bitcontrol® Digital TV Link

Software Development Kit (SDK)

Version 2.1

bitcontrol® Digital TV Link

Software Development Kit (SDK)

Version 2.1

Table of Contents

List of Figures

List of Tables

BitCtrl information

1. Copyright

The documentation reflects the present development stage of the software and its documentation. If you should come across any errors or unclear passages in the documentation please contact:

BitCtrl Systems GmbH

Weißenfelser Str. 67

04229 Leipzig, Germany

Tel. +49-341-490670

Fax +49-341-4906715

E-Mail: <info@bitctrl.de>

2. Liability

BitCtrl Systems GmbH (referred to as BitCtrl in the following) will not accept liability (whether specifically or implicitly) for the software and its components. This includes any claims regarding usage and suitability of the software for a specific purpose. BitCtrl will in no way accept liability for coincidental, indirect or consequential damage resulting from misuse or correct usage of the software. This also applies should BitCtrl be informed prior to this of such possible damage.

The general terms of business for BitCtrl Systems GmbH will apply. Rights to change software and documentation accrued through technical advancements are reserved.

Release levels

Date	Version	Description	Author
30.11.2007	1.0	Documents creation	Krosse
02.12.2008	1.1	Appendix A added	Hösel / Babin
31.03.2009	1.2	Appendix B added	Hösel / Babin
10.07.2014	2.1	Documentation renewal - valid for *bitcontrol® Digital TV Link* V3.8	Baranowsky / Winter

Chapter 1. Introduction

This documentation of the *bitcontrol® Digital TV Link* Software Development Kit (SDK) should help to understand the functionalities of *bitcontrol® Digital TV Link* on developers base. Severals functions of the *bitcontrol® Digital TV Link* and its interfaces will be described by means of some code examples. So it is possible to bind external software applications to the *bitcontrol® Digital TV Link*.

Note

To understand the code examples, advanced knowledge of the scripting language JScript is required, because this documentation does not contain the elemental architectures of this language. Furthermore it is necessary, to understand the basics of XML. Please use the internet to catch up on something about the mentioned topics.

Chapter 2. Usage of JScript interface

The *bitcontrol® Digital TV Link* provides a JScript interface, that allows a remote control of the program. This interface is realized through the Multistream Server plug-in. So the first constraint is that you have a started *bitcontrol® Digital TV Link* with a running Multistream Server plug-in. There is a *HTTP Media Session* where you can add or remove different URL. This parameterization can you change in the xml file. This will be explain in ??? . Most of the parameters of *bitcontrol® Digital TV Link* and its other plug-ins can be changed by the usage of this interface. So, the readout of information is also possible. In Appendix A, *JScript Snippets* [12] you can find some JScript examples.

The communication between server and program can be separated in two different JScript versions: *expression form* or *normal form* . The *expression form* is mandatory. The *normal form* can be used to set parameters. The differencebetween the two forms is the attribute *expression* inside the **script** tag.

The *normal form* looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>
    <DTV version="1.0">
         <script language="JScript">
              <![CDATA[
                ...
           ]]>
         </script>
    </DTV>
```

Figure 2.1. Code example - Server program communication / normal form

The *expression form* differs minimal.

```
<?xml version="1.0" encoding="UTF-8"?>
    <DTV version="1.0">
         <script language="JScript"  expression="1">
              <![CDATA[
              ...
           ]]>
         </script>
    </DTV>
```

Figure 2.2. Code example - Server program communication / expression form

The answer of the server follows also a defined scheme.

```
<?xml version="1.0" encoding="UTF-8"?>
    <DTV>
         <script errno="0" result="..."
         ...
         </script>
    </DTV>
```

Figure 2.3. Code example - Server program communication / answer of the server

The attribute **errno** has the value 0 in success case. In case of error, the value correlates the Windows® error code. During information readout, the information will be displayed inside the **script** attribute **result** .

The parser respectively interpreter that is included in the HTTP server stops the processing in case of error. So it is advisable to write a **try-catch** block around every task block. Occurred errors can be intercepted via this way.

2.1. *DTVLink* parameterization

The constant *DTVLink* inside the script represents the filter and allows the readout and the setting of parameters.

```
<Script Language="JScript" GlobalContext="true" Expression="true"
 Response="XMLResponse" Id="RequestNum1">
   DTVLink.EnableVideoOutPin=true;
   DTVLink.EnableAudioOutPin=true;
   DTVLink.LogMessage(3, "JScript", "Log Message", 0, 0);
 </Script>
```

Figure 2.4. Code example - DTVLink parameterization

The JavaScript part shows the usage of the variable *DTVLink* for the configuration and parameterization of *bitcontrol® Digital TV Link* . In ??? you can find some more snippets.

The following functions or variables are usable:

Name	**Description**	**Read**	**Write**
AudioStreamBaseTimestampSec	Access to the audio stream time stamp	yes	no
AutoloadPlugins	Access to the option for the automatic plug-in start	yes	yes
BufferingTime	Access to the buffer time of input streams	yes	yes
DataFolder	Access to the program directory	yes	no
EnabledAudioOutPin	Access to the audio pin	yes	yes

Name	Description	Read	Write
EnabledVideoOutPin	Access to the video pin	yes	yes
LicenseMode	Readout of the license mode	yes	no
LicenseName	Access to the license name	yes	no
LicenseRegnum	Access to the license number	yes	no
LogMessage(int loglevel,Sourcename,Message,clrBackground, clrForeground)	Access to the console text	no	yes
Plugins[String plugin]	Access to the plug-in list	yes	no
ShowConsole=true or false	Access to the console or hide the console.	yes	no
URL	Access to the URL	yes	yes
Version	Access to versions	yes	no
VideoStreamBaseTimestampSec	Access to the video stream base time stamp	yes	no
VideoDecoder	Access to the video decoder	yes	no

Table 2.1. bitcontrol® Digital TV Link functions and variables

2.2. Plug-in

2.2.1. Access to a plug-in

The access to the plug-ins takes place with help of the name and via the **DTVLink** attribute **Plugins** .

```
...
DTVLink.Plugins["Capture"].StartupType=1;
...
```

Figure 2.5. Code example - Access to a plug-in

2.2.1.1. Load and unload a plug-in

The functions **Load** and **Unload** are used to load or to unload a plug-in. The condition request takes place via the variable **State** and the numbers **0** and **1** .

```
DTVLink.Plugins["Capture"].State=1;
```

Figure 2.6. Code example - Load a plug-in

```
DTVLink.Plugins["Capture"].State=0;
```

Figure 2.7. Code example - Unload a plug-in

2.2.1.2. Start and stop a plug-in

The functions **Start** and **Stop** are used to start or to stop a plug-in. The condition request takes place via the variable **State** and the numbers **0** and **2** .

```
DTVLink.Plugins["Capture"].State=2;
```

Figure 2.8. Code example - Start a plug-in

The following functions or variables are available for all *bitcontrol® Digitial TV Link* plug-ins.

Name	**Description**	**Read**	**Write**
State	Starts, Loads and unloads a plug-in. (0 = disabled, 1 = auto, 2 = manual)	no	yes
Name	The choosen plug-in name.	yes	yes
Class	The Class of the plug-in.	yes	no
StartupType	The StartupType for this plug-in. (0 = disabled, 1 = auto, 2 = manual)	yes	yes
Priority	The priority of this plug-in on startup. (Lower value means higher priority)	yes	yes
Hidden	Indicating if the plug-in is shown in the *bitcontrol® Digital TV Link* options menu. You need the boolean values true and false.	yes	yes
InfoURL	An URL where you can find some more information about this plug-in.	yes	no

Table 2.2. Common plugin functions and variables

2.2.2. Access to specific methods of a plug-in interface

The following chapter lists only the methods that are provided by the plug-in. Furthermore the methods will be explained with help of a short code example.

2.2.2.1. Plug-in Recorder

```
DTVLink.Plugins["Recorder"].Plugin.LimitVolume=20;
DTVLink.Plugins["Recorder"].Plugin.LimitDuration=15;
DTVLink.Plugins["Recorder"].Plugin.AutoDelete = true;
DTVLink.Plugins["Recorder"].Plugin.AutoDeleteTime = 1440;
```

Figure 2.9. Different code examples - Access to specific methods of the plug-in Recorder

The following methods or variables are usable:

Name	**Description**	**Read**	**Write**
LimitDuration	Access to the time limits of recording files (min)	yes	yes
LimitVolume	Access to the size limits of recording files (MByte)	yes	yes
AutoDelete	Enable/Disable automatic deletion of files	yes	yes
AutoDeleteTime	The time (min) after that recorded files will be deleted if *AutoDelete* is true	yes	yes

Table 2.3. Recorder plug-in functions and variables

2.2.2.2. Plug-in DBox2

```
DTVLink.Plugins["DBox2"].Plugin.Address="192.168.2.108";
DTVLink.Plugins["DBox2"].Plugin.ChannelsRefreshInterval=1440;
```

Figure 2.10. Different code examples - Access to specific methods of the plug-in DBox2

The following methods and variables are usable:

Name	**Description**	**Read**	**Write**
Address	Access to the configured STB network address	yes	yes
ChannelsRefreshInterval	Access to the update interval	yes	yes
RefreshChannelList	Refreshing the channel list.	-	-

Table 2.4. DBox2 plug-in functions and variables

2.2.2.3. Plug-in DreamBox

```
DTVLink.Plugins["DreamBox"].Plugin.Address="192.168.1.50";
DTVLink.Plugins["DreamBox"].Plugin.Model=5;
DTVLink.Plugins["DreamBox"].Plugin.StreamMethod=1;
DTVLink.Plugins["DreamBox"].Plugin.ChannelsRefreshInterval=1440;
```

Figure 2.11. Different code examples - Access to specific methods of the plug-in DreamBox

The following methods or variables are usable:

Name	**Description**	**Read**	**Write**
Address	Access to the configured STB network address.	yes	yes
ChannelsRefreshInterval	Access to the update interval (Min).	yes	yes
Model	Access to the model type. (1 = DM500, 2 = DM600, 3 = DM800, 4 = DM7000, 5 = DM7020, 6 = DM7025)	yes	yes
StreamMethode	Access to the transmission method. (1 = PES, 2 = TS)	yes	yes
ModelFirmwareVersion	The Firmware version of the used Dreambox.	yes	yes
RefreshChannelList	Refreshing the channel list.	-	-

Table 2.5. Dreambox plug-in functions and variables

2.2.2.4. Plug-in ReelBox

```
DTVLink.Plugins["ReelBox"].Plugin.Address="192.168.2.105";
DTVLink.Plugins["ReelBox"].Plugin.ChannelsRefreshInterval=1440;
```

Figure 2.12. Different code examples - Access to specific methods of the plug-in ReelBox

The following methods or variables are usable:

Name	**Description**	**Read**	**Write**
Address	Access to the configured STB network address	yes	yes
ChannelsRefreshInterval	Access to the update interval	yes	yes
RefreshChannelList	Refreshing the channel list.	-	-

Table 2.6. ReelBox plugin functions and variables

2.2.2.5. Plug-in RTSP Receiver

```
DTVLink.Plugins["RTSP Receiver"].Plugin.
    Address="rtsp://192.168.134.22:41001/live";
```

Figure 2.13. Code example - Access to specific methods of the plug-in RTSP Receiver

The following methods or variables are usable:

Name	**Description**	**Read**	**Write**
Address	Access to the source address of the network stream	yes	yes

Table 2.7. RTSP Receiver plug-in functions and variables

2.2.2.6. Plug-in Multistream Server

```
DTVLink.Plugins["Server"].Plugin.Version
```

Figure 2.14. Different code examples - Access to specific methods of the plug-in Multistream Server

The plug-in combines the formerly HTTP server and RTSP server and contains the Media Sessions at plug-in statistic.

It is possible to change server properties (for example user rights) by editing the Server.xml file that can be found here *C:\ProgramData\BitCtrl\DTVLink\Plugins* .

Note

The Server.xml file should only be edited by experienced users!

2.2.2.7. Plug-in BDA Tuner

```
DTVLink.Plugins["BDA Tuner"].Plugin.
    BDATuner="Cinergy Hybrid T USB XS Digital Tuner";
DTVLink.Plugins["BDA Tuner"].Start();
DTVLink.SwitchToChannelByName("BDA Tuner","3sat");
```

Figure 2.15. Code example - Access to specific methods of the plug-in BDA Tuner

The following methods or variables are usable:

Name	Description	Read	Write
BDATuner	Defines a BDA tuner device name	yes	yes
BDATunerId	Defines a BDA tuner device unique hardware id	yes	yes
BDADemodulator	Defines a BDA demodulator device name. When tuner has no demodulator set empty value.	yes	yes
BDADemodulatorId	Defines a BDA demodulator device unique hardware id. When tuner has no demodulator set empty value.	yes	yes
BDACapture	Defines a BDA capture device name. When tuner has no capture set empty value.	yes	yes
BDACaptureId	Defines a BDA capture device unique hardware id. When tuner has no capture set empty value.	yes	yes
BDASystemType	The BDA tuner DVB system type. This parameter can be one of the following values: -1 - Unknown DVB system 1 - DVB-S (satellite system) 2 - DVB-C (cable system) 3 - DVB-T (terrestrial system)	yes	yes
DiSEqCviaInputRange	The BDA tuner initial DVB-S DiSEqC.	yes	yes
TotalReceivedBytes	The number of received bytes.	yes	no

Table 2.8. BDA Tuner plug-in functions and variables

Chapter 3. *bitcontrol Decoder*

3.1. Access to *bitcontrol® Decoder*

The Multistream Server plug-in allows the access to *bitcontrol® Decoder.*

