateVariable
`getAllowedValues, getDataType, getDefaultValue, getMaximumValue, getMinimumValue, getName, getStepValue, isEvented`

org.ocap.hn.upnp.common

Interface UPnPDevice

All Known Subinterfaces:
UPnPAdvertisedDevice, UPnPClientDevice, UPnPManagedDevice

```
public interface UPnPDevice
```

This interface is an abstract representation of a UPnP device. It provides the data constituting a UPnP device that is independent of the network interface on which the device is advertised.

Method Summary

java.lang.String	**getDeviceType**() Gets the UPnP deviceType of this device.
java.lang.String	**getFriendlyName**() Gets the UPnP "friendly name" of this device.
java.lang.String	**getManufacturer**() Gets the UPnP manufacturer of this device.
java.lang.String	**getManufacturerURL**() Gets the UPnP manufacturer URL of this device.
java.lang.String	**getModelDescription**() Gets the UPnP model description of this device.
java.lang.String	**getModelName**() Gets the UPnP model name of this device.
java.lang.String	**getModelNumber**() Gets the UPnP model number of this device.
java.lang.String	**getModelURL**() Gets the UPnP model URL of this device.
java.lang.String	**getSerialNumber**() Gets the UPnP serial number of this device.
java.lang.String	**getSpecVersion**() Gets the UPnP specVersion major and minor values of this UPnP device, or of the root UPnP device containing this device.
java.lang.String	**getUDN**() Gets the UPnP Unique Device Name of this device.
java.lang.String	**getUPC**() Gets the UPnP Universal Product Code of this device.
boolean	**isRootDevice**() Reports whether this UPnP device is a UPnP root device.

Method Detail

getDeviceType

```
java.lang.String getDeviceType()
```

Gets the UPnP deviceType of this device. This value is taken from the value of the `deviceType` element within the device description.

Returns:

The type of this device. If the `deviceType` is empty or not present, returns the empty String.

getFriendlyName

`java.lang.String` **`getFriendlyName`**`()`

Gets the UPnP "friendly name" of this device. This value is taken from the value of the `friendlyName` element within the device description.

Returns:

The `friendlyName` of this device. If the `friendlyName` is empty or not present, returns the empty String.

getManufacturer

`java.lang.String` **`getManufacturer`**`()`

Gets the UPnP manufacturer of this device. This value is taken from the value of the `manufacturer` element within the device description.

Returns:

The manufacturer of this device. If the manufacturer is empty or not present, returns the empty String.

getManufacturerURL

`java.lang.String` **`getManufacturerURL`**`()`

Gets the UPnP manufacturer URL of this device. This value is taken from the value of the `manufacturerURL` element within the device description. If the manufacturerURL is empty or not present, returns the empty String.

Returns:

The manufacturerURL of this device.

getModelDescription

`java.lang.String` **`getModelDescription`**`()`

Gets the UPnP model description of this device. This value is taken from the value of the `modelDescription` element within the device description. If the modelDescription is empty or not present, returns the empty String.

Returns:

The modelDescription of this device.

getModelName

`java.lang.String` **`getModelName`**`()`

Gets the UPnP model name of this device. This value is taken from the value of the `modelName` element within the device description.

Returns:

The `modelName` of this device. If the `modelName` is empty or not present, returns the empty String.

getModelNumber

`java.lang.String` **`getModelNumber`**`()`

Gets the UPnP model number of this device. This value is taken from the value of the `modelNumber` element within the device description. If the modelNumber is empty or not present, returns the empty String.

Returns:

The modelNumber of this device.

getModelURL

`java.lang.String` **`getModelURL`**`()`

Gets the UPnP model URL of this device. This value is taken from the value of the `modelURL` element within the device description. If the modelURL is empty or not present, returns the empty String.

Returns:

The modelURL of this device.

getSerialNumber

`java.lang.String` **`getSerialNumber`**`()`

Gets the UPnP serial number of this device. This value is taken from the value of the `serialNumber` element within the device description. If the serialNumber is empty or not present, returns the empty String.

Returns:

The serialNumber of this device.

getSpecVersion

`java.lang.String` **`getSpecVersion`**`()`

Gets the UPnP specVersion major and minor values of this UPnP device, or of the root UPnP device containing this device. This value is taken from the value of the major and minor sub elements of the `specVersion` element within the device description. The format of the returned String is the <major> value, followed by '.', followed by the <minor> value.

Returns:

The UPnP specVersion of this device.

getUDN

`java.lang.String` **`getUDN`**`()`

Gets the UPnP Unique Device Name of this device. This value is taken from the value of the `UDN` element within the device description.

Returns:

The UDN of this device. If the UDN is empty or not present, returns the empty String.

getUPC

`java.lang.String` **`getUPC`**`()`

Gets the UPnP Universal Product Code of this device. This value is taken from the value of the `UPC` element within the device description. If the UPC is empty or not present, returns the empty String.

Returns:

The UPC of this device.

isRootDevice

`boolean` **`isRootDevice`**`()`

Reports whether this UPnP device is a UPnP root device.

Returns:

true if this UPnP device represents a root device, false if not.

org.ocap.hn.upnp.common

Interface UPnPDeviceIcon

All Known Subinterfaces:
UPnPAdvertisedDeviceIcon, UPnPClientDeviceIcon
All Known Implementing Classes:
UPnPManagedDeviceIcon

```
public interface UPnPDeviceIcon
```

This interfaces is an abstract representation of a UPnP device icon. It provides the data constituting a device icon that is independent of the network interface on which it has been advertised.

Method Summary

int	**getColorDepth**() Gets the color depth of this icon in bits.
int	**getHeight**() Gets the height of this icon in pixels.
java.lang.String	**getMimeType**() Gets the mimetype for this UPnPDeviceIcon.
int	**getWidth**() Gets the width of this icon in pixels.

Method Detail

getColorDepth

```
int getColorDepth()
```

Gets the color depth of this icon in bits.
Returns:
The color depth of the icon in bits.

getHeight

```
int getHeight()
```

Gets the height of this icon in pixels.
Returns:
The height of the icon in pixels.

getMimeType

```
java.lang.String getMimeType()
```

Gets the mimetype for this UPnPDeviceIcon. For UPnPDeviceIcons conforming to UPnP Device Architecture 1.0, this should be of the form image/xxxx where xxxx is the specific image subtype.
Returns:
The mimetype string for this icon.

getWidth

`int` **`getWidth`**`()`

Gets the width of this icon in pixels.

Returns:

The width of the icon in pixels.

org.ocap.hn.upnp.common

Class UPnPErrorResponse

```
java.lang.Object
  └org.ocap.hn.upnp.common.UPnPResponse
      └org.ocap.hn.upnp.common.UPnPErrorResponse
```

```
public class UPnPErrorResponse
extends UPnPResponse
```

The class represents a response to an unsuccessfully completed UPnP action, where the server responded with HTTP code 500 (Internal Server Error). It carries the UPnP errorCode and errorDescription. Instances of this class are constructed by the `UPnPActionHandler` on a UPnP server, and are passed to a client in the `UPnPActionResponseHandler`.

Constructor Summary

`UPnPErrorResponse``(int errorCode, java.lang.String errorDescription, UPnPActionInvocation action)` Construct the instance.

Method Summary

`int`	**`getErrorCode`**`()` Get the error code.
`java.lang.String`	**`getErrorDescription`**`()` Get the error description.

Methods inherited from class org.ocap.hn.upnp.common.UPnPResponse
`getActionInvocation, getHTTPResponseCode`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

UPnPErrorResponse

```
public UPnPErrorResponse(int errorCode,
                         java.lang.String errorDescription,
                         UPnPActionInvocation action)
```

Construct the instance.

Parameters:

`errorCode` - The UPnP errorCode to be used for this error response.

`errorDescription` - The UPnP errorDescription to be used for this error response.

`action` - The action invocation that this error response relates to.

Method Detail

getErrorCode

`public int` **`getErrorCode`**`()`

Get the error code.

Returns:

The error code for this error response.

getErrorDescription

`public java.lang.String` **`getErrorDescription`**`()`

Get the error description.

Returns:

The error description for this error response.

org.ocap.hn.upnp.common

Class UPnPGeneralErrorResponse

```
java.lang.Object
  └org.ocap.hn.upnp.common.UPnPResponse
      └org.ocap.hn.upnp.common.UPnPGeneralErrorResponse
```

```
public class UPnPGeneralErrorResponse
extends UPnPResponse
```

The class represents a response to an unsuccessfully completed UPnP action, where the server either did not respond, or responded with an HTTP error code other than 500 (Internal Server Error). Instances of this class are constructed by the stack by the UPnPActionHandler, and are passed to a client in the `UPnPActionResponseHandler`.

Field Summary

`static int`	**`NETWORK_ERROR`** Indicates that the action was not able to be invoked due to a network error (could not reach server, or other non-timeout network error).
`static int`	**`NETWORK_TIMEOUT`** Indicates that the server did not respond.

Constructor Summary

`UPnPGeneralErrorResponse``(int errorCode, UPnPActionInvocation action)` Construct the instance.

Method Summary

`int`	**`getErrorCode`**`()` Get the error code.

Methods inherited from class org.ocap.hn.upnp.common.UPnPResponse
`getActionInvocation, getHTTPResponseCode`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Field Detail

NETWORK_TIMEOUT

```
public static final int NETWORK_TIMEOUT
```

Indicates that the server did not respond.

See Also:

Constant Field Values

NETWORK_ERROR

`public static final int` **`NETWORK_ERROR`**

Indicates that the action was not able to be invoked due to a network error (could not reach server, or other non-timeout network error).

See Also:

Constant Field Values

Constructor Detail

UPnPGeneralErrorResponse

```
public UPnPGeneralErrorResponse(int errorCode,
                                UPnPActionInvocation action)
```

Construct the instance.

Parameters:

`errorCode` - The error code that this general error response is to contain.

`action` - The `UPnPActionInvocation` that this general error response is in response to.

Method Detail

getErrorCode

`public int` **`getErrorCode`**`()`

Get the error code.

Returns:

The error code for this error response.

org.ocap.hn.upnp.common

Interface UPnPIncomingMessageHandler

```
public interface UPnPIncomingMessageHandler
```

This interface represents an incoming message handler that can monitor and modify any incoming messages to the UPnP stack. All messages targeting the UPnP stack go through the handler (if the handler is registered). This includes advertisements, action responses, device, service and icon retrieval responses on a client, UPnP action invocations, subscription requests, device searches, and device, service and icon retrieval requests on a server.

Method Summary

UPnPMessage	**handleIncomingMessage**(java.net.InetSocketAddress address, byte[] incomingMessage, UPnPIncomingMessageHandler defaultHandler) Handles an incoming message.

Method Detail

handleIncomingMessage

```
UPnPMessage handleIncomingMessage(java.net.InetSocketAddress address,
                                  byte[] incomingMessage,
                                  UPnPIncomingMessageHandler defaultHandler)
```

Handles an incoming message. The primary responsibility is to parse the incoming byte array and produce an XML document representing the incoming content. An application-provided `UPnPIncomingMessageHandler` may invoke the default, stack-provided message handler via the specified `defaultHandler`. The handler may also cause the incoming message to be discarded by returning `null`; subsequent processing SHALL continue as if the message had never been received.

Note that if the `UPnPMessage` returned by this method contains an HTTP `CONTENT-LENGTH` header, its value should describe the length of the raw XML data *before* it is parsed into the XML document reported by `UPnPMessage.getXML()`. See `UPnPMessage.getHeaders()`.

Parameters:
`address` - InetSocketAddress representing the network interface and port on which the message was received.
`incomingMessage` - The incoming UPnP message data, including any HTTP headers.
`defaultHandler` - The default stack-provided incoming message handler. If this `UPnPIncomingMessageHandler` is the default incoming message handler, this parameter SHALL be ignored.
Returns:
The UPnP message to be passed up the stack. The handler can cause the incoming message to be discarded by returning null.

org.ocap.hn.upnp.common

Class UPnPMessage

```
java.lang.Object
  └org.ocap.hn.upnp.common.UPnPMessage
```

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

The class represents a UPnP message comprising an HTTP start line, zero or more headers and an optional XML document.

Constructor Summary

UPnPMessage(java.lang.String startLine, java.lang.String[] headers, org.w3c.dom.Document xml) Public constructor for the message.

Method Summary

java.lang.String[]	**getHeaders**() Reports the headers from the message, including all HTTP headers but excluding trailing CR/LF characters.
java.lang.String	**getStartLine**() Reports the HTTP start line, excluding trailing CR/LF characters.
org.w3c.dom.Document	**getXML**() Gets the XML from the message.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

UPnPMessage

```
public UPnPMessage(java.lang.String startLine,
                   java.lang.String[] headers,
                   org.w3c.dom.Document xml)
```

Public constructor for the message.

