VNC AUTOMOTIVE

VNC® Network Advertiser SDK for Android

Reference Manual

Version 4.2.3.6776

http://automotive.realvnc.com/

The VNC Network Advertiser SDK allows you to create and manipulate VNC Network Advertiser objects.

See: Description

Packages

Package	Description
com.realvnc.networkadvertisersdk	The Android interface to the VNC Network Advertiser SDK.

The VNC Network Advertiser SDK allows you to create and manipulate VNC Network Advertiser objects. A VNC Network Advertiser uses UPnP to advertise RealVNC Mobile Solution VNC server on a network. RealVNC Mobile Solution VNC viewer applications can use the VNC Network Discoverer to discover this VNC server and connect to it.

Each VNC Network Advertiser object manages a UPnP device, which is used to advertise your server. This UPnP device also represents the device (e.g. a smartphone or tablet) on which your VNC server is running. When a VNC Network Discoverer requests connection details for your server, you shall be notified and must provide a VNC command string describing how a connection to your server may be made.

The Android interface to the SDK does not provide any user interface of its own.

Contents

Compiling your application

The VNC Android Network Advertiser SDK is provided as an Android "library project" which can be included in your application. Alternatively, it is possible to copy the libraries from the `sdk/libs/` directory into the `libs/` directory of your own Android project.

In order to use this library project, you should enable manifest merging and add a reference to the library project in your application's `project.properties` file. For example:

```
manifestmerger.enabled=true
android.library.reference.1=../VNCNetworkAdvertiserSDK
```

Alternatively, if copying the libraries directly, then you must declare the following permissions in your application's `AndroidManifest.xml` file:

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.CHANGE_WIFI_MULTICAST_STATE" />
```

Basic usage

Typically, you will want to allow a VNC Network Advertiser to run in the background, whilst the user is performing other tasks. To facilitate this, you should create an Android service and manage the VNC Network Advertiser from your Service object. This is not strictly required by the SDK, as a VNC Network Advertiser can run in any Android context, such as an activity. However, this is unlikely to be useful in practise. See the Android developer documentation for more information on services.

Call `VNCNetworkAdvertiserSDK.advertiserCreate(android.content.Context,VNCNetworkAdvertiserListener)` to create a VNC Network Advertiser object. You may reuse the same `VNCNetworkAdvertiser` object for a succession of sessions that require a similar configuration. There is no need to create a new object for each session.

When you create a VNC Network Advertiser object, you must set a `VNCNetworkAdvertiserListener` which will receive callbacks with information about advertiser events. `VNCNetworkAdvertiserListener` is an interface, which you could implement in your Service object. You must always implement this method:

> `VNCNetworkAdvertiserListener.globalServerCommandStringRequest(String)`, so that you are notified when a VNC Network Discoverer wishes to retrieve a VNC command string to allow its associated

VNC viewer to establish a connection to your VNC server.

You should also implement this method:

`VNCNetworkAdvertiserListener.error(int)`, so that you are notified when the UPnP device thread stops.

Licensing the SDK

In order to use the SDK, your application must provide each VNC Network Advertiser object with at least one license. Use `VNCNetworkAdvertiser.addLicense(InputStream)` or `VNCNetworkAdvertiser.addLicense(String)` to do this. Contact your RealVNC sales representative to obtain a license.

Configuring a VNC Network Advertiser

Use `VNCNetworkAdvertiser.setParameter(String,String)` to set any desired parameters on the VNC Network Advertiser. See `VNCNetworkAdvertiserParameter` for a full list of configuration options.

Use `VNCNetworkAdvertiser.setDeviceDetails(String,String,String,String,String,UUID,String)` to specify the basic details for the device that the VNC Network Advertiser is representing, such as the friendly name, manufacturer and model details. The UPnP Unique Device Name (UDN) is also set here. This information is used by the underlying UPnP device that the VNC Network Advertiser manages. This method must be called before `VNCNetworkAdvertiser.start()` can be used.

Use `VNCNetworkAdvertiser.setDeviceIdentity(VNCNetworkAdvertiserDeviceIdentity)` to specify identity information for the device that the VNC Network Advertiser is representing, such as the device's IMEI number or WiFi MAC address. This method must also be called before `VNCNetworkAdvertiser.start()` can be used.

Use `VNCNetworkAdvertiser.setGlobalServerDetails(VNCNetworkAdvertiserServerDetails,VNCNetworkAdvertiserIcon[])` to specify the details of the VNC server that you wish to advertise. This method must also be called before `VNCNetworkAdvertiser.start()` can be used. You may specify a series of icons that may be used by the VNC Network Discoverer to represent your server on a graphical display.

The UPnP device thread

Calling `VNCNetworkAdvertiser.start()` starts a new thread of execution. This thread is called the UPnP device thread.

Once started, the UPnP device thread continues to execute until one of the following occurs:

Your application calls `VNCNetworkAdvertiser.stop()` (including implicit calls to `VNCNetworkAdvertiser.stop()` made by `VNCNetworkAdvertiser.destroy()`.
An error occurs, which prevents the VNC Network Advertiser from continuing.

Before it exits, the UPnP device thread will always make exactly one call to your implementation of `VNCNetworkAdvertiserListener.error(int)`. This call will take place even if your application calls`VNCNetworkAdvertiser.stop()` or `VNCNetworkAdvertiser.destroy()`.

Interaction with your application

All SDK processing for each VNC Network Advertiser instance takes place within the UPnP device thread for that instance. This includes processing that your application requests by calling SDK APIs on VNC Network Advertiser instances. The majority of these SDK APIs place a command on a queue that the UPnP device thread services, and do not return to the caller until the command has been serviced by the UPnP device thread.

The only exception to this rule is `VNCNetworkAdvertiserListener.log(String,int,String)`, which the SDK may make from any thread, including calls made before the UPnP device thread is started. However, the SDK guarantees that no two threads will call `VNCNetworkAdvertiserListener.log(String,int,String)` at the same time, even if there are multiple advertiser instances.

This threading model greatly simplifies the interaction of the SDK with your application. It does, however, mean that there are two particular rules that your application must follow in order to avoid deadlock.

Don't call SDK API while holding locks. If an application thread calls an SDK API whilst holding a lock that is needed by a callback function, then the likelihood of deadlock is very high. The UPnP device thread may invoke that callback function, causing it to block until it acquires the lock. This prevents the UPnP device

thread from servicing the application thread's API calls, leading to deadlock.
Don't block the UPnP device thread to wait for a response from an application thread. It is common to want to communicate events from the UPnP device thread to the application thread. If the inter-thread RPC mechanism used to communicate the events to the application thread blocks until it is serviced, then the likelihood of deadlock is very high. Sooner or later, the application thread will be blocked on an SDK API call at the time that the UPnP device thread needs it to service the blocking RPC, and the application is deadlocked.

Network interfaces

One or more network interfaces may be associated with or disassociated from a VNC Network Advertiser at any time. This allows network interfaces which are transient, such as those based on a USB connection, to be supported. A VNC Network Advertiser may be running without any network interfaces associated with it, but will not do anything until a network interface is made available.

Call `VNCNetworkAdvertiser.registerInterface(String,int)` to associate a network interface with a VNC Network Advertiser. If running, the advertiser will start advertising its presence on the network and will service incoming requests. You may associate multiple network interfaces with a single VNC Network Advertiser, but a network interface must not be associated with more than one VNC Network Advertiser instance at any one time. When registering a network interface, you may optionally specify the TCP port that must be used to receive requests from VNC Network Discoverers. If this port is unavailable, then the network interface shall not be used by the SDK. Your application shall be notified of the TCP port in use for a network interface through `VNCNetworkAdvertiserListener.listening(String,int)`.

Call `VNCNetworkAdvertiser.unregisterInterface(String)` to disassociate a network interface from a VNC Network Advertiser. If possible, the device will announce its disappearance from the network. If the network interface still exists, then it may be safely registered with another VNC Network Advertiser instance.

If an error occurs when interacting with a network interface, then that interface is automatically disassociated from the VNC Network Advertiser. This is not considered a fatal error, as a transient network interface may disappear from the system at any time. The SDK shall call `VNCNetworkAdvertiserListener.interfaceUnregistered(String)` to notify your application when an interface has been automatically disassociated.

Note: A VNC Network Advertiser requires multicast packet reception, to allow it to receive search requests from UPnP contol points. It shall therefore hold an `android.net.wifi.WifiManager.MulticastLock` whilst the UPnP device thread is running. If your device is unable to provide support for multicast packet reception, or the network is unable to deliver multicast packets, then the VNC Network Advertiser may not be detected on the network by the VNC Network Discoverer.

Managing icon resources

A VNC Network Advertiser may present a series of icons to VNC Network Discoverers for use when representing your VNC server on a graphical display. To associate an icon with a VNC Network Advertiser, call `VNCNetworkAdvertiser.iconCreate(VNCNetworkAdvertiserIconDetails,byte[])` with the icon details and data. The SDK will return a `VNCNetworkAdvertiserIcon` object representing the icon, which can be used in subsequent API calls, such as `VNCNetworkAdvertiser.setGlobalServerDetails(VNCNetworkAdvertiserServerDetails,VNCNetworkAdvertiserIcon[])`.

When a `VNCNetworkAdvertiserIcon` object is no longer required by your application (as you do not require it for any future SDK API calls), call `VNCNetworkAdvertiser.iconRelease(VNCNetworkAdvertiserIcon)` to release your reference to it. The object is reference counted by the SDK, and shall be destroyed when no longer in use by either the VNC Network Advertiser or your application. When the VNC Network Advertiser is destroyed with `VNCNetworkAdvertiser.destroy()`, then all icon objects that remain are also destroyed.

VNC servers

A VNC Network Advertiser advertises a single VNC server to VNC Network Discoverers. Use `VNCNetworkAdvertiser.setGlobalServerDetails(VNCNetworkAdvertiserServerDetails,VNCNetworkAdvertiserIcon[])` to specify the details of your VNC server.

A VNC Network Discoverer may request a VNC command string for your VNC server, in order to allow its associated VNC viewer to establish a connection. When this occurs, the SDK shall notify your application by invoking `VNCNetworkAdvertiserListener.globalServerCommandStringRequest(String)`. You must perform the actions necessary to to obtain a suitable VNC command string for your VNC server. This may include determining whether or not the VNC server is busy with a connection to another VNC viewer, or ensuring that the VNC server is listening on the appropriate network interface. Once your attempt to obtain the VNC command string is complete, you must call `VNCNetworkAdvertiser.globalServerCommandStringResult(String,String)` to notify the SDK. An appropriate response will then be sent to the VNC Network Discoverer.

If your VNC server is busy, you should provide an empty string when a VNC command string is requested. VNC Network Discoverers shall interpret this to mean that the VNC server is present but currently not available to viewers. You should only stop your VNC Network Advertiser

object when your VNC server becomes inactive.

Legal information

Details of and copyright notices for third-party software that is used by the VNC Android Network Advertiser SDK can be found in the file `Acknowledgements.txt` in the SDK distribution.

Hierarchy For All Packages

Package Hierarchies:
com.realvnc.networkadvertisersdk

Class Hierarchy

- java.lang.Object
  - java.lang.Throwable (implements java.io.Serializable)
    - java.lang.Exception
      - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserException**
  - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserDeviceIdentity**
  - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserIconDetails**
  - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserParameter**
  - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserProperty**
  - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserSDK**
  - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserServerDetails**

Interface Hierarchy

- com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiser**
- com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserIcon**
- com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserListener**

Hierarchy For Package com.realvnc.networkadvertisersdk

Class Hierarchy

- java.lang.Object
  - java.lang.Throwable (implements java.io.Serializable)
    - java.lang.Exception
      - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserException**
  - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserDeviceIdentity**
  - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserIconDetails**
  - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserParameter**
  - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserProperty**
  - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserSDK**
  - com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserServerDetails**

Interface Hierarchy

- com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiser**
- com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserIcon**
- com.realvnc.networkadvertisersdk.**VNCNetworkAdvertiserListener**

Package com.realvnc.networkadvertisersdk

The Android interface to the VNC Network Advertiser SDK.

See: Description

Interface Summary

Interface	Description
VNCNetworkAdvertiser	Interface implemented by VNC Network Advertiser instances.
VNCNetworkAdvertiserIcon	Represents an icon used by a VNC Network Advertiser.
VNCNetworkAdvertiserListener	Callback interface used by VNC Network Advertisers to provide notifications to your application.

Class Summary

Class	Description
VNCNetworkAdvertiserDeviceIdentity	Specifies identifying information for a device.
VNCNetworkAdvertiserIconDetails	Describes the basic details of an icon.
VNCNetworkAdvertiserParameter	Defines constants representing configuration parameters that can be used with a VNC Network Advertiser.
VNCNetworkAdvertiserProperty	Defines properties of a VNC Network Advertiser that may be queried.
VNCNetworkAdvertiserSDK	SDK class handling network advertiser instantiation and providing various helper functions.
VNCNetworkAdvertiserServerDetails	Describes a VNC server.

Exception Summary

Exception	Description
VNCNetworkAdvertiserException	VNC Network Advertiser exception class indicating an error.

Package com.realvnc.networkadvertisersdk Description

The Android interface to the VNC Network Advertiser SDK.

Deprecated API

Contents

Constant Field Values

Contents

com.realvnc.*

com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserException

Modifier and Type	Constant Field	Value
public static final int	ALREADY_EXISTS	15
public static final int	FAILED	13
public static final int	FEATURE_NOT_LICENSED	19
public static final int	ILLEGAL_WHILE_NOT_RUNNING	3
public static final int	ILLEGAL_WHILE_RUNNING	2
public static final int	INVALID_PARAMETER	1
public static final int	LICENSE_NOT_VALID	18
public static final int	NETWORK_INTERFACE_IN_USE	5
public static final int	NO_DEVICE_DETAILS	9
public static final int	NO_DEVICE_IDENTITY	10
public static final int	NO_GLOBAL_SERVER_DETAILS	11
public static final int	NO_OUTSTANDING_REQUEST	12
public static final int	NONE	0
public static final int	NOT_FOUND	14
public static final int	NOT_IMPLEMENTED	17
public static final int	NOT_SUPPORTED	16
public static final int	PERMISSION_DENIED	7
public static final int	PORT_IN_USE	6
public static final int	RESOURCE_IN_USE	4
public static final int	STOPPED	8

com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserParameter

Modifier and Type	Constant Field	Value
public static final java.lang.String	DEFAULT_VALUE_LOG	""
public static final java.lang.String	DEFAULT_VALUE_SSDP_ADVERTISEMENT_EXPIRY_DURATION	"1800"
public static final java.lang.String	DEFAULT_VALUE_SSDP_ADVERTISEMENT_INTERVAL	"0"
public static final java.lang.String	LOG	"Log"
public static final java.lang.String	SSDP_ADVERTISEMENT_EXPIRY_DURATION	"SSDPAdvertisementExpiryDuration"
public static final java.lang.String	SSDP_ADVERTISEMENT_INTERVAL	"SSDPAdvertisementInterval"

com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserProperty

Modifier and Type	Constant Field	Value
public static final int	IS_RUNNING	0

Serialized Form

Package com.realvnc.networkadvertisersdk

Class com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserException extends java.lang.Exception implements Serializable

serialVersionUID: 1L

Serialized Fields

errorCode

