SES Platform Services

Media Platform Solutions | AS-20502

Novaspread-S

Reference manual

Version:

0.16

Date issued:

29 May 2015

Important Notice

This document has been produced by SES Platform Services GmbH (SES PS). Certain product names or brand names may be trademarks or designations of their respective owners.

Liability/Copyright

Cooperation

This document has been developed in cooperation with:

TARA Systems GmbH

Gmunder Str. 53

81379 München

Germany

TABLE OF CONTENT

1. Introduction 6
1.1. Purpose of document 6
1.2. Document history 6
1.3. References 6

2. Provided API 7
2.1. Novaspread Basic Types 7
2.2. NovaspreadServer 7
2.2.1. NovaspreadTServerInitParameter 8
2.2.2. NovaspreadServerInit 8
2.2.3. NovaspreadServerDone 8
2.2.4. NovaspreadServerProcess 9
2.2.5. NovaspreadServerFactoryReset 9
2.2.6. NovaspreadServerSetFriendlyName 9
2.2.7. NovaspreadServerGetDeviceList 10
2.2.8. NovaspreadServerSelectDevice 10
2.2.9. NovaspreadServerGetSelectedDevice 10
2.2.10. NovaspreadServerCreateSatIpTuner 11

2.3. NovaspreadDeviceList 11
2.3.1. NovaspreadDeviceListRelease 11
2.3.2. NovaspreadDeviceListGetLength 11
2.3.3. NovaspreadDeviceListGetDevice 12

2.4. NovaspreadDevice 12
2.4.1. NovaspreadDeviceRelease 12
2.4.2. NovaspreadDeviceGetIpAddress 13
2.4.3. NovaspreadDeviceGetFriendlyName 13
2.4.4. NovaspreadDeviceGetManufacturer 13
2.4.5. NovaspreadDeviceGetManufacturerUrl 14
2.4.6. NovaspreadDeviceGetModelDescription 14
2.4.7. NovaspreadDeviceGetModelNumber 14
2.4.8. NovaspreadDeviceGetModelUrl 14
2.4.9. NovaspreadDeviceGetSerialNumber 15
2.4.10. NovaspreadDeviceGetIconList 15

2.5. NovaspreadIconList 15
2.5.1. NovaspreadIconListRelease 16
2.5.2. NovaspreadIconListGetLength 16
2.5.3. NovaspreadIconListGetIcon 16

2.6. NovaspreadIcon 17
2.6.1. NovaspreadIconRelease 17
2.6.2. NovaspreadIconGetMimeType 17
2.6.3. NovaspreadIconGetWidth 17
2.6.4. NovaspreadIconGetHeight 18
2.6.5. NovaspreadIconGetDepth 18
2.6.6. NovaspreadIconGetUrl 18

2.7. NovaspreadTunerParameters 18
2.7.1. NovaspreadTTunerType 19
2.7.2. NovaspreadTTunerCodeRate 19
2.7.3. NovaspreadTTunerModulationSystem 20
2.7.4. NovaspreadTTunerModulation 20
2.7.5. NovaspreadTTunerParamDvbS 21
2.7.6. NovaspreadTTunerParamValue 22

2.7.7. NovaspreadTTunerParameters 22
2.7.8. NovaspreadTTunerSignalInfo 22

2.8. NovaspreadTranscoding 23
2.8.1. NOVASPREAD_PID_UNKNOWN 23
2.8.2. NovaspreadTVideoCodec 23
2.8.3. NovaspreadTVideoResolution 24
2.8.4. NovaspreadTAudioCodec 24
2.8.5. NovaspreadTTranscodingInput 25
2.8.6. NovaspreadTTranscodingOutput 25
2.8.7. NovaspreadTTranscoding 26
2.8.8. NovaspreadTVideoTranscodingCapability 26
2.8.9. NovaspreadTAudioTranscodingCapability 27

2.9. NovaspreadSatIpTuner 27
2.9.1. NovaspreadTSatIpTunerConnectionStatus 27
2.9.2. NovaspreadTSatIpTunerConnectionStatusChangeListener 28
2.9.3. NovaspreadTSatIpTunerDataAvailableListener 28
2.9.4. NovaspreadSatIpTunerDestroy 29
2.9.5. NovaspreadSatIpTunerSetParameters 29
2.9.6. NovaspreadSatIpTunerGetParameters 29
2.9.7. NovaspreadSatIpTunerConnect 30
2.9.8. NovaspreadSatIpTunerDisconnect 30
2.9.9. NovaspreadSatIpTunerGetConnectionStatus 31
2.9.10. NovaspreadSatIpTunerSetConnectionStatusChangeListener 31
2.9.11. NovaspreadSatIpTunerSetPids 32
2.9.12. NovaspreadSatIpTunerGetPids 32
2.9.13. NovaspreadSatIpTunerAddPids 33
2.9.14. NovaspreadSatIpTunerRemovePids 33
2.9.15. NovaspreadSatIpTunerIsLocked 33
2.9.16. NovaspreadSatIpTunerGetSignalInfo 34
2.9.17. NovaspreadSatIpTunerSetDataAvailableListener 34
2.9.18. NovaspreadSatIpTunerReadData 35

2.10. NovaspreadCaInfo 35
2.10.1. NovaspreadTCaInfoSmartcardStatus 35
2.10.2. NovaspreadCaInfoCreate 36
2.10.3. NovaspreadCaInfoDestroy 36
2.10.4. NovaspreadCaInfoSetChipsetUid 36
2.10.5. NovaspreadCaInfoSetChipsetType 37
2.10.6. NovaspreadCaInfoSetChipsetRevision 37
2.10.7. NovaspreadCaInfoSetCaVendor 38
2.10.8. NovaspreadCaInfoSetCaVersion 38
2.10.9. NovaspreadCaInfoSetCaNumber 38
2.10.10. NovaspreadCaInfoSetSmartcardInserted 39
2.10.11. NovaspreadCaInfoSetSmartcardSuitable 39
2.10.12. NovaspreadCaInfoSetSmartcardType 40
2.10.13. NovaspreadCaInfoSetSmartcardNumber 40
2.10.14. NovaspreadCaInfoSetSmartcardStatus 40
2.10.15. NovaspreadCaInfoSetExpirationDate 41

3. Required API 42
3.1. NovaspreadHost 42
3.1.1. NovaspreadTHostCapabilities 42
3.1.2. NovaspreadHostGetCapabilities 43
3.1.3. NovaspreadHostAllocateTuner 43
3.1.4. NovaspreadHostReleaseTuner 44

3.2. NovaspreadTuner 44
3.2.1. NovaspreadTTunerError 45
3.2.2. NovaspreadTTunerState 45
3.2.3. NovaspreadTTunerStateChangeListener 46
3.2.4. NovaspreadTTunerDataAvailableListener 46

3.2.5. NovaspreadTTunerAllocationParameter 46
3.2.6. NovaspreadTTunerAllocationFinishedListener 47
3.2.7. NovaspreadTTunerReleaseRequestedListener 47
3.2.8. NovaspreadTunerGetTransportSessionId 47
3.2.9. NovaspreadTunerSetTranscoding 48
3.2.10. NovaspreadTunerSetStateChangeListener 48
3.2.11. NovaspreadTunerGetError 49
3.2.12. NovaspreadTunerSetPids 49
3.2.13. NovaspreadTunerAddPids 50
3.2.14. NovaspreadTunerRemovePids 50
3.2.15. NovaspreadTunerStart 51
3.2.16. NovaspreadTunerStop 51
3.2.17. NovaspreadTunerIsLocked 51
3.2.18. NovaspreadTunerGetSignalInfo 52
3.2.19. NovaspreadTunerSetDataAvailableListener 52
3.2.20. NovaspreadTunerReadData 53

3.3. NovaspreadCa 53
3.3.1. NovaspreadTCaDvbId 53
3.3.2. NovaspreadTCaPlatformUsageRulesReceivedListener 54
3.3.3. NovaspreadTCaServiceUsageRulesReceivedListener 54
3.3.4. NovaspreadCaGetInfo 55
3.3.5. NovaspreadCaSetDvbId 55
3.3.6. NovaspreadCaSetPlatformUsageRulesReceivedListener 56
3.3.7. NovaspreadCaSetServiceUsageRulesReceivedListener 56

3.4. NovaspreadDrm 57
3.4.1. NovaspreadTDrmLicense 57
3.4.2. NovaspreadTDrmLicenseParameter 57
3.4.3. NovaspreadDrmStart 58
3.4.4. NovaspreadDrmStop 58

1. INTRODUCTION

1.1. Purpose of document

This document describes the required and provided interfaces of Novaspread-S in the scope of the Multiscreen product of SES Platform Services.

1.2. Document history

Version	Date	Author	Changes
0.16	2015-05-29	Manfred Schmidt Georg Kamjunke	Working draft

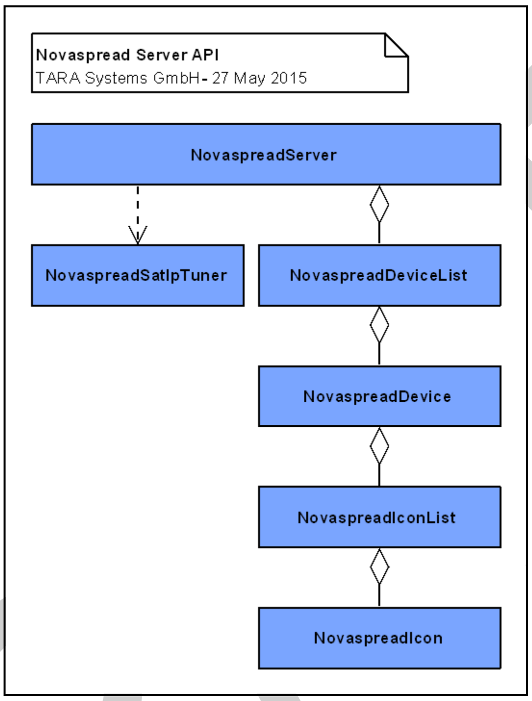
1.3. References

[1] SPS; "AS-20001: Multiscreen”
[2] SPS; "AS-20501: Novaspread-S"

2. PROVIDED API

The following section describes the Application Programming Interface (API) which is provided by Novaspread-S. To use the interface in an application include the NovaspreadServer.h.

The following diagram gives an overview of the classes provided by Novaspread-S.



2.1. Novaspread Basic Types

Novaspread uses the following basic types:

`NovaspreadTInt8`	– A 8-bit signed integer
`NovaspreadTInt16`	– A 16-bit signed integer
`NovaspreadTInt32`	– A 32-bit signed integer
`NovaspreadTUInt8`	– A 8-bit unsigned integer
`NovaspreadTUInt16`	– A 16-bit unsigned integer
`NovaspreadTUInt32`	– A 32-bit unsigned integer
`NovaspreadTBoolean`	– A boolean type which can have the values `NOVASPREAD_TRUE` or `NOVASPREAD_FALSE`
`NOVASPREAD_NULL`	– A null-reference

2.2. NovaspreadServer

A NovaspreadServer represents the Novaspread-S main class. Before the NovaspreadServer can be used it must be initialised with the function NovaspreadServerInit(). With the function NovaspreadServerDone() the NovaspreadServer is shutdown again.

With the function NovaspreadServerSetFriendlyName() the name of the SAT>IP server provided by the NovaspreadServer is defined.

To access other (standard) SAT>IP servers a device list of all SAT>IP servers can be retrieved with the function NovaspreadServerGetDeviceList(). One of these SAT>IP servers is selected with the function

NovaspreadServerSelectDevice(). SAT>IP tuners are only accessed from the selected SAT>IP server. A SAT>IP tuner can be allocated with the function NovaspreadServerCreateSatIpTuner() and used by the host device to receive parts of a transport streams.

2.2.1. NovaspreadTServerInitParameter

This type defines initialization parameters for NovaspreadServer. It is recommended to initialize this struct with all 0 (see example below).

SYNTAX

```
typedef struct
{
  const char * DataPath;

} NovaspreadTServerInitParameter;
```

COMPONENTS

DataPath
Path to a directory within the local file system, were NovaspreadServer can store its configuration data.