Parameters:

`startLine` - The HTTP start line, excluding trailing CR/LF characters. May be an empty string.

`headers` - The HTTP or other headers that this instance is to contain, excluding trailing CF/LF characters and the blank line that follows HTTP headers. The contents of the `headers` parameter are copied into the resulting `UPnPMessage` object. May be a zero-length array.

`xml` - The XML document that this instance is to contain. May be null.

Throws:

`java.lang.NullPointerException` - if `startLine`, `headers`, or any of the array elements within `headers` is `null`.

Method Detail

getStartLine

`public java.lang.String` **`getStartLine`**`()`

Reports the HTTP start line, excluding trailing CR/LF characters.

Returns:

The HTTP start line. If the UPnPMessage includes no start line, returns the empty string.

getHeaders

`public java.lang.String[]` **`getHeaders`**`()`

Reports the headers from the message, including all HTTP headers but excluding trailing CR/LF characters. The blank line following the last HTTP header is not included in the array.

Note that if the message includes an HTTP `CONTENT-LENGTH` header, its value describes the length of the raw XML data carried in the UPnP message body, not the size of the XML document provided by `getXML()`.

Returns:

An array containing a copy of the header lines contained in the `UPnPMessage` object. If the `UPnPMessage` has no headers, returns a zero-length array.

See Also:

`UPnPIncomingMessageHandler`, `UPnPOutgoingMessageHandler`

getXML

`public org.w3c.dom.Document` **`getXML`**`()`

Gets the XML from the message.

Returns:

The XML document of the message. May be null if the message contained no XML document.

org.ocap.hn.upnp.common

Interface UPnPOutgoingMessageHandler

```
public interface UPnPOutgoingMessageHandler
```

This interface represents an outgoing message handler that can monitor and modify any outgoing messages from the UPnP stack. All messages originating from the UPnP stack go through the handler (if the handler is registered). This includes advertisements, action responses and device, service and icon retrieval responses on a server, UPnP action invocations, subscription requests, device searches and device, service and icon retrieval requests on a client.

Method Summary

byte[]	**handleOutgoingMessage**(java.net.InetSocketAddress address, UPnPMessage message, UPnPOutgoingMessageHandler defaultHandler) Handles an outgoing message.

Method Detail

handleOutgoingMessage

```
byte[] handleOutgoingMessage(java.net.InetSocketAddress address,
                             UPnPMessage message,
                             UPnPOutgoingMessageHandler defaultHandler)
```

Handles an outgoing message. The primary responsibility is to process the provided `UPnPMessage` object and produce a composite byte array for the outbound message that complies with the UPnP Device Architecture specification. An application-provided `UPnPOutgoingMessageHandler` may invoke the default, stack-provided message handler via the specified `defaultHandler`. The handler may also cause the outgoing message to be discarded by returning `null`; the message SHALL NOT be sent, and subsequent processing SHALL continue as if any expected response to the message had never been received.

Note that if the `UPnPMessage` provided to this method contains an HTTP `CONTENT-LENGTH` header, its value is undefined. The handler must supply the correct value after "stringifying" the XML document provided by `UPnPMessage.getXML()` into XML data to be carried in returned byte array. See `UPnPMessage.getHeaders()`.

Parameters:
`address` - The InetSocketAddress to which the message is to be sent.
`message` - The UPnP message that is to be sent.
`defaultHandler` - The default stack-provided outgoing message handler. If this `UPnPOutgoingMessageHandler` is the default outgoing message handler, this parameter SHALL be ignored.

Returns:
Composite output byte array containing HTTP start line, headers, and body of the message. No further processing or parsing is performed by the stack on the output byte array prior to transmission. The handler can cause the outgoing message to be discarded by returning null.

org.ocap.hn.upnp.common

Class UPnPResponse

```
java.lang.Object
  └org.ocap.hn.upnp.common.UPnPResponse
```

Direct Known Subclasses:

UPnPActionResponse, UPnPErrorResponse, UPnPGeneralErrorResponse

```
public abstract class UPnPResponse
extends java.lang.Object
```

This class represents a response to a UPnP action.

Constructor Summary

protected	**UPnPResponse**() Protected constructor for the instance.

Method Summary

UPnPActionInvocation	**getActionInvocation**() Gets the UPnPActionInvocation that the response is for.
int	**getHTTPResponseCode**() Gets the HTTP response code from the response.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

UPnPResponse

protected **UPnPResponse**()

Protected constructor for the instance.

Method Detail

getHTTPResponseCode

public int **getHTTPResponseCode**()

Gets the HTTP response code from the response.

Returns:

The HTTP response code associated with the response, such as 200 (OK) or 500 (Internal Server Error).

getActionInvocation

public UPnPActionInvocation **getActionInvocation**()

Gets the UPnPActionInvocation that the response is for.

Returns:
The UPnP action invocation that prompted this response.

org.ocap.hn.upnp.common

Interface UPnPService

All Known Subinterfaces:

UPnPAdvertisedService, UPnPClientService, UPnPManagedService

```
public interface UPnPService
```

This interface is an abstract representation of a UPnP service. It provides the data constituting a UPnP service that is independent of the network interface on which it has been advertised.

Method Summary

UPnPAction	**getAction**(java.lang.String actionName) Gets the named action from this service.
UPnPAction[]	**getActions**() Gets the actions that can be used with this service.
java.lang.String	**getServiceId**() Gets the UPnP serviceId of this service.
java.lang.String	**getServiceType**() Gets the UPnP serviceType of this service.
java.lang.String	**getSpecVersion**() Gets the UPnP specVersion major and minor values of this service.

Method Detail

getAction

```
UPnPAction getAction(java.lang.String actionName)
```

Gets the named action from this service.

Parameters:

`actionName` - The name of the UPnPAction to retrieve.

Returns:

The UPnPAction object from this service with the matched name.

Throws:

`java.lang.IllegalArgumentException` - if the `actionName` does not match an action name in this service.

getActions

```
UPnPAction[] getActions()
```

Gets the actions that can be used with this service.

Returns:

An array of `UPnPAction`s. If the service has no actions, returns an zero-length array.

getServiceId

```
java.lang.String getServiceId()
```

Gets the UPnP serviceId of this service. This value is taken from the value of the `serviceId` element within the device description.

Returns:
The serviceId of this service.

getServiceType

`java.lang.String` **`getServiceType`**`()`

Gets the UPnP serviceType of this service. This value is taken from the value of the `serviceType` element within the device description.

Returns:
The type of this service.

getSpecVersion

`java.lang.String` **`getSpecVersion`**`()`

Gets the UPnP specVersion major and minor values of this service. This value is taken from the value of the major and minor sub-elements of the `specVersion` element within the service description. The format of the returned String is the <major> value, followed by '.', followed by the <minor> value.

Returns:
The UPnP specVersion of this service.

org.ocap.hn.upnp.common

Interface UPnPStateVariable

All Known Subinterfaces:

UPnPAdvertisedStateVariable, UPnPClientStateVariable, UPnPManagedStateVariable

```
public interface UPnPStateVariable
```

This interface is an abstract representation of a UPnP state variable. It provides the data constituting a state variable that is independent of the network interface on which it has been advertised.

Method Summary

`java.lang.String[]`	**`getAllowedValues`**`()` Gets the allowed values for this UPnP state variable.
`java.lang.String`	**`getDataType`**`()` Gets the data type of this UPnP state variable.
`java.lang.String`	**`getDefaultValue`**`()` Reports the default value of this UPnP state variable.
`java.lang.String`	**`getMaximumValue`**`()` Gets the allowedValueRange maximum value of this UPnP state variable.
`java.lang.String`	**`getMinimumValue`**`()` Gets the allowedValueRange minimum value for this UPnP state variable.
`java.lang.String`	**`getName`**`()` Gets the name of this UPnP state variable.
`java.lang.String`	**`getStepValue`**`()` Gets the allowedValueRange step value for this UPnP state variable.
`boolean`	**`isEvented`**`()` Indicates if this state variable is evented.

Method Detail

getAllowedValues

`java.lang.String[] getAllowedValues()`

Gets the allowed values for this UPnP state variable. The value returned is formatted per the UPnP Device Architecture specification, service description, `allowedValueList` element definition. If the `UPnPStateVariable` does not have an allowedValueList specified, returns zero length array.

Returns:

An array containing the allowed values for this state variable. Each element in the array contains the value of one allowedValue element in the allowedValueList. The array has the same order as the allowedValueList element.

getDefaultValue

`java.lang.String getDefaultValue()`

Reports the default value of this UPnP state variable. This value is taken from the `defaultValue` element in the UPnP service description that defines this state variable.

Returns:
The default value of this state variable. Returns an empty string if the variable does not have a defaultValue.

getMaximumValue

`java.lang.String` **`getMaximumValue`**`()`

Gets the allowedValueRange maximum value of this UPnP state variable. The value returned is formatted per the UPnP Device Architecture specification, service description, `allowedValueRange` maximum element definition.

Returns:
A `String` containing the maximum allowed value for this state variable. Returns an empty string if the variable does not have an allowedValueRange.

getMinimumValue

`java.lang.String` **`getMinimumValue`**`()`

Gets the allowedValueRange minimum value for this UPnP state variable. The value returned is formatted per the UPnP Device Architecture specification, service description, `allowedValueRange` minimum element definition.

Returns:
A `String` containing the minimum allowed value for this state variable. Returns an empty string if the variable does not have an allowedValueRange.

getName

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

Gets the name of this UPnP state variable. This value is taken from the `name` element of the UPnP service description `stateVariable` element.

Returns:
The name of the state variable.

getDataType

`java.lang.String` **`getDataType`**`()`

Gets the data type of this UPnP state variable. The value returned is formatted per the UPnP Device Architecture specification, service description, `dataType` element definition.

Returns:
The data type of the state variable.

getStepValue

`java.lang.String` **`getStepValue`**`()`

Gets the allowedValueRange step value for this UPnP state variable. The value returned is formatted per the UPnP Device Architecture specification, service description, `allowedValueRange step` element definition.

Note that if the `step` element is omitted and the data type of the state variable is an integer, the step value is considered to be 1.

Returns:
A `String` containing the `step` value for this state variable. Returns an empty string if service description of this variable does not specify a `step` value.

isEvented

`boolean` **`isEvented`**`()`

Indicates if this state variable is evented. The value is taken from the `sendEvents` attribute in the UPnP service description that defines this state variable.

Returns:

True if this UPnP state variable is evented, otherwise returns false.

Package org.ocap.hn.upnp.server

Provides UPnP server functionality, permitting management of devices and services in the local Host device.

See:

Description

Interface Summary

UPnPActionHandler	This class represents an action handler an application can use to receive action requests and respond to them.
UPnPManagedDevice	This interface represents a locally hosted UPnP device created when a privileged application uses `UPnPDeviceManager.createDevice` to create a device.
UPnPManagedDeviceListener	This interface represents a listener to UPnP local (server) device availability on a home network.
UPnPManagedService	This interface provides the server representation of a UPnP service created when a privileged application uses the `UPnPDeviceManager` in the local host.
UPnPManagedStateVariable	This interface provides the server representation of a UPnP state variable.
UPnPStateVariableHandler	This interface represents a handler for the evented UPnP state variables on a service.

Class Summary

UPnPDeviceManager	This class represents a manager that can create devices and services for devices.
UPnPManagedDeviceIcon	This class represents a UPnP Device Icon with associated binary data for a `UPnPManagedDevice`.

Package org.ocap.hn.upnp.server Description

Provides UPnP server functionality, permitting management of devices and services in the local Host device.

The UPnP Device Manager class (`UPnPDeviceManager`) provides the ability for an application to create devices and cause the Host device to expose them using the UPnP discovery process. Application created devices are UPnP compliant and may contain services with actions and state variables. An application uses the `UPnPDeviceManager.createDevice` method to create a device and passes in a `UPnPManagedDevice` object that contains a hierarchy of objects that represents all of the device information including service (`UPnPManagedService`) objects. The object hierarchy matches the UPnP device architecture definition, e.g. a device contains sub-devices, and services.

An application can set itself as an action handler by calling the `UPnPManagedService.setActionHandler(UPnPActionHandler)` method. The application will be notified when an action request is received for the managed service. The application will create a response and return it to the Host implementation which will send the UPnP response. Similarly, an application can set itself as a state variable handler by calling the `UPnPManagedService.setHandler(UPnPManagedStateVariableHandler)` method. The application will be notified when a control point subscribes to or unsubscribes from the service or has requested the value of a state variable through the QueryStateVariable action.

org.ocap.hn.upnp.server

Interface UPnPActionHandler

```
public interface UPnPActionHandler
```

This class represents an action handler an application can use to receive action requests and respond to them. A `UPnPActionHandler` must be provided with the service description in a call to `UPnPDeviceManager.createService()`, or in a call to `UPnPManagedService.setActionHandler()`.