```
DTVLink.VideoDecoder.AlwaysDeinterlace=true;
DTVLink.VideoDecoder.AspectRatioX=5;
DTVLink.VideoDecoder.AspectRatioY=3;
DTVLink.VideoDecoder.FrameSkipping=true;
```

Figure 3.1. Different code examples - Access to bitcontrol® Decoder

The following methods and variables are usable:

Name	Description	Read	Write
AlwaysDeinterlace	Access to the option, if deinterlacing always should be used. This can be set on or off with true or false.	yes	yes
AspectRatioX	The aspect ration in x direction. This must be use in connection with AspectRatioY.	yes	yes
AspectRatioY	The aspect ration in y direction. This must be use in connection with AspectRatioX.	yes	yes
ColorCorrection	Reads out, if color correction is set	yes	yes
Deinterlace	Reads out deinterlace mode (0 - auto, 1 - progrssive, 2 - weave, 3 - hardware bob, 4 - software bob, 5 - software bob double rate)	yes	yes
DXVA	Set DXVA and read it out. For example DXVA1MPEG1=true set DXVA at MPEG1 or DXVA1MPEG2=false don't set DXVA at MPEG2. You can use: MPEG1, MPEG2, MPEG4, H263, H264, WMV8, WMV9 and VC1.	yes	yes
FrameSkipping	Reads out, if pictures can be skipped. This can be set on or off with true or false.	yes	yes
FrameSkippingTolerance	Reads out how many deficient pictures are allowed	yes	yes
LicenseMode	Reads out the license mode. (0 - normal, 1 - demo, 2 - trial, 3 - expired)	yes	no
LicenseName	Reads out the license name	yes	no

Name	**Description**	**Read**	**Write**
LicenseRegnum	Reads out the license number	yes	no
OutputConnectionAllowList	Access to the list of filters, which are allowed to use the decoder		
OutputConnectionDenyList	Access to the list of filters, which are allowed to use the decoder	yes	yes
OutputTypesPriority	Access to the priorities of output formats	yes	yes
OutputTypesPriorityDXVA	Access to the priorities of DirectX® output formats. This connection must be used with DXVA values (for example OutputTypesPriorityDXVA1_ MPEG1 or OutputTypesPriorityDXVA2_MPEG2). The different video types are the same like DXVA.	yes	yes
OutputTypesPriorityHW	Access to the priorities of the output formats in deinterlace mode.	yes	yes
RendererLateness	Reads out the render delay	yes	no
UseHadwareAcceleratedDeinterlace	Access to the option, if hardware accelerated deinterlace should be used. This can be set on or off with true or false.	yes	yes
Version	Reads the decoder version.	yes	no

Table 3.1. bitcontrol® Decoder functions and variables

Appendix A. JScript Snippets

A.1. JScript Snippets for *bitcontrol® Digital TV Link* GET requests

Get requests can be used to read out information from *bitcontrol® Digital TV Link* . The source code examples show the output of the selected *Get request* .

A.1.1. Get Version

Returns the version of the used *bitcontrol® Digital TV Link* .

```
<?xml version="1.0" encoding="utf-8"?>
    <Result Id="RequestNum3">
          v3.8.0.0, Release(Unicode)X86, assembled on May 27 2014
    </Result>
```

Figure A.1. JScript Snippet output of >Get Version

A.1.2. Get All Channels

Returns all available Channels of started input plug-ins.

```
<?xml version="1.0" encoding="utf-8"?>
 <Sources>
  <Source>
   <Name>AXIS</Name>
    <Channels>
     <Fields Count="4">
      <Field Id="Id">Id</Field>
      <Field Id="Name">Name</Field>
      <Field Id="Network">Network</Field>
      <Field Id="URL">URL</Field>
     </Fields>
     <Channel Enable="true" Active="false">
      <Id>0</Id>
      <Name>AXIS M3011 - 00408C9492B7</Name>
      <Network>unicast</Network>
      <URL>rtsp://192.168.1.28/axis-media/media.amp?videocodec=mpeg4</URL>
     </Channel>
    </Channels>
  </Source>
  <Source>
   <Name>File</Name>
    <Channels>
     <Fields Count="4">
      <Field Id="Id">Id</Field>
      <Field Id="Name">Name</Field>
      <Field Id="URL">URL</Field>
      <Field Id="Duration">Duration</Field>
     </Fields>
     <Channel Enable="true" Active="false">
      <Id>2</Id>
      <Name>FIFASPOT_LVB_3gmp4_352x288.mp4</Name>
      <URL>C:\Video\video_samples\MPEG-4 3GP\
      FIFASPOT_LVB_3gmp4_352x288.mp4</URL>
     </Channel>
    </Channels>
  </Source>
 </Sources>
```

Figure A.2. JScript Snippet output for Get All Channels

The source code example shows two active input plug-ins (Axis plug-in and File plug-in) as video sources. Both plug-ins have one Channel that isn't active. Inside the Channel tag are further information about the Channel available.

A.1.3. Get Active Channels

Returns all Channels (of a bouquet) of the input plug-in that is active

```
<?xml version="1.0" encoding="utf-8"?>
 <Channels>
  <Channel SourceId="76472624" StreamId="1" UpTimeSec="9.4" OnAir="true"
  Record="0" Publish="0"
  Substreams="2" VideoSubstreams="1" AudioSubstreams="1">
   <SourceName>File</SourceName>
   <StreamName>FIFASPOT_LVB_3gmp4_352x288.mp4</StreamName>
   <EPGName>FIFASPOT_LVB_3gmp4_352x288</EPGName>
   <EPGDescription>C:\Videon\video_samples\MPEG-4 3GP\
   FIFASPOT_LVB_3gmp4_352x288.mp4</EPGDescription>
    <Substream SubstreamId="1" Type="Video" FCC="MP4V" OnAir="true">
     <Description>352 x 288</Description>
     <Profile>Advanced Simple</Profile>
     <Level>4</Level>
    </Substream>
    <Substream SubstreamId="2" Type="Audio" FCC=" AAC" OnAir="true">
     <Description>1 Channels, 32000 Hz, 16-bits</Description>
     <Profile>AAC LC</Profile>
    </Substream>
  </Channel>
 </Channels>
```

Figure A.3. JScript Snippet output for Get All Channels

The example shows the Channel of the plug-in File that is active at the moment. The Channel tag contains additional information for example that the video file consists of two substreams (video and audio).

The response of an active BDA plug-in would show all available Channels of the selected bouquet and additional information (source name, stream name, provider, EPG data, etc.) of each Channel.

A.1.4. Set Channel OnAir flag

Set the OnAir flag of the selected Channel true or false.

Note

Before using the JScript Snippet the SourceId and the StreamId must be known. The necessary information can be found with help of the *Get Active Channels* script. After executing the script the SourceId and the StreamId are displayed as attributes in the Channel tag (see Figure A.4, “SourceId (first attribute) and StreamId (second attribute) in Channel tag ” [14]).

```
<Channel SourceId="151496720" StreamId="2" UpTimeSec="667.7" OnAir="true"
Record="0" Publish="0" Substreams="3" VideoSubstreams="1" AudioSubstreams="2">
```

Figure A.4. SourceId (first attribute) and StreamId (second attribute) in Channel tag

```
/SetActiveChannelOnAir?SourceId=1234567890123456&StreamId=12345678&value=false
```

Figure A.5. JScript Snippet request for Set Channel OnAir flag (original source code with place holders)

The place holders for SourceId (*1234567890123456*) and StreamId (*12345678*) must be replaced with the SourceId and the StreamId that was determinated with the *Get Active Channels* script.

The default value for the OnAir flag is *true* . That means the OnAir state of the Channel is *on* . If the state should be changed to *off* by script, the value part of the code must be set *false* (see Figure A.6, " JScript Snippet request for Set Channel OnAir flag (place holders replaced with the SourceId and StreamId of the selected Channel) " [15]).

```
/SetActiveChannelOnAir?SourceId=151496720&StreamId=2&value=false
```

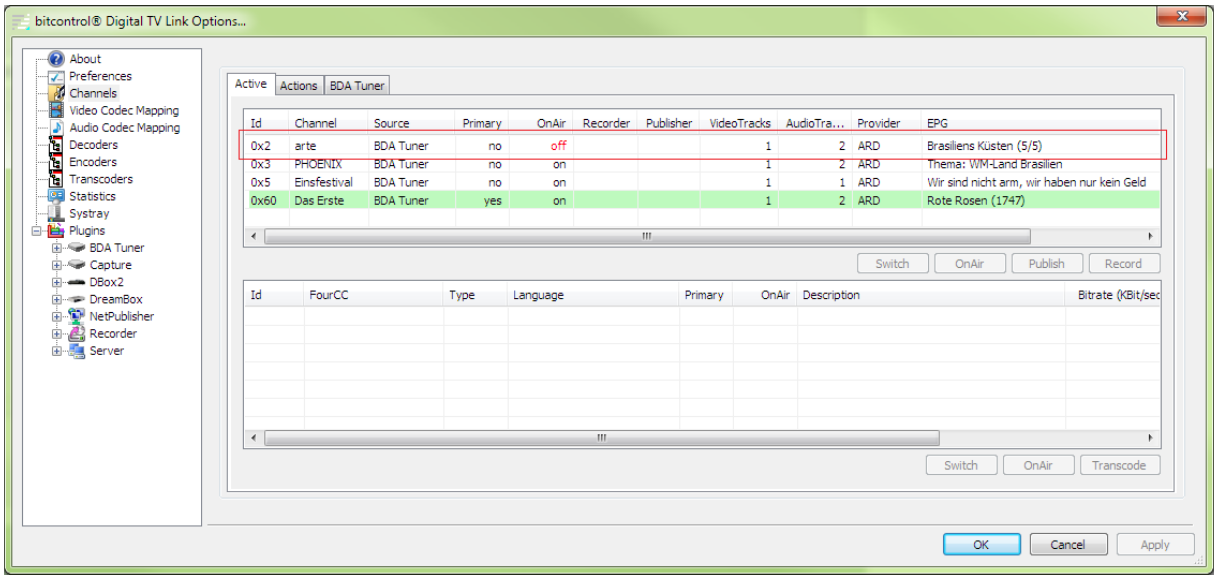
Figure A.6. JScript Snippet request for Set Channel OnAir flag (place holders replaced with the SourceId and StreamId of the selected Channel)

The server response will be *OK* (line 2).

```
<?xml version="1.0" encoding="utf-8"?>
<Result>Ok</Result>
```

Figure A.7. JScript Snippet response for Set Channel OnAir flag

After execution the respective OnAir entry in *bitcontrol® Digital TV Link* Channels window (slider Active) will be set *off* (see Figure A.8, " bitcontrol® Digital TV Link Channels window (slider Active) - OnAir off " [16]).



Id	Channel	Source	Primary	OnAir	Recorder	Publisher	VideoTracks	AudioTra...	Provider	EPG
0x2	arte	BDA Tuner	no	off			1	2	ARD	Brasiliens Küsten (5/5)
0x3	PHOENIX	BDA Tuner	no	on			1	2	ARD	Thema: WM-Land Brasilien
0x5	Einsfestival	BDA Tuner	no	on			1	1	ARD	Wir sind nicht arm, wir haben nur kein Geld
0x60	Das Erste	BDA Tuner	yes	on			1	2	ARD	Rote Rosen (1747)

Id	FourCC	Type	Language	Primary	OnAir	Description	Bitrate (KBit/sec

Figure A.8. bitcontrol® Digital TV Link Channels window (slider Active) - OnAir off

Changing the JScript Snippet code value to *true* set the OnAir table entry *on* .

A.1.5. Set Channel Record flag

Set the Record flag of the selected Channel true or false.

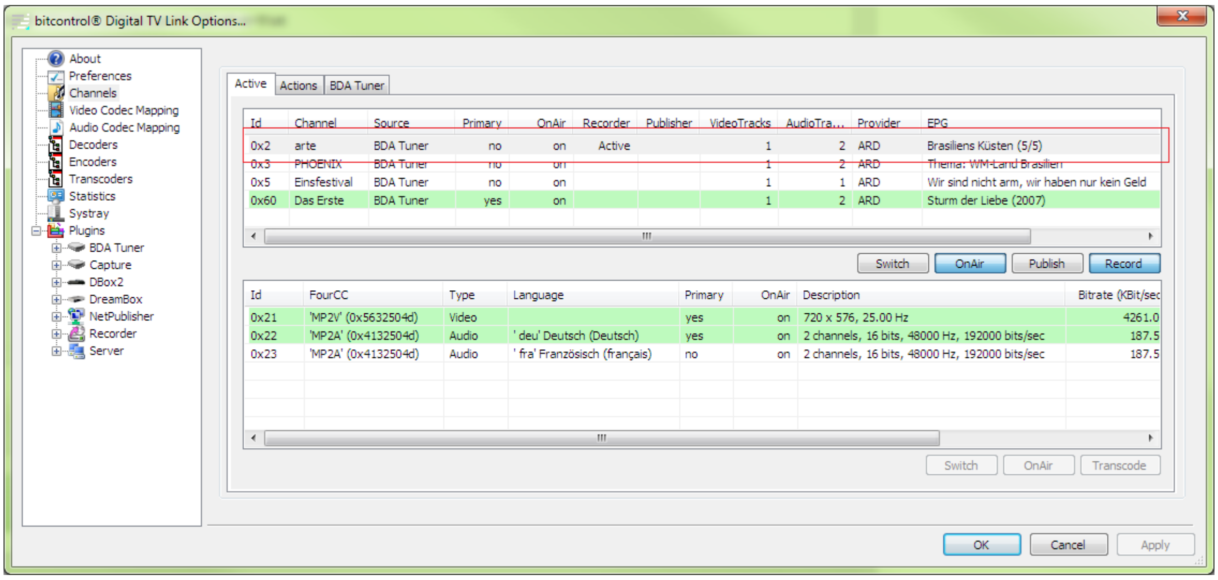
The JScript Snippet *Set Channel Record flag* works in the same way as the *Set Channel OnAir flag* JScript Snippet. The difference is the default value for the flag. In this case it is *false* .

The next code shows the input line.