EXAMPLE

```
NovaspreadTServerInitParameter initParameters;

 memset( &initParameters, 0, sizeof( initParameters ));
 initParameters.DataPath = "/data";
 NovaspreadServerInit( &initParameters );
```

2.2.2. NovaspreadServerInit

This function initializes the NovaspreadServer. It must be called once, before any other NovaspreadServer function is called.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadServerInit(
  NovaspreadTServerInitParameter * aInitParameter );
```

PARAMETERS

aInitParameter
The initialization parameter for NovaspreadServer.

RETURN VALUE

NOVASPREAD_TRUE
if successful.

NOVASPREAD_FALSE
otherwise.

SEE ALSO

NovaspreadTServerInitParameter
NovaspreadServerDone()

2.2.3. NovaspreadServerDone

This function shuts down the NovaspreadServer. After this function is called no other functions of the NovaspreadServer shall be called.

Syntax

```
PUBLIC void
NovaspreadServerDone(
  void );
```

See Also

```
NovaspreadServerInit()
```

2.2.4. NovaspreadServerProcess

This function must be called periodically to process the NovaspreadServer.

Syntax

```
PUBLIC NovaspreadTBoolean
NovaspreadServerProcess(
  void );
```

Return Value

NOVASPREAD_TRUE
  if there is still something to process and this function should be called again.

NOVASPREAD_FALSE
  if nothing is available to process.

2.2.5. NovaspreadServerFactoryReset

Resets the NovaspradServer to factory defaults.

Syntax

```
PUBLIC NovaspreadTBoolean
NovaspreadServerFactoryReset(
  void );
```

Return Value

NOVASPREAD_TRUE
  if successful.

NOVASPREAD_FALSE
  otherwise.

2.2.6. NovaspreadServerSetFriendlyName

Sets the friendly name used by the SAT>IP Server. If the friendly name is changed by calling this function the SAT>IP UPnP device is restarted to advertise the new name in the UPnP device description. Ongoing streamings may be stopped.

Syntax

```
PUBLIC NovaspreadTBoolean
NovaspreadServerSetFriendlyName(
  const char * aFriendlyName );
```

Parameters

aFriendlyName
  The friendly name as UTF-8 encoded string.

Return Value

NOVASPREAD_TRUE
  if successful.

NOVASPREAD_FALSE
otherwise.

2.2.7. NovaspreadServerGetDeviceList

Gets the currently available list of SAT>IP Servers. The DeviceList contains all (standard) SAT>IP server devices which have been detected in the local network at the moment when the function is called. The returned list is a copy and does not change, if e.g. a new SAT>IP server was found after getting the list. To get an updated list, the list must be released and retrieved again with this function. The SAT>IP Server provided by the NovaspreadServer is excluded from this list to avoid self-referencing.

When the DeviceList is no longer used, the function NovaspreadDeviceListRelease() must be called to release it.

SYNTAX

```
PUBLIC NovaspreadTDeviceList
NovaspreadServerGetDeviceList(
  void );
```

RETURN VALUE

A new DeviceList. NOVASPREAD_NULL if an error occurred.

SEE ALSO

```
NovaspreadTDeviceList
NovaspreadDeviceListRelease()
```

2.2.8. NovaspreadServerSelectDevice

With this function a SAT>IP server device is selected. SatIpTuners will only be used from this selected device.

If a call to this function changes the selected device, all currently connected SatIpTuners will be disconnected. If NOVASPREAD_NULL is passed as Device, SatIpTuners can no longer connect to a SAT>IP server.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadServerSelectDevice(
  NovaspreadTDevice aDevice );
```

PARAMETERS

aDevice
The SAT>IP server device to be selected.

RETURN VALUE

NOVASPREAD_TRUE
if successful.

NOVASPREAD_FALSE
otherwise.

SEE ALSO

```
NovaspreadServerGetDeviceList()
NovaspreadServerCreateSatIpTuner()
```

2.2.9. NovaspreadServerGetSelectedDevice

Gets the currently selected SAT>IP server device.

SYNTAX

```
PUBLIC NovaspreadTDevice
NovaspreadServerGetSelectedDevice(
    void );
```

RETURN VALUE

The selected device. NOVASPREAD_NULL if no device has been selected.

SEE ALSO

```
NovaspreadServerSelectDevice()
```

2.2.10. NovaspreadServerCreateSatIpTuner

Creates a new SatIpTuner. A SatIpTuner can be used to receive transport stream data from a SAT>IP server, that is available in the local network. If the SatIpTuner is no longer used, call NovaspreadSatIpTunerRelease() to free it. The function call is non-blocking, i.e. it returns immediately. To actually allocate a SatIpTuner from the selected SAT>IP server device call the function NovaspreadSatIpTunerConnect().

SYNTAX

```
PUBLIC NovaspreadTSatIpTuner
NovaspreadServerCreateSatIpTuner(
  void );
```

RETURN VALUE

A new SatIpTuner. NOVASPREAD_NULL if an error occurred.

SEE ALSO

```
NovaspreadTSatIpTuner
NovaspreadSatIpTunerConnect()
```

2.3. NovaspreadDeviceList

A DeviceList holds Devices which have been found via UPnP device detection in the network. Call NovaspreadServerGetDeviceList() to get a list of detected Devices. A DeviceList represents a snapshot of the Devices when the list is retrieved. The DeviceList is not changed afterwards even if new Devices are found or disappeared from the network. To update a list for the user interface simply retrieve the list again to get a current snapshot of this list.

2.3.1. NovaspreadDeviceListRelease

Releases this DeviceList. After calling this function the list shall no longer be accessed. Each DeviceList returned by the function NovaspreadServerGetDeviceList() must be released with this function when it is no longer used.

SYNTAX

```
PUBLIC void
NovaspreadDeviceListRelease(
  NovaspreadTDeviceList This );
```

PARAMETERS

This
The DeviceList.

SEE ALSO

```
NovaspreadServerGetDeviceList()
```

2.3.2. NovaspreadDeviceListGetLength

Gets the number of Devices stored in this list. If the list is empty 0 is returned.

SYNTAX

```
PUBLIC NovaspreadTUInt32
NovaspreadDeviceListGetLength(
  NovaspreadTDeviceList This );
```

PARAMETERS

This
The DeviceList.

RETURN VALUE

The number of Devices stored in this list.

2.3.3. NovaspreadDeviceListGetDevice

Gets the Device at the given index from the DeviceList. The returned Device must be released by calling NovaspreadDeviceRelease(), when it is no longer needed. The first Device in the list has the index 0.

SYNTAX

```
PUBLIC NovaspreadTDevice
NovaspreadDeviceListGetDevice(
  NovaspreadTDeviceList This,
  NovaspreadTUInt32     aIndex );
```

PARAMETERS

This
The DeviceList.

aIndex
The index of the Device to be returned.

RETURN VALUE

A Device. NOVASPREAD_NULL if the given index is invalid.

SEE ALSO

NovaspreadTDevice
NovaspreadDeviceListGetLength()

2.4. NovaspreadDevice

A Device represents a SAT>IP server which was found via UPnP in the local network. The Device provides information that are retrieved from the UPnP device description provided by the SAT>IP server.

The list of Devices can be retrieved with the function NovaspreadServerGetDeviceList(). To get access to the properties of a Device, use the function NovaspreadDeviceListGetDevice().

2.4.1. NovaspreadDeviceRelease

Release this Device. This function must be called for each Device retrieved with the functions NovaspreadDeviceListGetDevice() when the Device is no longer used.

SYNTAX

```
PUBLIC void
NovaspreadDeviceRelease(
  NovaspreadTDevice This );
```

PARAMETERS

This

The Device.

SEE ALSO

```
NovaspreadDeviceListGetDevice()
```

2.4.2. NovaspreadDeviceGetIpAddress

Gets the IP address of this Device.

SYNTAX

```
PUBLIC const char *
NovaspreadDeviceGetIpAddress(
  NovaspreadTDevice This );
```

PARAMETERS

This
The Device.

RETURN VALUE

The IP address as UTF-8 encoded string. NOVASPREAD_NULL is returned if not defined or if an error occurred.

2.4.3. NovaspreadDeviceGetFriendlyName

Gets the friendly name of this Device.

SYNTAX

```
PUBLIC const char *
NovaspreadDeviceGetFriendlyName(
  NovaspreadTDevice This );
```

PARAMETERS

This
The Device.

RETURN VALUE

The friendly name as UTF-8 encoded string. NOVASPREAD_NULL is returned if not defined or if an error occurred.

2.4.4. NovaspreadDeviceGetManufacturer

Gets the manufacturer information of this Device.

SYNTAX

```
PUBLIC const char *
NovaspreadDeviceGetManufacturer(
  NovaspreadTDevice This );
```

PARAMETERS

This
The Device.

RETURN VALUE

The manufacturer as UTF-8 encoded string. NOVASPREAD_NULL is returned if not defined or if an error occurred.

2.4.5. NovaspreadDeviceGetManufacturerUrl

Gets the manufacturer's URL.

SYNTAX

```
PUBLIC const char *
NovaspreadDeviceGetManufacturerUrl(
  NovaspreadTDevice This );
```

PARAMETERS

This
The Device.

RETURN VALUE

The manufacturer's URL as UTF-8 encoded string. NOVASPREAD_NULL is returned if not defined or if an error occurred.

2.4.6. NovaspreadDeviceGetModelDescription

Gets the model description.

SYNTAX

```
PUBLIC const char *
NovaspreadDeviceGetModelDescription(
  NovaspreadTDevice This );
```

PARAMETERS

This
The Device.

RETURN VALUE

The model description as UTF-8 encoded string. NOVASPREAD_NULL is returned if not defined or if an error occurred.

2.4.7. NovaspreadDeviceGetModelNumber

Gets the model number.

SYNTAX

```
PUBLIC const char *
NovaspreadDeviceGetModelNumber(
  NovaspreadTDevice This );
```

PARAMETERS

This
The Device

RETURN VALUE

The model number as UTF-8 encoded string. NOVASPREAD_NULL is returned if not defined or if an error occurred.

2.4.8. NovaspreadDeviceGetModelUrl

Gets the model's URL.

SYNTAX

```
PUBLIC const char *
NovaspreadDeviceGetModelUrl(
  NovaspreadTDevice This );
```

PARAMETERS

This
The Device.

RETURN VALUE

The model URL as UTF-8 encoded string. NOVASPREAD_NULL is returned if not defined or if an error occurred.

2.4.9. NovaspreadDeviceGetSerialNumber

Gets the serial number.

SYNTAX

```
PUBLIC const char *
NovaspreadDeviceGetSerialNumber(
  NovaspreadTDevice This );
```

PARAMETERS

This
The Device.

RETURN VALUE

The serial number as UTF-8 encoded string. NOVASPREAD_NULL is returned if not defined or if an error occurred.

2.4.10. NovaspreadDeviceGetIconList

Gets the IconList defined for this Device. The returned IconList must be released with the function NovaspreadIconListRelease() if it is no longer used.

SYNTAX

```
PUBLIC NovaspreadTIconList
NovaspreadDeviceGetIconList(
  NovaspreadTDevice This );
```

PARAMETERS

This
The Device.

RETURN VALUE

The IconList. NOVASPREAD_NULL if an error occurred.

SEE ALSO

```
NovaspreadTIconList
NovaspreadIconListRelease()
```

2.5. NovaspreadIconList

For each Device a list of Icons can be retrieved. An Icon is used for a graphical user interface. The IconList class represents a list of Icons.

2.5.1. NovaspreadIconListRelease

Releases this IconList. After calling this function the list shall no longer be accessed. Each time an IconList is retrieved with the function NovaspreadDeviceGetIconList() this function must be called to release the IconList.

SYNTAX

```
PUBLIC void
NovaspreadIconListRelease(
  NovaspreadTIconList This );
```

PARAMETERS

This
The IconList

SEE ALSO

```
NovaspreadDeviceGetIconList()
```

2.5.2. NovaspreadIconListGetLength

Gets the number of Icons stored in this list. If the IconList is empty this function returns 0.

SYNTAX

```
PUBLIC NovaspreadTUInt32
NovaspreadIconListGetLength(
  NovaspreadTIconList This );
```

PARAMETERS

This
The IconList

RETURN VALUE

The number of Icons stored in this list. 0 if the IconList is empty.