Method Summary

`void`	**`notifyActionHandlerReplaced`**`(UPnPActionHandler replacement)` Notifies an application registered as an action handler that it has been replaced by another `UPnPActionHandler`.
`UPnPResponse`	**`notifyActionReceived`**`(UPnPActionInvocation action)` Notifies an application registered as an action handler that an action invocation has been received for the `UPnPManagedService` it is associated with.

Method Detail

notifyActionReceived

`UPnPResponse` **`notifyActionReceived`**`(UPnPActionInvocation action)`

Notifies an application registered as an action handler that an action invocation has been received for the `UPnPManagedService` it is associated with. The handler must respond with a properly-formed `UPnPResponse`, per the UPnP Device Architecture specification.

Parameters:

`action` - The UPnP action invocation received.

Returns:

The response to the action: either a `UPnPActionResponse` or a `UPnPErrorResponse`. A null return indicates the action was not handled.

notifyActionHandlerReplaced

`void` **`notifyActionHandlerReplaced`**`(UPnPActionHandler replacement)`

Notifies an application registered as an action handler that it has been replaced by another `UPnPActionHandler`.

Parameters:

`replacement` - The replacement UPnPActionHandler.

org.ocap.hn.upnp.server

Class UPnPDeviceManager

```
java.lang.Object
  └org.ocap.hn.upnp.server.UPnPDeviceManager
```

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

This class represents a manager that can create devices and services for devices. Creating a device does not cause corresponding UPnP device advertisement until the `UPnPManagedDevice.sendByeBye()` method is called followed by a call to `UPnPManagedDevice.sendAlive()`.

Constructor Summary

protected	**UPnPDeviceManager**() Construct the instance.

Method Summary

void	**addDeviceListener**(UPnPManagedDeviceListener listener) Adds a listener for additions or removals of locally hosted server devices.
UPnPManagedDevice	**createDevice**(UPnPManagedDevice parent, java.io.InputStream description, UPnPManagedDeviceIcon[] icons) Creates a UPnP device in the local host.
UPnPManagedDevice[]	**getDevices**() Gets locally hosted UPnPManagedDevices.
UPnPManagedDevice[]	**getDevicesByServiceType**(java.lang.String type) Gets a server representation of any UPnPManagedDevices containing a service of the specified type, advertised by this host.
UPnPManagedDevice[]	**getDevicesByType**(java.lang.String type) Gets a server representation of all UPnP devices of the specified type advertised by this host.
UPnPManagedDevice[]	**getDevicesByUDN**(java.lang.String UDN) Gets a server representation of any UPnPManagedDevices of the specified UDN advertised by this host.
static UPnPDeviceManager	**getInstance**() Obtain the local UPnP device manager.
void	**removeDeviceListener**(UPnPManagedDeviceListener listener) Removes a previously registered UPnPManagedDeviceListener.
void	**setIncomingMessageHandler**(UPnPIncomingMessageHandler inHandler) Sets a message handler for incoming messages (advertisements, evented state variables, action responses, device and service descriptions).

Method Summary

void	**setOutgoingMessageHandler** (UPnPOutgoingMessageHandler outHandler) Sets a message handler for outgoing messages (action invocations, subscription requests, device and service retrievals).

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

UPnPDeviceManager

protected **UPnPDeviceManager**()

Construct the instance.

Method Detail

getInstance

public static UPnPDeviceManager **getInstance**()

Obtain the local UPnP device manager.

Returns:

The singleton UPnPDeviceManager.

Throws:

java.lang.UnsupportedOperationException - if this implementation does not support UPnP server-side operations.

getDevices

public UPnPManagedDevice[] **getDevices**()

Gets locally hosted UPnPManagedDevices.

Returns:

UPnPManagedDevice representations of UPnP root devices local to the host. Each element in the array of UPnPManagedDevices returned represents one root device exposed by the local host through UPnP advertisement. It does NOT return any UPnPManagedDevices which are not currently advertised on the home network. If there are no UPnPManagedDevices to return, returns a zero-length array.

getDevicesByType

public UPnPManagedDevice[] **getDevicesByType**(java.lang.String type)

Gets a server representation of all UPnP devices of the specified type advertised by this host. This does not cause a search to take place, but simply returns the currently known devices.

Parameters:

type - The type of devices to return. Of the form urn:schemas-upnp-org:device:deviceType:v where deviceType is replaced with a type specific to the device being requested, and v is a version specifier as defined in UPnP Device Architecture.

Returns:

The UPnPManagedDevices advertised on this host matching the type specified, of the specified version or lower version number. Each element in the array of UPnPManagedDevices returned represents one device advertised on the local host. If no devices matching the type are found, returns a zero-length array.

getDevicesByUDN

```
public UPnPManagedDevice[] getDevicesByUDN(java.lang.String UDN)
```

Gets a server representation of any UPnPManagedDevices of the specified UDN advertised by this host. This does not cause a search to take place, but simply returns the currently known devices.

Note that normally a UDN is unique and would return a single device. While it is not valid to have multiple UPnPManagedDevices with the same UDN, the stack does not enforce this and consequently there is the potential to return more than one matching UPnPManagedDevice

Parameters:
UDN - The UDN of the devices to return.
Returns:
The UPnP devices advertised on this host matching the UDN specified. Each element in the array of UPnPManagedDevices returned represents one device advertised by the local host. If no devices matching the UDN are found, returns a zero-length array.

getDevicesByServiceType

```
public UPnPManagedDevice[] getDevicesByServiceType(java.lang.String type)
```

Gets a server representation of any UPnPManagedDevices containing a service of the specified type, advertised by this host. This does not cause a search to take place, but simply returns the currently known devices.

Parameters:
type - The type of service to use in determining which devices to return. Of the form urn:schemas-upnp-org:service:serviceType:v where serviceType is replaced with a type specific to the service being requested, and v is a version specifier as defined in UPnP Device Architecture.
Returns:
The UPnPManagedDevices advertised by this host containing a service matching the type specified, of the specified version or lower version number. Each element in the array of UPnPManagedDevices returned represents one device advertised by the local host. Returns only devices directly containing a service of the matching type, not devices where only their embedded devices contain a service of the matching type. If no devices matching the criteria are found, returns a zero-length array.

createDevice

```
public UPnPManagedDevice createDevice(UPnPManagedDevice parent,
                                      java.io.InputStream description,
                                      UPnPManagedDeviceIcon[] icons)
                               throws java.io.IOException,
                                      java.lang.SecurityException
```

Creates a UPnP device in the local host. The object created implements the UPnPManagedDevice interface.

The description parameter applies to this device only and does not contain embedded device descriptions. Embedded devices are created when this method is called with the parent parameter referencing a UPnPManagedDevice object and a root device is created when the parent parameter is null.

When the parent is not null this device is added to the parent as an embedded device and the parent getManagedEmbeddedDevices method return value will include the new embedded device.

The `description` parameter must not include any serviceList or service elements. Services are added to the device description by calling `UPnPManagedDevice.createService(java.lang.String, java.io.InputStream, org.ocap.hn.upnp.server.UPnPActionHandler)` on the returned `UPnPManagedDevice`.

Since the stack does not provide any persistence of the device description, it is the responsibility of the application to provide the UDN element within the device description, and implement any persistence requirements for UDN as described by the UPnP Device architecture or other related specifications.

The `description` parameter must not include any iconList or icon elements. Icons are added to the device description based upon the passed `icons` parameter, or through calling setIcons() on the returned `UPnPManagedDevice`.

Within the description, the contents of the URLBase element (if present) are ignored and will be replaced by the stack if the element is present. As the URLBase element is optional, the stack will not add the element if it was absent from the passed description.

All other URL-type elements of the device description are unmodified. It is the responsibility of the application to handle requests for the manufacturerURL, modelURL and presentationURL where these are relative URLs as defined in UPnP Device Architecture.

Parameters:
`parent` - The parent device of this device. If the parent parameter is null then the device to be created is a root device, otherwise it is created as an embedded device of the parent. The device is advertised as defined by the UPnP Device Architecture specification.
`description` - The device description as defined by the UPnP Device Architecture specification. The `InputStream` format is an XML document representing the description.
`icons` - The icons to be associated with this device. Each icon in the array is copied into the resulting `UPnPManagedDevice`; subsequent calls to `UPnPManagedDevice.getIcons()` will return different instances than those specified by the `icons` parameter. May be a zero length array to create a device with no icons.
Returns:
The UPnPManagedDevice created from the description. Null is returned if the host platform does not support device advertisement.
Throws:
`java.lang.IllegalArgumentException` - if the `description` parameter does not comply with a device description as defined by the UPnP Device Architecture specification, or if the `description` parameter includes service descriptions, or if one or more of the services specified is already associated with another UPnPManagedDevice.
`java.io.IOException` - if an I/O error occurs on the description.
`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

addDeviceListener

`public void` **`addDeviceListener`**`(UPnPManagedDeviceListener listener)`

Adds a listener for additions or removals of locally hosted server devices. Each `UPnPManagedDeviceListener` is notified when a UPnPManagedDevice on the local host is added to or removed from a home network through calling the UPnPManagedDevice sendAlive() or sendByeBye() methods.

Adding a listener which is the same instance as a previously added (and not removed) listener has no effect.

Parameters:
`listener` - The listener to add.

removeDeviceListener

```
public void removeDeviceListener(UPnPManagedDeviceListener listener)
```

Removes a previously registered UPnPManagedDeviceListener.

Parameters:

`listener` - The listener to remove.

setIncomingMessageHandler

```
public void setIncomingMessageHandler(UPnPIncomingMessageHandler inHandler)
                               throws java.lang.SecurityException
```

Sets a message handler for incoming messages (advertisements, evented state variables, action responses, device and service descriptions). Calls to set the message handler replace any prior incoming message handler.

A message handler may be removed by passing null as the inHandler. In the absence of a registered message handler the stack will parse the incoming messages.

If the application-provided handler throws any exceptions during execution, the stack will attempt to process the message with the default (stack-provided) handler.

Parameters:

`inHandler` - The incoming message handler to set.

Throws:

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

setOutgoingMessageHandler

```
public void setOutgoingMessageHandler(UPnPOutgoingMessageHandler outHandler)
                               throws java.lang.SecurityException
```

Sets a message handler for outgoing messages (action invocations, subscription requests, device and service retrievals). Calls to set the message handler replace any prior outgoing message handler.

A message handler may be removed by passing null as the outHandler. In the absence of a registered message handler the stack will process the outgoing messages.

If the application-provided handler throws any exceptions during execution, the stack will attempt to process the message with the default (stack-provided) handler.

Parameters:

`outHandler` - The outgoing message handler to set.

Throws:

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

org.ocap.hn.upnp.server

Interface UPnPManagedDevice

All Superinterfaces:
UPnPDevice

```
public interface UPnPManagedDevice
extends UPnPDevice
```

This interface represents a locally hosted UPnP device created when a privileged application uses `UPnPDeviceManager.createDevice` to create a device.

The UPnPManagedDevice corresponds to a single UPnP server device with a single set of handlers and a single state. This server may be visible on multiple interfaces and/or multiple IP addresses, depending upon the list of InetAddresses that are associated with this server device. The network representations of this device (`UPnPAdvertisedDevice`), one per `InetAddress`, are available for query via the `getAdvertisedDevices()` method.

Updating a device causes corresponding UPnP device byebye/alive pairs to be sent. To institute multiple changes without multiple removals/advertisements, the application should call the `sendByeBye()` method prior to the first change, calling `sendAlive()` following the last change.

Method Summary

UPnPManagedService	**createService**(java.lang.String serviceType, java.io.InputStream description, UPnPActionHandler handler) Creates a UPnP service associated with this device.
UPnPAdvertisedDevice[]	**getAdvertisedDevices**() Gets the representations of this device on the network interfaces on which is it advertised.
java.lang.String	**getDeviceType**() Gets the UPnP deviceType of this device.
UPnPManagedDevice[]	**getEmbeddedDevices**() Gets any `UPnPManagedDevice` embedded devices for this `UPnPManagedDevice`.
java.lang.String	**getFriendlyName**() Gets the UPnP "friendly name" of this device.
UPnPManagedDeviceIcon[]	**getIcons**() Gets the icons for this device.
java.net.InetAddress[]	**getInetAddresses**() Gets the `InetAddress`es that this `UPnPManagedDevice` is associated with.
java.lang.String	**getManufacturer**() Gets the UPnP manufacturer of this device.
java.lang.String	**getManufacturerURL**() Gets the UPnP manufacturer URL of this device.