```
/SetActiveChannelRecord?SourceId=151496720&StreamId=2&value=true
```

Figure A.9. JScript Snippet request for Set Channel Record flag

The next picture shows the result in *bitcontrol® Digital TV Link* Channels window (slider Active).



Id	Channel	Source	Primary	OnAir	Recorder	Publisher	VideoTracks	AudioTra...	Provider	EPG
0x2	arte	BDA Tuner	no	on	Active		1	2	ARD	Brasiliens Küsten (5/5)
0x3	PHOENIX	BDA Tuner	no	on			1	2	ARD	Thema: WM-Land Brasilien
0x5	Einsfestival	BDA Tuner	no	on			1	1	ARD	Wir sind nicht arm, wir haben nur kein Geld
0x60	Das Erste	BDA Tuner	yes	on			1	2	ARD	Sturm der Liebe (2007)

Id	FourCC	Type	Language	Primary	OnAir	Description	Bitrate (KBit/sec
0x21	'MP2V' (0x5632504d)	Video		yes	on	720 x 576, 25.00 Hz	4261.0
0x22	'MP2A' (0x4132504d)	Audio	' deu' Deutsch (Deutsch)	yes	on	2 channels, 16 bits, 48000 Hz, 192000 bits/sec	187.5
0x23	'MP2A' (0x4132504d)	Audio	' fra' Französisch (français)	no	on	2 channels, 16 bits, 48000 Hz, 192000 bits/sec	187.5

Figure A.10. bitcontrol® Digital TV Link Channels window (slider Active) - Recorder Active

A.1.6. Set Channel Publish flag

Set the Publish flag of the selected Channel true or false.

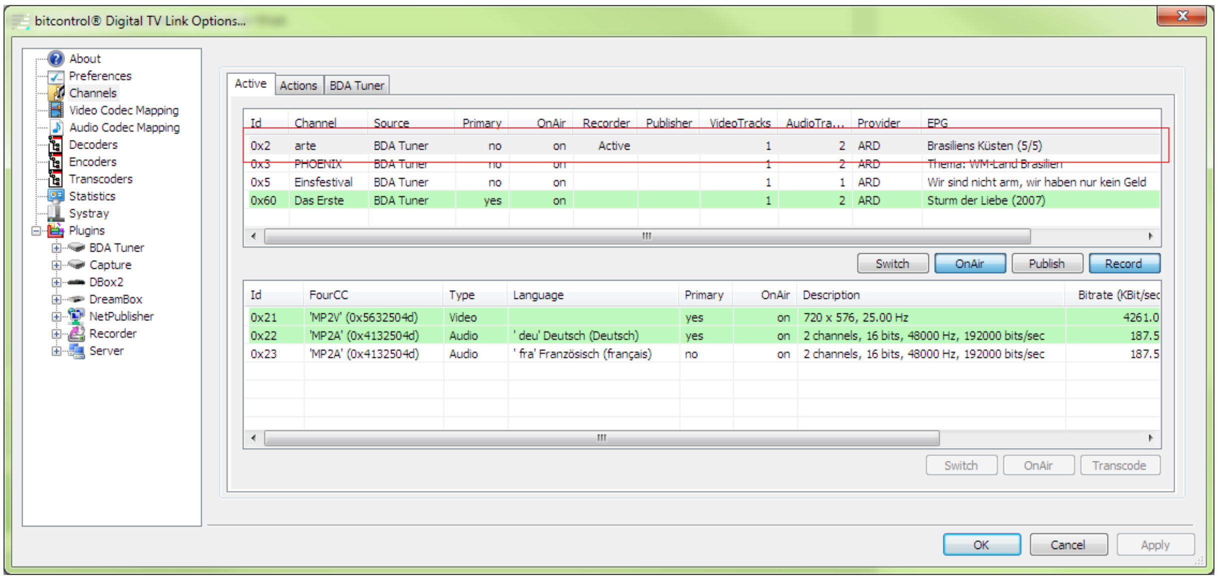
The JScript Snippet *Set Channel Plublish flag* works in the same way as the *Set Channel OnAir flag* JScript Snippet. The difference is the default value for the flag. In this case it is *false* .

The next code shows the input line.

```
/SetActiveChannelPublish?SourceId=151496720&StreamId=2&value=true
```

Figure A.11. JScript Snippet request for Set Channel Plublish flag

The next picture shows the result in *bitcontrol® Digital TV Link* Channels window (slider Active).



Id	Channel	Source	Primary	OnAir	Recorder	Publisher	VideoTracks	AudioTra...	Provider	EPG
0x2	arte	BDA Tuner	no	on	Active		1	2	ARD	Brasiliens Küsten (5/5)
0x3	PHOENIX	BDA Tuner	no	on			1	2	ARD	Thema: WM-Land Brasilien
0x5	Einsfestival	BDA Tuner	no	on			1	1	ARD	Wir sind nicht arm, wir haben nur kein Geld
0x60	Das Erste	BDA Tuner	yes	on			1	2	ARD	Sturm der Liebe (2007)

Id	FourCC	Type	Language	Primary	OnAir	Description	Bitrate (KBit/sec
0x21	'MP2V' (0x5632504d)	Video		yes	on	720 x 576, 25.00 Hz	4261.0
0x22	'MP2A' (0x4132504d)	Audio	' deu' Deutsch (Deutsch)	yes	on	2 channels, 16 bits, 48000 Hz, 192000 bits/sec	187.5
0x23	'MP2A' (0x4132504d)	Audio	' fra' Französisch (français)	no	on	2 channels, 16 bits, 48000 Hz, 192000 bits/sec	187.5

Figure A.12. bitcontrol® Digital TV Link Channels window (slider Active) - Publisher Active

A.2. JScript Snippets for *bitcontrol® Digital TV Link* core functions (POST scripts)

These JScript Snippets use the POST method to read out core function of *bitcontrol® Digital TV Link* . The example codes show the request and the result of core function (POST script) requests.

A.2.1. Get Version

Returns the version of the used *bitcontrol® Digital TV Link*

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum0">
 <![CDATA[
 DTVLink.Version
 ]]>
</Script>
```

Figure A.13. JScript Snippet request for Get version (core function)

```
<?xml version="1.0" encoding="utf-8"?>
 <Result Id="RequestNum0">v3.8.0.0, Release(Unicode)X86,
 assembled on May 27 2014
 </Result>
```

Figure A.14. JScript Snippet server response for Get version (core function)

The result is the same as the output of the *Get version* GET request (see Section A.1, “ JScript Snippets for bitcontrol® Digital TV Link GET requests ”[12]). So the difference is not the result but the way of server request.

A.2.2. Write Log Message

Writes a log message that is displayed in Console output.

```
<Script Language="JScript" GlobalContext="true" Expression="false"
Response="XMLResponse" Id="RequestNum1">
 <![CDATA[
 DTVLink.LogMessage(3, "JScript", "Hello World", 0, 0);
 ]]>
</Script>
```

Figure A.15. JScript Snippet request for Write Log Message (core function)

GlobalContext is true (line 1), so the global variable for the *LogMessage* must be addressed explicit.

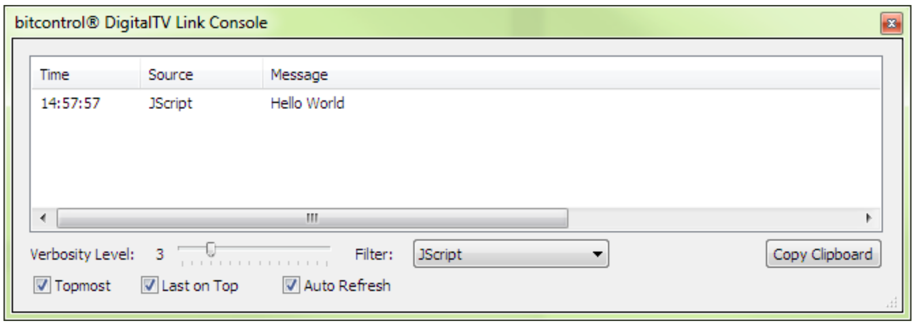
Note

At the moment there is only one global variable available: *DTVLink* .

The values inside the brackets (line 4) mean:

- Verbose level (This verbose level must be set minimum to see the log message in the Console.)
- JScript
- Content of the log message
- Background color
- Text color

The example code contains the string *Hello World* (line 4). This string is displayed as log message in Console (see Figure A.16, “ Log message output in bitcontrol® Digital TV Link Console window”[20]) from verbose level 3, not colored.


bitcontrol® DigitalTV Link Console

```
Time       Source    Message
14:57:57   JScript   Hello World
```

Verbosity Level: 3 Filter: JScript Copy Clipboard
[✓] Topmost [✓] Last on Top [✓] Auto Refresh

Figure A.16. Log message output in bitcontrol® Digital TV Link Console window

A.2.3. Write Log Message (local context)

Writes a log message that is displayed colored in Console output.

```
<Script Language="JScript" GlobalContext="false" Expression="false"
Response="XMLResponse" Id="RequestNum2">
 <![CDATA[
 LogMessage(3, "JScript", "Hello World", 0x0000ff, 0xffff00);
 ]]>
</Script>
```

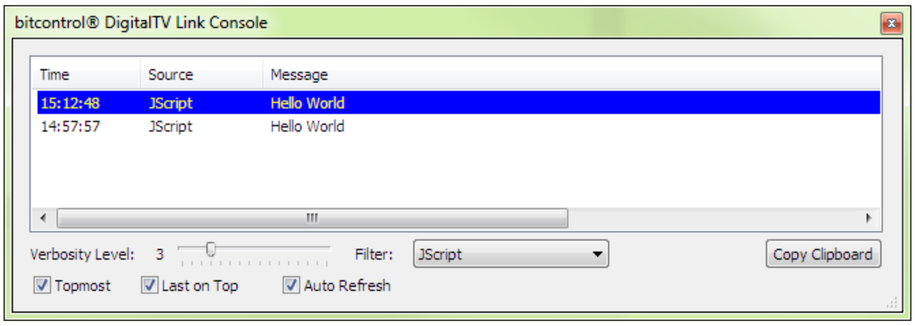
Figure A.17. JScript Snippet request for Write Log Message (local context) (core function)

If *GlobalContext* is false (line 1), the *LogMessage* script runs in *DTVLink* context. It is not necessary to define a global variable.

The *LogMessage* properties in the brackets are the same as the example before.

This example code contains the string *Hello World* (line 4) but also a background color (first hex value) and a text color (second hex value).

Hello World is displayed as log message in Console in yellow on blue (see Figure A.18, “ Colored log message output in bitcontrol® Digital TV Link Console window ” [20]) from verbose level 3.


bitcontrol® DigitalTV Link Console

```
Time       Source    Message
15:12:48   JScript   Hello World
14:57:57   JScript   Hello World
```

Verbosity Level: 3 Filter: JScript Copy Clipboard
[✓] Topmost [✓] Last on Top [✓] Auto Refresh

Figure A.18. Colored log message output in bitcontrol® Digital TV Link Console window

A.2.4. Read Log Messages

Returns *bitcontrol® Digital TV Link* Log Messages

```
<Script Language="JScript" GlobalContext="true" Expression="false"
Response="XMLResponse" Id="RequestNum2">
 <![CDATA[
  var entry = DTVLink.LogLastEntry;
  while (entry != null) {
   var xmlentry = XMLResponse.ownerDocument.createElement('Entry');
   XMLResponse.appendChild(xmlentry);
   xmlentry.setAttribute('Number', entry.Number);
   xmlentry.setAttribute('Level', entry.Level);
   xmlentry.setAttribute('ProcessId', entry.ProcessId);
   xmlentry.setAttribute('ThreadId', entry.ThreadId);
   xmlentry.setAttribute('Date', new Date(entry.Date).toLocaleString());
   xmlentry.setAttribute('Source', entry.Source);
   xmlentry.text = entry.Message;
   entry = DTVLink.FindLogEntryByNumber(entry.Number - 1);
  }
 ]]>
</Script>
```

Figure A.19. JScript Snippet request for Read Log Message (core function)

If the JScript Snippet is used without changing defaults, all *bitcontrol® Digital TV Link* log messages will be returned.

The xmlentry attributes can be used to specify the log message output:

- Number (index number of the log message, last log message has the highest number)
- Level (*bitcontrol® Digital TV Link* verbose level)
- ProcessId (Windows® process Id)
- ThreadId (Windows® thread Id)
- Date (timestamp)
- Source

For the next example the default code was modified with some JavaScript code (line 6) to find out all log messages with verbose level 3.