SEE ALSO

```
NovaspreadIconListGetIcon()
```

2.5.3. NovaspreadIconListGetIcon

Gets the Icon at the given index from the IconList. The returned Icon must be released by calling NovaspreadIconRelease(), when it is no longer needed. The index starts with 0 for the first icon in the IconList.

SYNTAX

```
PUBLIC NovaspreadTIcon
NovaspreadIconListGetIcon(
  NovaspreadTIconList This,
  NovaspreadTUInt32   aIndex );
```

PARAMETERS

This
The IconList

aIndex
The index of the Icon to be returned.

RETURN VALUE

The Icon at the given index position. NOVASPREAD_NULL if the given index is invalid.

2.6. NovaspreadIcon

Defines an Icon used for a Device. Properties of an Icon are the width, height and color depth and the URL from where the icon image file can be loaded.

2.6.1. NovaspreadIconRelease

Releases the Icon. After calling this function, the Icon shall no longer be accessed. Each Icon that is retrieved with the function NovaspreadIconListGetIcon() must be released with this function.

SYNTAX

```
PUBLIC void
NovaspreadIconRelease(
  NovaspreadTIcon This );
```

PARAMETERS

This
The Icon.

SEE ALSO

`NovaspreadIconListGetIcon()`

2.6.2. NovaspreadIconGetMimeType

Gets the MIME type of the Icon. The MIME type is a null-terminated ASCII string that defines the format of the Icon, e.g. "image/png", "image/jpeg".

SYNTAX

```
PUBLIC const char *
NovaspreadIconGetMimeType(
  NovaspreadTIcon This );
```

PARAMETERS

This
The Icon.

RETURN VALUE

The MIME type as null-terminated ASCII string.

2.6.3. NovaspreadIconGetWidth

Gets the width of the Icon in pixels.

SYNTAX

```
PUBLIC NovaspreadTUInt16
NovaspreadIconGetWidth(
  NovaspreadTIcon This );
```

PARAMETERS

This
The Icon.

RETURN VALUE

The width of the Icon.

SEE ALSO

```
NovaspreadIconGetHeight()
```

2.6.4. NovaspreadIconGetHeight

Gets the height of the Icon in pixels.

SYNTAX

```
PUBLIC NovaspreadTUInt16
NovaspreadIconGetHeight(
  NovaspreadTIcon This );
```

PARAMETERS

This
The Icon.

RETURN VALUE

The height of the Icon.

SEE ALSO

```
NovaspreadIconGetWidth()
```

2.6.5. NovaspreadIconGetDepth

Gets the color depth of the Icon. The returned value indicates the number of colors of the Icon.

SYNTAX

```
PUBLIC NovaspreadTUInt32
NovaspreadIconGetDepth(
  NovaspreadTIcon This );
```

PARAMETERS

This
The Icon.

RETURN VALUE

The color depth of the Icon.

2.6.6. NovaspreadIconGetUrl

Gets the URL of the Icon. From this URL the Icon can be downloaded.

SYNTAX

```
PUBLIC const char *
NovaspreadIconGetUrl(
  NovaspreadTIcon This );
```

PARAMETERS

This
The Icon.

RETURN VALUE

The URL as UTF-8 encoded string.

2.7. NovaspreadTunerParameters

TunerParameters define the types for tuning parameters.

2.7.1. NovaspreadTTunerType

NovaspreadTunerType defines the different types of tuners. Which tuners are actually supported depends on the target platform. Currently only DVB-S tuners are supported.

SYNTAX

```
typedef enum
{
  NOVASPREAD_TUNER_TYPE_DVB_S

} NovaspreadTTunerType;
```

COMPONENTS

NOVASPREAD_TUNER_TYPE_DVB_S
A DVB-S tuner receives data from a satellite

2.7.2. NovaspreadTTunerCodeRate

This type defines the code rates.

SYNTAX

```
typedef enum
{
  NOVASPREAD_TUNER_CODE_RATE_UNKNOWN,
  NOVASPREAD_TUNER_CODE_RATE_AUTO,
  NOVASPREAD_TUNER_CODE_RATE_1_2,
  NOVASPREAD_TUNER_CODE_RATE_1_3,
  NOVASPREAD_TUNER_CODE_RATE_1_4,
  NOVASPREAD_TUNER_CODE_RATE_2_3,
  NOVASPREAD_TUNER_CODE_RATE_2_5,
  NOVASPREAD_TUNER_CODE_RATE_3_4,
  NOVASPREAD_TUNER_CODE_RATE_3_5,
  NOVASPREAD_TUNER_CODE_RATE_4_5,
  NOVASPREAD_TUNER_CODE_RATE_5_6,
  NOVASPREAD_TUNER_CODE_RATE_6_7,
  NOVASPREAD_TUNER_CODE_RATE_7_8,
  NOVASPREAD_TUNER_CODE_RATE_8_9,
  NOVASPREAD_TUNER_CODE_RATE_9_10,
  NOVASPREAD_TUNER_CODE_RATE_LAST

} NovaspreadTTunerCodeRate;
```

COMPONENTS

NOVASPREAD_TUNER_CODE_RATE_UNKNOWN
The code rate is unknown. This value shall not be used for setting the tuner parameters.

NOVASPREAD_TUNER_CODE_RATE_AUTO
If this code rate is used, the tuner tries to find out the correct code rate automatically.

NOVASPREAD_TUNER_CODE_RATE_1_2
Represents a code rate of 1/2

NOVASPREAD_TUNER_CODE_RATE_1_3
Represents a code rate of 1/3

NOVASPREAD_TUNER_CODE_RATE_1_4
Represents a code rate of 1/4

NOVASPREAD_TUNER_CODE_RATE_2_3
Represents a code rate of 2/3

NOVASPREAD_TUNER_CODE_RATE_2_5
Represents a code rate of 2/5

NOVASPREAD_TUNER_CODE_RATE_3_4
Represents a code rate of 3/4

NOVASPREAD_TUNER_CODE_RATE_3_5
Represents a code rate of 3/5

NOVASPREAD_TUNER_CODE_RATE_4_5
Represents a code rate of 4/5

NOVASPREAD_TUNER_CODE_RATE_5_6
Represents a code rate of 5/6

NOVASPREAD_TUNER_CODE_RATE_6_7
Represents a code rate of 6/7

NOVASPREAD_TUNER_CODE_RATE_7_8
Represents a code rate of 7/8

NOVASPREAD_TUNER_CODE_RATE_8_9
Represents a code rate of 8/9

NOVASPREAD_TUNER_CODE_RATE_9_10
Represents a code rate of 9/10

NOVASPREAD_TUNER_CODE_RATE_LAST
The last code rate parameter. For internal use only.

SEE ALSO

`NovaspreadTTunerParameters`

2.7.3. NovaspreadTTunerModulationSystem

This type represents the supported modulation systems which are necessary for DVB-S2 tuners.

SYNTAX

```
typedef enum
{
  NOVASPREAD_TUNER_MODULATION_SYSTEM_DVB_S,
  NOVASPREAD_TUNER_MODULATION_SYSTEM_DVB_S2

} NovaspreadTTunerModulationSystem;
```

COMPONENTS

NOVASPREAD_TUNER_MODULATION_SYSTEM_DVB_S
The modulation system 'DVB-S'.

NOVASPREAD_TUNER_MODULATION_SYSTEM_DVB_S2
The modulation system 'DVB-S2'.

2.7.4. NovaspreadTTunerModulation

This enumeration type defines the supported modulation types for DVB tuners. For each tuner type (e.g. DVB-S) only a selection of the listed modulation types can be used.

SYNTAX

```
typedef enum
{
  NOVASPREAD_TUNER_MODULATION_UNKNOWN,
  NOVASPREAD_TUNER_MODULATION_AUTO,
  NOVASPREAD_TUNER_MODULATION_QPSK,
  NOVASPREAD_TUNER_MODULATION_8PSK,
  NOVASPREAD_TUNER_MODULATION_QAM_16,
  NOVASPREAD_TUNER_MODULATION_QAM_32,
  NOVASPREAD_TUNER_MODULATION_QAM_64,
  NOVASPREAD_TUNER_MODULATION_QAM_128,
  NOVASPREAD_TUNER_MODULATION_QAM_256,
  NOVASPREAD_TUNER_MODULATION_LAST

} NovaspreadTTunerModulation;
```

COMPONENTS

NOVASPREAD_TUNER_MODULATION_UNKNOWN
The modulation is unknown. This value shall not be used for setting the tuner parameters.

NOVASPREAD_TUNER_MODULATION_AUTO
If this modulation is used, the tuner tries to find out the correct modulation automatically.

NOVASPREAD_TUNER_MODULATION_QPSK
Represents a QPSK modulation.

NOVASPREAD_TUNER_MODULATION_8PSK
Represents a 8PSK modulation.

NOVASPREAD_TUNER_MODULATION_QAM_16
Represents a 16-QAM modulation.

NOVASPREAD_TUNER_MODULATION_QAM_32
Represents a 32-QAM modulation.

NOVASPREAD_TUNER_MODULATION_QAM_64
Represents a 64-QAM modulation.

NOVASPREAD_TUNER_MODULATION_QAM_128
Represents a 128-QAM modulation.

NOVASPREAD_TUNER_MODULATION_QAM_256
Represents a 256-QAM modulation.

NOVASPREAD_TUNER_MODULATION_LAST
The last modulation parameter. For internal use only.

2.7.5. NovaspreadTTunerParamDvbS

This structure contains the tuning parameters a DVB-S tuner.

SYNTAX

```
  typedef struct
  {
    NovaspreadTUInt8                 SourceId;
    NovaspreadTInt16                 OrbitalPosition;
    NovaspreadTUInt32                Frequency;
    NovaspreadTUInt32                SymbolRate;
    NovaspreadTTunerCodeRate         CodeRate;
    NovaspreadTTunerModulationSystem ModulationSystem;
    NovaspreadTTunerModulation       Modulation;

  } NovaspreadTTunerParamDvbS;
```

COMPONENTS

SourceId
This sourceId is passed to the SAT>IP server. Set to 0 if not used.

OrbitalPosition
In 1/10 degrees. e.g. Astra 19.2E = 192

Frequency
The transponder frequency in KHz to tune to.

SymbolRate
Kilo-symbols per second.

CodeRate
The code rate of the transponder.

ModulationSystem
The used modulation system. This is only necessary for DVB-S2 tuner. DVB-S tuner will ignore it.

Modulation
The modulation of the transponder.

2.7.6. NovaspreadTTunerParamValue

This union contains the parameters for the different types of tuners. Currently only DVB-S/S2 is supported.

SYNTAX

```
typedef union
{
  NovaspreadTTunerParamDvbS DvbS;

} NovaspreadTTunerParamValue;
```

COMPONENTS

DvbS
The tuning parameters specific for DVB-S/S2 reception.

2.7.7. NovaspreadTTunerParameters

This data structure defines the tuning parameters to be set at a tuner. It defines the type of tuner (currently only DVB-S) for which the parameters are to be set. Depending on this type the Value is interpreted.

SYNTAX

```
typedef struct
{
  NovaspreadTTunerType       Type;
  NovaspreadTTunerParamValue Value;

} NovaspreadTTunerParameters;
```

COMPONENTS

Type
The type of the tuner.

Value
Structure containing the tuning parameters specific for a type of tuner.

SEE ALSO

`NovaspreadHostAllocateTuner()`

2.7.8. NovaspreadTTunerSignalInfo

This type defines the SignalInfo of a tuner. The SignalInfo contains the Level and the Quality of the signal received by the Tuner. For a specification of the SignalInfo see SAT>IP Protocol Specification V1.2.2.

SYNTAX

```
typedef struct
{
  NovaspreadTUInt8 Level;
  NovaspreadTUInt8 Quality;

} NovaspreadTTunerSignalInfo;
```

COMPONENTS

Level

Numerical value between 0 and 255. An incoming L-band satellite signal of -25dBm corresponds to 224, -65dBm corresponds to 32 and no signal corresponds to 0.

Quality

Numerical value between 0 and 15. Lower values indicate to higher error rates. The value 15 indicates a BER lower than 2.0E-4 after Viterbi for DVB-S, a BER lower than 10.0E-7 for DVB-S2.