Method Summary

java.lang.String	**getModelDescription**() Gets the UPnP model description of this device.
java.lang.String	**getModelName**() Gets the UPnP model name of this device.
java.lang.String	**getModelNumber**() Gets the UPnP model number of this device.
java.lang.String	**getModelURL**() Gets the UPnP model URL of this device.
UPnPManagedDevice	**getParentDevice**() Returns the parent device of this UPnPManagedDevice.
java.lang.String	**getSerialNumber**() Gets the UPnP serial number of this device.
UPnPManagedService[]	**getServices**() Gets the services supported by this device.
java.lang.String	**getSpecVersion**() Gets the UPnP specVersion major and minor values of this UPnP device, or of the root UPnP device containing this device.
java.lang.String	**getUDN**() Gets the UPnP Unique Device Name of this device.
java.lang.String	**getUPC**() Gets the UPnP Universal Product Code of this device.
boolean	**isAlive**() Gets whether the UPnPManagedDevice is active.
boolean	**isRootDevice**() Reports whether this UPnP device is a UPnP root device.
void	**sendAlive**() Sends UPnP ssdp:alive messages from the physical device.
void	**sendByeBye**() Sends UPnP ssdp:byebye messages from this device.
boolean	**setDeviceType**(java.lang.String type) Sets the UPnP deviceType of this device.
boolean	**setFriendlyName**(java.lang.String friendlyName) Sets the UPnP friendlyName of this device.
UPnPManagedDeviceIcon[]	**setIcons**(UPnPManagedDeviceIcon[] icons) Sets the Icons for this device.
java.net.InetAddress[]	**setInetAddresses**(java.net.InetAddress[] addresses) Sets the InetAddresses that this UPnPManagedDevice is associated with.
boolean	**setManufacturer**(java.lang.String manufacturer) Sets the UPnP manufacturer of this device.
boolean	**setManufacturerURL**(java.lang.String manufacturerURL) Sets the UPnP manufacturer URL of this device.

Method Summary

boolean	**setModelDescription**(java.lang.String modelDescription) Sets the UPnP model description of this device.
boolean	**setModelName**(java.lang.String modelName) Sets the UPnP model name of this device.
boolean	**setModelNumber**(java.lang.String modelNumber) Sets the UPnP model number of this device.
boolean	**setModelURL**(java.lang.String modelURL) Sets the UPnP model URL of this device.
boolean	**setSerialNumber**(java.lang.String serialNumber) Sets the UPnP serial number of this device.
boolean	**setUDN**(java.lang.String UDN) Sets the UPnP Unique Device Number of this device.
boolean	**setUPC**(java.lang.String UPC) Sets the UPnP Universal Product Code of this device.

Method Detail

sendAlive

```
void sendAlive()
            throws java.lang.SecurityException
```

Sends UPnP ssdp:alive messages from the physical device. Sends a 3+2D+k series of alive messages as defined by the UPnP Device Architecture specification on each IP address and network interface associated with the `UPnPManagedDevice`. Enters the "alive" state once the first advertisements are initiated on the first InetAddress/interface.

If the `UPnPManagedDevice` has no associated `InetAddress`, no advertisements are sent and the device does not enter the "alive" state.

If the device is already in the "alive" state, the device still sends advertisements and remains in the "alive" state.

Throws:
`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").
See Also:
`isAlive()`

sendByeBye

```
void sendByeBye()
            throws java.lang.SecurityException
```

Sends UPnP ssdp:byebye messages from this device. Sends a byebye message for each device and service corresponding to alive messages that would be sent by the `sendAlive` method.

Once the last byebye message has been sent on the last `InetAddress`, the device enters the "not alive" state and is no longer discoverable via `UPnPDeviceManager.getDevices()`. At this point, the UPnP implementation holds no internal references to this `UPnPManagedDevice` object.

Throws:

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

See Also:
`isAlive()`

setInetAddresses

```
java.net.InetAddress[] setInetAddresses(java.net.InetAddress[] addresses)
                                        throws java.lang.SecurityException
```

Sets the InetAddresses that this `UPnPManagedDevice` is associated with. The device will send advertisements (and de-advertise) on the most appropriate interface for each of the addresses specified, and will respond to actions on each of those interfaces.

The passed array replaces any prior addresses that the device was associated with, including any stack-provided defaults. If the device is already active on one or more addresses, it must de-advertise itself at any addresses that are being removed, and advertise itself at any addresses that are being added.

The device description advertised on each interface will be identical with the exception of the URLs related to the device and its components, which will reflect the IP address on which the device is advertised.

Note that it is the responsibility of the application to monitor for changes in IP addresses that the host responds to, and to update the `UPnPManagedDevice` with any modifications using this method.

Parameters:
`addresses` - Array of `InetAddress` objects representing the IP addresses that this device is to advertise as. May be zero length.

Returns:
Array of prior addresses that were associated with the `UPnPManagedDevice`. If there were no prior addresses, returns a zero-length array.

Throws:
`java.lang.NullPointerException` - if `addresses` or any of its elements are `null`.
`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

See Also:
`getInetAddresses()`

getInetAddresses

```
java.net.InetAddress[] getInetAddresses()
```

Gets the `InetAddress`es that this `UPnPManagedDevice` is associated with. If `setInetAddresses(java.net.InetAddress[])` has not yet been called on this `UPnPManagedDevice`, this method will report the default InetAddresses provided by the stack.

Returns:
Array of `InetAddress` objects representing the network interfaces that this device is to advertise on. If the device has no associated network interfaces, returns a zero length array.

isAlive

```
boolean isAlive()
```

Gets whether the `UPnPManagedDevice` is active. The device is active if it has valid advertisements on one or more network interfaces.

While the device is in the "alive" state, the device:

- responds to SSDP discovery requests on the home network, conforming to the UPnP Device Architecture

- continues to automatically send SSDP "alive" messages conforming to the UPnP Device Architecture, prior to the expiration of prior alive messages
- may be discovered through `UPnPDeviceManager.getDevices()`.

Returns:
True if the `UPnPManagedDevice` has valid advertisements on one or more network interfaces, else false.

getAdvertisedDevices

`UPnPAdvertisedDevice[] getAdvertisedDevices()`

Gets the representations of this device on the network interfaces on which is it advertised. Since the UPnP device description contains network-dependent information, there can be multiple `UPnPAdvertisedDevice` objects associated with a single `UPnPManagedDevice`. The returned array corresponds to the set of UPnP device descriptions published for this UPnP device.

Returns:
The network representations of this `UPnPManagedDevice`. Returns a zero-length array if this device has not been advertised on a network interface.

getEmbeddedDevices

`UPnPManagedDevice[] getEmbeddedDevices()`

Gets any `UPnPManagedDevice` embedded devices for this `UPnPManagedDevice`. Returns an array of `UPnPManagedDevices`, one per device appearing in this device's `deviceList` element. Does not recurse through embedded devices of this device.

Returns:
The embedded devices for this device. If there are no embedded devices of this device then returns a zero length array.

getParentDevice

`UPnPManagedDevice getParentDevice()`

Returns the parent device of this `UPnPManagedDevice`.

Returns:
A UPnPManagedDevice representing this device's parent device. Returns null if this device has no parent.

setDeviceType

```
boolean setDeviceType(java.lang.String type)
                      throws java.lang.SecurityException
```

Sets the UPnP deviceType of this device. This value is placed in the `deviceType` element within the device description. Adds the element if it is not already present. If the device is currently published on one or more network addresses, this causes a byebye/alive cycle for this device on all published addresses.

Parameters:
`type` - The deviceType to be set for this device.

Returns:
True if the setting caused a byebye/alive cycle for this device, false if not.

Throws:
`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

setFriendlyName

```
boolean setFriendlyName(java.lang.String friendlyName)
                        throws java.lang.SecurityException
```

Sets the UPnP friendlyName of this device. This value is placed in the `friendlyName` element within the device description. Adds the element if it is not already present. If the device is currently published on one or more network addresses, this causes a byebye/alive cycle for this device on all published addresses.

Parameters:

`friendlyName` - The friendlyName to set for this device

Returns:

True if the setting caused a byebye/alive cycle for this device, false if not.

Throws:

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

setManufacturer

```
boolean setManufacturer(java.lang.String manufacturer)
                        throws java.lang.SecurityException
```

Sets the UPnP manufacturer of this device. This value is placed in the `manufacturer` element within the device description. Adds the element if it is not already present. If the device is currently published on one or more network addresses, this causes a byebye/alive cycle for this device on all published addresses.

Parameters:

`manufacturer` - The manufacturer to set for this device

Returns:

True if the setting caused a byebye/alive cycle for this device, false if not.

Throws:

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

setManufacturerURL

```
boolean setManufacturerURL(java.lang.String manufacturerURL)
                           throws java.lang.SecurityException
```

Sets the UPnP manufacturer URL of this device. This value is placed in the `manufacturerURL` element within the device description. Adds the element if it is not already present. If the device is currently published on one or more network addresses, this causes a byebye/alive cycle for this device on all published addresses.

Parameters:

`manufacturerURL` - The manufacturerURL to set for this device

Returns:

True if the setting caused a byebye/alive cycle for this device, false if not.

Throws:

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

setModelDescription

```
boolean setModelDescription(java.lang.String modelDescription)
                            throws java.lang.SecurityException
```

Sets the UPnP model description of this device. This value is placed in the `modelDescription` element within the device description. Adds the element if it is not already present. If the device is currently published on one or more network addresses, this causes a byebye/alive cycle for this device on all published addresses.

Parameters:

`modelDescription` - The modelDescription to set for this device

Returns:

True if the setting caused a byebye/alive cycle for this device, false if not.

Throws:

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

setModelName

```
boolean setModelName(java.lang.String modelName)
                     throws java.lang.SecurityException
```

Sets the UPnP model name of this device. This value is placed in the `modelName` element within the device description. Adds the element if it is not already present. If the device is currently published on one or more network addresses, this causes a byebye/alive cycle for this device on all published addresses.

Parameters:

`modelName` - The modelName to set for this device

Returns:

True if the setting caused a byebye/alive cycle for this device, false if not.

Throws:

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

setModelNumber

```
boolean setModelNumber(java.lang.String modelNumber)
                       throws java.lang.SecurityException
```

Sets the UPnP model number of this device. This value is placed in the `modelNumber` element within the device description. Adds the element if it is not already present. If the device is currently published on one or more network addresses, this causes a byebye/alive cycle for this device on all published addresses.

Parameters:

`modelNumber` - The modelNumber to set for this device.

Returns:

True if the setting caused a byebye/alive cycle for this device, false if not.

Throws:

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

setModelURL

```
boolean setModelURL(java.lang.String modelURL)
                    throws java.lang.SecurityException
```

Sets the UPnP model URL of this device. This value is placed in the `modelURL` element within the device description. Adds the element if it is not already present. If the device is currently published on one or more network addresses, this causes a byebye/alive cycle for this device on all published addresses.

Parameters:

`modelURL` - The modelURL to set for this device.

Returns:

True if the setting caused a byebye/alive cycle for this device, false if not.

Throws:

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

setSerialNumber

```
boolean setSerialNumber(java.lang.String serialNumber)
                        throws java.lang.SecurityException
```

Sets the UPnP serial number of this device. This value is placed in the `serialNumber` element within the device description. Adds the element if it is not already present. If the device is currently published on one or more network addresses, this causes a byebye/alive cycle for this device on all published addresses.

Parameters:
`serialNumber` - The serialNumber to set for this device.
Returns:
True if the setting caused a byebye/alive cycle for this device, false if not.
Throws:
`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

setUDN

```
boolean setUDN(java.lang.String UDN)
               throws java.lang.SecurityException
```

Sets the UPnP Unique Device Number of this device. This value is placed in the UDN element within the device description. Adds the element if it is not already present. If the device is currently published on one or more network addresses, this causes a byebye/alive cycle for this device on all published addresses.
Parameters:
`UDN` - The UDN to set for this device.
Returns:
True if the setting caused a byebye/alive cycle for this device, false if not.
Throws:
`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

setUPC

```
boolean setUPC(java.lang.String UPC)
               throws java.lang.SecurityException
```

Sets the UPnP Universal Product Code of this device. This value is placed in the UPC element within the device description. Adds the element if it is not already present. If the device is currently published on one or more network addresses, this causes a byebye/alive cycle for this device on all published addresses.
Parameters:
`UPC` - The UPC to set for this device.
Returns:
True if the setting caused a byebye/alive cycle for this device, false if not.
Throws:
`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

setIcons

```
UPnPManagedDeviceIcon[] setIcons(UPnPManagedDeviceIcon[] icons)
                                 throws java.lang.SecurityException
```

Sets the Icons for this device. The passed array of UPnPManagedDeviceIcons is used to generate the `iconList` sub-document of the device. Replaces any previous `iconList` for this device. If the array is of zero length, removes any prior `iconList` from the device description. If the device is currently published on one or more network addresses, this causes a byebye/alive cycle for this device on all published addresses.
Parameters:
`icons` - An array of UPnPManagedDeviceIcons to be used for the IconList of this device. Each icon in the array is copied into the `UPnPManagedDevice`; subsequent calls to `getIcons()` will return different instances than those specified by the `icons` parameter. May be a zero-length array to indicate no icons.
Returns:
An array of UPnPManagedDeviceIcons representing the previous device icons. If there were no previous device icons, returns a zero-length array.
Throws:

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

getIcons

```
UPnPManagedDeviceIcon[] getIcons()
```

Gets the icons for this device.