```
int errorCode
```

The error code describing the reason for this exception.

Possible values for this and the associated meaning are defined as constants in this class.

See Also:

`VNCNetworkAdvertiserSDK.getErrorName(int)`

com.realvnc.networkadvertisersdk

Class VNCNetworkAdvertiserDeviceIdentity

java.lang.Object
    com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserDeviceIdentity

---

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

Specifies identifying information for a device.

Construct an instance of this class to pass device identity details to a VNC Network Advertiser instance using `VNCNetworkAdvertiser.setDeviceIdentity(VNCNetworkAdvertiserDeviceIdentity)`.

See Also:

`VNCNetworkAdvertiser`

Field Summary

Fields

Modifier and Type	Field and Description
`java.lang.String`	**`deviceId`** A device-specific identifier.
`java.lang.String`	**`imei`** The IMEI number for the device.
`java.lang.String`	**`wifiMacAddr`** The WiFi MAC address of the device.

Constructor Summary

Constructors

Constructor and Description
`VNCNetworkAdvertiserDeviceIdentity``(java.lang.String deviceId, java.lang.String imei, java.lang.String wifiMacAddr)` Constructs a new VNCNetworkAdvertiserDeviceIdentity object with the given identity details.

Method Summary

Methods inherited from class java.lang.Object

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

Field Detail

deviceId

```
public final java.lang.String deviceId
```

A device-specific identifier.

It is recommended to use the `ANDROID_ID`. This value must be unique to the device.

imei

```
public final java.lang.String imei
```

The IMEI number for the device.

This may be `null` if the device has no IMEI number, or if access to the IMEI number is restricted by Android.

wifiMacAddr

```
public final java.lang.String wifiMacAddr
```

The WiFi MAC address of the device.

This may be `null` if the device has no WiFi adapter, or if access to the (real) MAC address restricted by Android. Note that from API level 23 Android will return a constant value for the MAC address to safeguard privacy.

Constructor Detail

VNCNetworkAdvertiserDeviceIdentity