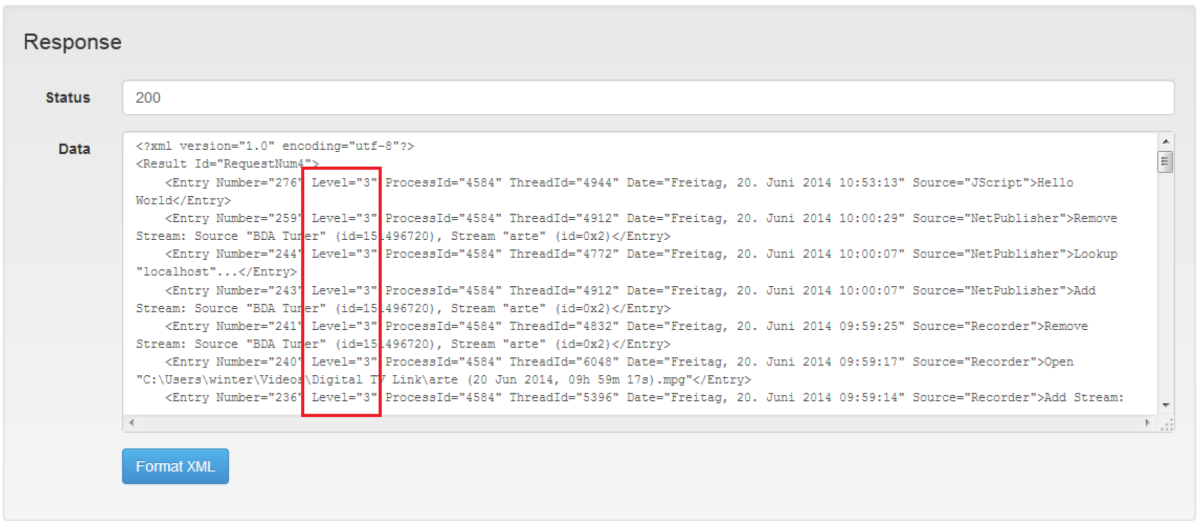
```
<Script Language="JScript" GlobalContext="true" Expression="false"
Response="XMLResponse" Id="RequestNum4">
 <![CDATA[
  var entry = DTVLink.LogLastEntry;
  while (entry != null) {
  if (entry.Level == 3) {
   var xmlentry = XMLResponse.ownerDocument.createElement('Entry');
   XMLResponse.appendChild(xmlentry);
   xmlentry.setAttribute('Number', entry.Number);
   xmlentry.setAttribute('Level', entry.Level);
   xmlentry.setAttribute('ProcessId', entry.ProcessId);
   xmlentry.setAttribute('ThreadId', entry.ThreadId);
   xmlentry.setAttribute('Date', new Date(entry.Date).toLocaleString());
   xmlentry.setAttribute('Source', entry.Source);
   xmlentry.text = entry.Message;
  }
  entry = DTVLink.FindLogEntryByNumber(entry.Number - 1);
  }
 ]]>
</Script>
```

Figure A.20. Example code: All log messages with verbose level 3

The next picture shows the result in the response area (see Figure A.21, “ Selected verbose level ” [22]).

Note

Use the XML button to format the output!



Response

Status: 200

Data:

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum4">
    <Entry Number="276" Level="3" ProcessId="4584" ThreadId="4944" Date="Freitag, 20. Juni 2014 10:53:13" Source="JScript">Hello
World</Entry>
    <Entry Number="259" Level="3" ProcessId="4584" ThreadId="4912" Date="Freitag, 20. Juni 2014 10:00:29" Source="NetPublisher">Remove
Stream: Source "BDA Tuner" (id=151496720), Stream "arte" (id=0x2)</Entry>
    <Entry Number="244" Level="3" ProcessId="4584" ThreadId="4772" Date="Freitag, 20. Juni 2014 10:00:07" Source="NetPublisher">Lookup
"localhost"...</Entry>
    <Entry Number="243" Level="3" ProcessId="4584" ThreadId="4912" Date="Freitag, 20. Juni 2014 10:00:07" Source="NetPublisher">Add
Stream: Source "BDA Tuner" (id=151496720), Stream "arte" (id=0x2)</Entry>
    <Entry Number="241" Level="3" ProcessId="4584" ThreadId="4832" Date="Freitag, 20. Juni 2014 09:59:25" Source="Recorder">Remove
Stream: Source "BDA Tuner" (id=151496720), Stream "arte" (id=0x2)</Entry>
    <Entry Number="240" Level="3" ProcessId="4584" ThreadId="6048" Date="Freitag, 20. Juni 2014 09:59:17" Source="Recorder">Open
"C:\Users\winter\Videos\Digital TV Link\arte (20 Jun 2014, 09h 59m 17s).mpg"</Entry>
    <Entry Number="236" Level="3" ProcessId="4584" ThreadId="5396" Date="Freitag, 20. Juni 2014 09:59:14" Source="Recorder">Add Stream:
```

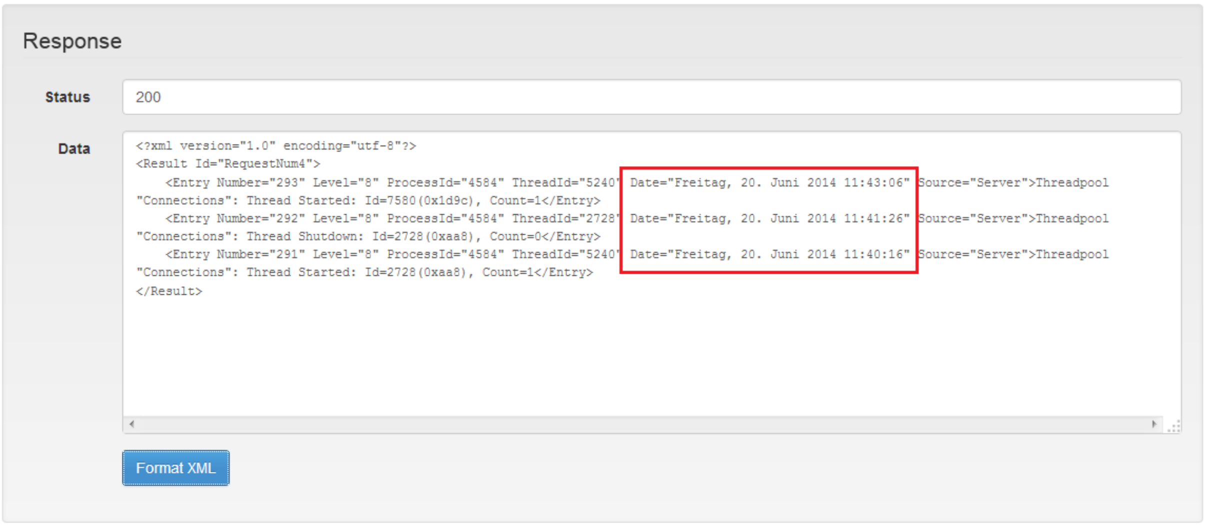
Figure A.21. Selected verbose level

For the next example all log messages with timstamp greater than 11:40 should be issued. For this the default code was modified with JavaScript in line 5 and line 7.

```
<Script Language="JScript" GlobalContext="true" Expression="false"
Response="XMLResponse" Id="RequestNum4">
 <![CDATA[
  var entry = DTVLink.LogLastEntry;
  var now = new Date(2014, 5, 20, 11, 40, 00, 000);
  while (entry != null) {
  if (entry.Date > now) {
   var xmlentry = XMLResponse.ownerDocument.createElement('Entry');
   XMLResponse.appendChild(xmlentry);
   xmlentry.setAttribute('Number', entry.Number);
   xmlentry.setAttribute('Level', entry.Level);
   xmlentry.setAttribute('ProcessId', entry.ProcessId);
   xmlentry.setAttribute('ThreadId', entry.ThreadId);
   xmlentry.setAttribute('Date', new Date(entry.Date).toLocaleString());
   xmlentry.setAttribute('Source', entry.Source);
   xmlentry.text = entry.Message;
  }
  entry = DTVLink.FindLogEntryByNumber(entry.Number - 1);
  }
 ]]>
</Script>
```

Figure A.22. Example code: All log messages with timestamp greater than 11:40

Figure A.23, “ All log messages with timestamp greater than 11:40 [”23] shows the result in the response area.



```
200
```

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum4">
    <Entry Number="293" Level="8" ProcessId="4584" ThreadId="5240" Date="Freitag, 20. Juni 2014 11:43:06" Source="Server">Threadpool
"Connections": Thread Started: Id=7580(0x1d9c), Count=1</Entry>
    <Entry Number="292" Level="8" ProcessId="4584" ThreadId="2728" Date="Freitag, 20. Juni 2014 11:41:26" Source="Server">Threadpool
"Connections": Thread Shutdown: Id=2728(0xaa8), Count=0</Entry>
    <Entry Number="291" Level="8" ProcessId="4584" ThreadId="5240" Date="Freitag, 20. Juni 2014 11:40:16" Source="Server">Threadpool
"Connections": Thread Started: Id=2728(0xaa8), Count=1</Entry>
</Result>
```

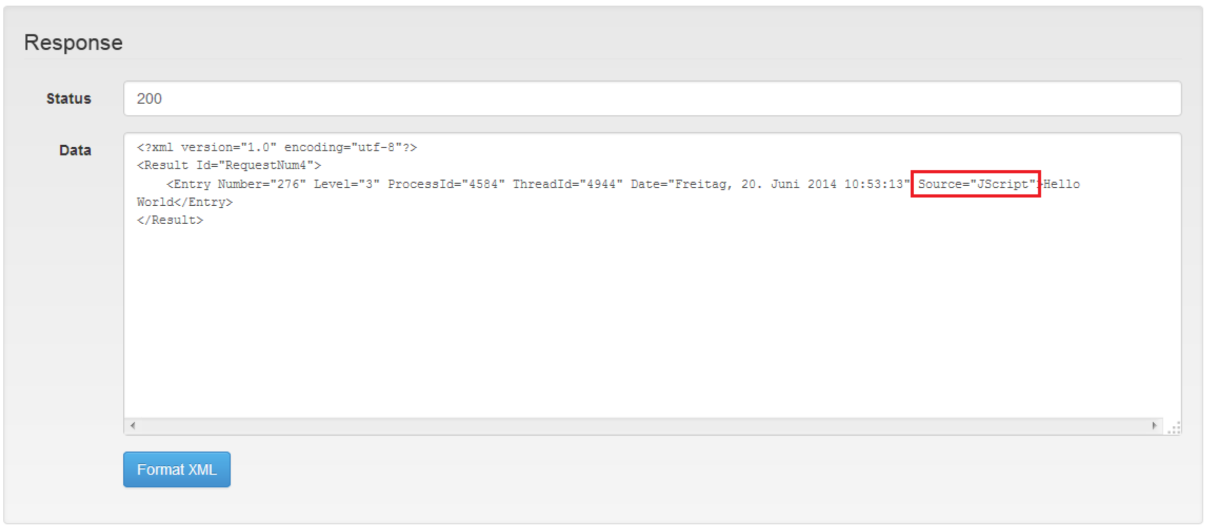
Figure A.23. All log messages with timestamp greater than 11:40

The next code is used to find out all log messages for the source *JScript*. The default code was modified with JavaScript in line 6.

```
<Script Language="JScript" GlobalContext="true" Expression="false"
Response="XMLResponse" Id="RequestNum4">
 <![CDATA[
  var entry = DTVLink.LogLastEntry;
  while (entry != null) {
  if (entry.Source == "JScript") {
   var xmlentry = XMLResponse.ownerDocument.createElement('Entry');
   XMLResponse.appendChild(xmlentry);
   xmlentry.setAttribute('Number', entry.Number);
   xmlentry.setAttribute('Level', entry.Level);
   xmlentry.setAttribute('ProcessId', entry.ProcessId);
   xmlentry.setAttribute('ThreadId', entry.ThreadId);
   xmlentry.setAttribute('Date', new Date(entry.Date).toLocaleString());
   xmlentry.setAttribute('Source', entry.Source);
   xmlentry.text = entry.Message;
  }
  entry = DTVLink.FindLogEntryByNumber(entry.Number - 1);
  }
 ]]>
</Script>
```

Figure A.24. Example code: Find out all log messages for source JScript

Figure A.25, " All log messages from the source JScript " [24] shows the result in the response area.