Returns:

An array of UPnPManagedDeviceIcons representing the current device icons. If no icons are present on the device, returns a zero-length array.

getServices

```
UPnPManagedService[] getServices()
```

Gets the services supported by this device. Does not return services contained in embedded devices of this device.

Returns:

An array of `UPnPManagedService` objects. If there are no services associated with this `UPnPManagedDevice`, this method returns a zero length array.

createService

```
UPnPManagedService createService(java.lang.String serviceType,
                                 java.io.InputStream description,
                                 UPnPActionHandler handler)
                                 throws java.io.IOException,
                                        java.lang.SecurityException
```

Creates a UPnP service associated with this device. If this device is currently advertised, causes a byebye/alive process to be followed. Note that the `USN` and `serviceId` elements are generated by the stack and not provided as parameters.

Parameters:

`serviceType` - The UPnP service type of the service to be populated in the serviceType element of the device description.

`description` - The service description as defined by the UPnP Device Architecture specification. The `InputStream` format is an XML document representing the description.

`handler` - The action handler that will handle action requests for the service.

Returns:

The UPnP service created from the description. Returns null if the host platform does not support service advertisement.

Throws:

`java.lang.IllegalArgumentException` - if the `description` parameter does not comply with a service description as defined by the UPnP Device Architecture specification.

`java.io.IOException` - if an I/O error occurs on the description.

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

getDeviceType

```
java.lang.String getDeviceType()
```

Gets the UPnP deviceType of this device. This value is taken from the value of the `deviceType` element within the device description.

Specified by:

`getDeviceType` in interface `UPnPDevice`

Returns:

The type of this device. If the device has been advertised, this method returns the same value as `getAdvertisedDevices()[0].getDeviceType()`.

getFriendlyName

`java.lang.String` **`getFriendlyName`**`()`

Gets the UPnP "friendly name" of this device. This value is taken from the value of the `friendlyName` element within the device description.

Specified by:
`getFriendlyName` in interface `UPnPDevice`

Returns:
The `friendlyName` of this device. If the device has been advertised, this method returns the same value as `getAdvertisedDevices()[0].getFriendlyName()`.

getManufacturer

`java.lang.String` **`getManufacturer`**`()`

Gets the UPnP manufacturer of this device. This value is taken from the value of the `manufacturer` element within the device description.

Specified by:
`getManufacturer` in interface `UPnPDevice`

Returns:
the manufacturer of this device. If the device has been advertised, this method returns the same value as `getAdvertisedDevices()[0].getManufacturer()`.

getManufacturerURL

`java.lang.String` **`getManufacturerURL`**`()`

Gets the UPnP manufacturer URL of this device. This value is taken from the value of the `manufacturerURL` element within the device description. If the manufacturerURL is empty or not present, returns the empty String.

Specified by:
`getManufacturerURL` in interface `UPnPDevice`

Returns:
The manufacturerURL of this device. If the device has been advertised, this method returns the same value as `getAdvertisedDevices()[0].getManufacturerURL()`.

getModelDescription

`java.lang.String` **`getModelDescription`**`()`

Gets the UPnP model description of this device. This value is taken from the value of the `modelDescription` element within the device description. If the modelDescription is empty or not present, returns the empty String.

Specified by:
`getModelDescription` in interface `UPnPDevice`

Returns:
The modelDescription of this device. If the device has been advertised, this method returns the same value as `getAdvertisedDevices()[0].getModelDescription()`.

getModelName

`java.lang.String` **`getModelName`**`()`

Gets the UPnP model name of this device. This value is taken from the value of the `modelName` element within the device description.

Specified by:
`getModelName` in interface `UPnPDevice`
Returns:
The `modelName` of this device. If the device has been advertised, this method returns the same value as `getAdvertisedDevices()[0].getModelName()`.

getModelNumber

`java.lang.String` **`getModelNumber`**`()`

Gets the UPnP model number of this device. This value is taken from the value of the `modelNumber` element within the device description. If the modelNumber is empty or not present, returns the empty String.
Specified by:
`getModelNumber` in interface `UPnPDevice`
Returns:
The modelNumber of this device. If the device has been advertised, this method returns the same value as `getAdvertisedDevices()[0].getModelNumber()`.

getModelURL

`java.lang.String` **`getModelURL`**`()`

Gets the UPnP model URL of this device. This value is taken from the value of the `modelURL` element within the device description. If the modelURL is empty or not present, returns the empty String.
Specified by:
`getModelURL` in interface `UPnPDevice`
Returns:
The modelURL of this device. If the device has been advertised, this method returns the same value as `getAdvertisedDevices()[0].getModelURL()`.

getSerialNumber

`java.lang.String` **`getSerialNumber`**`()`

Gets the UPnP serial number of this device. This value is taken from the value of the `serialNumber` element within the device description. If the serialNumber is empty or not present, returns the empty String.
Specified by:
`getSerialNumber` in interface `UPnPDevice`
Returns:
The serialNumber of this device. If the device has been advertised, this method returns the same value as `getAdvertisedDevices()[0].getSerialNumber()`.

getSpecVersion

`java.lang.String` **`getSpecVersion`**`()`

Gets the UPnP specVersion major and minor values of this UPnP device, or of the root UPnP device containing this device. This value is taken from the value of the major and minor sub elements of the `specVersion` element within the device description. The format of the returned String is the <major> value, followed by '.', followed by the <minor> value.
Specified by:
`getSpecVersion` in interface `UPnPDevice`
Returns:
The UPnP specVersion of this device. If the device has been advertised, this method returns the same value as `getAdvertisedDevices()[0].getSpecVersion()`.

getUDN

`java.lang.String` **`getUDN`**`()`

Gets the UPnP Unique Device Name of this device. This value is taken from the value of the `UDN` element within the device description.

Specified by:

`getUDN` in interface `UPnPDevice`

Returns:

The UDN of this device. If the device has been advertised, this method returns the same value as `getAdvertisedDevices()[0].getUDN()`.

getUPC

`java.lang.String` **`getUPC`**`()`

Gets the UPnP Universal Product Code of this device. This value is taken from the value of the `UPC` element within the device description. If the UPC is empty or not present, returns the empty String.

Specified by:

`getUPC` in interface `UPnPDevice`

Returns:

The UPC of this device. If the device has been advertised, this method returns the same value as `getAdvertisedDevices()[0].getUPC()`.

isRootDevice

`boolean` **`isRootDevice`**`()`

Reports whether this UPnP device is a UPnP root device.

Specified by:

`isRootDevice` in interface `UPnPDevice`

Returns:

true if this UPnP device represents a root device, false if not. If the device has been advertised, this method returns the same value as `getAdvertisedDevices()[0].isRootDevice()`.

org.ocap.hn.upnp.server

Class UPnPManagedDeviceIcon

```
java.lang.Object
  └org.ocap.hn.upnp.server.UPnPManagedDeviceIcon
```

All Implemented Interfaces:

UPnPDeviceIcon

```
public class UPnPManagedDeviceIcon
extends java.lang.Object
implements UPnPDeviceIcon
```

This class represents a UPnP Device Icon with associated binary data for a `UPnPManagedDevice`.

Constructor Summary

`UPnPManagedDeviceIcon``(java.lang.String mimetype, int width, int height, int colordepth, byte[] data)` Construct the instance.

Method Summary

`UPnPAdvertisedDeviceIcon[]`	**`getAdvertisedDeviceIcons`**`()` Gets the network representations of this `UPnPManagedDeviceIcon`.
`int`	**`getColorDepth`**`()` Gets the color depth of this icon in bits.
`byte[]`	**`getData`**`()` Gets the binary data that represents this icon.
`int`	**`getHeight`**`()` Gets the height of this icon in pixels.
`java.lang.String`	**`getMimeType`**`()` Gets the mimetype for this UPnPDeviceIcon.
`int`	**`getWidth`**`()` Gets the width of this icon in pixels.

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

UPnPManagedDeviceIcon

```
public UPnPManagedDeviceIcon(java.lang.String mimetype,
                             int width,
                             int height,
```

```
                        int colordepth,
                        byte[] data)
```

Construct the instance.

Parameters:

`mimetype` - The mimetype of this icon in the form image/xxxx.

`width` - The width of this icon in pixels.

`height` - The height of this icon in pixels.

`colordepth` - The color depth of this icon in bits.

`data` - A byte array containing the binary icon data. The contents of the array are copied into the resulting `UPnPManagedDeviceIcon` object. No validation is performed on the array, but it should contain data consistent with the other parameters to the constructor.

Method Detail

getData

`public byte[] **getData**()`

Gets the binary data that represents this icon.

Returns:

An array containing a copy of the binary icon data.

getAdvertisedDeviceIcons

`public UPnPAdvertisedDeviceIcon[] **getAdvertisedDeviceIcons**()`

Gets the network representations of this `UPnPManagedDeviceIcon`. Since the UPnP device description `iconList` element contains information specific to the network interface on which it is advertised, there can be multiple `UPnPAdvertisedDeviceIcon` objects associated with a single `UPnPManagedDeviceIcon`.

Returns:

The network representations of this `UPnPManagedDeviceIcon`. Returns a zero-length array if the corresponding UPnP device has not been advertised on a network interface.

getColorDepth

`public int **getColorDepth**()`

Gets the color depth of this icon in bits.

Specified by:

`getColorDepth` in interface `UPnPDeviceIcon`

Returns:

The color depth of the icon in bits. If the corresponding UPnP device has been advertised, this method returns the same value as `getAdvertisedDeviceIcons()[0].getColorDepth()`.

getHeight

`public int **getHeight**()`

Gets the height of this icon in pixels.

Specified by:

`getHeight` in interface `UPnPDeviceIcon`

Returns:

The height of the icon in pixels. If the corresponding UPnP device has been advertised, this method returns the same value as `getAdvertisedDeviceIcons()[0].getHeight()`.

getMimeType

`public java.lang.String` **`getMimeType`**`()`

Gets the mimetype for this UPnPDeviceIcon. For UPnPDeviceIcons conforming to UPnP Device Architecture 1.0, this should be of the form image/xxxx where xxxx is the specific image subtype.

Specified by:

`getMimeType` in interface `UPnPDeviceIcon`

Returns:

The mimetype string for this icon. If the corresponding UPnP device has been advertised, this method returns the same value as `getAdvertisedDeviceIcons()[0].getMimeType()`.

getWidth

`public int` **`getWidth`**`()`

Gets the width of this icon in pixels.

Specified by:

`getWidth` in interface `UPnPDeviceIcon`

Returns:

The width of the icon in pixels. If the corresponding UPnP device has been advertised, this method returns the same value as `getAdvertisedDeviceIcons()[0].getWidth()`.

org.ocap.hn.upnp.server

Interface UPnPManagedDeviceListener

All Superinterfaces:
java.util.EventListener

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

This interface represents a listener to UPnP local (server) device availability on a home network.

See Also:
`UPnPClientDeviceListener`

Method Summary

`void`	**`notifyDeviceAdded`**`(UPnPManagedDevice device)` Notifies the listener that a `UPnPManagedDevice` is about to be added to the home network by the local host.
`void`	**`notifyDeviceRemoved`**`(UPnPManagedDevice device)` Notifies the listener that a UPnPManagedDevice on the local host has been removed from the home network.

Method Detail

notifyDeviceAdded

`void` **`notifyDeviceAdded`**`(UPnPManagedDevice device)`

Notifies the listener that a `UPnPManagedDevice` is about to be added to the home network by the local host. This listener method is called in response to the `UPnPManagedDevice.sendAlive()` method, and prior to the device being advertised on the home network. This allows an application to prepare for the advertisement of the new device, at a point where it can modify the device prior to the advertisement taking place.

Parameters:
`device` - The `UPnPManagedDevice` that is about to be added.

notifyDeviceRemoved

`void` **`notifyDeviceRemoved`**`(UPnPManagedDevice device)`

Notifies the listener that a UPnPManagedDevice on the local host has been removed from the home network. This allows an application that is monitoring the managed devices to clean up after the device has been removed.