```
public VNCNetworkAdvertiserDeviceIdentity(java.lang.String deviceId,
                                          java.lang.String imei,
                                          java.lang.String wifiMacAddr)
```

Constructs a new VNCNetworkAdvertiserDeviceIdentity object with the given identity details.

com.realvnc.networkadvertisersdk

Class VNCNetworkAdvertiserException

java.lang.Object
    java.lang.Throwable
        java.lang.Exception
            com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserException

All Implemented Interfaces:

java.io.Serializable

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

VNC Network Advertiser exception class indicating an error.

The error codes defined in this class are also used in calls to `VNCNetworkAdvertiserListener.error(int)`, in which case they indicate that the VNC Network Advertiser has stopped and that the UPnP device thread is about to exit.

See Also:

`VNCNetworkAdvertiser`, `VNCNetworkAdvertiserSDK`, Serialized Form

Field Summary

Fields

Modifier and Type	Field and Description
static int	ALREADY_EXISTS The operation failed because the thing being added or registered already exists.
int	errorCode The error code describing the reason for this exception.
static int	FAILED The operation failed for an unknown reason.
static int	FEATURE_NOT_LICENSED A feature of the SDK cannot be used because it is not licensed.
static int	ILLEGAL_WHILE_NOT_RUNNING The application called an SDK API that may not be called unless the UPnP device thread is running.
static int	ILLEGAL_WHILE_RUNNING The application called an SDK API that may not be called while the UPnP device thread is running.
static int	INVALID_PARAMETER A parameter supplied to an SDK API method is not valid.
static int	LICENSE_NOT_VALID A license could not be added because it is invalid.
static int	NETWORK_INTERFACE_IN_USE The request failed because the network interface in question is already in use.
static int	NO_DEVICE_DETAILS The request failed because the necessary device details have not been provided.
static int	NO_DEVICE_IDENTITY The request failed because the necessary device identity information has not been provided.
static int	NO_GLOBAL_SERVER_DETAILS The request failed because the necessary VNC server details have not been provided.
static int	NO_OUTSTANDING_REQUEST The application attempted to provided a result for a request, but no request is currently outstanding.
static int	NONE No error.

static int	NOT_FOUND The required object, entity or data was not found.
static int	NOT_IMPLEMENTED This API function is not implemented for the current platform.
static int	NOT_SUPPORTED The operation failed because it is not supported.
static int	PERMISSION_DENIED Permission to access the specified resource or entity was denied by the operating system.
static int	PORT_IN_USE The request failed because the network port in question is already in use.
static int	RESOURCE_IN_USE The request failed because the resource in question is actively being used by a VNC Network Advertiser instance.
static int	STOPPED VNCNetworkAdvertiser.stop() was called (either explicitly or from VNCNetworkAdvertiser.destroy()).

Constructor Summary

Constructors

Constructor and Description
VNCNetworkAdvertiserException(int errorCode) Constructs a new VNC Network Advertiser exception instance, with the specified error code.
VNCNetworkAdvertiserException(int errorCode, java.lang.String message) Constructs a new VNC Network Advertiser exception instance, with the specified error code.

Method Summary

Methods inherited from class java.lang.Throwable

addSuppressed, fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, getSuppressed, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString

Methods inherited from class java.lang.Object

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

Field Detail

NONE

```
public static final int NONE
```

No error.

See Also:

Constant Field Values

INVALID_PARAMETER

public static final int INVALID_PARAMETER

A parameter supplied to an SDK API method is not valid.

See Also:

Constant Field Values

ILLEGAL_WHILE_RUNNING

public static final int ILLEGAL_WHILE_RUNNING

The application called an SDK API that may not be called while the UPnP device thread is running.

See Also:

Constant Field Values

ILLEGAL_WHILE_NOT_RUNNING

public static final int ILLEGAL_WHILE_NOT_RUNNING

The application called an SDK API that may not be called unless the UPnP device thread is running.

See Also:

Constant Field Values

RESOURCE_IN_USE

public static final int RESOURCE_IN_USE

The request failed because the resource in question is actively being used by a VNC Network Advertiser instance.

See Also:

Constant Field Values

NETWORK_INTERFACE_IN_USE

public static final int NETWORK_INTERFACE_IN_USE

The request failed because the network interface in question is already in use.

See Also:

Constant Field Values

PORT_IN_USE

public static final int PORT_IN_USE

The request failed because the network port in question is already in use.

See Also:

Constant Field Values

PERMISSION_DENIED

public static final int PERMISSION_DENIED

Permission to access the specified resource or entity was denied by the operating system.

See Also:

Constant Field Values

STOPPED

`public static final int STOPPED`

`VNCNetworkAdvertiser.stop()` was called (either explicitly or from `VNCNetworkAdvertiser.destroy()`).

See Also:

Constant Field Values

NO_DEVICE_DETAILS

`public static final int NO_DEVICE_DETAILS`

The request failed because the necessary device details have not been provided.

See Also:

Constant Field Values

NO_DEVICE_IDENTITY

`public static final int NO_DEVICE_IDENTITY`

The request failed because the necessary device identity information has not been provided.

See Also:

Constant Field Values

NO_GLOBAL_SERVER_DETAILS

`public static final int NO_GLOBAL_SERVER_DETAILS`

The request failed because the necessary VNC server details have not been provided.

See Also:

Constant Field Values

NO_OUTSTANDING_REQUEST

`public static final int NO_OUTSTANDING_REQUEST`

The application attempted to provided a result for a request, but no request is currently outstanding.

See Also:

Constant Field Values

FAILED

`public static final int FAILED`

The operation failed for an unknown reason.

See Also:

Constant Field Values

NOT_FOUND

```
public static final int NOT_FOUND
```

The required object, entity or data was not found.

See Also:

Constant Field Values

ALREADY_EXISTS

```
public static final int ALREADY_EXISTS
```

The operation failed because the thing being added or registered already exists.

See Also:

Constant Field Values

NOT_SUPPORTED

```
public static final int NOT_SUPPORTED
```

The operation failed because it is not supported.

See Also:

Constant Field Values

NOT_IMPLEMENTED

```
public static final int NOT_IMPLEMENTED
```

This API function is not implemented for the current platform.

See Also:

Constant Field Values

LICENSE_NOT_VALID

```
public static final int LICENSE_NOT_VALID
```

A license could not be added because it is invalid.

See Also:

Constant Field Values

FEATURE_NOT_LICENSED

```
public static final int FEATURE_NOT_LICENSED
```

A feature of the SDK cannot be used because it is not licensed.

See Also:

Constant Field Values

errorCode

```
public final int errorCode
```

The error code describing the reason for this exception.

Possible values for this and the associated meaning are defined as constants in this class.

See Also:

`VNCNetworkAdvertiserSDK.getErrorName(int)`

Constructor Detail

VNCNetworkAdvertiserException

```
public VNCNetworkAdvertiserException(int errorCode)
```

Constructs a new VNC Network Advertiser exception instance, with the specified error code.

Parameters:

`errorCode` - The reason for the exception.

VNCNetworkAdvertiserException

```
public VNCNetworkAdvertiserException(int errorCode,
                                     java.lang.String message)
```

Constructs a new VNC Network Advertiser exception instance, with the specified error code.

Parameters:

`errorCode` - The reason for the exception.

`message` - The message for the exception.

com.realvnc.networkadvertisersdk

Interface VNCNetworkAdvertiser

`public interface` **`VNCNetworkAdvertiser`**

Interface implemented by VNC Network Advertiser instances.

To create a VNC Network Advertiser instance, call `VNCNetworkAdvertiserSDK.advertiserCreate(android.content.Context,VNCNetworkAdvertiserListener)`.

See Also:

`VNCNetworkAdvertiserSDK`, `VNCNetworkAdvertiserListener`

Method Summary

Methods

Modifier and Type	Method and Description
`byte[]`	**`addLicense`**`(java.io.InputStream is)` Adds a VNC license to this VNC Network Advertiser.
`byte[]`	**`addLicense`**`(java.lang.String licenseText)` Adds a VNC license to this VNC Network Advertiser.
`void`	**`destroy`**`()` Destroys this VNC Network Advertiser.
`java.lang.String`	**`getParameter`**`(java.lang.String parameter)` Queries the value of a VNC Network Advertiser parameter.
`int`	**`getProperty`**`(int property)` Queries the value of an integer-valued runtime property.
`java.lang.String`	**`getPropertyString`**`(int property)` Queries the value of a string-valued runtime property.
`void`	**`globalServerCommandStringResult`**`(java.lang.String interfaceName, java.lang.String commandString)` Notifies the SDK of the result of a command string request for the VNC server.
`VNCNetworkAdvertiserIcon`	**`iconCreate`**`(VNCNetworkAdvertiserIconDetails iconDetails, byte[] iconData)` Create an icon resource for this VNC Network Advertiser.
`void`	**`iconRelease`**`(VNCNetworkAdvertiserIcon icon)` Release an icon resource associated with this VNC Network Advertiser.
`void`	**`registerInterface`**`(java.lang.String interfaceName, int port)` Registers a network interface with this VNC Network Advertiser.
`void`	**`setDeviceDetails`**`(java.lang.String friendlyName, java.lang.String manufacturer, java.lang.String modelName, java.lang.String modelDescription, java.lang.String modelNumber, java.util.UUID uniqueDeviceName, java.lang.String productDetailString)` Set basic device details for this VNC Network Advertiser.
`void`	**`setDeviceIdentity`**`(VNCNetworkAdvertiserDeviceIdentity identity)` Sets device identity information for this VNC Network Advertiser.
`void`	**`setGlobalServerDetails`**`(VNCNetworkAdvertiserServerDetails details, VNCNetworkAdvertiserIcon[] icons)` Sets the details of the VNC server to be advertised by this VNC Network Advertiser.
`void`	**`setParameter`**`(java.lang.String parameter, java.lang.String value)` Sets the value of a VNC Network Advertiser parameter.
`void`	**`start`**`()` Starts this VNC Network Advertiser.
`void`	**`stop`**`()` Stops this VNC Network Advertiser.

void	unregisterInterface(java.lang.String interfaceName) Unregisters a network interface from this VNC Network Advertiser.

Method Detail

destroy