200

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum4">
    <Entry Number="276" Level="3" ProcessId="4584" ThreadId="4944" Date="Freitag, 20. Juni 2014 10:53:13" Source="JScript">Hello
World</Entry>
</Result>
```

Figure A.25. All log messages from the source JScript

A.2.5. Get InstanceId

Returns the id of the active bitcontrol® Digital TV Link instance.

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum0">
 <![CDATA[
  DTVLink.InstanceId;
 ]]>
</Script>
```

Figure A.26. JScript Snippet request for Get Instance Id

The result contains the *Instance ID* that is the Windows® process id (pid) and the Windows® thread id (tid). The id will be built when bitcontrol® Digital TV Link starts. It is unique and identifies the current instance.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum0">
DTVLink: pid 2300, tid 4216
</Result>
```

Figure A.27. JScript Snippet server response for Get Instance Id (core function)

Note

If there are several bitcontrol® Digital TV Link instances active, the result of the request will be a number of different *Instance Id's*. Then it isn't possible to identify the corresponding id for a bitcontrol® Digital TV Link instance.

A.2.6. Get Data Folder

Returns the path of the used *bitcontrol® Digital TV Link*

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum1">
 <![CDATA[
 DTVLink.DataFolder
]]>
</Script>
```

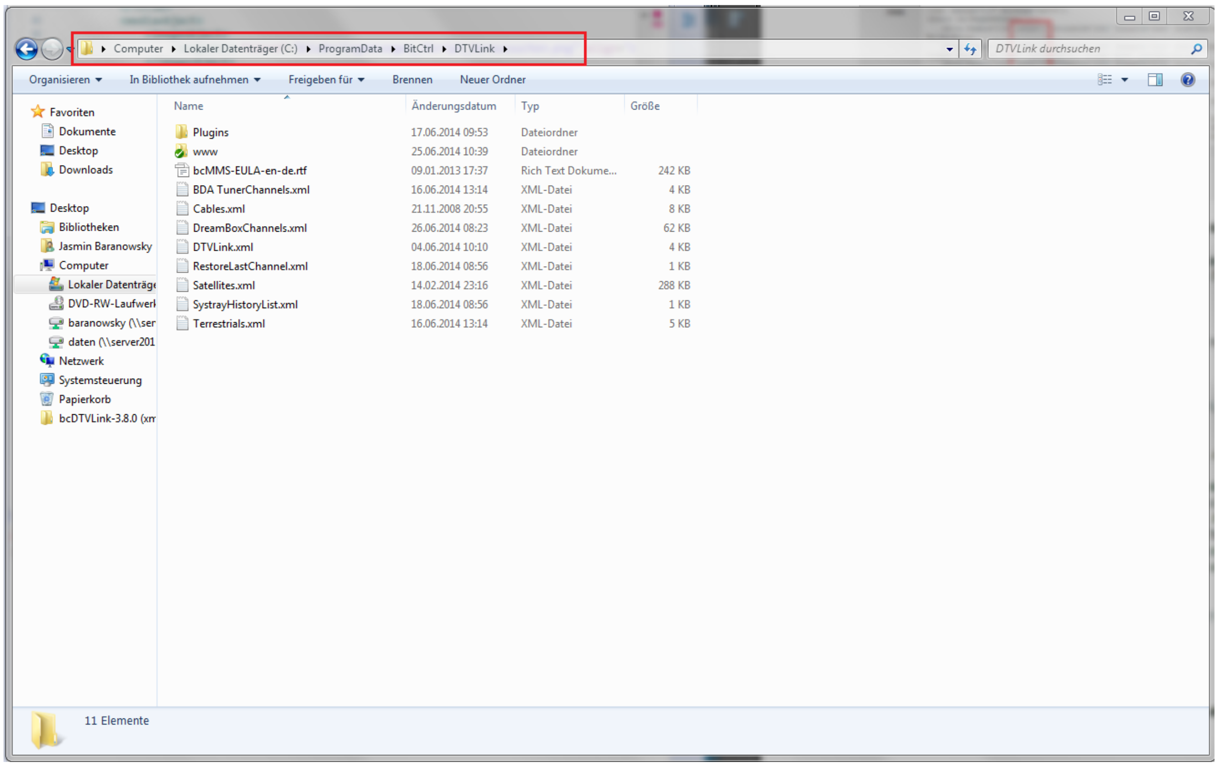
Figure A.28. JScript Snippet request for Get DataFolder

The result is the path url of the folder that contains the bitcontrol® Digital TV Link program data (all plug-in configurations and the website).

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum1">C:\ProgramData\BitCtrl\DTVLink\</Result>
```

Figure A.29. JScript Snippet server response for Get DataFolder (core function)

The path url in *Windows Explorer* opens the bitcontrol® Digital TV Link folder (see Figure A.30, “ The disk space of the bitcontrol® Digital TV Link ” [26]).



Name	Änderungsdatum	Typ	Größe
Plugins	17.06.2014 09:53	Dateiordner	
www	25.06.2014 10:39	Dateiordner	
bcMMS-EULA-en-de.rtf	09.01.2013 17:37	Rich Text Dokume...	242 KB
BDA TunerChannels.xml	16.06.2014 13:14	XML-Datei	4 KB
Cables.xml	21.11.2008 20:55	XML-Datei	8 KB
DreamBoxChannels.xml	26.06.2014 08:23	XML-Datei	62 KB
DTVLink.xml	04.06.2014 10:10	XML-Datei	4 KB
RestoreLastChannel.xml	18.06.2014 08:56	XML-Datei	1 KB
Satellites.xml	14.02.2014 23:16	XML-Datei	288 KB
SystrayHistoryList.xml	18.06.2014 08:56	XML-Datei	1 KB
Terrestrials.xml	16.06.2014 13:14	XML-Datei	5 KB

Figure A.30. The disk space of the bitcontrol® Digital TV Link

A.2.7. Get LicenseRegnum

Returns the license number of the used *bitcontrol*® Digital TV Link

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum2">
 <![CDATA[
 DTVLink.LicenseRegnum
 ]]>
</Script>
```

Figure A.31. JScript Snippet request for Get LicenseRegnum (core function)

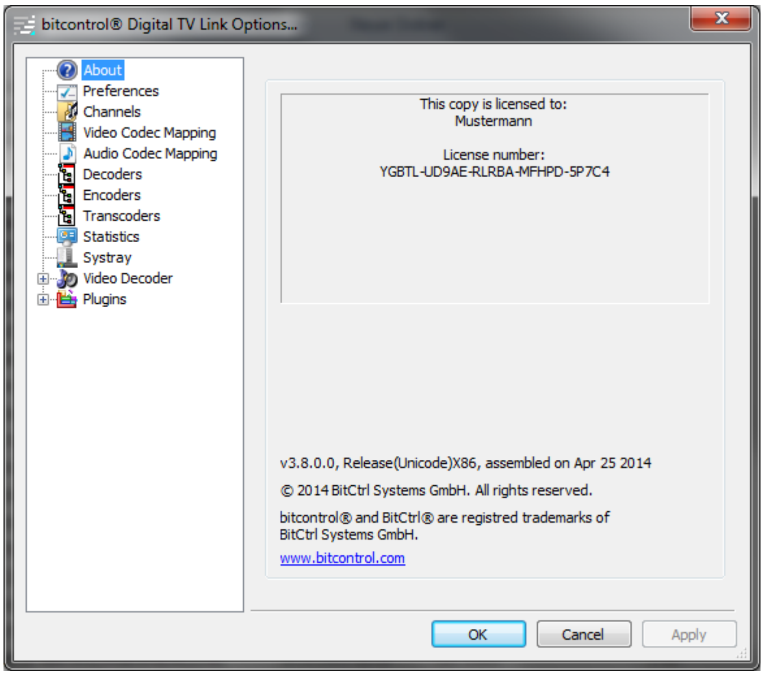
The licence number of *bitcontrol® Digital TV Link* is displayed as result in the response area.

Note

To see a license number you must have a license key.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum2">YGBTL-UD9AE-RLRBA-MFHPD-5P7C4</Result>
```

Figure A.32. JScript Snippet server response for Get LicenseRegnum (core function)



This copy is licensed to:
Mustermann

License number:
YGBTL-UD9AE-RLRBA-MFHPD-5P7C4

v3.8.0.0, Release(Unicode)X86, assembled on Apr 25 2014

© 2014 BitCtrl Systems GmbH. All rights reserved.

bitcontrol® and BitCtrl® are registred trademarks of
BitCtrl Systems GmbH.

www.bitcontrol.com

Figure A.33. License key in About area via bitcontrol® Digital TV Link Options menu

A.2.8. Get LicenseName

Returns the name of the *bitcontrol® Digital TV Link* licensee

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum0">
 <![CDATA[
 DTVLink.LicenseName
 ]]>
</Script>
```

Figure A.34. JScript Snippet request for Get LicenseRegnum (core function)

The result is the name of the *bitcontrol® Digital TV Link* licensee. It's the same name as in the *About* area that can be opened via *bitcontrol® Digital TV Link Options* menu (see Figure A.33, “ License key in About area via bitcontrol® Digital TV Link Options menu ” [27]).

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum3">Mustermann</Result>
```

Figure A.35. JScript server for Get LicenseName (core function)

A.2.9. Get URL

Returns the URL of the *bitcontrol® Digital TV Link* server

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum0">
 <![CDATA[
 DTVLink.URL
 ]]>
</Script>
```

Figure A.36. JScript Snippet request for Get URL (core function)

The result server URL has no port number. The default port is 80. The port can be changed by editing the *Server.xml* file.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum0">dtvl://localhost</Result>
```

Figure A.37. JScript server response for Get URL (core function)

A.2.10. Get EnablePlugins / Get AutoloadPlugins /Get AutostartPlugins

These JScript snippets are used to get information about the plug-in states of a *bitcontrol® Digital TV Link* instance that are configured by a third party application (e.g. bitcontrol® Video Streaming Server).

Note

The use of the JScript snippets only makes sense when starting *bitcontrol® Digital TV Link*.

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum1">
 <![CDATA[
 DTVLink.EnablePlugins
 ]]>
</Script>
```

Figure A.38. Example JScript Snippet request for Get Enable Plugins

The response is a boolean value that is valid for all *bitcontrol® Digital TV Link* plug-ins of the instance.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum1">true</Result>
```

Figure A.39. Example JScript Snippet server response for Get Enable Plugins (core function)

JScript Snippet	Response
Get EnablePlugins	If the response is *true*, all plug-ins that are marked as *enabled* will be immediately available but must be probably started by hand.
Get AutoloadPlugins	If the response is *true*, all plug-ins that are marked as *Manual* or *Auto* will loaded.
Get AutostartPlugins	If the response is *true*, all plug-ins that are marked as *Auto* will started automatically.

A.2.11. Get RestoreLastChannel / Get AutoJitter / Get PreferAC3 / Get SetDefaultOnAir / Get ShowConsole

Each of these JScript snippets return the state of a *bitcontrol® Digital TV Link* preference option.

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum14">
 <![CDATA[
 DTVLink.RestoreLastChannel
 ]]>
</Script>
```

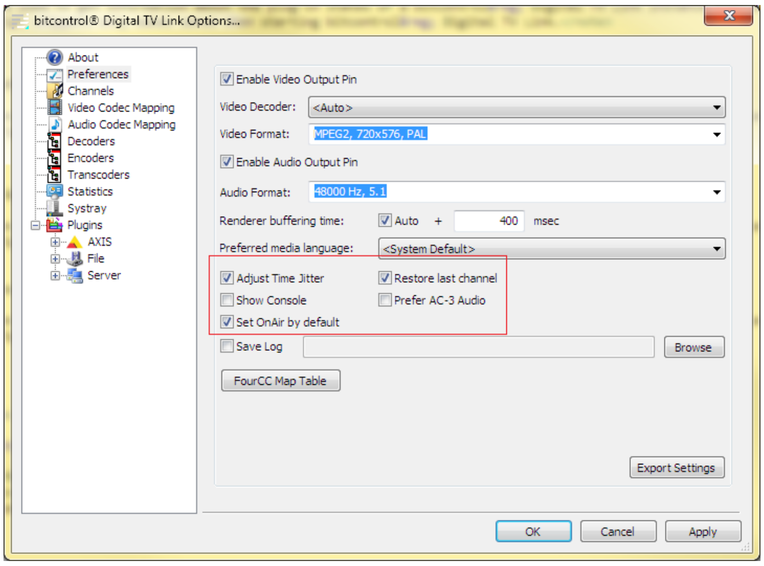
Figure A.40. Example JScript Snippet request for Get RestoreLastChannel (core function)

The response is a boolean value that means the option is enabled (true) or disabled (false).

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum14">true</Result>
```

Figure A.41. Example JScript Snippet server response for Get RestoreLastChannel (core function)

The next picture shows preferences area of the *bitcontrol® Digital TV Link Options* dialogues. Red marked area (see Figure A.42, “ Functions in bitcontrol® Digital TV Link Options preferences area that are read out by the JScript Snippits ”[31]) contains the functions that are read out by the JScript Snippits.


bitcontrol® Digital TV Link Options...

About
Preferences
Channels
Video Codec Mapping
Audio Codec Mapping
Decoders
Encoders
Transcoders
Statistics
Systray
Plugins
AXIS
File
Server

[✓] Enable Video Output Pin
Video Decoder: <Auto>
Video Format: MPEG2, 720x576, PAL
[✓] Enable Audio Output Pin
Audio Format: 48000 Hz, 5.1
Renderer buffering time: [✓] Auto + 400 msec
Preferred media language: <System Default>
[✓] Adjust Time Jitter
[✓] Restore last channel
[ ] Show Console
[ ] Prefer AC-3 Audio
[✓] Set OnAir by default
[ ] Save Log

Figure A.42. Functions in bitcontrol® Digital TV Link Options preferences area that are read out by the JScript Snippits

A.2.12. Get EnableVideoOutPin / Get EnableAudioOutPin

Returns the state of the *Video Output Pin* / *Audio Output Pin* option

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum15">
 <![CDATA[
 DTVLink.EnableVideoOutPin
 ]]>
</Script>
```

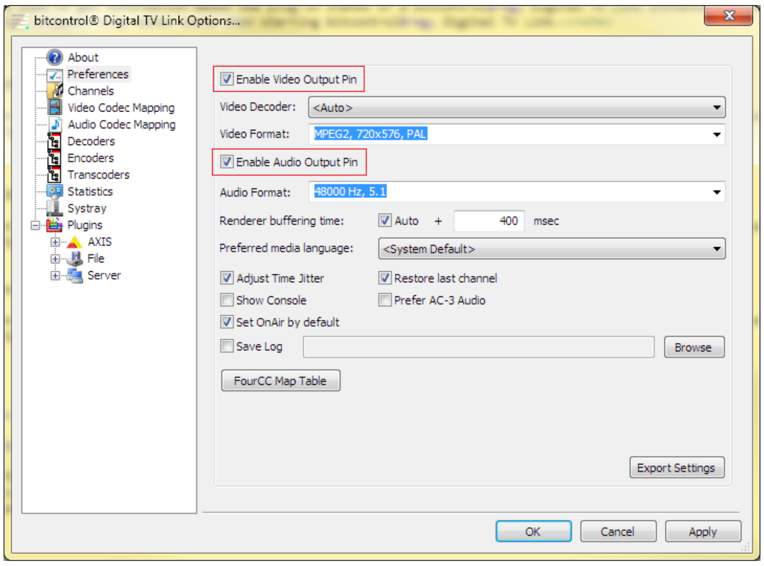
Figure A.43. Example JScript Snippet request for Get EnableVideoOutPin (core function)

The response is a boolean value that means the option is enabled (true) or disabled (false).