SEE ALSO

```
NovaspreadTunerGetSignalInfo()
NovaspreadSatIpTunerGetSignalInfo()
```

2.8. NovaspreadTranscoding

NovaspreadTranscoding defines all types which are used for transcoding. Transcoding is controlled with the Tuner by the function NovaspreadTunerSetTranscoding(). The NovaspreadTTranscoding structure defined in this section contains all necessary parameters.

2.8.1. NOVASPREAD_PID_UNKNOWN

This constant defines the value to be used for an unknown PID. It is used e.g. in NovaspreadTTranscodingInput, if no audio stream resp. video stream shall be transcoded.

SYNTAX

```
#define NOVASPREAD_PID_UNKNOWN 0xFFFF
```

SEE ALSO

```
NovaspreadTTranscodingInput
```

2.8.2. NovaspreadTVideoCodec

This enumeration type defines all available video codecs used for transcoding. The video codec is used for the TranscodingInput parameters to indicate the used video codec of the input video stream. It is also used for the TranscodingOutput parameters to indicate the video codec to be output by the Transcoder. The list of supported video codecs depends on the platform.

SYNTAX

```
typedef enum
{
  NOVASPREAD_VIDEO_CODEC_MPEG_2,
  NOVASPREAD_VIDEO_CODEC_AVC,
  NOVASPREAD_VIDEO_CODEC_HEVC,
  NOVASPREAD_VIDEO_CODEC_LAST

} NovaspreadTVideoCodec;
```

COMPONENTS

NOVASPREAD_VIDEO_CODEC_MPEG_2

MPEG-2 video

NOVASPREAD_VIDEO_CODEC_AVC

H.264 video (MPEG-4 AVC). The following profile shall be used: HP@L4.

NOVASPREAD_VIDEO_CODEC_HEVC

High Efficiency Video Coding (HEVC). The following profile shall be used: MP@L4.1 Main Tier.

NOVASPREAD_VIDEO_CODEC_LAST

For internal use only.

SEE ALSO

```
NovaspreadTTranscodingInput
```

```
NovaspreadTTranscodingOutput
```

2.8.3. NovaspreadTVideoResolution

This enumeration type defines the possible video resolutions to be used by the Transcoder for the video output. The supported video resolutions depend on the platform.

SYNTAX

```
typedef enum
{
  NOVASPREAD_VIDEO_RESOLUTION_176_144P,
  NOVASPREAD_VIDEO_RESOLUTION_352_288P,
  NOVASPREAD_VIDEO_RESOLUTION_720_576P,
  NOVASPREAD_VIDEO_RESOLUTION_720_576I,
  NOVASPREAD_VIDEO_RESOLUTION_1280_720P,
  NOVASPREAD_VIDEO_RESOLUTION_1920_1080P,
  NOVASPREAD_VIDEO_RESOLUTION_1920_1080I,
  NOVASPREAD_VIDEO_RESOLUTION_LAST

} NovaspreadTVideoResolution;
```

COMPONENTS

NOVASPREAD_VIDEO_RESOLUTION_176_144P
176x144 progressive

NOVASPREAD_VIDEO_RESOLUTION_352_288P
352x288 progressive

NOVASPREAD_VIDEO_RESOLUTION_720_576P
720x576 progressive

NOVASPREAD_VIDEO_RESOLUTION_720_576I
720x576 interlaced

NOVASPREAD_VIDEO_RESOLUTION_1280_720P
1280x720 progressive

NOVASPREAD_VIDEO_RESOLUTION_1920_1080P
1920x1080 progressive

NOVASPREAD_VIDEO_RESOLUTION_1920_1080I
1920x1080 interlaced

NOVASPREAD_VIDEO_RESOLUTION_LAST
For internal use only

SEE ALSO

```
NovaspreadTTranscodingOutput
```

2.8.4. NovaspreadTAudioCodec

This enumeration type defines the different audio codecs used for transcoding. The audio codec is used for the TranscodingInput parameters to indicate the used audio codec of the input audio stream. It is also used for the TranscodingOutput parameters to indicate the audio codec to be output by the Transcoder. The list of supported audio codecs depends on the platform.

SYNTAX

```
typedef enum
{
  NOVASPREAD_AUDIO_CODEC_MP2,
  NOVASPREAD_AUDIO_CODEC_AC3,
  NOVASPREAD_AUDIO_CODEC_AAC,
  NOVASPREAD_AUDIO_CODEC_HE_AAC,
  NOVASPREAD_AUDIO_CODEC_LAST

} NovaspreadTAudioCodec;
```

COMPONENTS

`NOVASPREAD_AUDIO_CODEC_MP2`
MPEG-1 Audio Layer II

`NOVASPREAD_AUDIO_CODEC_AC3`
Dolby Digital

`NOVASPREAD_AUDIO_CODEC_AAC`
Advanced Audio Coding (AAC)

`NOVASPREAD_AUDIO_CODEC_HE_AAC`
High-Efficiency Advanced Audio Coding (HE-AAC). The following profile shall be used: HE-AAC v1.

`NOVASPREAD_AUDIO_CODEC_LAST`
For internal use only

SEE ALSO

```
NovaspreadTTranscodingInput
NovaspreadTTranscodingOutput
```

2.8.5. NovaspreadTTranscodingInput

This TranscodingInput type defines the parameters of the input stream to be transcoding. The PIDs for audio, video and PCR as well as the audio and video codecs of the input stream are defined.

SYNTAX

```
typedef struct
{
  NovaspreadTUInt16      AudioPid;
  NovaspreadTAudioCodec AudioCodec;
  NovaspreadTUInt16      VideoPid;
  NovaspreadTVideoCodec VideoCodec;
  NovaspreadTUInt16      PcrPid;

} NovaspreadTTranscodingInput;
```

COMPONENTS

`AudioPid`
The pid of the audio stream. If set to NOVASPREAD_PID_UNKNOWN, no audio stream shall be transcoded.

`AudioCodec`
The codec of the stream.

`VideoPid`
The pid of the video stream. If set to NOVASPREAD_PID_UNKNOWN, no video stream shall be transcoded.

`VideoCodec`
The codec of the stream.

`PcrPid`
The pid containing the PCR information.

SEE ALSO

```
NovaspreadTTranscoding
```

2.8.6. NovaspreadTTranscodingOutput

The TranscodingOuput type defines the output properties of the transcoded stream.

SYNTAX

```
typedef struct
{
  NovaspreadTAudioCodec       AudioCodec;
  NovaspreadTUInt32           AudioBitrate;
  NovaspreadTVideoCodec       VideoCodec;
  NovaspreadTVideoResolution  VideoResolution;
  NovaspreadTUInt32           VideoBitrate;

} NovaspreadTTranscodingOutput;
```

COMPONENTS

AudioCodec
The AudioCodec of the transcoded audio stream. See NovaspreadTAudioCodec for a list of possible values.

AudioBitrate
The bit rate of the transcoded audio stream in kbits/sec.

VideoCodec
The VideoCodec of the transcoded video stream. See NovaspreadTVideoCodec for a list of possible values.

VideoResolution
The resolution of the transcoded video stream.

VideoBitrate
The maximum bit rate of the transcoded video stream in kbits/sec.

SEE ALSO

```
NovaspreadTAudioCodec
NovaspreadTVideoCodec
NovaspreadTTranscoding
```

2.8.7. NovaspreadTTranscoding

This type defines all transcoding parameters for the input and output audio and video streams.

SYNTAX

```
typedef struct
{
  NovaspreadTTranscodingInput  Input;
  NovaspreadTTranscodingOutput Output;

} NovaspreadTTranscoding;
```

COMPONENTS

Input
The input parameters of the stream to transcode.

Output
The output parameters of the transcoded stream.

SEE ALSO

```
NovaspreadTTranscodingInput
NovaspreadTTranscodingOutput
NovaspreadTHostCapabilities
NovaspreadTunerSetTranscoding()
```

2.8.8. NovaspreadTVideoTranscodingCapability

VideoTranscodingCapability describes the capability of a video transcoder for transcoding to a particular VideoCodec.

SYNTAX

```
typedef struct
{
  NovaspreadTVideoCodec  VideoCodec;
  NovaspreadTUInt32      MinBitrate;
  NovaspreadTUInt32      MaxBitrate;

} NovaspreadTVideoTranscodingCapability;
```

COMPONENTS

VideoCodec
The destination VideoCodec.

MinBitrate
The minimal bit rate in Kilobits/sec.

MaxBitrate
The maximal bit rate in Kilobits/sec.

2.8.9. NovaspreadTAudioTranscodingCapability

AudioTranscodingCapability describes the capability of an audio transcoder for transcoding to a particular AudioCodec.

SYNTAX

```
  typedef struct
  {
    NovaspreadTAudioCodec  AudioCodec;
    NovaspreadTUInt32      MinBitrate;
    NovaspreadTUInt32      MaxBitrate;

  } NovaspreadTAudioTranscodingCapability;
```

COMPONENTS

AudioCodec
The destination AudioCodec.

MinBitrate
The minimal bit rate in Kilobits/sec.

MaxBitrate
The maximal bit rate in Kilobits/sec.

2.9. NovaspreadSatIpTuner

A SatIpTuner can be used to receive transport stream data from a SAT>IP server which is available in the local network.

To create SatIpTuner call the function NovaspreadSeverCreateSatIpTuner(). Only SatIpTuners from the selected SAT>IP server device are used.

After creation a SatIpTuner is not connected to a SAT>IP server. To connect to a SAT>IP server, tuner parameters must be set at the SatIpTuner with the function NovaspreadSatIpTunerSetParameters() and then NovaspreadSatIpTunerConnect() must be called. By calling NovaspreadSatIpTunerSetPids(), the pids that shall be received from the SAT>IP server are defined.

As soon as the SatIpTuner has changed its ConnectionStatus to CONNECTED, NovaspreadSatIpTunerReadData() will provide transport stream data.

2.9.1. NovaspreadTSatIpTunerConnectionStatus

A SatIpTuner is in one of the following ConnectionStatus.

SYNTAX

```
typedef enum
{
  NOVASPREAD_SAT_IP_TUNER_CONNECTION_STATUS_NOT_CONNECTED,
  NOVASPREAD_SAT_IP_TUNER_CONNECTION_STATUS_CONNECTING,
  NOVASPREAD_SAT_IP_TUNER_CONNECTION_STATUS_CONNECTED

} NovaspreadTSatIpTunerConnectionStatus;
```

COMPONENTS

NOVASPREAD_SAT_IP_TUNER_CONNECTION_STATUS_NOT_CONNECTED
After creating a SatIpTuner, the tuner is in ConnectionStatus NOT_CONNECTED. It is not connected to any SAT>IP server. This ConnectionStatus is also reached, when NovaspreadSatIpTunerDisconnect() is called, resp. from CONNECTING, when it was not possible to establish a connection.

NOVASPREAD_SAT_IP_TUNER_CONNECTION_STATUS_CONNECTING
When NovaspreadSatIpTunerConnect() was called successfully, the SatIpTuner is in ConnectionStatus CONNECTING.

NOVASPREAD_SAT_IP_TUNER_CONNECTION_STATUS_CONNECTED
When a SatIpTuner connected successfully to a SAT>IP server, it changed to ConnectionStatus CONNECTED. Transport stream data can be read from the tuner by calling NovaspreadSatIpTunerReadData().

SEE ALSO

```
NovaspreadSatIpTunerConnect()
NovaspreadSatIpTunerDisconnect()
NovaspreadSatIpTunerGetConnectionStatus()
```

2.9.2. NovaspreadTSatIpTunerConnectionStatusChangeListener

A function of this type can be set at a SatIpTuner. It is called every time the ConnectionStatus of the SatIpTuner changes.

SYNTAX

```
typedef void
(* NovaspreadTSatIpTunerConnectionStatusChangeListener ) (
  void *                                aContext,
  NovaspreadTSatIpTuner                 aSatIpTuner,
  NovaspreadTSatIpTunerConnectionStatus aNewStatus );
```

PARAMETERS

aContext
This context is passed unchanged from NovaspreadSatIpTunerSetConnectionStatusChangeListener().

aSatIpTuner
The status of this tuner has changed.

aNewStatus
The new status.