```
void destroy()
```

Destroys this VNC Network Advertiser.

This method destroys this VNC Network Advertiser instance. If the VNC Network Advertiser is still running, it will be stopped first. Any icons still associated with this instance will also be destroyed.

After calling this method, you should not attempt to interact further with this VNC Network Advertiser.

Note: It is not mandatory that the destroy is called as this is performed within the finalize method. However on low-memory systems, this ensures that destruction is performed immediately which may be important for memory management.

addLicense

```
byte[] addLicense(java.io.InputStream is)
                  throws VNCNetworkAdvertiserException
```

Adds a VNC license to this VNC Network Advertiser.

The VNC Network Advertiser must be licensed before it can be started. It recommended that this method is called soon after creating the VNC Network Advertiser instance.

To obtain a license for use with the VNC Network Advertiser SDK, contact your RealVNC sales representative.

Parameters:

`is` - An InputStream supplying the entire license text.

Returns:

The serial number for the license.

Throws:

`VNCNetworkAdvertiserException` - with the following errors:

- `IllegalWhileRunning` The license could not be added, as the UPnP device thread is running.
- `InvalidParameter` The license could not be added, as `is` is `null`.
- `LicenseNotValid` The license could not be added, as `is` does not supply a valid VNC license.
- `Failed` The license could not be added, for an unknown reason.

See Also:

`addLicense(String)`

addLicense

```
byte[] addLicense(java.lang.String licenseText)
                  throws VNCNetworkAdvertiserException
```

Adds a VNC license to this VNC Network Advertiser.

The VNC Network Advertiser must be licensed before it can be started. It recommended that this method is called soon after creating the VNC Network Advertiser instance.

To obtain a license for use with the VNC Network Advertiser SDK, contact your RealVNC sales representative.

You must provide the text of the entire license to this method, including the header and footer. If your license has been supplied to you in a `.vnclicense` file, then you should provide the entire contents of that file.

Parameters:

`licenseText` - The text of the VNC license.

Returns:

The serial number for the license.

Throws:

`VNCNetworkAdvertiserException` - with the following errors:

- `IllegalWhileRunning` The license could not be added, as the UPnP device thread is running.
- `InvalidParameter` The license could not be added, as `licenseText` is `null`.
- `LicenseNotValid` The license could not be added, as `licenseText` does not contain a valid VNC license.
- `Failed` The license could not be added, for an unknown reason.

getProperty

```
int getProperty(int property)
                throws VNCNetworkAdvertiserException
```

Queries the value of an integer-valued runtime property.

You can use this method to find out information about this VNC Network Advertiser, such as whether or not the UPnP device thread is running.

To query string-valued runtime parameters, use `getPropertyString(int)` instead.

Parameters:

`property` - The property to query from `VNCNetworkAdvertiserProperty`.

Returns:

The value of the property.

Throws:

`VNCNetworkAdvertiserException` - with the following error:

- `InvalidParameter` The requested property is not a valid integer property.

See Also:

`VNCNetworkAdvertiserProperty`

getPropertyString

```
java.lang.String getPropertyString(int property)
                                   throws VNCNetworkAdvertiserException
```

Queries the value of a string-valued runtime property.

You can use this method to find out information about this VNC Network Advertiser, such as whether or not the UPnP device thread is running.

To query integer-valued runtime parameters, use `getProperty(int)` instead.

Parameters:

`property` - The property to query from `VNCNetworkAdvertiserProperty`.

Returns:

The String value of the property.

Throws:

`VNCNetworkAdvertiserException` - with the following errors:

- `InvalidParameter` The requested property is not a valid integer property.
- `Failed` The value of the property could not be retrieved for an unknown reason.

See Also:

VNCNetworkAdvertiserProperty

setParameter

```
void setParameter(java.lang.String parameter,
                  java.lang.String value)
                    throws VNCNetworkAdvertiserException
```

Sets the value of a VNC Network Advertiser parameter.

If the UPnP device thread is running, then the new value will take effect immediately, provided that this makes sense. Otherwise, the new value will be used the next time the UPnP device thread is started.

VNC Network Advertiser parameters may never have `null` values. To 'clear' the value of a parameter, pass an empty string as the value.

Parameters whose values are integers are formatted in decimal.

VNC Network Advertiser parameter names are case-insensitive.

Parameters:

`parameter` - The name of the VNC Network Advertiser parameter to set. See `VNCNetworkAdvertiserParameter` for an explanation of the supported parameters.

`value` - The new parameter value.

Throws:

`VNCNetworkAdvertiserException` - with the following errors:

- `InvalidParameter` Either the parameter name is not recognised, or the given value is not valid for this parameter.
- `Failed` The parameter could not be set, for an unknown reason.

See Also:

VNCNetworkAdvertiserParameter

getParameter

```
java.lang.String getParameter(java.lang.String parameter)
                              throws VNCNetworkAdvertiserException
```

Queries the value of a VNC Network Advertiser parameter.

The configuration of a VNC Network Advertiser is stored internally as a list of name-value pairs. This method queries the value of a VNC Network Advertiser parameter and returns it. The parameter names are case-insensitive.

Parameters:

`parameter` - The name of the VNC Network Advertiser parameter to query. See `VNCNetworkAdvertiserParameter` for an explanation of the supported parameters.

Returns:

The parameter's value.

Throws:

`VNCNetworkAdvertiserException` - with the following errors:

- `InvalidParameter` The parameter name is not recognised.
- `Failed` The parameter value could not be retrieved, for an unknown reason.

See Also:

VNCNetworkAdvertiserParameter

registerInterface

```
void registerInterface(java.lang.String interfaceName,
                       int port)
                         throws VNCNetworkAdvertiserException
```

Registers a network interface with this VNC Network Advertiser.

This method may be called at any time to associate a network interface with the advertiser. This allows support for transient network interfaces, such as those based on USB connections. A network interface must only be registered with a single VNC Network Advertiser at any one time.

The network interface is described using the platform-specific name. Valid interface names may be `"eth0"`, `"wlan0"` or something else, depending on what your Android device supports.

If a non-zero TCP port is specified, then this shall be used to receive requests from VNC Network Discoverers by the VNC Network Advertiser on this network interface and shall be presented in all URIs advertised on this interface. If the port is not available, then the network interface shall be automatically unregistered by the SDK. Your application shall be notified if this occurs by `VNCNetworkAdvertiserListener.interfaceUnregistered(String)`.

Alternatively, if the specified TCP port is zero, then the SDK shall choose an available TCP port. Your application shall be notified of the chosen port by `VNCNetworkAdvertiserListener.listening(String,int)`.

In addition to the TCP port, a VNC Network Advertiser shall always use UDP port 1900 for SSDP messages. This port is mandated by SSDP, and is not configurable.

Parameters:

`interfaceName` - The platform-specific name identifying the interface to register, such as `"eth0"` or `"wlan0"`.

`port` - The TCP port to use to receive requests from VNC Network Discoverers, or zero if the SDK should choose an available TCP port.

Throws:

`VNCNetworkAdvertiserException` - with the following errors:

- `InvalidParameter` The network interface or port is invalid.
- `PermissionDenied` Access to the network interface was denied by the operating system.
- `NetworkInterfaceInUse` The network interface could not be registered as it is already associated with this VNC Network Advertiser.
- `PortInUse` A non-zero port has been given which cannot currently be accessed.
- `Failed` The network interface could not be registered for an unknown reason.

unregisterInterface

```
void unregisterInterface(java.lang.String interfaceName)
```

Unregisters a network interface from this VNC Network Advertiser.

This method may be called at any time to disassociate a network interface from the advertiser. If possible, the advertiser will announce its disappearance from the network before dropping the interface. When this method returns, the network interface shall no longer be used by the VNC Network Advertiser.

Once unregistered from one VNC Network Advertiser, a network interface can safely be registered with any other VNC Network Advertiser instance.

The network interface is described using the platform-specific name. Valid interface names may be `"eth0"`, `"wlan0"` or something else, depending on what your Android device supports. If the interface identified is not associated, then this method does nothing.

Parameters:

`interfaceName` - The platform-specific name identifying the interface to unregister, such as `"eth0"` or `"wlan0"`.

iconCreate