Parameters:
`device` - The `UPnPManagedDevice` that was removed.

org.ocap.hn.upnp.server

Interface UPnPManagedService

All Superinterfaces:
UPnPService

```
public interface UPnPManagedService
extends UPnPService
```

This interface provides the server representation of a UPnP service created when a privileged application uses the `UPnPDeviceManager` in the local host.

Method Summary

`UPnPAction`	**`getAction`**`(java.lang.String actionName)` Gets the named action from this service.
`UPnPActionHandler`	**`getActionHandler`**`()` Gets the current action handler for this service.
`UPnPAction[]`	**`getActions`**`()` Gets the actions that can be used with this service.
`UPnPAdvertisedService[]`	**`getAdvertisedServices`**`()` Gets the representations of this service on the network interfaces on which is it advertised.
`UPnPManagedDevice`	**`getDevice`**`()` Gets the UPnP device that this service is a part of.
`java.lang.String`	**`getServiceId`**`()` Gets the UPnP serviceId of this service.
`java.lang.String`	**`getServiceType`**`()` Gets the UPnP serviceType of this service.
`java.lang.String`	**`getSpecVersion`**`()` Gets the UPnP specVersion major and minor values of this service.
`UPnPManagedStateVariable[]`	**`getStateVariables`**`()` Gets the state variables that are part of this service definition.
`boolean`	**`getSubscribedStatus`**`()` Reports the subscription status of the service.
`void`	**`respondToQueries`**`(boolean respond)` Control whether the service responds to UPnP QueryStateVariable actions.
`UPnPActionHandler`	**`setActionHandler`**`(UPnPActionHandler handler)` Sets an action handler for this service, replacing any prior action handler.
`UPnPStateVariableHandler`	**`setHandler`**`(UPnPStateVariableHandler handler)` Sets a `UPnPStateVariableHandler` to this `UPnPManagedService`.

Method Detail

setActionHandler

```
UPnPActionHandler setActionHandler(UPnPActionHandler handler)
                                   throws java.lang.SecurityException
```

Sets an action handler for this service, replacing any prior action handler.

Parameters:
`handler` - UPnPActionHandler to be set for this managed service.

Returns:
Previous action handler, if any; null, if none.

Throws:
`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

See Also:
`UPnPActionHandler`

getActionHandler

```
UPnPActionHandler getActionHandler()
```

Gets the current action handler for this service. If no action server is registered, returns null.

Returns:
Current action handler, if any; null if none.

See Also:
`UPnPActionHandler`

respondToQueries

```
void respondToQueries(boolean respond)
                      throws java.lang.SecurityException
```

Control whether the service responds to UPnP QueryStateVariable actions.

If respond is true, the UPnP stack will respond to the QueryStateVariable action invocation with a value from any registered `UPnPManagedStateVariableHandler`, or with the most recently set value for the state value if no handler is registered, or with the default value of the state variable if no handler is registered and no calls to `UPnPManagedStateVariable.setValue()` have been made. If respond is false, responds with an error as defined by UPnP Device Architecture 1.0, with UPnPError errorCode of 401 (Invalid Action).

If respond is true and a request is received with a state variable name that is not part of this service, responds with an error as defined by UPnP Device Architecture 1.0, with UPnPError errorCode of 404 (Invalid Var).

Parameters:
`respond` - True to cause the stack to respond to QueryStateVariable actions, returning the current state variable value to the UPnP control point. False to refuse QueryStateVariable actions with UPnP errorCode of 401 (Invalid Action).

Throws:
`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

getAdvertisedServices

```
UPnPAdvertisedService[] getAdvertisedServices()
```

Gets the representations of this service on the network interfaces on which is it advertised. Since the UPnP device and service descriptions contain network-dependent information, there can be multiple `UPnPAdvertisedService` objects associated with a single `UPnPManagedService`.

Returns:

The network representations of this `UPnPManagedService`. Returns a zero-length array if the service has not been advertised on a network interface.

getStateVariables

`UPnPManagedStateVariable[] getStateVariables()`

Gets the state variables that are part of this service definition.

Returns:

The state variables that are part of this service definition. If this service defines no state variables, returns a zero-length array.

getDevice

`UPnPManagedDevice getDevice()`

Gets the UPnP device that this service is a part of.

Returns:

The device that this service is a part of.

setHandler

`UPnPStateVariableHandler setHandler(UPnPStateVariableHandler handler)`

Sets a `UPnPStateVariableHandler` to this `UPnPManagedService`. The handler provides the ability to respond dynamically to the QueryStateVariable action, and to be notified of subscription requests.

Only a single handler may be registered at any point in time. Subsequent requests to add a handler replace any existing handler.

Parameters:

`handler` - The handler to add. May be null, removing any previously set handler.

Returns:

The previously set UPnPStateVariableHandler, if any. Returns null if no prior handler set.

getSubscribedStatus

`boolean getSubscribedStatus()`

Reports the subscription status of the service.

Returns:

True if any control points are presently subscribed to this service.

getAction

`UPnPAction getAction(java.lang.String actionName)`

Gets the named action from this service.

Specified by:

`getAction` in interface `UPnPService`

Parameters:

`actionName` - The name of the UPnPAction to retrieve.

Returns:

The UPnPAction object from this service with the matched name. If the service has been advertised, this method returns the same value as `getAdvertisedServices()[0].getAction()`.

Throws:

`java.lang.IllegalArgumentException` - if the `actionName` does not match an action name in this service.

getActions

`UPnPAction[] getActions()`

Gets the actions that can be used with this service.

Specified by:

`getActions` in interface `UPnPService`

Returns:

An array of `UPnPAction`s. If the service has no actions, returns an zero-length array. If the service has been advertised, this method returns the same value as `getAdvertisedServices()[0].getActions()`.

getServiceId

`java.lang.String getServiceId()`

Gets the UPnP serviceId of this service. This value is taken from the value of the `serviceId` element within the device description.

Specified by:

`getServiceId` in interface `UPnPService`

Returns:

The serviceId of this service. If the service has been advertised, this method returns the same value as `getAdvertisedServices()[0].getServiceId()`.

getServiceType

`java.lang.String getServiceType()`

Gets the UPnP serviceType of this service. This value is taken from the value of the `serviceType` element within the device description.

Specified by:

`getServiceType` in interface `UPnPService`

Returns:

The type of this service. If the service has been advertised, this method returns the same value as `getAdvertisedServices()[0].getServiceType()`.

getSpecVersion

`java.lang.String getSpecVersion()`

Gets the UPnP specVersion major and minor values of this service. This value is taken from the value of the major and minor sub-elements of the `specVersion` element within the service description. The format of the returned String is the <major> value, followed by '.', followed by the <minor> value.

Specified by:

`getSpecVersion` in interface `UPnPService`

Returns:

The UPnP specVersion of this service. If the service has been advertised, this method returns the same value as `getAdvertisedServices()[0].getSpecVersion()`.

org.ocap.hn.upnp.server

Interface UPnPManagedStateVariable

All Superinterfaces:
UPnPStateVariable

```
public interface UPnPManagedStateVariable
extends UPnPStateVariable
```

This interface provides the server representation of a UPnP state variable.

Method Summary

`UPnPAdvertisedStateVariable[]`	**`getAdvertisedStateVariables`**`()` Gets the network representations of this `UPnPManagedStateVariable`.
`java.lang.String[]`	**`getAllowedValues`**`()` Gets the allowed values for this UPnP state variable.
`java.lang.String`	**`getDataType`**`()` Gets the data type of this UPnP state variable.
`java.lang.String`	**`getDefaultValue`**`()` Reports the default value of this UPnP state variable.
`java.lang.String`	**`getMaximumValue`**`()` Gets the allowedValueRange maximum value of this UPnP state variable.
`java.lang.String`	**`getMinimumValue`**`()` Gets the allowedValueRange minimum value for this UPnP state variable.
`int`	**`getModerationDelta`**`()` Gets the moderation delta of this state variable.
`int`	**`getModerationInterval`**`()` Gets the moderation interval of this state variable, in milliseconds.
`java.lang.String`	**`getName`**`()` Gets the name of this UPnP state variable.
`UPnPManagedService`	**`getService`**`()` Gets the service that this state variable is a member of.
`java.lang.String`	**`getStepValue`**`()` Gets the allowedValueRange step value for this UPnP state variable.
`java.lang.String`	**`getValue`**`()` Reports the current value of this `UPnPManagedStateVariable`.
`boolean`	**`isEvented`**`()` Indicates if this state variable is evented.

Method Summary

void	**setModerationDelta**(int delta) Sets the moderation delta of this state variable.
void	**setModerationInterval**(int interval) Sets the moderation interval of this state variable, in milliseconds.
void	**setValue**(java.lang.String value) Sets the value of this state variable.

Method Detail

setValue

```
void setValue(java.lang.String value)
              throws java.lang.SecurityException
```

Sets the value of this state variable. If the UPnP state variable corresponding to this `UPnPManagedStateVariable` is evented the host will send the appropriate UPnP event, subject to the moderation constraints set for this `UPnPManagedStateVariable`.

Parameters:

`value` - The new value of the state variable.

Throws:

`java.lang.IllegalArgumentException` - if `value` violates the constraints of this variable's `allowedValueList` or `allowedValueRange`, or does not conform to the required format of the state variable's `dataType` as indicated by the UPnP Device Architecture Specification.

`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

See Also:

`getAllowedValues()`, `getMinimumValue()`, `getMaximumValue()`, `getDataType()`

getValue

```
java.lang.String getValue()
```

Reports the current value of this `UPnPManagedStateVariable`. This method will return the most recent value specified by `setValue(java.lang.String)`, or the variable's default value if its value has not been set.

Returns:

The value of this `UPnPManagedStateVariable`.

setModerationInterval

```
void setModerationInterval(int interval)
                           throws java.lang.SecurityException
```

Sets the moderation interval of this state variable, in milliseconds. Corresponds to the UPnP DA 1.0 "maximumRate".

Parameters:

`interval` - New moderation interval of the state variable, in milliseconds. A value of zero indicates unmoderated.

Throws:

`java.lang.UnsupportedOperationException` - if the corresponding `UPnPStateVariable` is not evented.

`java.lang.IllegalArgumentException` - if `interval` is negative.
`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

getModerationInterval

`int` **`getModerationInterval`**`()`

Gets the moderation interval of this state variable, in milliseconds. Corresponds to the UPnP DA 1.0 "maximumRate".

Returns:
The current moderation interval for this `UPnPManagedStateVariable` in milliseconds. A value of zero indicates unmoderated.

Throws:
`java.lang.UnsupportedOperationException` - if the corresponding `UPnPStateVariable` is not evented.

setModerationDelta

`void` **`setModerationDelta`**`(int delta)`
`throws java.lang.SecurityException`

Sets the moderation delta of this state variable. Corresponds to the UPnP DA 1.0 "minimumDelta".

Parameters:
`delta` - New moderation delta of the state variable. A value of zero indicates unmoderated.

Throws:
`java.lang.UnsupportedOperationException` - if the state variable is not evented or if the data type of this variable is not numeric.
`java.lang.IllegalArgumentException` - if `delta` is negative.
`java.lang.SecurityException` - if the calling application has not been granted MonitorAppPermission("handler.homenetwork").

See Also:
`getDataType()`

getModerationDelta

`int` **`getModerationDelta`**`()`

Gets the moderation delta of this state variable. Corresponds to the UPnP DA 1.0 "minimumDelta".

Returns:
The current moderation delta for this `UPnPManagedStateVariable`. A value of zero indicates unmoderated.

Throws:
`java.lang.UnsupportedOperationException` - if the corresponding `UPnPStateVariable` is not evented.

getAdvertisedStateVariables

`UPnPAdvertisedStateVariable[]` **`getAdvertisedStateVariables`**`()`

Gets the network representations of this `UPnPManagedStateVariable`. Since the UPnP service description contains information specific to the network interface on which it is advertised, there can be multiple `UPnPAdvertisedStateVarable` objects associated with a single `UPnPManagedStateVariable`.

Returns:
The network representations of this `UPnPManagedStateVariable`. Returns a zero-length array if the corresponding UPnP service has not been advertised on a network interface.

getService

`UPnPManagedService` **`getService`**`()`

Gets the service that this state variable is a member of.

Returns:

The `UPnPManagedService` that this state variable is a member of.

getAllowedValues

`java.lang.String[]` **`getAllowedValues`**`()`

Gets the allowed values for this UPnP state variable. The value returned is formatted per the UPnP Device Architecture specification, service description, `allowedValueList` element definition. If the `UPnPStateVariable` does not have an allowedValueList specified, returns zero length array.