SEE ALSO

```
NovaspreadSatIpTunerSetConnectionStatusChangeListener()
```

2.9.3. NovaspreadTSatIpTunerDataAvailableListener

A function of this type can be set at the SatIpTuner. When NovaspreadSatIpTunerReadData() returns 0, because no data is available, the registered DataAvailableListener will be called as soon as data is available again.

SYNTAX

```
typedef void
(* NovaspreadTSatIpTunerDataAvailableListener ) (
  void * aContext );
```

Parameters

aContext
This context is passed unchanged from the NovaspreadSatIpTunerSetDataAvailableListener() function.

See Also

```
NovaspreadSatIpTunerSetDataAvailableListener()
NovaspreadSatIpTunerReadData()
```

2.9.4. NovaspreadSatIpTunerDestroy

Destroys the given SatIpTuner. The SatIpTuner may not be accessed after calling this function.

Syntax

```
PUBLIC void
NovaspreadSatIpTunerDestroy(
  NovaspreadTSatIpTuner This );
```

Parameters

This
The SatIpTuner.

2.9.5. NovaspreadSatIpTunerSetParameters

Sets the tuning parameters of this SatIpTuner. Tuning parameters define the transponder from where the transport stream is to be received.

Syntax

```
PUBLIC NovaspreadTBoolean
NovaspreadSatIpTunerSetParameters(
  NovaspreadTSatIpTuner          This,
  NovaspreadTTunerParameters * aParameter );
```

Parameters

This
The SatIpTuner.

aParameter
The TunerParametesr. See data type NovaspreadTTunerParameters for a description of all tuning parameters.

Return Value

NOVASPREAD_TRUE
if the parameters were set successfully.

NOVASPREAD_FALSE
if an error occurred.

See Also

```
NovaspreadTTunerParameters
NovaspreadSatIpTunerGetParameters()
```

2.9.6. NovaspreadSatIpTunerGetParameters

Gets the currently set TunerParameters.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadSatIpTunerGetParameters(
  NovaspreadTSatIpTuner        This,
  NovaspreadTTunerParameters * aParameter );
```

PARAMETERS

This
The SatIpTuner.

aParameter
OUT: Pointer to variable of type NovaspreadTTunerParameters, where the function returns the currently set TunerParameters.

RETURN VALUE

NOVASPREAD_TRUE
if the TunerParameters are returned successfully.

NOVASPREAD_FALSE
if an error occurred. In this case the variable aParameters points to is not unchanged.

SEE ALSO

```
NovaspreadTTunerParameters
NovaspreadSatIpTunerSetParameters()
```

2.9.7. NovaspreadSatIpTunerConnect

This function is called to establish a connection of the SatIpTuner with a SAT>IP server in the network. During this call the SatIpTuner changes its ConnectionStatus to CONNECTING.

As soon as the connection is established successfully, the ConnectionStatus is changed to CONNECTED. From this point in time received transport stream packets can be retrieved with the function NovaspreadSatIpTunerReadData().

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadSatIpTunerConnect(
  NovaspreadTSatIpTuner This );
```

PARAMETERS

This
The SatIpTuner.

RETURN VALUE

NOVASPREAD_TRUE
if the tuner started connecting successfully.

NOVASPREAD_FALSE
otherwise.

SEE ALSO

```
NovaspreadTSatIpTunerConnectionStatus
NovaspreadSatIpTunerDisconnect()
```

2.9.8. NovaspreadSatIpTunerDisconnect

Disconnects a SatIpTuner from a SAT>IP server.

SYNTAX

```
PUBLIC void
```

```
NovaspreadSatIpTunerDisconnect(
  NovaspreadTSatIpTuner This );
```

PARAMETERS

This
The SatIpTuner.

SEE ALSO

```
NovaspreadTSatIpTunerConnectionStatus
NovaspreadSatIpTunerConnect()
```

2.9.9. NovaspreadSatIpTunerGetConnectionStatus

Gets the ConnectionStatus of a SatIpTuner.

SYNTAX

```
PUBLIC NovaspreadTSatIpTunerConnectionStatus
NovaspreadSatIpTunerGetConnectionStatus(
  NovaspreadTSatIpTuner This );
```

PARAMETERS

This
The SatIpTuner.

RETURN VALUE

The current ConnectionStatus of the SatIpTuner.

SEE ALSO

```
NovaspreadTSatIpTunerConnectionStatus
NovaspreadSatIpTunerConnect()
```

2.9.10. NovaspreadSatIpTunerSetConnectionStatusChangeListener

This function sets a ConnectionStatusChangeListener at a SatIpTuner.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadSatIpTunerSetConnectionStatusChangeListener(
  NovaspreadTSatIpTuner                               This,
  NovaspreadTSatIpTunerConnectionStatusChangeListener aListener,
  void *                                              aContext );
```

PARAMETERS

This
The SatIpTuner

aListener
The listener to be set. PASS NOVASPREAD_NULL to unset the listener.

aContext
This context is passed unchanged to the listener.

RETURN VALUE

NOVASPREAD_TRUE
if successful

NOVASPREAD_FALSE
otherwise

SEE ALSO

```
NovaspreadTSatIpTunerConnectionStatusChangeListener
```

2.9.11. NovaspreadSatIpTunerSetPids

Sets the pids which shall be available in the stream received by this Tuner. This function overwrites the pids previously enabled for the SatIpTuner. To reset all pids, pass aPids=NOVASPREAD_NULL and aNoOfPids=0.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadSatIpTunerSetPids(
  NovaspreadTSatIpTuner This,
  NovaspreadTUInt16 *    aPids,
  NovaspreadTUInt32      aNoOfPids );
```

PARAMETERS

This
The SatIpTuner.

aPids
The array of pids.

aNoOfPids
The number of pids in the array.

RETURN VALUE

NOVASPREAD_TRUE
if successful

NOVASPREAD_FALSE
otherwise

SEE ALSO

```
NovaspreadSatIpTunerGetPids()
```

2.9.12. NovaspreadSatIpTunerGetPids

Gets the pids that are currently enabled for streaming.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadSatIpTunerGetPids(
  NovaspreadTSatIpTuner This,
  NovaspreadTUInt16 *    aPids,
  NovaspreadTUInt32      aMaxNoOfPids,
  NovaspreadTUInt32 *    aNoOfPids );
```

PARAMETERS

This
The SatIpTuner.

aPids
OUT: Pointer to an array of UInt16 where the function stores the currently enabled pids

aMaxNoOfPids
The maximal number of pids that can be copied into the aPids array.

aNoOfPids
OUT: The number of pids that are copied into the aPids array.

Return Value

NOVASPREAD_TRUE
if successful

NOVASPREAD_FALSE
otherwise

See Also

`NovaspreadSatIpTunerSetPids()`

2.9.13. NovaspreadSatIpTunerAddPids

Adds pids, which shall additionally be received by this Tuner.

Syntax

```
PUBLIC NovaspreadTBoolean
NovaspreadSatIpTunerAddPids(
  NovaspreadTSatIpTuner This,
  NovaspreadTUInt16 *    aPids,
  NovaspreadTUInt32      aNoOfPids );
```

Parameters

This
The SatIpTuner.

aPids
The array of pids that shall additionally be received.

aNoOfPids
The number of pids in the array.

Return Value

NOVASPREAD_TRUE
if successful.

NOVASPREAD_FALSE
otherwise.

2.9.14. NovaspreadSatIpTunerRemovePids

Removes pids, which should no longer be received by this Tuner.

Syntax

```
PUBLIC void
NovaspreadSatIpTunerRemovePids(
  NovaspreadTSatIpTuner This,
  NovaspreadTUInt16 *    aPids,
  NovaspreadTUInt32      aNoOfPids );
```

Parameters

This
The SatIpTuner.

aPids
The array of pids that should no longer be streamed.

aNoOfPids
The number of pids in the array.

2.9.15. NovaspreadSatIpTunerIsLocked

Returns the lock status of the tuner. A Tuner is locked if a signal is detected for the set TunerParameter and the demodulator is able to decode the signal. A Tuner receives data only if it is locked.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadSatIpTunerIsLocked(
  NovaspreadTSatIpTuner This );
```

PARAMETERS

This
The SatIpTuner.

RETURN VALUE

NOVASPREAD_TRUE
if the SatIpTuner is locked.

NOVASPREAD_FALSE
otherwise.

2.9.16. NovaspreadSatIpTunerGetSignalInfo

Gets the current SignalInfo of the SatIpTuner. See data type NovaspreadTTunerSignalInfo for a description of the returned data.

SYNTAX

```
PUBLIC NovaspreadTTunerSignalInfo
NovaspreadSatIpTunerGetSignalInfo(
  NovaspreadTSatIpTuner This );
```

PARAMETERS

This
The SatIpTuner.

RETURN VALUE

The current SignalInfo.

SEE ALSO

`NovaspreadTTunerSignalInfo`

2.9.17. NovaspreadSatIpTunerSetDataAvailableListener

Sets a DataAvailableListener. Only one DataAvailableListener can be set at a SatIpTuner.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadSatIpTunerSetDataAvailableListener(
  NovaspreadTSatIpTuner                      This,
  NovaspreadTSatIpTunerDataAvailableListener aListener,
  void *                                     aContext );
```

PARAMETERS

This
The SatIpTuner.

aListener
The DataAvailableListener. Pass NOVASPREAD_NULL to unset the listener.

aContext
This context is passed unchanged to the listener.

RETURN VALUE

NOVASPREAD_TRUE
  if successful.

NOVASPREAD_FALSE
  otherwise.

SEE ALSO

```
NovaspreadTSatIpTunerDataAvailableListener
NovaspreadSatIpTunerReadData()
```

2.9.18. NovaspreadSatIpTunerReadData

As soon as the SatIpTuner is in ConnectionStatus CONNECTED, this function will write 188 bytes long transport stream packets to the buffer. This function must be called periodically to avoid a SatIpTuner internal buffer overflow.

If this function is called when the SatIpTuner is in a different ConnectionStatus, it will not write data to the buffer and return 0.

If NovaspreadSatIpTunerReadData() is called in ConnectionStatus CONNECTED and no data is available, 0 will be returned. As soon as data is available again, a previously set DataAvailableListener will be called. Do not call NovaspreadSatIpTunerReadData() in the context of the DataAvailableListener.

SYNTAX

```
PUBLIC NovaspreadTUInt32
NovaspreadSatIpTunerReadData(
  NovaspreadTSatIpTuner This,
  NovaspreadTUInt8 *    aBuffer,
  NovaspreadTUInt32     aBufferSize );
```

PARAMETERS

This
  The SatIpTuner.

aBuffer
  Transport stream packets are written to this buffer.

aBufferSize
  The size of the buffer. Any buffer size is allowed.

RETURN VALUE

The number of bytes written to the buffer. If there are no transport stream packets available 0 is returned.

2.10. NovaspreadCaInfo

A NovaspreadCaInfo represent all information that is returned via a "GET /rc/ca" request as defined in "FreeTV Remote Control Specification v1.0" and "FREETVA-RC Profile AS-30102 HD+ Platform v1.0".

2.10.1. NovaspreadTCaInfoSmartcardStatus

This type defines various smartcard status.

SYNTAX

```
typedef enum
{
  NOVASPREAD_CA_INFO_SMARTCARD_STATUS_ACTIVATING,
  NOVASPREAD_CA_INFO_SMARTCARD_STATUS_NOT_ACTIVATED,
  NOVASPREAD_CA_INFO_SMARTCARD_STATUS_ACTIVATED,
  NOVASPREAD_CA_INFO_SMARTCARD_STATUS_TUNE,
  NOVASPREAD_CA_INFO_SMARTCARD_STATUS_EXPIRED

} NovaspreadTCaInfoSmartcardStatus;
```

COMPONENTS

NOVASPREAD_CA_INFO_SMARTCARD_STATUS_ACTIVATING
The smartcard is currently activating.

NOVASPREAD_CA_INFO_SMARTCARD_STATUS_NOT_ACTIVATED
The smartcard is not yet activated.

NOVASPREAD_CA_INFO_SMARTCARD_STATUS_ACTIVATED
The smartcard is activated.

NOVASPREAD_CA_INFO_SMARTCARD_STATUS_TUNE
Tune to a specific channel.

NOVASPREAD_CA_INFO_SMARTCARD_STATUS_EXPIRED
The smartcard is expired.

2.10.2. NovaspreadCaInfoCreate

Creates a new CaInfo.

SYNTAX

```
PUBLIC NovaspreadTCaInfo
NovaspreadCaInfoCreate(
  void );
```

RETURN VALUE

A new CaInfo if successful. NOVASPREAD_NULL otherwise.

SEE ALSO

`NovaspreadCaInfoDestroy()`

2.10.3. NovaspreadCaInfoDestroy

Destroys the given CaInfo.

SYNTAX

```
PUBLIC void
NovaspreadCaInfoDestroy(
  NovaspreadTCaInfo This );
```

PARAMETERS

This
This CaInfo.

SEE ALSO

`NovaspreadCaInfoCreate()`

2.10.4. NovaspreadCaInfoSetChipsetUid

Sets the chipset unique ID.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadCaInfoSetChipsetUid(
  NovaspreadTCaInfo This,
  const char *      aChipsetUid );
```

Parameters

This
This CaInfo.

aChipsetUid
The chipset unique ID.