```
VNCNetworkAdvertiserIcon iconCreate(VNCNetworkAdvertiserIconDetails iconDetails,
                                    byte[] iconData)
                                      throws VNCNetworkAdvertiserException
```

Create an icon resource for this VNC Network Advertiser.

An icon can be retrieved by VNC Network Discoverers over HTTP. The icon may be used by the discoverer to represent a VNC server on a graphical display. This method may be called at any time to create a new icon resource.

The icon resource is represented as a `VNCNetworkAdvertiserIcon` object, which may subsequently be passed to other SDK APIs. This object is reference counted by the SDK. When you have finished with the object, you should release your reference to it with `iconRelease(VNCNetworkAdvertiserIcon)`. For example, you may release the object after associating it with a global server, if you have no further use for it. `VNCNetworkAdvertiserIcon` objects which are no longer being used by either the VNC Network Advertiser or your application shall be destroyed by the SDK. All `VNCNetworkAdvertiserIcon` objects shall be destroyed when `destroy()` is called.

Parameters:

`iconDetails` - The icon's details.

`iconData` - A buffer holding the data for the icon.

Returns:

A `VNCNetworkAdvertiserIcon` object representing the new icon.

Throws:

`VNCNetworkAdvertiserException` - with the following errors:

- `InvalidParameter` The icon details were invalid or no buffer was provided with icon data.
- `Failed` The icon could not be created for an unknown reason.

See Also:

`VNCNetworkAdvertiserIconDetails`

iconRelease

```
void iconRelease(VNCNetworkAdvertiserIcon icon)
                 throws VNCNetworkAdvertiserException
```

Release an icon resource associated with this VNC Network Advertiser.

This method should be called when a `VNCNetworkAdvertiserIcon` object is no longer required by your application. This invalidates your reference to the object. When the icon resource is no longer being used by the VNC Network Advertiser, it shall be destroyed by the SDK.

Parameters:

`icon` - The `VNCNetworkAdvertiserIcon` to release.

Throws:

`VNCNetworkAdvertiserException` - with the following errors:

- `InvalidParameter` The `VNCNetworkAdvertiserIcon` object does not identify a valid icon resource.
- `Failed` The icon could not be released for an unknown reason.

setDeviceDetails

```
void setDeviceDetails(java.lang.String friendlyName,
                      java.lang.String manufacturer,
                      java.lang.String modelName,
                      java.lang.String modelDescription,
                      java.lang.String modelNumber,
                      java.util.UUID uniqueDeviceName,
                      java.lang.String productDetailString)
                        throws VNCNetworkAdvertiserException
```

Set basic device details for this VNC Network Advertiser.

This sets details about your device that shall be advertised to VNC Network Discoverers through UPnP device description document generated by this VNC Network Advertiser. Refer to the UPnP 1.0 specification for further information on the exact meaning of these details.

This method must be called before `start()` can be invoked. This method may be called multiple times, but must not be called whilst the UPnP device thread is running. The details from the most recent invocation of this method shall be used when the UPnP device thread starts.

The product detail string is included alongside the OS name, OS version and UPnP version in the SERVER header of UPnP HTTP notifications and responses sent by the VNC Network Advertiser. The recommended format for the product detail string is `"ProductName/ProductVersion"`. Any additional information given will also be appended to the SERVER header string. For example, if the product detail string is set to `"RealVNC-MobileSolution-Server/1.0"`, then the SERVER header reported by this VNC Network Advertiser will be similar to the following:

```
Server: Android UPnP/1.0 RealVNC-MobileSolution-Server/1.0
```

Parameters:

`friendlyName` - The friendly name for your device. This string must be less than 64 characters in length.

`manufacturer` - The manufacturer name for your device. This string must be less than 64 characters in length.

`modelName` - The model name for your device. This string must be less than 32 characters in length.

`modelDescription` - A long description for your device. This string must be less than 128 characters in length.

`modelNumber` - The model number for your device. This string must be less than 32 characters in length.

`uniqueDeviceName` - The Universally Unique IDentifier for your device. This string must contain a parsable UUID value. An individual device must consistently provide the same UUID value over time, and that value must be unique to the individual device. The SDK will not check the UUID for consistency or uniqueness.

`productDetailString` - The product detail string for your device. This string must not be empty.

Throws:

`VNCNetworkAdvertiserException` - with the following errors:

- `IllegalWhileRunning` This method was called whilst the UPnP device was running.
- `InvalidParameter` Not all required details were provided, or one or more device details could not be accepted.
- `Failed` The device details could not be accepted for an unknown reason.

setDeviceIdentity

```
void setDeviceIdentity(VNCNetworkAdvertiserDeviceIdentity identity)
                       throws VNCNetworkAdvertiserException
```

Sets device identity information for this VNC Network Advertiser.

This sets identifying information for your device, required by VNC Network Discoverers, which is additional to the general device information provided by UPnP to control points. This additional identity information includes a unique identifier string for your device, which would usually be the `ANDROID_ID`.

This method must be called before `start()` can be invoked. This method may be called multiple times, but must not be called whilst the UPnP device thread is running. The details from the most recent invocation of this method shall be used when the UPnP device thread starts.

Parameters:

`identity` - The device identity information.

Throws:

`VNCNetworkAdvertiserException` - with the following errors:

- `IllegalWhileRunning` This method was called whilst the UPnP device was running.
- `InvalidParameter` Not all required identity information was provided, or one or more elements could not be accepted.
- `Failed` The identity information could not be accepted for an unknown reason.

See Also:

`VNCNetworkAdvertiserDeviceIdentity`

setGlobalServerDetails

```
void setGlobalServerDetails(VNCNetworkAdvertiserServerDetails details,
```

```
                                VNCNetworkAdvertiserIcon[] icons)
                                  throws VNCNetworkAdvertiserException
```

Sets the details of the VNC server to be advertised by this VNC Network Advertiser.

You should call this method to specify the details of the Mobile Solution VNC server that you wish to advertise with this VNC Network Advertiser. This method must be called before `start()` can be invoked. This method may be called multiple times, but must not be called whilst the UPnP device thread is running. The details from the most recent invocation of this method shall be used when the UPnP device thread starts.

Parameters:

`details` - The VNC server details.

`icons` - An array of icons that may be used to represent the VNC server, or `null` if the VNC server has no icons.

Throws:

`VNCNetworkAdvertiserException` - with the following errors:

- `IllegalWhileRunning` This method was called whilst the UPnP device was running.
- `InvalidParameter` Not all required details were provided, or one or more details could not be accepted.
- `Failed` The VNC server details could not be accepted for an unknown reason.

See Also:

`VNCNetworkAdvertiserServerDetails`

start

```
void start()
           throws VNCNetworkAdvertiserException
```

Starts this VNC Network Advertiser.

This starts the UPnP device thread. The VNC Network Advertiser will become discoverable on all registered network interfaces. You must add at least one license with `addLicense(InputStream)` or `addLicense(String)` before you can start the VNC Network Advertiser. You must also call `setDeviceDetails(String,String,String,String,String,UUID,String)`, `setDeviceIdentity(VNCNetworkAdvertiserDeviceIdentity)` and `setGlobalServerDetails(VNCNetworkAdvertiserServerDetails,VNCNetworkAdvertiserIcon[])` before invoking this method.

When the UPnP device thread is about to stop, `VNCNetworkAdvertiserListener.error(int)` will be invoked with an error code describing the reason for the stoppage. The UPnP device thread will stop when a non-recoverable error occurs, or either `stop()` or `destroy()` is called.

If the UPnP device thread is already running, this method does nothing.

Throws:

`VNCNetworkAdvertiserException` - with the following errors:

- `NoDeviceDetails` The UPnP device thread could not be started as `setDeviceDetails(String,String,String,String,String,UUID,String)` has not been called.
- `NoDeviceIdentity` The UPnP device thread could not be started as `setDeviceIdentity(VNCNetworkAdvertiserDeviceIdentity)` has not been called.
- `NoGlobalServerDetails` The UPnP device thread could not be started as `setGlobalServerDetails(VNCNetworkAdvertiserServerDetails,VNCNetworkAdvertiserIcon[])` has not been called.

stop

```
void stop()
```

Stops this VNC Network Advertiser.

This stops the UPnP device thread asynchronously. `VNCNetworkAdvertiserListener.error(int)` will be invoked with the error `Stopped` when the UPnP device thread is about to stop. Before stopping, the VNC Network Advertiser will broadcast its removal from all registered network interfaces where this is possible.

Calling this method does not destroy the VNC Network Advertiser object itself. You may either call `destroy()` to destroy it, or, once it has finished stopping, call `start()` to start it again.

If the UPnP device thread is not running, this method does nothing.

globalServerCommandStringResult