Specified by:

`getAllowedValues` in interface `UPnPStateVariable`

Returns:

An array containing the allowed values for this state variable. Each element in the array contains the value of one allowedValue element in the allowedValueList. The array has the same order as the allowedValueList element. If the corresponding UPnP service has been advertised, this method returns the same value as `getAdvertisedStateVariables()[0].getAllowedValues()`.

getDefaultValue

`java.lang.String` **`getDefaultValue`**`()`

Reports the default value of this UPnP state variable. This value is taken from the `defaultValue` element in the UPnP service description that defines this state variable.

Specified by:

`getDefaultValue` in interface `UPnPStateVariable`

Returns:

The default value of this state variable. Returns an empty string if the variable does not have a defaultValue. If the corresponding UPnP service has been advertised, this method returns the same value as `getAdvertisedStateVariables()[0].getDefaultValue()`.

getMaximumValue

`java.lang.String` **`getMaximumValue`**`()`

Gets the allowedValueRange maximum value of this UPnP state variable. The value returned is formatted per the UPnP Device Architecture specification, service description, `allowedValueRange` maximum element definition.

Specified by:

`getMaximumValue` in interface `UPnPStateVariable`

Returns:

A `String` containing the maximum allowed value for this state variable. Returns an empty string if the variable does not have an allowedValueRange. If the corresponding UPnP service has been advertised, this method returns the same value as `getAdvertisedStateVariables()[0].getMaximumValue()`.

getMinimumValue

`java.lang.String` **`getMinimumValue`**`()`

Gets the allowedValueRange minimum value for this UPnP state variable. The value returned is formatted per the UPnP Device Architecture specification, service description, `allowedValueRange` minimum element definition.

Specified by:

`getMinimumValue` in interface `UPnPStateVariable`

Returns:
A `String` containing the minimum allowed value for this state variable. Returns an empty string if the variable does not have an allowedValueRange. If the corresponding UPnP service has been advertised, this method returns the same value as
`getAdvertisedStateVariables()[0].getMinimumValue()`.

getName

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

Gets the name of this UPnP state variable. This value is taken from the `name` element of the UPnP service description `stateVariable` element.

Specified by:
`getName` in interface `UPnPStateVariable`

Returns:
The name of the state variable. If the corresponding UPnP service has been advertised, this method returns the same value as `getAdvertisedStateVariables()[0].getName()`.

getDataType

`java.lang.String` **`getDataType`**`()`

Gets the data type of this UPnP state variable. The value returned is formatted per the UPnP Device Architecture specification, service description, `dataType` element definition.

Specified by:
`getDataType` in interface `UPnPStateVariable`

Returns:
The data type of the state variable. If the corresponding UPnP service has been advertised, this method returns the same value as
`getAdvertisedStateVariables()[0].getStateVariableType()`.

getStepValue

`java.lang.String` **`getStepValue`**`()`

Gets the allowedValueRange step value for this UPnP state variable. The value returned is formatted per the UPnP Device Architecture specification, service description, `allowedValueRange step` element definition.

Note that if the `step` element is omitted and the data type of the state variable is an integer, the step value is considered to be 1.

Specified by:
`getStepValue` in interface `UPnPStateVariable`

Returns:
A `String` containing the `step` value for this state variable. Returns an empty string if service description of this variable does not specify a `step` value. If the corresponding UPnP service has been advertised, this method returns the same value as `getAdvertisedStateVariables()[0].getStepValue()`.

isEvented

`boolean` **`isEvented`**`()`

Indicates if this state variable is evented. The value is taken from the `sendEvents` attribute in the UPnP service description that defines this state variable.

Specified by:
`isEvented` in interface `UPnPStateVariable`

Returns:

True if this UPnP state variable is evented, otherwise returns false. If the corresponding UPnP service has been advertised, this method returns the same value as `getAdvertisedStateVariables()[0].isEvented()`.

org.ocap.hn.upnp.server

Interface UPnPStateVariableHandler

```
public interface UPnPStateVariableHandler
```

This interface represents a handler for the evented UPnP state variables on a service. The hander is called as a result of incoming subscription and query actions, and supplies the state variable values to be evented.

Method Summary

java.lang.String	**getValue**(UPnPManagedStateVariable variable) Notifies the listener that a control point has requested the value of a state variable through the QueryStateVariable action.
void	**notifySubscribed**(UPnPManagedService service) Notifies the listener that a control point has subscribed to state variable eventing on the specified service.
void	**notifyUnsubscribed**(UPnPManagedService service, int remainingSubs) Notifies the listener that a control point has successfully unsubscribed from state variable eventing on the specified service, or that a prior subscription has expired.

Method Detail

getValue

```
java.lang.String getValue(UPnPManagedStateVariable variable)
```

Notifies the listener that a control point has requested the value of a state variable through the QueryStateVariable action. The handler must return the current value of the requested state variable.

Parameters:

variable - The UPnP state variable that was queried.

Returns:

The current value of the state variable.

notifySubscribed

```
void notifySubscribed(UPnPManagedService service)
```

Notifies the listener that a control point has subscribed to state variable eventing on the specified service. This method is called subsequent to the transmission of subscription response message, but prior to the transmission of the initial event message. The eventing process blocks until this method returns, permitting the handler to set the initial values of the service's state variables as desired.

Parameters:

service - The UPnP service that was subscribed to.

notifyUnsubscribed

```
void notifyUnsubscribed(UPnPManagedService service,
                        int remainingSubs)
```

Notifies the listener that a control point has successfully unsubscribed from state variable eventing on the specified service, or that a prior subscription has expired. This method is called subsequent to the transmission of the unsubscription response message.

Parameters:
`service` - The UPnP service that was unsubscribed from.
`remainingSubs` - The number of remaining active subscriptions to this service.

Annex I Resource API

Package org.ocap.hn.resource

Interface Summary

NetResourceUsage	This interface represents a usage of resources on a specific home network activity.

org.ocap.hn.resource
Interface NetResourceUsage

All Superinterfaces:
ResourceUsage

```
public interface NetResourceUsage
extends ResourceUsage
```

This interface represents a usage of resources on a specific home network activity.

Field Summary

`static java.lang.String`	**`USAGE_TYPE_PRESENTATION`** Usage type is presentation.

Method Summary

`AppID`	**`getAppID`**`()` Returns `null`.
`java.net.InetAddress`	**`getInetAddress`**`()` Returns the network address of the device from which the request was originated for this usage.
`OcapLocator`	**`getOcapLocator`**`()` Gets the `OcapLocator` of the service associated with this usage.
`java.lang.String`	**`getUsageType`**`()` Gets the usage type associated with this usage.

Methods inherited from interface org.ocap.resource.ResourceUsage
`getResource, getResourceNames`

Field Detail

USAGE_TYPE_PRESENTATION

`static final java.lang.String` **`USAGE_TYPE_PRESENTATION`**
Usage type is presentation.
See Also:
Constant Field Values

Method Detail

getInetAddress

`java.net.InetAddress` **`getInetAddress`**`()`

Returns the network address of the device from which the request was originated for this usage.

Returns:

The network address of the device requesting resources.

getUsageType

`java.lang.String` **`getUsageType`**`()`

Gets the usage type associated with this usage.

This method is intended to eventually report one of multiple usage types; at present, it returns only `USAGE_TYPE_PRESENTATION`.

Returns:

Returns `USAGE_TYPE_PRESENTATION`.

getOcapLocator

`OcapLocator` **`getOcapLocator`**`()`

Gets the `OcapLocator` of the service associated with this usage.

Returns:

The `OcapLocator`.

getAppID

`AppID` **`getAppID`**`()`

Returns `null`.

Specified by:

`getAppID` in interface `ResourceUsage`

Returns:

`null`

Annex J Remote UI API

Package org.ocap.hn.ruihsrc

Class Summary	
RemoteUIServerManager	This class represents a Manager that can be used to configure the Remote User Interfaces published by the Host on a home network.

org.ocap.hn.ruihsrc
Class RemoteUIServerManager

```
java.lang.Object
  └org.ocap.hn.ruihsrc.RemoteUIServerManager
```

```
public abstract class RemoteUIServerManager
extends java.lang.Object
```

This class represents a Manager that can be used to configure the Remote User Interfaces published by the Host on a home network.

Constructor Summary	
protected	**RemoteUIServerManager**() Protected constructor not callable by applications.

Method Summary	
static RemoteUIServerManager	**getInstance**() Get the RemoteUIServerManager singleton.
abstract void	**setUIList**(java.io.InputStream uiList) Sets a remote user interface list.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

RemoteUIServerManager

protected **RemoteUIServerManager**()

Protected constructor not callable by applications.

Method Detail

setUIList

```
public abstract void setUIList(java.io.InputStream uiList)
                        throws java.io.IOException
```

Sets a remote user interface list. If the parameter is null any RUI's that were previously set are removed.

Parameters:

`uiList` - An `InputStream` representing the XML document for the UI List.

Throws:

`java.lang.IllegalArgumentException` - if the parameter is not null and the XML document is detected to be invalid.

`java.io.IOException` - if an I/O error occurs on the uiList

`java.lang.SecurityException` - if the calling application does not have MonitorAppPermission("handler.homenetwork")

getInstance

```
public static RemoteUIServerManager getInstance()
```

Get the RemoteUIServerManager singleton.

Returns:

The RemoteUIServerManager singleton.

Annex K Transformation API

Package org.ocap.hn.transformation

Interface Summary

Transformation	This interface represents a transformation from an input content format to an output content format.
TransformationListener	Listener interface for classes interested in getting notifications from the `TransformationManager`.

Class Summary

TransformationManager	This class is a singleton manager that can be used to get transformation capabilities of the Host device and to manage the content item transformation configuration.

org.ocap.hn.transformation
Interface Transformation

`public interface` **`Transformation`**

This interface represents a transformation from an input content format to an output content format. Instances implementing this interface are created by the `TransformationManager`.

Method Summary

`ContentFormat`	**`getInputContentFormat`**`()` Returns the input content format that can be transformed.
`ContentFormat`	**`getOutputContentFormat`**`()` Returns the output content format the input format can be transformed into.

Method Detail

getInputContentFormat

`ContentFormat` **`getInputContentFormat`**`()`

Returns the input content format that can be transformed.

Returns:

The input content format.

getOutputContentFormat

`ContentFormat` **`getOutputContentFormat`**`()`

Returns the output content format the input format can be transformed into. For content formats, that include video the method SHALL return an `OutputVideoContentFormat`.

Returns:

The output content format.

org.ocap.hn.transformation

Interface TransformationListener

All Superinterfaces:

java.util.EventListener

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

Listener interface for classes interested in getting notifications from the `TransformationManager`. Only one of the notify callbacks will be received for each (`ContentItem`, `Transformation`) tuple.

Field Summary

`static int`	**`REASON_CONTENTITEM_DELETED`** ReasonCode: The specific content item for which the transformation was requested has been deleted.
`static int`	**`REASON_NONMATCHING_INPUT_PROFILE`** ReasonCode: The content item native format isn't compatible with the requested transformation's input content profile.
`static int`	**`REASON_RESOURCE_UNAVAILABLE`** ReasonCode: Some resource was not available to create the transformation.
`static int`	**`REASON_UNKNOWN`** ReasonCode: Transformation was not successful due to unknown reason(s).

Method Summary

`void`	**`notifyTransformationFailed`**`(ContentItem contentItem, Transformation transformation, int reasonCode)` Callback indicating the content binary representation for the transformation could not be created.
`void`	**`notifyTransformationReady`**`(ContentItem contentItem, Transformation transformation)` Callback indicating the `ContentResource` for the transformation has been created.

Field Detail

REASON_UNKNOWN

`static final int` **`REASON_UNKNOWN`**

ReasonCode: Transformation was not successful due to unknown reason(s).

See Also:

Constant Field Values

REASON_RESOURCE_UNAVAILABLE

`static final int` **`REASON_RESOURCE_UNAVAILABLE`**

ReasonCode: Some resource was not available to create the transformation.

See Also:

Constant Field Values

REASON_CONTENTITEM_DELETED

`static final int` **`REASON_CONTENTITEM_DELETED`**

ReasonCode: The specific content item for which the transformation was requested has been deleted.

See Also:

Constant Field Values

REASON_NONMATCHING_INPUT_PROFILE

`static final int` **`REASON_NONMATCHING_INPUT_PROFILE`**

ReasonCode: The content item native format isn't compatible with the requested transformation's input content profile.

See Also:

Constant Field Values

Method Detail

notifyTransformationReady

```
void notifyTransformationReady(ContentItem contentItem,
                               Transformation transformation)
```

Callback indicating the `ContentResource` for the transformation has been created.

Parameters:

`contentItem` - affected contentItem

`transformation` - requested transformation on contentItem

notifyTransformationFailed

```
void notifyTransformationFailed(ContentItem contentItem,
                                Transformation transformation,
                                int reasonCode)
```

Callback indicating the content binary representation for the transformation could not be created.