Return Value

NOVASPREAD_TRUE
if successful.

NOVASPREAD_FALSE
otherwise.

2.10.5. NovaspreadCaInfoSetChipsetType

Sets the chipset type.

Syntax

```
PUBLIC NovaspreadTBoolean
NovaspreadCaInfoSetChipsetType(
  NovaspreadTCaInfo This,
  const char *      aChipsetType );
```

Parameters

This
This CaInfo.

aChipsetType
The type of the chipset.

Return Value

NOVASPREAD_TRUE
if successful.

NOVASPREAD_FALSE
otherwise.

2.10.6. NovaspreadCaInfoSetChipsetRevision

Sets the chipset revision.

Syntax

```
PUBLIC NovaspreadTBoolean
NovaspreadCaInfoSetChipsetRevision(
  NovaspreadTCaInfo This,
  const char *      aChipsetRevision );
```

Parameters

This
This CaInfo

aChipsetRevision
The chipset revision.

Return Value

NOVASPREAD_TRUE
if successful.

NOVASPREAD_FALSE

otherwise.

2.10.7. NovaspreadCaInfoSetCaVendor

Sets the CAS vendor.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadCaInfoSetCaVendor(
  NovaspreadTCaInfo This,
  const char *      aCaVendor );
```

PARAMETERS

This
This CaInfo.

aCaVendor
The CAS vendor.

RETURN VALUE

NOVASPREAD_TRUE
if successful.

NOVASPREAD_FALSE
otherwise.

2.10.8. NovaspreadCaInfoSetCaVersion

Sets the CAS version.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadCaInfoSetCaVersion(
  NovaspreadTCaInfo This,
  const char *      aCaVersion );
```

PARAMETERS

This
This CaInfo.

aCaVersion
The CAs version.

RETURN VALUE

NOVASPREAD_TRUE
if successful.

NOVASPREAD_FALSE
otherwise.

2.10.9. NovaspreadCaInfoSetCaNumber

Sets the CAS serial number.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadCaInfoSetCaNumber(
  NovaspreadTCaInfo This,
  const char *      aCaNumber );
```

PARAMETERS

This
This CaInfo.

aCaNumber
The CAS serial number.

RETURN VALUE

NOVASPREAD_TRUE
if successful.

NOVASPREAD_FALSE
otherwise.

2.10.10. NovaspreadCaInfoSetSmartcardInserted

Sets whether a smartcard is inserted or not.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadCaInfoSetSmartcardInserted(
  NovaspreadTCaInfo   This,
  NovaspreadTBoolean aInserted );
```

PARAMETERS

This
This CaInfo.

aInserted
NOVASPREAD_TRUE if a smartcard is inserted. NOVASPREAD_FALSE otherwise.

RETURN VALUE

NOVASPREAD_TRUE
if successful.

NOVASPREAD_FALSE
otherwise.

2.10.11. NovaspreadCaInfoSetSmartcardSuitable

Sets if the smartcard is suitable for the Operator.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadCaInfoSetSmartcardSuitable(
  NovaspreadTCaInfo   This,
  NovaspreadTBoolean aSuitable );
```

PARAMETERS

This
This CaInfo.

aSuitable
NOVASPREAD_TRUE if the inserted smartcard is suitable. NOVASPREAD_FALSE otherwise.

RETURN VALUE

NOVASPREAD_TRUE
if successful.

NOVASPREAD_FALSE

otherwise.

2.10.12. NovaspreadCaInfoSetSmartcardType

Sets the type and/or version of the smartcard.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadCaInfoSetSmartcardType(
  NovaspreadTCaInfo This,
  const char *      aType );
```

PARAMETERS

This
This CaInfo.

aType
The type and/or version.

RETURN VALUE

NOVASPREAD_TRUE
if successful.

NOVASPREAD_FALSE
otherwise.

2.10.13. NovaspreadCaInfoSetSmartcardNumber

Sets the smartcard's serial number.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadCaInfoSetSmartcardNumber(
  NovaspreadTCaInfo This,
  const char *      aNumber );
```

PARAMETERS

This
This CaInfo.

aNumber
The serial number.

RETURN VALUE

NOVASPREAD_TRUE
if successful.

NOVASPREAD_FALSE
otherwise.

2.10.14. NovaspreadCaInfoSetSmartcardStatus

Sets the smartcard status information as defined by the operator.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadCaInfoSetSmartcardStatus(
  NovaspreadTCaInfo               This,
  NovaspreadTCaInfoSmartcardStatus aStatus );
```

Parameters

`This`
This CaInfo.

`aStatus`
The status.

Return Value

`NOVASPREAD_TRUE`
if successful.

`NOVASPREAD_FALSE`
otherwise.

See Also

`NovaspreadTCaInfoSmartcardStatus`

2.10.15. NovaspreadCaInfoSetExpirationDate

Sets the expiration date. The expiration date shall be set if the status of the smartcard is ACTIVATED.

Syntax

```
PUBLIC NovaspreadTBoolean
NovaspreadCaInfoSetExpirationDate(
  NovaspreadTCaInfo This,
  NovaspreadTUInt32 aDate );
```

Parameters

`This`
The CaInfo

`aDate`
The expiration date in UTC (seconds since 1 Jan 1970).

Return Value

`NOVASPREAD_TRUE`
if successful.

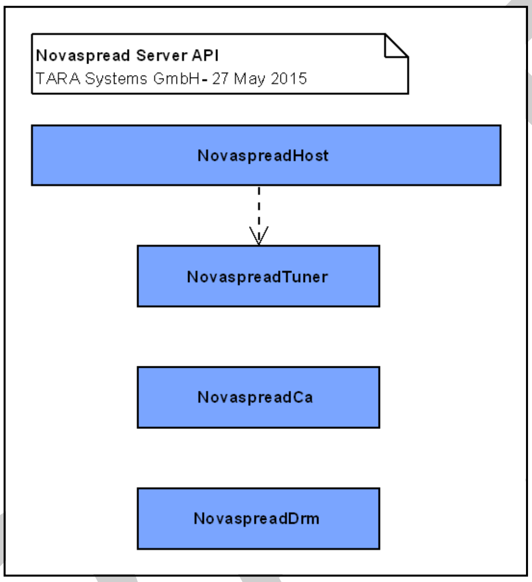
`NOVASPREAD_FALSE`
otherwise.

3. REQUIRED API

The following section describes the Application Programming Interface (API) which is required by Novaspread-S. All functions described in this section must be implemented for the target platform to which Novaspread-S is ported.

To use the interface include the file NovaspreadHost.h.

The following diagram gives an overview of the classes required by Novaspread-S.



3.1. NovaspreadHost

The Host is a required interface used by the NovaspreadServer. All methods must be implemented on target platforms to which NovaspreadServer is ported.

3.1.1. NovaspreadTHostCapabilities

The HostCapabilities describe the capabilities of the tuners, transcoders and transcryptors. The capabilities can be retrieved from the Host by calling NovaspreadHostGetCapabilities().

SYNTAX

```
typedef struct
{
  NovaspreadTUInt8                      NoOfTuners;
  NovaspreadTUInt8                      NoOfTranscoders;
  NovaspreadTUInt8                      NoOfTranscryptors;
  NovaspreadTUInt8                      NoOfVideoTranscodingCapabilities;
  NovaspreadTVideoTranscodingCapability VideoTranscodingCapabilities[
NOVASPREAD_VIDEO_CODEC_LAST ];
  NovaspreadTUInt8                      NoOfAudioTranscodingCapabilities;
  NovaspreadTAudioTranscodingCapability AudioTranscodingCapabilities[
NOVASPREAD_AUDIO_CODEC_LAST ];

} NovaspreadTHostCapabilities;
```

COMPONENTS

NoOfTuners
The total number of tuners that are managed by the Host.

NoOfTranscoders
The total number of transcoders. This shall be set to 0, if transcoding is not supported at all.

NoOfTranscryptors
The total number of transcryptors. This shall be set to 0, if transcryption is not supported at all.

NoOfVideoTranscodingCapabilities
The number of TranscodingCapabilities returned in VideoTranscodingCapabilities.

VideoTranscodingCapabilities[NOVASPREAD_VIDEO_CODEC_LAST]
A list of supported VideoTranscodings.

NoOfAudioTranscodingCapabilities
The number of TranscodingCapabilities returned in AudioTranscodingCapabilities.

AudioTranscodingCapabilities[NOVASPREAD_AUDIO_CODEC_LAST]
A list of supported AudioTranscodings.

SEE ALSO

```
NovaspreadTVideoTranscodingCapability
NovaspreadTAudioTranscodingCapability
```

3.1.2. NovaspreadHostGetCapabilities

This function gets the capabilities of this Host.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadHostGetCapabilities(
  NovaspreadTHostCapabilities * aCapabilities );
```

PARAMETERS

aCapabilities
OUT: Pointer to a variable of type NovaspreadTHostCapabilities, where the function returns the capabilities.

RETURN VALUE

NOVASPREAD_TRUE
if the Host successfully filled all Capabilities members.

NOVASPREAD_FALSE
otherwise

3.1.3. NovaspreadHostAllocateTuner

When a SAT>IP client connects to NovaspreadServer, NovaspreadServer calls this function to allocate a tuner for the given TunerAllocationParameter.

When the allocation request is fulfilled, the Tuner must be tuned to the given parameters.

When the Host is able to fulfill the allocation request, the Host shall call the given AllocationFinishedListener and pass the Tuner to NovaspreadServer via this listener.

When the Host denies the allocation request, e.g. because all Tuners are in use by higher priority usages, the Host shall call the AllocationFinishedListener and pass NOVASPREAD_NULL as Tuner via this listener.

NovaspreadHostAllocateTuner() shall not block and return immediately. If the Host can decide during the call of this function whether the request can be fulfilled or must be denied, it may call the AllocationFinishedListener directly. If it is not possible to process the request while calling this function, this function shall return and the Host shall call the AllocationFinishedListener later.

If no local tuner is available, the Host can create a NovaspreadSatIpTuner by calling NovaspreadServerCreateSatIpTuner() and try to connect it to a SAT>IP LNB by calling NovaspreadSatIpTunerSetParameter() and NovaspreadSatIpTunerConnect(). If the connection is established, this tuner can also be provided via the AllocationFinishedListener.

SYNTAX

```
PUBLIC void
NovaspreadHostAllocateTuner(
  NovaspreadTTunerAllocationParameter *      aTunerAllocationParameter,
  NovaspreadTTunerAllocationFinishedListener aAllocationFinishedListener,
  NovaspreadTTunerReleaseRequestedListener   aReleaseRequestedListener,
  void *                                     aContext );
```

PARAMETERS

aTunerAllocationParameter
A Tuner is requested, which can fulfill this parameter.

aAllocationFinishedListener
The Host must call this function, when either the allocation request could be fulfilled, or the allocation request is denied.

aReleaseRequestedListener
NovaspreadServer provides this ReleaseRequestedListener. It can be called by the Host, if a provided tuner is needed for a different usage with a higher priority (e.g. for performing a PVR recording). When this listener is called, NovaspreadServer will call NovaspreadHostReleaseTuner() from within this listener.

aContext
This context shall be passed to the AllocationFinishedListener and ReleaseRequestedListener when the listener is called.

SEE ALSO