```
void globalServerCommandStringResult(java.lang.String interfaceName,
                                     java.lang.String commandString)
                                       throws VNCNetworkAdvertiserException
```

Notifies the SDK of the result of a command string request for the VNC server.

This method must be called after your application has received the `VNCNetworkAdvertiserListener.globalServerCommandStringRequest(String)` callback, and when the result of the request is known. This method may be called from within or outside of your implementation of `VNCNetworkAdvertiserListener.globalServerCommandStringRequest(String)`. The command string request is received from a VNC Network Discoverer for the VNC server described in the call to `setGlobalServerDetails(VNCNetworkAdvertiserServerDetails,VNCNetworkAdvertiserIcon[])`.

If the VNC server is listening, then a suitable command string should be provided. If the VNC server is no longer listening, for example because another VNC viewer has connected to it, then empty-string may be provided. The VNC Network Discoverer shall interpret this to mean that the server is no longer available. If a command string could not be provided because of an error, then the commandString parameter should be set to `null`. This will cause the VNC Network Advertiser to return an error to the VNC Network Discoverer.

Parameters:

`interfaceName` - The platform-specific name identifying the network interface being used to communicate with the VNC Network Discoverer. This must be the interface specified in the call to `VNCNetworkAdvertiserListener.globalServerCommandStringRequest(String)`.

`commandString` - A valid VNC command string if the VNC server is listening, empty-string if the VNC server is now busy or `null` if an error has occurred.

Throws:

`VNCNetworkAdvertiserException` - with the following errors:

- `InvalidParameter` The result was not accepted, as the command string was malformed.
- `IllegalWhileNotRunning` The result was not accepted, as the UPnP device thread is not running.
- `NoOutstandingRequest` The result was not accepted, as the SDK currently has no oustanding request for the given network interface.
- `Failed` The result could not be accepted for an unknown reason.

com.realvnc.networkadvertisersdk

Class VNCNetworkAdvertiserIconDetails

java.lang.Object
    com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserIconDetails

---

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

Describes the basic details of an icon.

Construct an instance of this class to pass icon details to a VNC Network Advertiser instance using `VNCNetworkAdvertiser.iconCreate(VNCNetworkAdvertiserIconDetails,byte[])`.

See Also:

`VNCNetworkAdvertiser`

Field Summary

Fields

Modifier and Type	Field and Description
`int`	**`depth`** The colour depth of the icon, in bits per pixel.
`int`	**`height`** The height of the icon, in pixels.
`java.lang.String`	**`mimeType`** The MIME type of the icon image.
`int`	**`width`** The width of the icon, in pixels.

Constructor Summary

Constructors

Constructor and Description
`VNCNetworkAdvertiserIconDetails``(java.lang.String mimeType, int width, int height, int depth)` Constructs a new VNCNetworkAdvertiserIconDetails object with the given icon details.

Method Summary

Methods inherited from class java.lang.Object

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

Field Detail

mimeType

```
public final java.lang.String mimeType
```

The MIME type of the icon image.

For example, `"image/png"`.

width

```
public final int width
```

The width of the icon, in pixels.

height

```
public final int height
```

The height of the icon, in pixels.

depth

```
public final int depth
```

The colour depth of the icon, in bits per pixel.

Constructor Detail

VNCNetworkAdvertiserIconDetails

```
public VNCNetworkAdvertiserIconDetails(java.lang.String mimeType,
                                       int width,
                                       int height,
                                       int depth)
```

Constructs a new VNCNetworkAdvertiserIconDetails object with the given icon details.

com.realvnc.networkadvertisersdk

Interface VNCNetworkAdvertiserIcon

`public interface` **`VNCNetworkAdvertiserIcon`**

Represents an icon used by a VNC Network Advertiser.

This is an opaque object that may be used in various methods defined in `VNCNetworkAdvertiser`. Use `VNCNetworkAdvertiser.iconCreate(VNCNetworkAdvertiserIconDetails,byte[])` to create an instance of this object.

See Also:

`VNCNetworkAdvertiser`

com.realvnc.networkadvertisersdk

Interface VNCNetworkAdvertiserListener

`public interface` **`VNCNetworkAdvertiserListener`**

Callback interface used by VNC Network Advertisers to provide notifications to your application.

Threading

Almost all listener methods are always invoked by the UPnP device thread. You must therefore avoid including code in your implementation of this interface that may disrupt the UPnP device thread's activities.

The only exception to this rule is `log(String,int,String)`, which the SDK may call from any thread. The SDK may also call `log(String,int,String)` before the UPnP device thread is started. However, the SDK guarantees that no two threads call this method at the same time, even if there are multiple VNC Network Advertiser instances.

As the listener methods are invoked on threads other than your Android application's UI thread, you must take care not to call Android APIs which must only be invoked on the UI thread. Instead, you may wish create an `Handler` on the UI thread, and then send it an `Message` from your listener method implementation. However, you must never block in a listener method implementation, waiting for a response from the UI thread.

See Also:

`VNCNetworkAdvertiser`, `VNCNetworkAdvertiserSDK`

Method Summary

Methods

Modifier and Type	Method and Description
`void`	**`error`**`(int error)` Called by the SDK to notify of an error.
`void`	**`globalServerCommandStringRequest`**`(java.lang.String interfaceName)` Called by the SDK when a VNC Network Discoverer requests the command string for the VNC server.
`void`	**`interfaceUnregistered`**`(java.lang.String interfaceName)` Called by the SDK when a network interface has been automatically disassociated from a VNC Network Advertiser.
`void`	**`listening`**`(java.lang.String interfaceName, int port)` Called by the SDK when it has started listening on a network interface.
`void`	**`log`**`(java.lang.String category, int severity, java.lang.String text)` Called by the SDK to allow logging from the SDK's internals.
`void`	**`threadStarted`**`()` Called by the SDK immediately after the UPnP device thread has started.

Method Detail

log

```
void log(java.lang.String category,
         int severity,
         java.lang.String text)
```

Called by the SDK to allow logging from the SDK's internals.

The information logged by this callback is intended for use by application developers and RealVNC support engineers. It is not localized.

Use this callback in conjunction with `VNCNetworkAdvertiserParameter.LOG` to select and gather logging information from the SDK.

Note: The SDK may invoke this callback from any thread, including calls made before the UPnP device thread is started. However, the SDK guarantees that no two threads will call this method at the same time, even if there are multiple VNC Network Advertiser instances.

Parameters:

`category` - The category of the log message.

`severity` - The severity of the log message (see `VNCNetworkAdvertiserParameter.LOG` for an explanation).

`text` - The text of the log message.

See Also:

`VNCNetworkAdvertiserParameter.LOG`, `VNCNetworkAdvertiserParameter.DEFAULT_VALUE_LOG`

error

```
void error(int error)
```

Called by the SDK to notify of an error.

If you explicitly stop a running VNC Network Advertiser with `VNCNetworkAdvertiser.stop()`, then this method will be called with with the error code `VNCNetworkAdvertiserException.STOPPED`. This allows you to use the same cleanup code for this case as for error cases.

See `VNCNetworkAdvertiserException` for a list of possible error codes.

Parameters:

`error` - The error code.

See Also:

`VNCNetworkAdvertiserException`

threadStarted

```
void threadStarted()
```

Called by the SDK immediately after the UPnP device thread has started.

This callback is guaranteed to be the first callback from the UPnP device thread.

listening

```
void listening(java.lang.String interfaceName,
               int port)
```

Called by the SDK when it has started listening on a network interface.

The SDK shall invoke this callback each time it starts listening on a new network interface. This callback shall only be invoked more than once for a particular interface if:

- The UPnP device thread is stopped and restarted, or
- That interface has been unregistered and re-registered.

The TCP port that the SDK is listening on is provided to this callback. This shall be the port specified in the call to `VNCNetworkAdvertiser.registerInterface(String,int)`, if non-zero. Otherwise, this shall be the port chosen by the SDK for this network interface.

Parameters:

`interfaceName` - The platform-specific name identifying the interface. This shall be an interface associated with the VNC Network Advertiser instance, which was registered using `VNCNetworkAdvertiser.registerInterface(String,int)`.

`port` - The TCP port that the SDK is listening on.

interfaceUnregistered

```
void interfaceUnregistered(java.lang.String interfaceName)
```

Called by the SDK when a network interface has been automatically disassociated from a VNC Network Advertiser.

The SDK will automatically disassociate a network interface when it encounters an error interacting with that interface. This may be caused by a transient network interface having been removed from the system, or by the non-zero TCP port specified in the call to `VNCNetworkAdvertiser.registerInterface(String,int)` not being accessible. Such errors are not considered fatal, and the UPnP device thread will continue running even when there are no network interfaces associated with the VNC Network Advertiser.

If the network interface becomes available again, then you will need to explicitly re-associate it with `VNCNetworkAdvertiser.registerInterface(String,int)` if you wish the VNC Network Advertiser to use the interface again.

The SDK shall not invoke this callback for interfaces unregistered by the application, using `VNCNetworkAdvertiser.unregisterInterface(String)`. However, it is possible that the SDK may invoke this callback before the call to `VNCNetworkAdvertiser.unregisterInterface(String)` has been handled.

Parameters:

`interfaceName` - The platform-specific name identifying the interface. This shall be an interface associated with the VNC Network Advertiser instance, which was registered using `VNCNetworkAdvertiser.registerInterface(String,int)`.

globalServerCommandStringRequest

```
void globalServerCommandStringRequest(java.lang.String interfaceName)
```

Called by the SDK when a VNC Network Discoverer requests the command string for the VNC server.

You must implement this callback to perform the actions necessary to obtain a suitable VNC command string for the VNC server, which the VNC Network Discoverer may then use to connect on the given network interface. Examples of such actions may include:

- Determining if the VNC server is already connected to another VNC viewer, and is therefore busy.
- Ensuring that the VNC server to listen for incoming connections on the network interface.

A VNC command string is a URI whose scheme is 'vnccmd'. The scheme-specific part of the URI (i.e. the section following the 'vnccmd:') is a semicolon-separated sequence of field names and values. Each field name is separated from its value by an equals sign. You will most likely provide a command string which uses the following form:

```
vnccmd:v=1;t=C;a=192.168.42.129;p=5900
```

This command string will instruct the VNC Network Discoverer's associated RealVNC Mobile Solution VNC viewer to connect to the VNC server listening on IP address 192.168.42.129 and TCP port 5900. The IP address must be reachable from the network on which the VNC Network Discoverer is situated. If appropriate, you may instead provide a command string which identifies an alternative transport. Please refer to the VNC Viewer SDK for more information on VNC command strings.

Once the result of the attempt to retrieve the VNC command string is known, your application must call `VNCNetworkAdvertiser.globalServerCommandStringResult(String,String)` to notify the SDK. In this call, you may provide either of the following as a successful response:

- A valid VNC command string to report to the VNC Network Discoverer.
- An empty string, to indicate to the VNC Network Discoverer that the VNC server is running, but is currently busy and so may not accept a connection at this time.

If an error occurs, you may call `VNCNetworkAdvertiser.globalServerCommandStringResult(String,String)` with the `commandString` parameter set to `null`. This will cause the VNC Network Advertiser to report an error to the VNC Network Discoverer for the request.

Parameters:

`interfaceName` - The platform-specific name identifying the interface being used to communicate with the VNC Network Discoverer. This shall be an interface associated with the VNC Network Advertiser instance, which was registered using `VNCNetworkAdvertiser.registerInterface(String,int)`.

com.realvnc.networkadvertisersdk

Class VNCNetworkAdvertiserParameter

java.lang.Object
    com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserParameter

---

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

Defines constants representing configuration parameters that can be used with a VNC Network Advertiser.

All parameter values are passed to and from the SDK as strings. If a parameter is listed as having an integer value, then the string should contain the value formatted in decimal. If a parameter is listed as having a Boolean value, the string should have the value "0" or "1".

Constants are also provided which define the default value of the parameters. For example, the default value of `LOG` is given by `DEFAULT_VALUE_LOG`.

See `VNCNetworkAdvertiser.setParameter(String,String)` and `VNCNetworkAdvertiser.getParameter(String)` for further information on configuration parameters.

See Also:

`VNCNetworkAdvertiser`

Field Summary

Fields

Modifier and Type	Field and Description
`static java.lang.String`	`DEFAULT_VALUE_LOG` The default value for the logging parameter.
`static java.lang.String`	`DEFAULT_VALUE_SSDP_ADVERTISEMENT_EXPIRY_DURATION` The default value for the SSDP advertisement expiry duration parameter.
`static java.lang.String`	`DEFAULT_VALUE_SSDP_ADVERTISEMENT_INTERVAL` The default value for the SSDP advertisement interval parameter.
`static java.lang.String`	`LOG` Controls the SDK's logging.
`static java.lang.String`	`SSDP_ADVERTISEMENT_EXPIRY_DURATION` Sets the SSDP advertisement expiry duration.
`static java.lang.String`	`SSDP_ADVERTISEMENT_INTERVAL` Sets the interval between successive multicasts of SSDP advertisements.

Constructor Summary

Constructors

Constructor and Description
`VNCNetworkAdvertiserParameter()`

Method Summary

Methods inherited from class java.lang.Object

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

Field Detail

LOG

`public static final java.lang.String LOG`

Controls the SDK's logging.

If the value is an empty string, then all logging is disabled. If the string is non-empty, then it is a comma-separated list of directives of the form `category:severity`.

`category` may be `"*"`, which means that the selected severity is to be used for all log categories. The category identifies the internal component reporting the log message. Normally, you'll want to include all log categories.

`severity` is an integer from 0 to 100 inclusive. The SDK defines these logging severities, but may use other values in between:

- `0` - serious errors
- `10` - warning and connection status messages
- `30` - informational messages
- `100` - verbose debugging

Changes to this parameter will take effect even if the UPnP device thread is already running.

String.

See Also:

`VNCNetworkAdvertiserListener.log(String,int,String)`, Constant Field Values

SSDP_ADVERTISEMENT_EXPIRY_DURATION

`public static final java.lang.String SSDP_ADVERTISEMENT_EXPIRY_DURATION`

Sets the SSDP advertisement expiry duration.

This specifies the number of seconds that SSDP advertisements sent by the VNC Network Advertiser remain valid for. When an SSDP advertisement expires, it is automatically refreshed. The frequency of SSDP advertisements sent before expiry can be controlled with `SSDP_ADVERTISEMENT_INTERVAL`.

Integer, minimum 5.

See Also:

Constant Field Values

SSDP_ADVERTISEMENT_INTERVAL

`public static final java.lang.String SSDP_ADVERTISEMENT_INTERVAL`

Sets the interval between successive multicasts of SSDP advertisements.

This specifies the minimum number of seconds that the VNC Network Advertiser should wait before repeating a multicast of an SSDP advertisement before its expiry. If set to zero, or set to a value greater than `SSDP_ADVERTISEMENT_EXPIRY_DURATION`, then no repeat multicasts are sent until expiry of the SSDP advertisement.

Integer, minimum 0.

See Also:

Constant Field Values

DEFAULT_VALUE_LOG

public static final java.lang.String DEFAULT_VALUE_LOG

The default value for the logging parameter.

With this value, logging is disabled.

See LOG.

See Also:

Constant Field Values

DEFAULT_VALUE_SSDP_ADVERTISEMENT_EXPIRY_DURATION

public static final java.lang.String DEFAULT_VALUE_SSDP_ADVERTISEMENT_EXPIRY_DURATION

The default value for the SSDP advertisement expiry duration parameter.

See SSDP_ADVERTISEMENT_EXPIRY_DURATION.

See Also:

Constant Field Values

DEFAULT_VALUE_SSDP_ADVERTISEMENT_INTERVAL

public static final java.lang.String DEFAULT_VALUE_SSDP_ADVERTISEMENT_INTERVAL

The default value for the SSDP advertisement interval parameter.

See SSDP_ADVERTISEMENT_INTERVAL.

See Also:

Constant Field Values

Constructor Detail

VNCNetworkAdvertiserParameter

public VNCNetworkAdvertiserParameter()

com.realvnc.networkadvertisersdk

Class VNCNetworkAdvertiserProperty

java.lang.Object
    com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserProperty

---

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

Defines properties of a VNC Network Advertiser that may be queried.

Integer-valued properties must be queried with `VNCNetworkAdvertiser.getProperty(int)`. String-valued properties must be queried with `VNCNetworkAdvertiser.getPropertyString(int)`.

See Also:

`VNCNetworkAdvertiser`

Field Summary

Fields

Modifier and Type	Field and Description
`static int`	`IS_RUNNING` Whether or not the UPnP device thread is running (integer).

Constructor Summary

Constructors

Constructor and Description
`VNCNetworkAdvertiserProperty()`

Method Summary

Methods inherited from class java.lang.Object

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

Field Detail

IS_RUNNING

```
public static final int IS_RUNNING
```

Whether or not the UPnP device thread is running (integer).

The value is non-zero if the VNC Network Advertiser has been started, and not yet stopped. Otherwise, the value is zero.

See Also:

Constant Field Values

Constructor Detail

VNCNetworkAdvertiserProperty

```
public VNCNetworkAdvertiserProperty()
```

com.realvnc.networkadvertisersdk

Class VNCNetworkAdvertiserSDK

java.lang.Object
    com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserSDK

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

SDK class handling network advertiser instantiation and providing various helper functions.

The VNC Network Advertiser SDK allows you to create and manipulate VNC Network Advertiser objects. A VNC Network Advertiser uses UPnP to advertise a RealVNC Mobile Solution VNC server on a network. RealVNC Mobile Solution VNC viewer applications can use the VNC Network Discoverer to discover this VNC server and connect to it.

Creating a VNC Network Advertiser object

- Create an object implementing the `VNCNetworkAdvertiserListener` interface.
- Call `advertiserCreate(android.content.Context,VNCNetworkAdvertiserListener)` to create a `VNCNetworkAdvertiser` object.

Legal information

Details of and copyright notices for third-party software that is used by the VNC Android Network Advertiser SDK can be found in the file `Acknowledgements.txt` in the SDK distribution.

See Also:

`VNCNetworkAdvertiser`, `VNCNetworkAdvertiserListener`

Constructor Summary

Constructors

Constructor and Description
`VNCNetworkAdvertiserSDK()`

Method Summary

Methods

Modifier and Type	Method and Description
`static VNCNetworkAdvertiser`	`advertiserCreate(android.content.Context context, VNCNetworkAdvertiserListener listener)` Creates a new VNC Network Advertiser instance.
`static java.lang.String`	`getErrorName(int error)` Returns a symbolic name corresponding to an SDK error code.
`static int`	`getVersionBuild()` Provides the build number component of the SDK's version number.
`static int`	`getVersionMajor()`

	Provides the major component of the SDK's version number.
static int	**getVersionMinor**() Provides the minor component of the SDK's version number.
static int	**getVersionPatch**() Provides the patch component of the SDK's version number.

Methods inherited from class java.lang.Object

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

Constructor Detail

VNCNetworkAdvertiserSDK