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum15">true</Result>
```

Figure A.44. Example JScript Snippet server response for Get EnableVideoOutPin (core function)

The next picture shows preferences area of the *bitcontrol® Digital TV Link Options* dialogues. Red marked (see Figure A.45, “ Video Output Pin / Audio Output Pin options in bitcontrol® Digital TV Link Options preferences dialogue ” [32]) are the functions that are read out by the JScript Snippits.


bitcontrol® Digital TV Link Options...

[✓] Enable Video Output Pin
Video Decoder: <Auto>
Video Format: MPEG2, 720x576, PAL
[✓] Enable Audio Output Pin
Audio Format: 48000 Hz, 5.1
Renderer buffering time: [✓] Auto + 400 msec
Preferred media language: <System Default>
[✓] Adjust Time Jitter
[✓] Restore last channel
[ ] Show Console
[ ] Prefer AC-3 Audio
[✓] Set OnAir by default
[ ] Save Log

Figure A.45. Video Output Pin / Audio Output Pin options in bitcontrol® Digital TV Link Options preferences dialogue

A.2.13. Get VideoDecId / Get AudioDecId

Returns the Id and the name of the currently used *Video Decoder* / *Audio Decoder*

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum1">
 <![CDATA[
 DTVLink.VideoDecoderId
 ]]>
</Script>
```

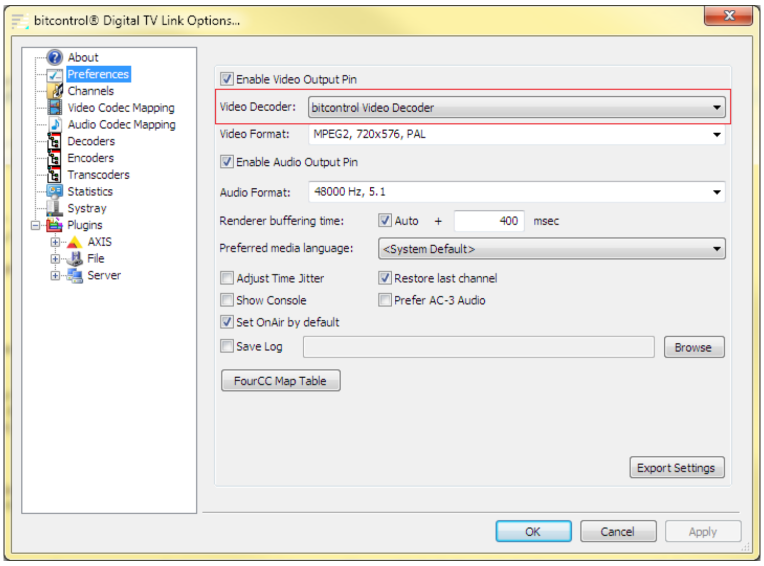
Figure A.46. Example JScript Snippet request for Get VideoDecoderId (core function)

The response is the registry number and the name of the currently used video decoder / audio decoder.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum1">
{BECD3686-53B3-4C9E-935F-A69E06BFB44A}/bitcontrol Video Decoder
</Result>
```

Figure A.47. Example JScript Snippet server response for Get VideoDeocederId (core function)

The next picture shows preferences area of the *bitcontrol® Digital TV Link Options* dialogues. Red marked (see Figure A.48, “ Video Decoder field in bitcontrol® Digital TV Link Options preferences dialogue ” [33]) is the field that is read out by the JScript Snippits.



☑ Enable Video Output Pin
Video Decoder: bitcontrol Video Decoder
Video Format: MPEG2, 720x576, PAL
☑ Enable Audio Output Pin
Audio Format: 48000 Hz, 5.1
Renderer buffering time: ☑ Auto + 400 msec
Preferred media language: <System Default>
☐ Adjust Time Jitter ☑ Restore last channel
☐ Show Console ☐ Prefer AC-3 Audio
☑ Set OnAir by default
☐ Save Log

Figure A.48. Video Decoder field in bitcontrol® Digital TV Link Options preferences dialogue

A.2.14. Get VideoFormatDescription / Get AudioFormatDescription

Returns some format values of the currently used *Video Decoder* / *Audio Decoder*

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum2">
 <![CDATA[
 DTVLink.VideoFormatDescription
 ]]>
</Script>
```

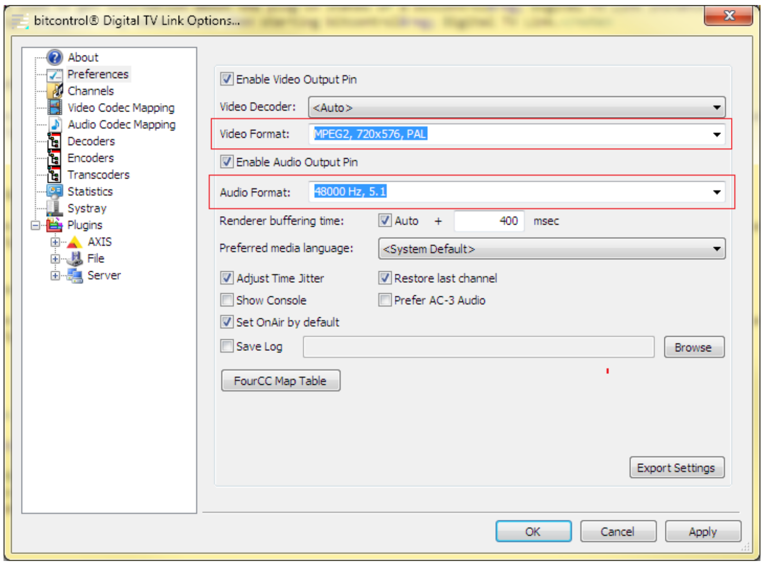
Figure A.49. Example JScript Snippet request for Get VideoFormatDescription (core function)

The response contains the format, the resolution and the transmission method of the currently used video decoder / audio decoder.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum2">MPEG2, 720x576, PAL</Result>
```

Figure A.50. Example JScript Snippet server response for Get VideoFormatDescription (core function)

The next picture shows preferences area of the *bitcontrol® Digital TV Link Options* dialogues. Red marked (see Figure A.51, “ Video Decoder field in bitcontrol® Digital TV Link Options preferences dialogue ” [34]) are the fields that are read out by the JScript Snippits.


bitcontrol® Digital TV Link Options...

- About
- Preferences
- Channels
- Video Codec Mapping
- Audio Codec Mapping
- Decoders
- Encoders
- Transcoders
- Statistics
- Systray
- Plugins
  - AXIS
  - File
  - Server

[✓] Enable Video Output Pin
Video Decoder: <Auto>
Video Format: MPEG2, 720x576, PAL
[✓] Enable Audio Output Pin
Audio Format: 48000 Hz, 5.1
Renderer buffering time: [✓] Auto + 400 msec
Preferred media language: <System Default>
[✓] Adjust Time Jitter [✓] Restore last channel
[ ] Show Console [ ] Prefer AC-3 Audio
[✓] Set OnAir by default
[ ] Save Log

Figure A.51. Video Decoder field in bitcontrol® Digital TV Link Options preferences dialogue

A.2.15. Get BufferingTime

Returns the *Renderer buffering time*

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum3">
 <![CDATA[
 DTVLink.BufferingTime
 ]]>
</Script>
```

Figure A.52. Example JScript Snippet request for Get BufferingTime (core function)

The response is the buffering time value in milliseconds.

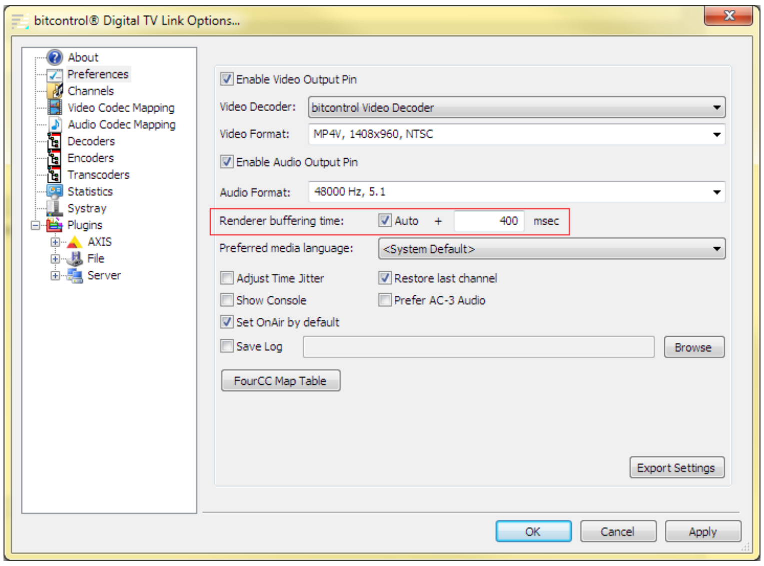
Note

If the response value is negative, the *Auto* option is not active.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum3">400</Result>
```

Figure A.53. Example JScript Snippet server response for Get BufferingTime (core function)

The next picture shows preferences area of the *bitcontrol® Digital TV Link Options* dialogues. Red marked (see ???) are the fields that are read out by the JScript Snippit.



[✓] Enable Video Output Pin
Video Decoder: bitcontrol Video Decoder
Video Format: MP4V, 1408x960, NTSC
[✓] Enable Audio Output Pin
Audio Format: 48000 Hz, 5.1
Renderer buffering time: [✓] Auto + 400 msec
Preferred media language: <System Default>
[ ] Adjust Time Jitter
[✓] Restore last channel
[ ] Show Console
[ ] Prefer AC-3 Audio
[✓] Set OnAir by default
[ ] Save Log

Figure A.54. Renderer buffering time option and selection field in bitcontrol® Digital TV Link Options preferences dialogue

A.2.16. Get FOURCCMap

Returns the contents of the FOURCC mapping table

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum4">
 <![CDATA[
 DTVLink.FOURCCMap
 ]]>
</Script>
```

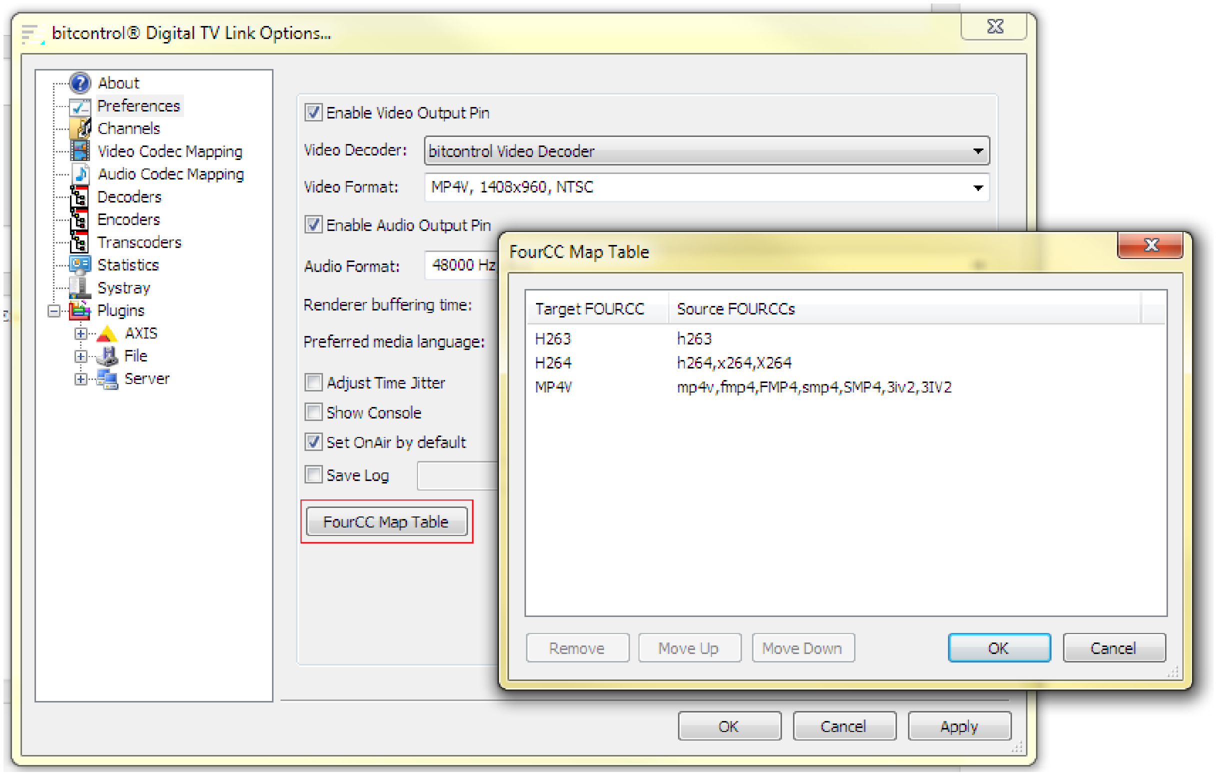
Figure A.55. Example JScript Snippet request for Get FOURCCMap (core function)