```
NovaspreadTTunerAllocationParameter
NovaspreadTTunerAllocationFinishedListener
NovaspreadTTunerReleaseRequestedListener
```

3.1.4. NovaspreadHostReleaseTuner

This function is called by NovaspreadServer if a tuner is no longer used by NovaspreadServer, e.g. if a SAT>IP client closed the connection.

The Host can use the tuner for another usage. The Host shall not call the ReleaseRequestedListener, which was given during NovaspreadHostAllocateTuner() for this tuner, when NovaspreadHostReleaseTuner() returned.

SYNTAX

```
PUBLIC void
NovaspreadHostReleaseTuner(
  NovaspreadTTuner aTuner );
```

PARAMETERS

aTuner
The Tuner which is no longer accessed by NovaspreadServer.

3.2. NovaspreadTuner

The NovaspreadTuner is an interface required by the NovaspreadServer and must be implemented on target platforms to which NovaspreadServer is ported.

The NovaspreadServer allocates a Tuner by calling the NovaspreadHostAllocateTuner() function of the NovaspreadHost. A Tuner is always allocated for a particular transponder. So NovaspreadServer cannot change the tuner parameters (i.e. the transponder) of an already allocated Tuner. Instead NovaspreadServer will release the Tuner and call NovaspreadHostAllocateTuner() for the new tuning parameters.

A Tuner is a combination of a local tuner, transcoder,and transcryptor.

So when a Tuner is allocated, a complete transport stream processing pipeline must be available for this Tuner. See NovaspreadHostAllocateTuner() for details. NovaspreadTunerReadData() can be called to receive transport stream packets from the tuner. Only transport stream packets for pids are received which were set before by calling one of the functions NovaspreadTunerSetPids(), NovaspreadTunerAddPids() or NovaspreadTunerRemovePids(). NovaspreadServer will send these transport stream packets via RTP/UDP to the SAT>IP client.

When a transcoder was requested during NovaspreadHostAllocateTuner(), it shall be possible to change the transcoding parameters of the Tuner at run-time, to allow streaming of a different audio stream. The transcryption parameter can also be changed at run-time.

3.2.1. NovaspreadTTunerError

This type defines various error codes. As long as no error occurred, NovaspreadTunerGetError() shall return NOVASPREAD_TUNER_ERROR_NONE.

SYNTAX

```
typedef enum
{
  NOVASPREAD_TUNER_ERROR_NONE

} NovaspreadTTunerError;
```

COMPONENTS

NOVASPREAD_TUNER_ERROR_NONE
No error occurred.

3.2.2. NovaspreadTTunerState

This type defines various states of a tuner.

SYNTAX

```
typedef enum
{
  NOVASPREAD_TUNER_STATE_STOPPED,
  NOVASPREAD_TUNER_STATE_STREAMING,
  NOVASPREAD_TUNER_STATE_ERROR

} NovaspreadTTunerState;
```

COMPONENTS

NOVASPREAD_TUNER_STATE_STOPPED
The tuner is stopped. No data can be read via the tuner's ReadData() function.

NOVASPREAD_TUNER_STATE_STREAMING
The tuner was started successfully. Data can be read via the tuner's ReadData() function.

NOVASPREAD_TUNER_STATE_ERROR
An error occurred. When this state is reached, an error code shall be returned when NovaspreadTunerGetError() is called. To leave this state, NovaspreadTunerStop() must be called.

SEE ALSO

`NovaspreadTTunerStateChangeListener`

3.2.3. NovaspreadTTunerStateChangeListener

A listener of this type can be set at a Tuner. It is called every time the tuner's state changes.

SYNTAX

```
typedef void
(* NovaspreadTTunerStateChangeListener ) (
  void *                 aContext,
  NovaspreadTTunerState aNewState );
```

PARAMETERS

aContext
This context is passed unchanged from the NovaspreadTunerSetStateChangeListener() function.

aNewState
The new state of the tuner.

SEE ALSO

`NovaspreadTTunerState`

3.2.4. NovaspreadTTunerDataAvailableListener

A function of this type can be set at the Tuner. When NovaspreadTunerReadData() returns 0, because no data is available, the registered DataAvailableListener will be called as soon as data is available.

SYNTAX

```
typedef void
(* NovaspreadTTunerDataAvailableListener ) (
  void * aContext );
```

PARAMETERS

aContext
This context is passed unchanged from the NovaspreadTunerSetDataAvailableListener() function.

SEE ALSO

```
NovaspreadTunerSetDataAvailableListener()
NovaspreadTunerReadData()
```

3.2.5. NovaspreadTTunerAllocationParameter

A Tuner is allocated for particular tuner parameters. Additionally it is defined whether a transcoder and a transcryptor are required.

When the allocation request is fulfilled, the Tuner must be tuned to the given tuner parameters.

SYNTAX

```
typedef struct
{
  NovaspreadTTunerParameters TunerParameters;
  NovaspreadTBoolean         AllocateTranscoder;
  NovaspreadTBoolean         AllocateTranscryptor;

} NovaspreadTTunerAllocationParameter;
```

COMPONENTS

TunerParameters
A Tuner is requested, which can fulfill this TunerParameter.

AllocateTranscoder
NOVASPREAD_TRUE if a transcoder shall be allocated. NOVASPREAD_FALSE if no transcoder is required.

AllocateTranscryptor
NOVASPREAD_TRUE if a transcryptor shall be allocated. NOVASPREAD_FALSE if no transcryptor is required.

SEE ALSO

```
NovaspreadTTunerParameters
```

3.2.6. NovaspreadTTunerAllocationFinishedListener

An AllocationFinishedListener is passed to NovaspreadHostAllocateTuner(). The listener shall be called when a Tuner is available, resp. when the allocation request is denied. This callback shall be called only once per allocation request.

SYNTAX

```
typedef void
(* NovaspreadTTunerAllocationFinishedListener ) (
  void *            aContext,
  NovaspreadTTuner aTuner );
```

PARAMETERS

aContext
The context which was given to NovaspreadHostAllocateTuner() shall be passed unchanged to this listener.

aTuner
A Tuner, if the allocation request could be fulfilled successfully. NOVASPREAD_NULL if the request was denied.

3.2.7. NovaspreadTTunerReleaseRequestedListener

A ReleaseRequestedListener is passed to NovaspreadHostAllocateTuner(). The listener can be called by the Host if the tuner is needed for a usage with higher priority. When this listener is called, the NovaspreadServer will soon call NovaspreadHostReleaseTuner() to release the tuner. This callback shall be called only once per allocation request.

SYNTAX

```
typedef void
(* NovaspreadTTunerReleaseRequestedListener ) (
  void *            aContext,
  NovaspreadTTuner aTuner );
```

PARAMETERS

aContext
The context which was given to NovaspreadHostAllocateTuner() shall be passed unchanged to this listener.

aTuner
The Tuner which shall be released.

3.2.8. NovaspreadTunerGetTransportSessionId

Gets the 32-bit TransportSessionId of the Tuner. The returned ID uniquely identifies the stream received by the tuner. This ID is used to indicate the stream to be decrypted with NovaspreadCa functions and for control DRM specific re-encryption with NovaspreadDrm.

SYNTAX

```
PUBLIC NovaspreadTUInt32
NovaspreadTunerGetTransportSessionId(
  NovaspreadTTuner This );
```

Parameters

This
The Tuner.

Return Value

The 32-bit TransportSessionId.

See Also

```
NovaspreadTCaServiceUsageRulesReceivedListener
NovaspreadCaSetServiceUsageRulesReceivedListener()
NovaspreadDrmStart()
```

3.2.9. NovaspreadTunerSetTranscoding

Sets the Transcoding of the Tuner. This function will be called by NovaspreadServer only, if the Tuner was requested for transcoding during NovaspreadHostAllocateTuner().

The transcoding can be changed even if the Tuner is already started. In this case the tuner must check, which part of the transcoding parameters has changed. E.g. if the transcoding parameters for video did not change and only the parameters for audio changed, the video stream shall not be stopped. This is necessary to allow clients to request a different audio stream, e.g. in a different language, without interrupting the video during this change.

Syntax

```
PUBLIC NovaspreadTBoolean
NovaspreadTunerSetTranscoding(
  NovaspreadTTuner          This,
  NovaspreadTTranscoding *  aTranscoding );
```

Parameters

This
The Tuner.

aTranscoding
The Transcoding to be set.

Return Value

NOVASPREAD_TRUE
if the Transcoding was set successfully.

NOVASPREAD_FALSE
otherwise.

See Also

```
NovaspreadTTranscoding
```

3.2.10. NovaspreadTunerSetStateChangeListener

Sets a StateChangeListener at this tuner.

Syntax

```
PUBLIC NovaspreadTBoolean
NovaspreadTunerSetStateChangeListener(
  NovaspreadTTuner                    This,
  NovaspreadTTunerStateChangeListener aListener,
  void *                              aContext );
```

Parameters

This
The tuner.

aListener
The listener. Pass NOVASPREAD_NULL to unset the listener.

aContext
This context shall be passed unchanged to the listener.

RETURN VALUE

NOVASPREAD_TRUE
if successful

NOVASPREAD_FALSE
otherwise

SEE ALSO

```
NovaspreadTTunerStateChangeListener
NovaspreadTTunerState
```

3.2.11. NovaspreadTunerGetError

If the Tuner changed to state ERROR, this function shall return an error code.

SYNTAX

```
PUBLIC NovaspreadTTunerError
NovaspreadTunerGetError(
  NovaspreadTTuner This );
```

PARAMETERS

This
The Tuner.

RETURN VALUE

The error code.

SEE ALSO

```
NovaspreadTTunerError
```

3.2.12. NovaspreadTunerSetPids

Sets the pids of the transport stream packets that shall be available in the Tuner's received stream. All previously set pids are replaced by this list. To reset all pids, pass NOVASPREAD_NULL for aPids and set aNoOfPids to 0. This function can be called when the tuner is stopped as well as when the tuner is started.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadTunerSetPids(
  NovaspreadTTuner     This,
  NovaspreadTUInt16 *  aPids,
  NovaspreadTUInt32    aNoOfPids );
```

PARAMETERS

This
The Tuner.

aPids
The array of pids.

aNoOfPids
The number of pids in the array.

Return Value

`NOVASPREAD_TRUE`
if successful

`NOVASPREAD_FALSE`
otherwise

3.2.13. NovaspreadTunerAddPids

Adds pids, which shall additionally be received by this Tuner. This function can be called when the tuner is stopped as well as when the Tuner is started.

Syntax

```
PUBLIC NovaspreadTBoolean
NovaspreadTunerAddPids(
  NovaspreadTTuner     This,
  NovaspreadTUInt16 *  aPids,
  NovaspreadTUInt32    aNoOfPids );
```

Parameters

`This`
The Tuner.

`aPids`
The array of pids that shall additionally be received.

`aNoOfPids`
The number of pids in the list.

Return Value

`NOVASPREAD_TRUE`
if successful

`NOVASPREAD_FALSE`
otherwise

See Also

`NovaspreadTunerRemovePids()`

3.2.14. NovaspreadTunerRemovePids

Removes pids, which should no longer be received by this Tuner. This function can be called when the Tuner is stopped as well as when the tuner is started.

Syntax

```
PUBLIC void
NovaspreadTunerRemovePids(
  NovaspreadTTuner     This,
  NovaspreadTUInt16 *  aPids,
  NovaspreadTUInt32    aNoOfPids );
```

Parameters

`This`
The Tuner.

`aPids`
The array of pids that should no longer be received.

`aNoOfPids`
The number of pids in the array.

SEE ALSO

```
NovaspreadTunerAddPids()
```

3.2.15. NovaspreadTunerStart

This function starts the Tuner. When TunerStart() was called, NovaspreadTunerReadData() can be called to receive transport stream data.