```
public VNCNetworkAdvertiserSDK()
```

Method Detail

advertiserCreate

```
public static VNCNetworkAdvertiser advertiserCreate(android.content.Context context,
                                    VNCNetworkAdvertiserListener listener)
                                             throws VNCNetworkAdvertiserException
```

Creates a new VNC Network Advertiser instance. This method creates and returns a new, uninitialized `VNCNetworkAdvertiser` object.

Parameters:

`context` - The Android context to use. Normally, you should create and control the VNC Network Advertiser from an Android Service; this parameter should be your service object.

`listener` - The `VNCNetworkAdvertiserListener` that should be used by the SDK for this instance.

Returns:

The new `VNCNetworkAdvertiser` instance.

Throws:

`VNCNetworkAdvertiserException` - if an error occurs.

getVersionMajor

```
public static int getVersionMajor()
```

Provides the major component of the SDK's version number.

Returns:

The major component of the version number.

getVersionMinor

```
public static int getVersionMinor()
```

Provides the minor component of the SDK's version number.

Returns:

The minor component of the version number.

getVersionPatch

```
public static int getVersionPatch()
```

Provides the patch component of the SDK's version number.

Returns:

The patch component of the version number.

getVersionBuild

```
public static int getVersionBuild()
```

Provides the build number component of the SDK's version number.

Returns:

The build number component of the version number.

getErrorName

```
public static java.lang.String getErrorName(int error)
```

Returns a symbolic name corresponding to an SDK error code.

For example, if the error is `VNCNetworkAdvertiserException.INVALID_PARAMETER`, then the returned string is "InvalidParameter". Such strings may be useful as internationalization keys or in log entries.

The strings returned by this method are brief and are not internationalized. They are therefore not suitable for display to an end user.

Parameters:

`error` - The error code.

Returns:

A short string corresponding to the error code. If the error code is not recognized by the SDK, then the return value is `null`.

com.realvnc.networkadvertisersdk

Class VNCNetworkAdvertiserServerDetails

java.lang.Object
    com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserServerDetails

---

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

Describes a VNC server.

Construct an instance of this class to pass server details to a VNC Network Advertiser instance using `VNCNetworkAdvertiser.setGlobalServerDetails(VNCNetworkAdvertiserServerDetails,VNCNetworkAdvertiserIcon[])`.

See Also:

`VNCNetworkAdvertiser`

Field Summary

Fields

Modifier and Type	Field and Description
`java.lang.String`	`manufacturer` The manufacturer of the server.
`java.lang.String`	`name` The name of the server.
`java.lang.String`	`version` The version of the server.

Constructor Summary

Constructors

Constructor and Description
`VNCNetworkAdvertiserServerDetails(java.lang.String name, java.lang.String version, java.lang.String manufacturer)` Constructs a new VNCNetworkAdvertiserServerDetails object with the given server details.

Method Summary

Methods inherited from class java.lang.Object

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

Field Detail

name

```
public final java.lang.String name
```

The name of the server.

version

```
public final java.lang.String version
```

The version of the server.

manufacturer

```
public final java.lang.String manufacturer
```

The manufacturer of the server.

Constructor Detail

VNCNetworkAdvertiserServerDetails

```
public VNCNetworkAdvertiserServerDetails(java.lang.String name,
                                         java.lang.String version,
                                         java.lang.String manufacturer)
```

Constructs a new VNCNetworkAdvertiserServerDetails object with the given server details.

How This API Document Is Organized

This API (Application Programming Interface) document has pages corresponding to the items in the navigation bar, described as follows.

Overview

The Overview page is the front page of this API document and provides a list of all packages with a summary for each. This page can also contain an overall description of the set of packages.

Package

Each package has a page that contains a list of its classes and interfaces, with a summary for each. This page can contain six categories:

- Interfaces (italic)
- Classes
- Enums
- Exceptions
- Errors
- Annotation Types

Class/Interface

Each class, interface, nested class and nested interface has its own separate page. Each of these pages has three sections consisting of a class/interface description, summary tables, and detailed member descriptions:

- Class inheritance diagram
- Direct Subclasses
- All Known Subinterfaces
- All Known Implementing Classes
- Class/interface declaration
- Class/interface description
- Nested Class Summary
- Field Summary
- Constructor Summary
- Method Summary
- Field Detail
- Constructor Detail
- Method Detail

Each summary entry contains the first sentence from the detailed description for that item. The summary entries are alphabetical, while the detailed descriptions are in the order they appear in the source code. This preserves the logical groupings established by the programmer.

Annotation Type

Each annotation type has its own separate page with the following sections:

- Annotation Type declaration
- Annotation Type description
- Required Element Summary
- Optional Element Summary
- Element Detail

Enum

Each enum has its own separate page with the following sections:

- Enum declaration
- Enum description
- Enum Constant Summary
- Enum Constant Detail

Tree (Class Hierarchy)

There is a Class Hierarchy page for all packages, plus a hierarchy for each package. Each hierarchy page contains a list of classes and a list of interfaces. The classes are organized by inheritance structure starting with `java.lang.Object`. The interfaces do not inherit from `java.lang.Object`.

- When viewing the Overview page, clicking on "Tree" displays the hierarchy for all packages.
- When viewing a particular package, class or interface page, clicking "Tree" displays the hierarchy for only that package.

Deprecated API

The Deprecated API page lists all of the API that have been deprecated. A deprecated API is not recommended for use, generally due to improvements, and a replacement API is usually given. Deprecated APIs may be removed in future implementations.

Index

The Index contains an alphabetic list of all classes, interfaces, constructors, methods, and fields.

Prev/Next

These links take you to the next or previous class, interface, package, or related page.

Frames/No Frames

These links show and hide the HTML frames. All pages are available with or without frames.

All Classes

The All Classes link shows all classes and interfaces except non-static nested types.

Serialized Form

Each serializable or externalizable class has a description of its serialization fields and methods. This information is of interest to re-implementors, not to developers using the API. While there is no link in the navigation bar, you can get to this information by going to any serialized class and clicking "Serialized Form" in the "See also" section of the class description.

Constant Field Values

The Constant Field Values page lists the static final fields and their values.

This help file applies to API documentation generated using the standard doclet.

A C D E F G H I L M N P R S T U V W

A

C

D

E

F

G

H

I

L

M

N

P

R

S

T

U

Unregisters a network interface from this VNC Network Advertiser.

V

version - Variable in class com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserServerDetails

The version of the server.

VNCNetworkAdvertiser - Interface in com.realvnc.networkadvertisersdk

Interface implemented by VNC Network Advertiser instances.

VNCNetworkAdvertiserDeviceIdentity - Class in com.realvnc.networkadvertisersdk

Specifies identifying information for a device.

VNCNetworkAdvertiserDeviceIdentity(String, String, String) - Constructor for class com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserDeviceIdentity

Constructs a new VNCNetworkAdvertiserDeviceIdentity object with the given identity details.

VNCNetworkAdvertiserException - Exception in com.realvnc.networkadvertisersdk

VNC Network Advertiser exception class indicating an error.

VNCNetworkAdvertiserException(int) - Constructor for exception com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserException

Constructs a new VNC Network Advertiser exception instance, with the specified error code.

VNCNetworkAdvertiserException(int, String) - Constructor for exception com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserException

Constructs a new VNC Network Advertiser exception instance, with the specified error code.

VNCNetworkAdvertiserIcon - Interface in com.realvnc.networkadvertisersdk

Represents an icon used by a VNC Network Advertiser.

VNCNetworkAdvertiserIconDetails - Class in com.realvnc.networkadvertisersdk

Describes the basic details of an icon.

VNCNetworkAdvertiserIconDetails(String, int, int, int) - Constructor for class com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserIconDetails

Constructs a new VNCNetworkAdvertiserIconDetails object with the given icon details.

VNCNetworkAdvertiserListener - Interface in com.realvnc.networkadvertisersdk

Callback interface used by VNC Network Advertisers to provide notifications to your application.

VNCNetworkAdvertiserParameter - Class in com.realvnc.networkadvertisersdk

Defines constants representing configuration parameters that can be used with a VNC Network Advertiser.

VNCNetworkAdvertiserParameter() - Constructor for class com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserParameter

VNCNetworkAdvertiserProperty - Class in com.realvnc.networkadvertisersdk

Defines properties of a VNC Network Advertiser that may be queried.

VNCNetworkAdvertiserProperty() - Constructor for class com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserProperty

VNCNetworkAdvertiserSDK - Class in com.realvnc.networkadvertisersdk

SDK class handling network advertiser instantiation and providing various helper functions.

VNCNetworkAdvertiserSDK() - Constructor for class com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserSDK

VNCNetworkAdvertiserServerDetails - Class in com.realvnc.networkadvertisersdk

Describes a VNC server.

VNCNetworkAdvertiserServerDetails(String, String, String) - Constructor for class com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserServerDetails

Constructs a new VNCNetworkAdvertiserServerDetails object with the given server details.

W

width - Variable in class com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserIconDetails

The width of the icon, in pixels.

wifiMacAddr - Variable in class com.realvnc.networkadvertisersdk.VNCNetworkAdvertiserDeviceIdentity

The WiFi MAC address of the device.

A C D E F G H I L M N P R S T U V W