Parameters:

`contentItem` - affected contentItem

`transformation` - requested transformation on contentItem

`reasonCode` - reason for the failure

org.ocap.hn.transformation

Class TransformationManager

```
java.lang.Object
  └org.ocap.hn.transformation.TransformationManager
```

```
public abstract class TransformationManager
extends java.lang.Object
```

This class is a singleton manager that can be used to get transformation capabilities of the Host device and to manage the content item transformation configuration. Transformation capabilities indicate the transformations from input content format to output content format that the device supports.

Constructor Summary

protected	**TransformationManager**() Constructor protected from erroneous application access.

Method Summary

abstract void	**addTransformationListener**(TransformationListener listener) Adds a TransformationListener to receive callbacks from the TransformationManager.
abstract Transformation[]	**getDefaultTransformations**() Returns the currently-set default transformation.
static TransformationManager	**getInstance**() Gets an instance of the TransformationManager.
abstract Transformation[]	**getSupportedTransformations**() Gets all of the transformation permutations the Host device supports.
abstract Transformation[]	**getTransformations**(ContentItem item) Returns the applied transformations for the content item.
abstract void	**removeTransformationListener**(TransformationListener listener) Removes the specified TransformationListener.
abstract Transformation[]	**setDefaultTransformations**(Transformation[] transformations) Sets the default transformations.
abstract void	**setTransformations**(ContentItem[] items) Applies the default transformations for a set of content items.
abstract void	**setTransformations**(ContentItem[] items, Transformation[] transformations) Applies specific transformations for a set of content items.

Method Summary

abstract void	**setTransformations**(Transformation[] transformations) Applies the transformations to all existing local content items that represent network operator content.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

TransformationManager

`protected` **`TransformationManager`**`()`

Constructor protected from erroneous application access.

Method Detail

getInstance

`public static TransformationManager` **`getInstance`**`()`

Gets an instance of the `TransformationManager`.

Throws:

`java.lang.SecurityException` - if the calling application has not been granted HomeNetPermission("contentmanagement").

addTransformationListener

`public abstract void`
`addTransformationListener``(TransformationListener listener)`

Adds a `TransformationListener` to receive callbacks from the `TransformationManager`. The `TransformationListener` will be notified whenever transformations are applied for the `ContentItem`. Subsequent calls to register the same listener will be ignored.

Parameters:

`listener` - The listener that will receive the callbacks

Throws:

`java.lang.IllegalArgumentException` - if the listener parameter is null.

removeTransformationListener

`public abstract void`
`removeTransformationListener``(TransformationListener listener)`

Removes the specified TransformationListener. If the listener specified is not registered or is null, then this method has no effect

Parameters:

`listener` - The listener to remove

getSupportedTransformations

`public abstract Transformation[]` **`getSupportedTransformations`**`()`

Gets all of the transformation permutations the Host device supports. See [OC-BUNDLE] for additional mapping of this method.

Returns:

Device supported transformations. If the device does not support transformations an empty array is returned.

setDefaultTransformations

```
public abstract Transformation[]
setDefaultTransformations(Transformation[] transformations)
```

Sets the default transformations. Default transformations will be applied to newly-created ContentItems only and certain calls to `TransformationManager`. A call to setDefaultTransformation over-rides any previously-set default transformations and passing an empty array disables any previously-set default transformations. See [OC-BUNDLE] for additional mapping of this method.

Parameters:

`transformations` - The new default transformations.

Returns:

The default transformations.

Throws:

`java.lang.IllegalArgumentException` - if the transformations parameter is null.

getDefaultTransformations

```
public abstract Transformation[] getDefaultTransformations()
```

Returns the currently-set default transformation.

Returns:

The default transformations.

getTransformations

```
public abstract Transformation[] getTransformations(ContentItem item)
```

Returns the applied transformations for the content item.

Parameters:

`item` - The content item.

Returns:

The applied transformations

Throws:

`java.lang.IllegalArgumentException` - if the item parameter is null.

setTransformations

```
public abstract void setTransformations(Transformation[] transformations)
```

Applies the transformations to all existing local content items that represent network operator content. A call to this method will remove any existing transformations before setting the transformations. See [OC-BUNDLE] for additional mapping of this method.

Parameters:

`transformations` - The array of transformations to be applied.

Throws:

`java.lang.IllegalArgumentException` - if the transformations parameter is null.

setTransformations

```
public abstract void setTransformations(ContentItem[] items)
```

Applies the default transformations for a set of content items. Configures metadata indicating content transformation support for each `ContentItem` in the items array parameter. If a content item in the array

parameter is not local or does not represent MSO local content it is skipped without change or notification. A call to this method will remove any existing transformations before setting the transformations. See [OC-BUNDLE] for additional mapping of this method.

Parameters:

`items` - The array of content items the transformation metadata will be configured in.

Throws:

`java.lang.IllegalArgumentException` - if the parameter is null or empty.

setTransformations

```
public abstract void setTransformations(ContentItem[] items,
                                        Transformation[] transformations)
```

Applies specific transformations for a set of content items. Configures metadata indicating content transformation support that matches any of the transformations in the transformations array parameter for each content item in the items array parameter. If a ContentItem in the array parameter is not local or does not represent MSO local content it is skipped without change or notification. A call to this method will remove any existing transformations before setting the transformations. See [OC-BUNDLE] for additional mapping of this method.

Parameters:

`items` - The array of content items the transformation metadata will be configured in.

`transformations` - The array of transformations to apply.

Throws:

`java.lang.IllegalArgumentException` - if items parameter is null or empty or the transformations parameter is null.

Appendix I Revision History

The following ECNs were incorporated into OC-SP-OCAP-HNEXT-I02-071220:

ECN	Date Accepted	Title of EC
OCAP-HNEXT-N-05.0844-1	12/19/05	System property identifying Home Networking extension
OCAP-HNEXT-N-06.0859-1	2/14/06	HN Mapping
OCAP-HNEXT-N-06.0865-1	2/7/06	Correction of System property name identifying Home Networking extension
OCAP-HNEXT-N-07.1079-2	9/25/07	OCAP Home Networking usability cleanup

The following ECNs were incorporated into OC-SP-OCAP-HNEXT-I03-080418:

ECN	Date Accepted	Title of EC
OCAP-HNEXT-N-08.1164-6	4/1/08	OCAP Home Networking Extension Phase II Version 1
OCAP-HNEXT-N-08.1241-1	4/15/08	UPnP Constants additions

The following ECNs were incorporated into OC-SP-OCAP-HNEXT-I04-091217:

ECN	Date Accepted	Title of EC
OCAP-HNEXT-N-08.1253-1	7/14/08	Home Networking Device InetAddress
OCAP-HNEXT-N-08.1261-2	7/14/08	HN Overlapping Methods
OCAP-HNEXT-N-08.1280-1	12/17/09	HN Security API
OCAP-HNEXT-N-08.1302-5	12/17/09	HN Authorization API
OCAP-HNEXT-N-08.1306-4	12/17/09	Miscellaneous HNEXT Phase II cleanup
OCAP-HNEXT-N-08.1333-2	12/17/09	Clarify the behavior of the method of NetRecordingEntry
OCAP-HNEXT-N-08.1336-2	12/17/09	Clarify the behavior of the method of ContentContainer
OCAP-HNEXT-N-08.1337-2	12/17/09	Remove ContentManagementNetModule interface
OCAP-HNEXT-N-08.1338-1	12/17/09	Add and clarify the methods of RecordingContentItem
OCAP-HNEXT-N-08.1339-1	12/17/09	Remove ContentDatabase class from HNEXT

ECN	Date Accepted	Title of EC
OCAP-HNEXT-N-08.1344-1	12/17/09	Clarify SRS related methods in OCAP HNExt
OCAP-HNEXT-N-08.1345-2	12/17/09	Remove a recording parameter from RecordingContentItem.requestSetMediaTime()
OCAP-HNEXT-N-08.1357-4	12/17/09	Home Networking Phase 2.0 API update
OCAP-HNEXT-N-08.1367-4	12/17/09	Add IllegalArgumentException conditions to RecordingNetModule
OCAP-HNEXT-N-09.1370-1	12/17/09	Modify action state definitions of NetActionEvent
OCAP-HNEXT-N-09.1384-2	12/17/09	Add Device.setProperties method
OCAP-HNEXT-N-09.1392-5	12/17/09	Remove HN Unchecked Exceptions
OCAP-HNEXT-N-09.1425-2	12/17/09	Update OCAP Reference
OCAP-HNEXT-N-09.1434-1	12/17/09	Corrections to make consistent use of scheduledStartDateTime
OCAP-HNEXT-N-09.1452-1	12/17/09	Fix inconsistencies in ObjectID handling between the APIs and UPnP behaviors
OCAP-HNEXT-N-09.1460-1	12/17/09	Fix AudioResource language APIs
OCAP-HNEXT-N-09.1462-1	12/17/09	Fix DatabaseQuery.or() throws clause
OCAP-HNEXT-N-09.1463-1	12/17/09	Correct Monitor App Permission Statement in 6.5.3
OCAP-HNEXT-N-09.1465-1	12/17/09	Clarify ContentContainer.deleteRecursive()
OCAP-HNEXT-N-09.1469-1	12/17/09	NetActionRequest.getError() clarification

The following ECNs were incorporated into OC-SP-OCAP-HNEXT-I05-100603:

ECN	Date Accepted	Title of EC
OCAP-HNEXT-N-09.1483-1	6/3/10	Remove Section 8
OCAP-HNEXT-N-10.1545-1	6/3/10	Home Networking Asset In Use Detection

The following ECNs were incorporated into OC-SP-OCAP-HNEXT-I06-110224:

ECN	Date Accepted	Title of EC
OCAP-HNEXT-N-10.1604-1	2/24/11	Correct signature of NetSecurityManager.revokeAuthorization
OCAP-HNEXT-N-10.1606-1	2/24/11	HN-EXT: Clarify type of spaceRequired
OCAP-HNEXT-N-10.1608-1	2/24/11	Content Metadata Caching clarification
OCAP-HNEXT-N-10.1631-2	2/24/11	ContentContainer API updates to clarify content deletion
OCAP-HNEXT-N-11.1637-2	2/24/11	Resolution of inter-spec contradictions about metadata property types
OCAP-HNEXT-N-11.1641-2	2/24/11	Multi-valued Metadata

The following ECNs were incorporated into OC-SP-OCAP-HNEXT-I07-110512:

ECN	Date Accepted	Title of EC
OCAP-HNEXT-N-11.1648-2	5/12/11	UPnP Diagnostics Feature
OCAP-HNEXT-N-11.1668-2	5/12/11	Reference edits for OpenCable bundle inclusion

The following ECNs were incorporated into OC-SP-OCAP-HNEXT-I08-120112:

ECN	Date Accepted	Title of EC
OCAP-HNEXT-N-10.1612-2	1/12/12	NetAuthorizationHandler2 API
OCAP-HNEXT-N-11.1693-3	1/12/12	Live Streaming to DLNA 1.5 Devices
OCAP-HNEXT-N-11.1720-2	1/12/12	NetResourceUsage API
OCAP-HNEXT-N-12.1746-1	1/12/12	ServiceResolutionHandler API

The following ECNs were incorporated into OC-SP-OCAP-HNEXT-I09-120531:

ECN	Date Accepted	Title of EC
OCAP-HNEXT-N-12.1776-1	5/31/12	Add VIDEO_ITEM_VPOP to HNExt
OCAP-HNEXT-N-12.1784-3	5/31/12	HTML5 RUI Support API
OCAP-HNEXT-N-12.1790-1	5/31/12	HTTP Request Resolution Handler

The following ECNs were incorporated into OC-SP-OCAP-HNEXT-I10-130418:

ECN	Author	Date Accepted	Title of EC
OCAP-HNEXT-N-12.1801-1	Michon	4/18/13	OCAP Application LAN Interface
OCAP-HNEXT-N-13.1815-3	Ladd	4/18/13	Transformation API
OCAP-HNEXT-N-13.1834-1	Yelekyathanhalli	4/18/13	Add VIDEO_ITEM_BROADCAST_VOD to ContentItem

The following ECNs were incorporated into OC-SP-OCAP-HNEXT-I11-130530:

ECN	Author	Date Accepted	Title of EC
OCAP-HNEXT-N-13.1828-2	Millard	5/30/13	NetManager NetList Enhancement
OCAP-HNEXT-N-13.1830-1	Ladd	5/30/13	Multiple IP Address Support
OCAP-HNEXT-N-13.1845-1	Millard	5/30/13	Return Value Clarification for ContentResource Subinterfaces
OCAP-HNEXT-N-13.1850-1	Pratt	5/30/13	ContentEntryFactory support for VIDEO_ITEM_BROADCAST_VOD