The response contains all entries of the FOURCC mapping table.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum4">h263=H263,h264=H264,x264=H264,X264=H264,
mp4v=MP4V,fmp4=MP4V,FMP4=MP4V,smp4=MP4V,SMP4=MP4V,3iv2=MP4V,3IV2=MP4V</Result>
```

Figure A.56. Example JScript Snippet server response for Get FOURCCMap (core function)

The next picture shows preferences area of the *bitcontrol® Digital TV Link Options* dialogues. Red marked (see Figure A.57, " FOURCC Map Table button in bitcontrol® Digital TV Link Options preferences dialogue and FOURCC map window "[36]) is the button to open the FOURCC table window that is read out by the JScript Snippit.



☑ Enable Video Output Pin
Video Decoder: bitcontrol Video Decoder
Video Format: MP4V, 1408x960, NTSC
☑ Enable Audio Output Pin
Audio Format: 48000 Hz
Renderer buffering time:
Preferred media language:
☐ Adjust Time Jitter
☐ Show Console
☑ Set OnAir by default
☐ Save Log

FourCC Map Table

Target FOURCC	Source FOURCCs
H263	h263
H264	h264,x264,X264
MP4V	mp4v,fmp4,FMP4,smp4,SMP4,3iv2,3IV2

Figure A.57. FOURCC Map Table button in bitcontrol® Digital TV Link Options preferences dialogue and FOURCC map window

A.2.17. Get MediaLanguageLANGID

Returns the Id of the currently set *Preffered media language*

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum5">
 <![CDATA[
 DTVLink.MediaLanguageLANGID
 ]]>
</Script>
```

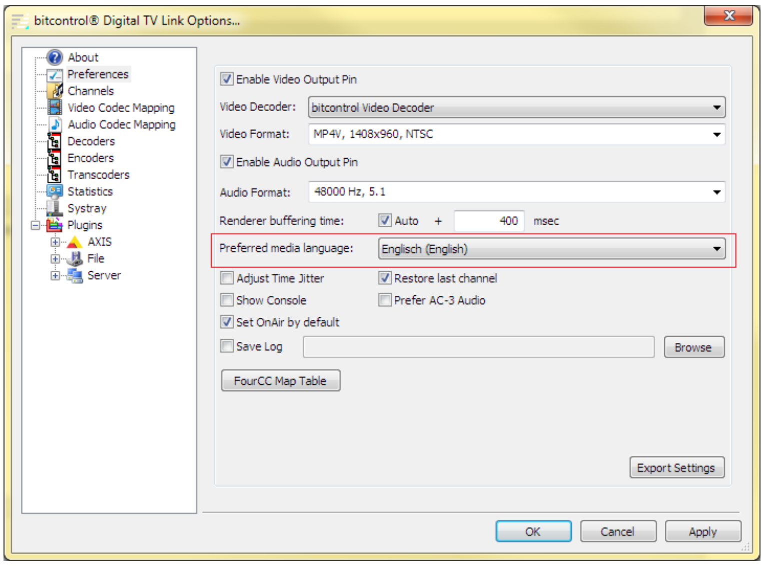
Figure A.58. Example JScript Snippet request for MediaLanguageLANGID (core function)

The response is the Windows system number that represents the id of currently set *Preffered media language*. In the example case it is 1033 for English.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum5">1033</Result>
```

Figure A.59. Example JScript Snippet server response for Get MediaLanguageLANGID (core function)

The next picture shows preferences area of the *bitcontrol® Digital TV Link Options* dialogues. Red marked (see Figure A.60, “ Preffered media language field in bitcontrol® Digital TV Link Options preferences dialogue ” [37]) is the fields that are read out by the JScript Snippit.


bitcontrol® Digital TV Link Options...

About
Preferences
Channels
Video Codec Mapping
Audio Codec Mapping
Decoders
Encoders
Transcoders
Statistics
Systray
Plugins
AXIS
File
Server

[✓] Enable Video Output Pin
Video Decoder: bitcontrol Video Decoder
Video Format: MP4V, 1408x960, NTSC
[✓] Enable Audio Output Pin
Audio Format: 48000 Hz, 5.1
Renderer buffering time: [✓] Auto + 400 msec
Preferred media language: Englisch (English)
[ ] Adjust Time Jitter
[✓] Restore last channel
[ ] Show Console
[ ] Prefer AC-3 Audio
[✓] Set OnAir by default
[ ] Save Log
Browse
FourCC Map Table
Export Settings
OK
Cancel
Apply

Figure A.60. Preffered media language field in bitcontrol® Digital TV Link Options preferences dialogue

A.2.18. Get VideoDecoderList / Get AudioDecoderList

Returns a list of all available *Video Decoder* / *Audio Decoder*

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum2">
 <![CDATA[
 DTVLink.VideoDecodersList
 ]]>
</Script>
```

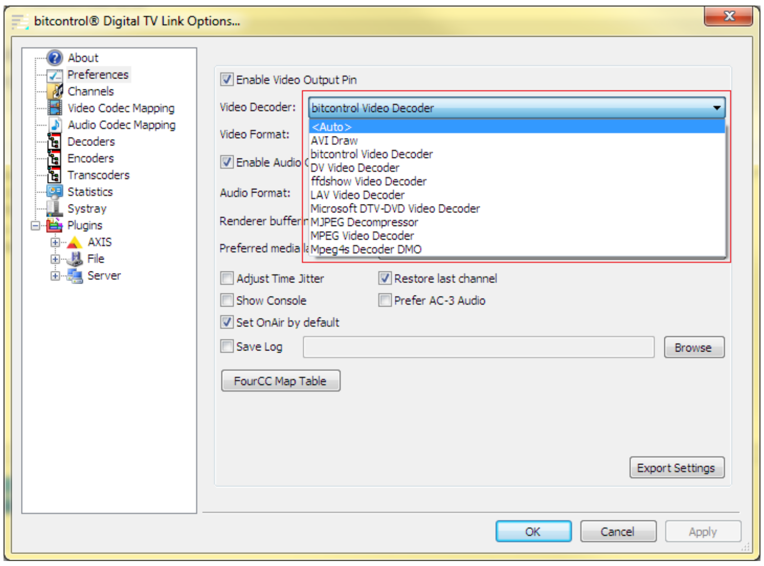
Figure A.61. Example JScript Snippet request for Get VideoDecoderList (core function)

The response lists the registry number and the corresponding name of all available video decoder / audio decoder.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum2">
{04FE9017-F873-410E-871E-AB91661A4EF7}/ffdshow Video Decoder
{FEB50740-7BEF-11CE-9BD9-0000E202599C}/MPEG Video Decoder
{EE30215D-164F-4A92-A4EB-9D4C13390F9F}/LAV Video Decoder
{4A69B442-28BE-4991-969C-B500ADF5D8A8}/Mpeg4s Decoder DMO
{4A69B442-28BE-4991-969C-B500ADF5D8A8}/WMVideo Decoder DMO
{4A69B442-28BE-4991-969C-B500ADF5D8A8}/Mpeg43 Decoder DMO
{4A69B442-28BE-4991-969C-B500ADF5D8A8}/Mpeg4 Decoder DMO
{B1B77C00-C3E4-11CF-AF79-00AA00B67A42}/DV Video Decoder
{A888DF60-1E90-11CF-AC98-00AA004C0FA9}/AVI Draw
{301056D0-6DFF-11D2-9EEB-006008039E37}/MJPEG Decompressor
{BECD3686-53B3-4C9E-935F-A69E06BFB44A}/bitcontrol Video Decoder
{212690FB-83E5-4526-8FD7-74478B7939CD}/Microsoft DTV-DVD Video Decoder
</Result>
```

Figure A.62. Example JScript Snippet server response for Get VideoDecoderList (core function)

The next pictures show preferences area of the *bitcontrol® Digital TV Link Options* dialogues. Red marked (see Figure A.63, “ Video decoder list in bitcontrol® Digital TV Link Options preferences dialogue ” [39] and ???) is the decoder list that are read out by the JScript Snippit.



[✓] Enable Video Output Pin

Video Decoder: bitcontrol Video Decoder

- <Auto>
- AVI Draw
- bitcontrol Video Decoder
- DV Video Decoder
- ffdshow Video Decoder
- LAV Video Decoder
- Microsoft DTV-DVD Video Decoder
- MJPEG Decompressor
- MPEG Video Decoder
- Mpeg4s Decoder DMO

Video Format:

[✓] Enable Audio

Audio Format:

Renderer buffer

Preferred media

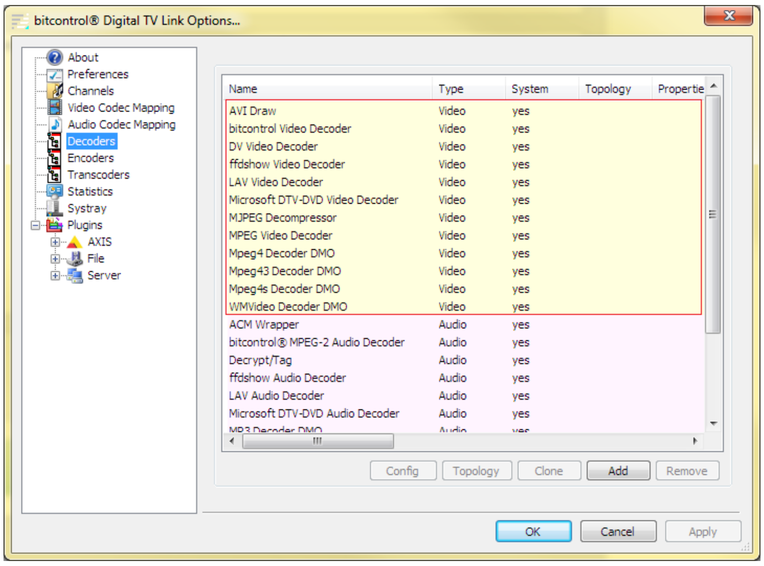
[ ] Adjust Time Jitter [✓] Restore last channel

[ ] Show Console [ ] Prefer AC-3 Audio

[✓] Set OnAir by default

[ ] Save Log

Figure A.63. Video decoder list in bitcontrol® Digital TV Link Options preferences dialogue



Name	Type	System	Topology	Propertie
AVI Draw	Video	yes		
bitcontrol Video Decoder	Video	yes		
DV Video Decoder	Video	yes		
ffdshow Video Decoder	Video	yes		
LAV Video Decoder	Video	yes		
Microsoft DTV-DVD Video Decoder	Video	yes		
MJPEG Decompressor	Video	yes		
MPEG Video Decoder	Video	yes		
Mpeg4 Decoder DMO	Video	yes		
Mpeg43 Decoder DMO	Video	yes		
Mpeg4s Decoder DMO	Video	yes		
WMVideo Decoder DMO	Video	yes		
ACM Wrapper	Audio	yes		
bitcontrol® MPEG-2 Audio Decoder	Audio	yes		
Decrypt/Tag	Audio	yes		
ffdshow Audio Decoder	Audio	yes		
LAV Audio Decoder	Audio	yes		
Microsoft DTV-DVD Audio Decoder	Audio	yes		
MP3 Decoder DMO	Audio	yes		

Figure A.64. Video decoder list in bitcontrol® Digital TV Link Options decoder dialogue

A.2.19. Get Storage

If *Get Storage* is used with default settings, the snippit response will be the same as *Get StorageFilename*.

If *Storage* is defined as object, the snippit return will be the object content as string.

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum0">
 <![CDATA[
 DTVLink.Storage
 ]]>
</Script>
```

Figure A.65. Example JScript Snippet request for Get Storage (core function)

A.2.20. Get StorageFilename

Returns the name of the *Storage* file.

If the default settings are used, the name of the *Storage* file will be *DTVLink.xml* (*bitcontrol® Digital TV Link* configuration file).

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum0">
 <![CDATA[
 DTVLink.StorageFilename
 ]]>
</Script>
```

Figure A.66. Example JScript Snippet request for Get StorageFilename (core function)

The response is a path that ends with the name of the *Storage* file *DTVLink.xml*.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum0">C:\ProgramData\BitCtrl\DTVLink\DTVLink.xml</Result>
```

Figure A.67. Example JScript Snippet server response for Get Storage (core function)

A.2.21. Get StorageXMLDocument / Get StorageXMLElement / Get StorageXMLElementProperties

Returns the content of the *bitcontrol® Digital TV Link* configuration file (*DTVLink.xml*)

- *Get StorageXMLDocument*: Returns the whole content of the xml document
- *Get StorageXMLElement*: Returns only the content of the *DTVLink tag*

- *Get StorageXMLElementProperties*: Returns the properties of the *DTVLink tag*

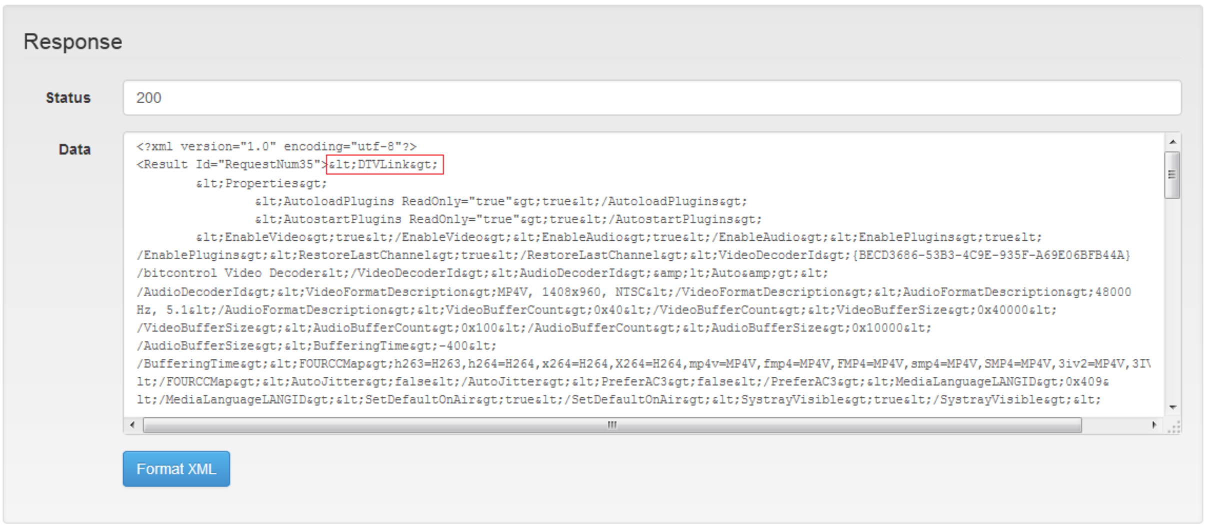
```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum0">
 <![CDATA[
 DTVLink.StorageXMLElement.xml
 ]]>
</Script>
```

Figure A.68. Example JScript Snippet request for Get StorageXMLElement (core function)

Note

The extention *.xml* behind *StorageXMLElement* means that the output will be a xml file. Without that extention, there will be no result.