Transcryption can be changed only when the tuner is stopped. Pids can be added, removed and set when the tuner is stopped as well as when the tuner is started.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadTunerStart(
  NovaspreadTTuner This );
```

PARAMETERS

This
The Tuner.

RETURN VALUE

NOVASPREAD_TRUE
if successful

NOVASPREAD_FALSE
otherwise

SEE ALSO

```
NovaspreadTunerStop()
```

3.2.16. NovaspreadTunerStop

This function stops the data reception of this Tuner.

When this function returns, a registered DataAvailableListener shall no longer be called.

SYNTAX

```
PUBLIC void
NovaspreadTunerStop(
  NovaspreadTTuner This );
```

PARAMETERS

This
The Tuner.

SEE ALSO

```
NovaspreadTunerStart()
```

3.2.17. NovaspreadTunerIsLocked

Returns whether the Tuner is locked or not. A Tuner is locked if a signal is detected for the set TunerParameter and the demodulator is able to decode the signal. This means a Tuner receives data only if it is locked. A valid lock status is returned if the tuner is started as well as when the tuner is stopped.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadTunerIsLocked(
  NovaspreadTTuner This );
```

PARAMETERS

This
The Tuner.

RETURN VALUE

NOVASPREAD_TRUE
if the Tuner is locked.

NOVASPREAD_FALSE
otherwise

3.2.18. NovaspreadTunerGetSignalInfo

Gets the SignalInfo of this Tuner. See the data type NovaspreadTTunerSignalInfo for a full description of the SignalInfo.

SYNTAX

```
PUBLIC NovaspreadTTunerSignalInfo
NovaspreadTunerGetSignalInfo(
  NovaspreadTTuner This );
```

PARAMETERS

This
The Tuner.

RETURN VALUE

The SignalInfo

SEE ALSO

`NovaspreadTTunerSignalInfo`

3.2.19. NovaspreadTunerSetDataAvailableListener

Sets a DataAvailableListener. Only one DataAvailableListener can be set at a Tuner.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadTunerSetDataAvailableListener(
  NovaspreadTTuner                     This,
  NovaspreadTTunerDataAvailableListener aListener,
  void *                               aContext );
```

PARAMETERS

This
The Tuner.

aListener
The DataAvailableListener. NOVASPREAD_NULL is passed to unset the listener.

aContext
This context shall be passed unchanged to the listener.

RETURN VALUE

NOVASPREAD_TRUE
if successful

NOVASPREAD_FALSE
otherwise

See Also

```
NovaspreadTTunerDataAvailableListener
NovaspreadTunerReadData()
```

3.2.20. NovaspreadTunerReadData

As soon as the Tuner is started, this function will write 188 bytes long transport stream packets to the buffer. This function must be called periodically to avoid a Tuner internal buffer overflow.

It is not required to write only complete transport stream packets to the buffer. If e.g. a buffer size of 200 bytes is given, and 200 bytes are available, they shall be written to the buffer.

If this function is called when the Tuner is stopped, it will not write data to the buffer and return 0.

If the tuner is started, NovaspreadTunerReadData() will return 0, if no data is available. As soon as data is available again, a previously set DataAvailableListener shall be called.

NovaspreadTunerReadData() will not be called in the context of the DataAvailableListener.

Syntax

```
PUBLIC NovaspreadTUInt32
NovaspreadTunerReadData(
  NovaspreadTTuner   This,
  NovaspreadTUInt8 * aBuffer,
  NovaspreadTUInt32  aBufferSize );
```

Parameters

This
The Tuner.

aBuffer
Transport stream packets are written to this buffer.

aBufferSize
The size of the buffer.

Return Value

The number of bytes written to the buffer. If there are no transport stream packets available 0 is returned.

3.3. NovaspreadCa

The NovaspreadCa interface contains functions Novaspread requires from the CA system. The main purpose of this interface is to retrieve the UsageRules on platform and service level from the CA system for a specific stream.

3.3.1. NovaspreadTCaDvbId

The DvbId identifies a DVB service.

Syntax

```
typedef struct
{
  NovaspreadTUInt16 OriginalNetworkId;
  NovaspreadTUInt16 TransportStreamId;
  NovaspreadTUInt16 ServiceId;

} NovaspreadTCaDvbId;
```

Components

OriginalNetworkId

The OriginalNetworkId

TransportStreamId
The TransportStreamId

ServiceId
The ServiceId

3.3.2. NovaspreadTCaPlatformUsageRulesReceivedListener

This listener must be called, when platform dependent UsageRules have been received. The structure of the passed UsageRules depends on the used CA system.

In case of a Nagra CA system, the payload of the IRD Command defined by tag=0x64 must be passed.

SYNTAX

```
typedef void
(* NovaspreadTCaPlatformUsageRulesReceivedListener ) (
  void *              aContext,
  NovaspreadTUInt8 *  aPlatformUsageRules,
  NovaspreadTUInt32   aLength );
```

PARAMETERS

aContext
The context which was given to NovaspreadCaSetPlatformUsageRulesReceivedListener() shall be passed unchanged to this listener.

aPlatformUsageRules
The UsageRules on platform level.

aLength
The length of the UsageRules buffer.

SEE ALSO

`NovaspreadCaSetPlatformUsageRulesReceivedListener()`

3.3.3. NovaspreadTCaServiceUsageRulesReceivedListener

A listener of this type can be registered at NovaspreadCa. It is to be called whenever new UsageRules for the particular service are received.

If UsageRules are only received if they are updated, this listener must be called at least once when it is registered with NovaspreadCa.

SYNTAX

```
typedef void
(* NovaspreadTCaServiceUsageRulesReceivedListener ) (
  void *              aContext,
  NovaspreadTUInt32   aTransportSessionId,
  NovaspreadTUInt8 *  aServiceUsageRules,
  NovaspreadTUInt32   aLength );
```

PARAMETERS

aContext
This context is passed unchanged from the NovaspreadCaSetUsageRulesReceivedListener() function.

aTransportSessionId
The UsageRules of this TransportSession have been updated.

aServiceUsageRules
The UsageRules on service level. For Nagra these are 3 bytes extracted from the ECM.

aLength
The length of the aServiceUsageRules.

3.3.4. NovaspreadCaGetInfo

This function returns information about the CA system. The returned CaInfo will be destroyed by NovaspreadServer.

SYNTAX

```
PUBLIC NovaspreadTCaInfo
NovaspreadCaGetInfo(
  void );
```

RETURN VALUE

A new CaInfo if successful. NOVASPREAD_NULL otherwise.

SEE ALSO

`NovaspreadTCaInfo`

EXAMPLE

```
// An implementation of this function shall proceed as follows:

PUBLIC NovaspreadTCaInfo
NovaspreadCaGetInfo ( void )
{
  NovaspradTCaInfo  caInfo;
  const char *      caVendor = "Nagra";

  caInfo = NovaspreadCaInfoCreate();
  if ( ! caInfo )
    return NOVASPREAD_NULL;

  // For Nagra, set the NUId by calling
  // NovaspreadCaInfoSetChipsetUid().

  NovaspreadCaInfoSetCaVendor( caInfo, caVendor );

  // Call NovaspreadCaInfoSet..() functions here to set
  // information about the CA system and the smartcard.

  return caInfo;
}
```

3.3.5. NovaspreadCaSetDvbId

Sets the DvbId which is needed for transcryption.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadCaSetDvbId(
  NovaspreadTUInt32  aTransportSessionId,
  NovaspreadTCaDvbId aDvbId );
```

PARAMETERS

aTransportSessionId
For this TransportSession the DvbId is set.

aDvbId
The DvbId to be set.

RETURN VALUE

NOVASPREAD_TRUE
if successful

NOVASPREAD_FALSE
otherwise

3.3.6. NovaspreadCaSetPlatformUsageRulesReceivedListener

This functions sets a listener, which shall be called when platform specific usage rules are received.

SYNTAX

```
PUBLIC void
NovaspreadCaSetPlatformUsageRulesReceivedListener(
  NovaspreadTCaPlatformUsageRulesReceivedListener * aListener,
  void *                                            aContext );
```

PARAMETERS

aListener
The PlatformUsageRulesReceivedListener to be set. NOVASPREAD_NULL is passed to unset the listener.

aContext
The context which shall be passed unchanged to the listener.

SEE ALSO

NovaspreadTunerGetTransportSessionId()
NovaspreadTCaPlatformUsageRulesReceivedListener

3.3.7. NovaspreadCaSetServiceUsageRulesReceivedListener

Sets a ServiceUsageRulesReceivedListener. The TransportSessionId identifies the stream received by a Tuner for which the UsageRules should be acquired. The TransportSessionId can be retrieved with the function NovaspreadTunerGetTransportSessionId().

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadCaSetServiceUsageRulesReceivedListener(
  NovaspreadTUInt32                              aTransportSessionId,
  NovaspreadTCaServiceUsageRulesReceivedListener aListener,
  void *                                         aContext );
```

PARAMETERS

aTransportSessionId
For this TransportSession the listener is set.

aListener
The ServiceUsageRulesReceivedListener to be set. NOVASPREAD_NULL is passed to unset the listener.

aContext
This context shall be passed unchanged to the listener.

RETURN VALUE

NOVASPREAD_TRUE
if successful

NOVASPREAD_FALSE
otherwise

SEE ALSO

```
NovaspreadTunerGetTransportSessionId()
NovaspreadTCaServiceUsageRulesReceivedListener
```

3.4. NovaspreadDrm

The NovaspreadDrm interface contains functions to control the DRM system. With the functions of this interface the encryption of the stream can be controlled. The license for the streamed content can be extracted by the function NovaspreadDrmStart(). This license is passed to Novaspread-C.

3.4.1. NovaspreadTDrmLicense

A license returned by the DRM system.

SYNTAX

```
typedef struct
{
  NovaspreadTUInt8 * License;
  NovaspreadTUInt32  LicenseLength;

} NovaspreadTDrmLicense;
```

COMPONENTS

License
The License as byte array.

LicenseLength
The length of the License.

3.4.2. NovaspreadTDrmLicenseParameter

This parameter is passed to the NovaspreadDrmStart() function.

If an OldLicense is passed, this license shall be re-used. If this is not possible or if no OldLicense is used, the other components shall be used to create a new license.

SYNTAX

```
typedef struct
{
  NovaspreadTDrmLicense OldLicense;
  NovaspreadTUInt32     Duration;
  NovaspreadTUInt8 *    UsageRules;
  NovaspreadTUInt32     UsageRulesLength;

} NovaspreadTDrmLicenseParameter;
```

COMPONENTS

OldLicense
A license previously returned by a call to NovaspreadDrmStart(). If no OldLicense is available, the content of this OldLicense is 0.

Duration
Defines how long a new license shall be valid. In seconds.

UsageRules
These UsageRules shall be set in TDvlRecordSessionParameters.pSpecifcMetadata.

UsageRulesLength
This UsageRulesLength shall be set in TDvlRecordSessionParameters.specificMetadataSize.

3.4.3. NovaspreadDrmStart

This function starts re-encryption and returns a license.

The TransportSessionId, which is passed to this function, can be got by a call to NovaspreadTunerGetTransportSessionId().

dvlStartRecordEx() shall be called with the given aLicenseParameter and the new license shall be returned.

SYNTAX

```
PUBLIC NovaspreadTBoolean
NovaspreadDrmStart(
  NovaspreadTUInt32                aTransportSessionId,
  NovaspreadTDrmLicenseParameter * aLicenseParameter,
  NovaspreadTDrmLicense *          aLicense );
```

PARAMETERS

aTransportSessionId
For this TransportSession the re-encryption is started.

aLicenseParameter
The parameters which shall be passed to dvlStartRecordEx()

aLicense
OUT: The new license

RETURN VALUE

NOVASPREAD_TRUE
if successful

NOVASPREAD_FALSE
otherwise. In this case there is no need to call NovaspreadDrmStop().

SEE ALSO

```
NovaspreadTunerGetTransportSessionId()
NovaspreadDrmStop()
NovaspreadTDrmLicenseParameter
NovaspreadTDrmLicense
```

3.4.4. NovaspreadDrmStop

Stops the re-encryption of the stream.

For Nagra dvlStopRecord() shall be called.

SYNTAX

```
PUBLIC void
NovaspreadDrmStop(
  NovaspreadTUInt32 aTransportSessionId );
```

PARAMETERS

aTransportSessionId
For this TransportSession the encryption is stopped.

SEE ALSO

```
NovaspreadDrmStart
NovaspreadTunerGetTransportSessionId()
```

Published by:

SES Platform Services GmbH

Beta Straße 1-10

85774 Unterföhring

Germany

For more information about SES, visit

www.ses-ps.com or email **info@ses-ps.com**