The next picture shows a part of the JScript Snippit server response. Red marked (see Figure A.69, " Server response for the example. Red marked is the DTVLink tag. " [41]) is the *DTVLink tag*.



```
200
```

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum35">&lt;DTVLink&gt;
        &lt;Properties&gt;
                &lt;AutoloadPlugins ReadOnly="true"&gt;true&lt;/AutoloadPlugins&gt;
                &lt;AutostartPlugins ReadOnly="true"&gt;true&lt;/AutostartPlugins&gt;
        &lt;EnableVideo&gt;true&lt;/EnableVideo&gt;&lt;EnableAudio&gt;true&lt;/EnableAudio&gt;&lt;EnablePlugins&gt;true&lt;
/EnablePlugins&gt;&lt;RestoreLastChannel&gt;true&lt;/RestoreLastChannel&gt;&lt;VideoDecoderId&gt;{BECD3686-53B3-4C9E-935F-A69E06BFB44A}
/bitcontrol Video Decoder&lt;/VideoDecoderId&gt;&lt;AudioDecoderId&gt;&amp;lt;Auto&amp;gt;&lt;
/AudioDecoderId&gt;&lt;VideoFormatDescription&gt;MP4V, 1408x960, NTSC&lt;/VideoFormatDescription&gt;&lt;AudioFormatDescription&gt;48000
Hz, 5.1&lt;/AudioFormatDescription&gt;&lt;VideoBufferCount&gt;0x40&lt;/VideoBufferCount&gt;&lt;VideoBufferSize&gt;0x40000&lt;
/VideoBufferSize&gt;&lt;AudioBufferCount&gt;0x100&lt;/AudioBufferCount&gt;&lt;AudioBufferSize&gt;0x10000&lt;
/AudioBufferSize&gt;&lt;BufferingTime&gt;-400&lt;
/BufferingTime&gt;&lt;FOURCCMap&gt;h263=H263,h264=H264,x264=H264,X264=H264,mp4v=MP4V,fmp4=MP4V,FMP4=MP4V,smp4=MP4V,SMP4=MP4V,3iv2=MP4V,3I\
lt;/FOURCCMap&gt;&lt;AutoJitter&gt;false&lt;/AutoJitter&gt;&lt;PreferAC3&gt;false&lt;/PreferAC3&gt;&lt;MediaLanguageLANGID&gt;0x409&
lt;/MediaLanguageLANGID&gt;&lt;SetDefaultOnAir&gt;true&lt;/SetDefaultOnAir&gt;&lt;SystrayVisible&gt;true&lt;/SystrayVisible&gt;&lt;
```

Figure A.69. Server response for the example. Red marked is the DTVLink tag.

A.2.22. Get Scenario

Note

This snippit requires a saved *bitcontrol® Digital TV Link* scenario. More information about *bitcontrol® Digital TV Link* scenarios can be found in *bitcontrol® Digital TV Link* user manual.

Note

bitcontrol® Digital TV Link must be started with a saved scenario!

Returns the path to the scenario file

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum0">
 <![CDATA[
 DTVLink.Scenario
 ]]>
</Script>
```

Figure A.70. Example JScript Snippet request for Get Scenario (core function)

The response is the path to the scenario that was used to start *bitcontrol® Digital TV Link*.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum0">C:\scenarios\test_scenario.dtv</Result>
```

Figure A.71. JScript Snippet server response for Get Scenario (core function)

A.2.23. Get EnableLogFile

Returns the state of the *Save Log* option

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum15">
 <![CDATA[
 DTVLink.EnableLogFile
 ]]>
</Script>
```

Figure A.72. JScript Snippet request for Get EnableLogFile (core function)

The response is a boolean value that means the option is enabled (true) or disabled (false).

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum15">true</Result>
```

Figure A.73. JScript Snippet server response for Get EnableLogFile (core function)

Picture Figure A.76, “ bitcontrol® Digital TV Link log option and log preferences[43] shows the *Save Log* option in the preferences area of the *bitcontrol® Digital TV Link Options* dialogues (red marked).

A.2.24. Get LogFilename

Returns the path to the log file

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum15">
 <![CDATA[
 DTVLink.LogFilename
 ]]>
</Script>
```

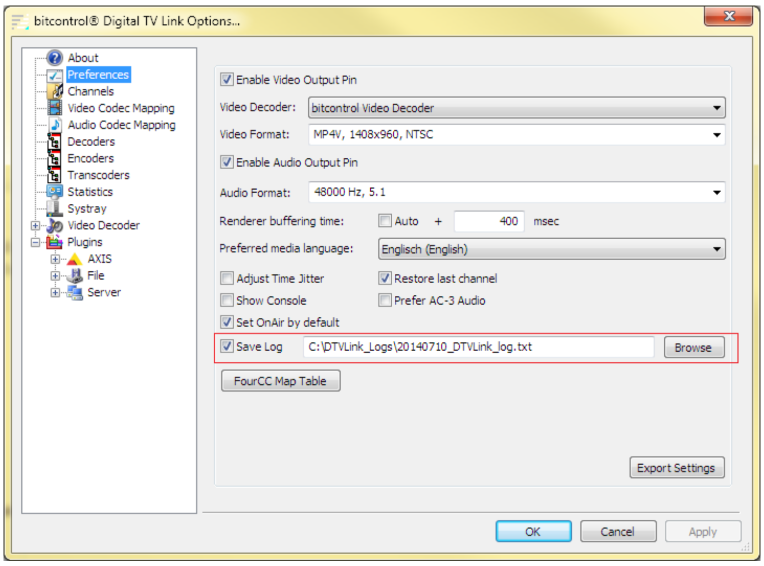
Figure A.74. JScript Snippet request for Get LogFilename (core function)

The response is the path to the log file..

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum15">C:\DTVLink_Logs\20140710_DTVLink_log.txt</Result>
```

Figure A.75. JScript Snippet server response for Get LogFilename (core function)

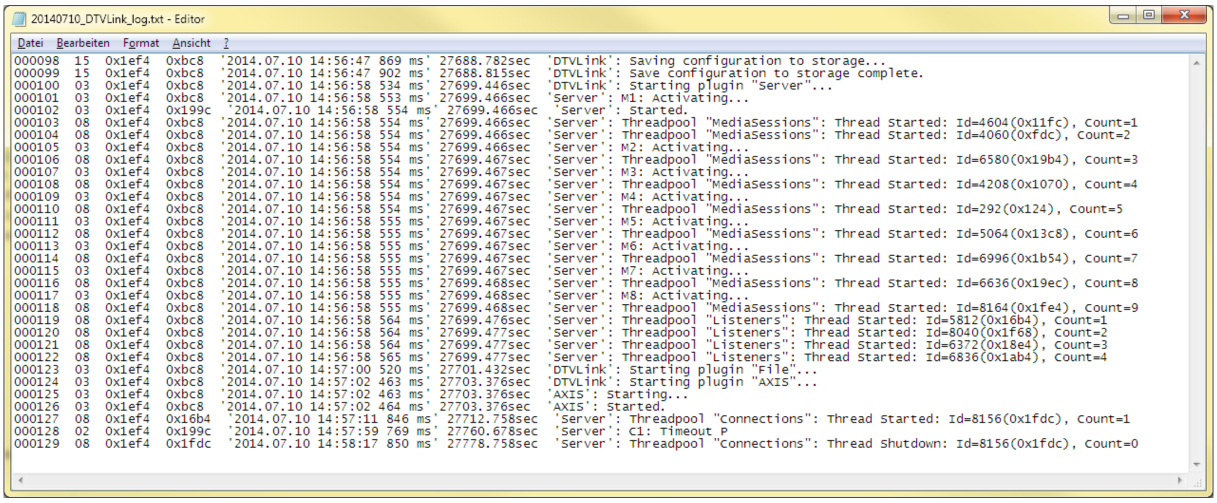
The next picture shows the log file path in the preferences area of the *bitcontrol® Digital TV Link Options* dialogues (red marked).



[✓] Enable Video Output Pin
Video Decoder: bitcontrol Video Decoder
Video Format: MP4V, 1408x960, NTSC
[✓] Enable Audio Output Pin
Audio Format: 48000 Hz, 5.1
Renderer buffering time: [ ] Auto + 400 msec
Preferred media language: Englisch (English)
[ ] Adjust Time Jitter [✓] Restore last channel
[ ] Show Console [ ] Prefer AC-3 Audio
[✓] Set OnAir by default
[✓] Save Log C:\DTVLink_Logs\20140710_DTVLink_log.txt

Figure A.76. bitcontrol® Digital TV Link log option and log preferences

Picture Figure A.77, “ Example of a bitcontrol® Digital TV Link log file [44] shows an example of a *bitcontrol® Digital TV Link* log file.


20140710_DTVLink_log.txt - Editor
Datei Bearbeiten Format Ansicht ?

```
000098  15  0x1ef4  0xbc8  '2014.07.10 14:56:47 869 ms' 27688.782sec  'DTVLink': Saving configuration to storage...
000099  15  0x1ef4  0xbc8  '2014.07.10 14:56:47 902 ms' 27688.815sec  'DTVLink': Save configuration to storage complete.
000100  03  0x1ef4  0xbc8  '2014.07.10 14:56:58 534 ms' 27699.446sec  'DTVLink': Starting plugin "Server"...
000101  03  0x1ef4  0xbc8  '2014.07.10 14:56:58 553 ms' 27699.466sec  'Server': M1: Activating...
000102  03  0x1ef4  0x199c  '2014.07.10 14:56:58 554 ms' 27699.466sec  'Server': Started.
000103  08  0x1ef4  0xbc8  '2014.07.10 14:56:58 554 ms' 27699.466sec  'Server': Threadpool "MediaSessions": Thread Started: Id=4604(0x11fc), Count=1
000104  08  0x1ef4  0xbc8  '2014.07.10 14:56:58 554 ms' 27699.466sec  'Server': Threadpool "MediaSessions": Thread Started: Id=4060(0xfdc), Count=2
000105  03  0x1ef4  0xbc8  '2014.07.10 14:56:58 554 ms' 27699.466sec  'Server': M2: Activating...
000106  08  0x1ef4  0xbc8  '2014.07.10 14:56:58 554 ms' 27699.467sec  'Server': Threadpool "MediaSessions": Thread Started: Id=6580(0x19b4), Count=3
000107  03  0x1ef4  0xbc8  '2014.07.10 14:56:58 554 ms' 27699.467sec  'Server': M3: Activating...
000108  08  0x1ef4  0xbc8  '2014.07.10 14:56:58 554 ms' 27699.467sec  'Server': Threadpool "MediaSessions": Thread Started: Id=4208(0x1070), Count=4
000109  03  0x1ef4  0xbc8  '2014.07.10 14:56:58 554 ms' 27699.467sec  'Server': M4: Activating...
000110  08  0x1ef4  0xbc8  '2014.07.10 14:56:58 554 ms' 27699.467sec  'Server': Threadpool "MediaSessions": Thread Started: Id=292(0x124), Count=5
000111  03  0x1ef4  0xbc8  '2014.07.10 14:56:58 555 ms' 27699.467sec  'Server': M5: Activating...
000112  08  0x1ef4  0xbc8  '2014.07.10 14:56:58 555 ms' 27699.467sec  'Server': Threadpool "MediaSessions": Thread Started: Id=5064(0x13c8), Count=6
000113  03  0x1ef4  0xbc8  '2014.07.10 14:56:58 555 ms' 27699.467sec  'Server': M6: Activating...
000114  08  0x1ef4  0xbc8  '2014.07.10 14:56:58 555 ms' 27699.467sec  'Server': Threadpool "MediaSessions": Thread Started: Id=6996(0x1b54), Count=7
000115  03  0x1ef4  0xbc8  '2014.07.10 14:56:58 555 ms' 27699.467sec  'Server': M7: Activating...
000116  08  0x1ef4  0xbc8  '2014.07.10 14:56:58 555 ms' 27699.468sec  'Server': Threadpool "MediaSessions": Thread Started: Id=6636(0x19ec), Count=8
000117  03  0x1ef4  0xbc8  '2014.07.10 14:56:58 555 ms' 27699.468sec  'Server': M8: Activating...
000118  08  0x1ef4  0xbc8  '2014.07.10 14:56:58 555 ms' 27699.468sec  'Server': Threadpool "MediaSessions": Thread Started: Id=8164(0x1fe4), Count=9
000119  08  0x1ef4  0xbc8  '2014.07.10 14:56:58 564 ms' 27699.476sec  'Server': Threadpool "Listeners": Thread Started: Id=5812(0x16b4), Count=1
000120  08  0x1ef4  0xbc8  '2014.07.10 14:56:58 564 ms' 27699.477sec  'Server': Threadpool "Listeners": Thread Started: Id=8040(0x1f68), Count=2
000121  08  0x1ef4  0xbc8  '2014.07.10 14:56:58 564 ms' 27699.477sec  'Server': Threadpool "Listeners": Thread Started: Id=6372(0x18e4), Count=3
000122  08  0x1ef4  0xbc8  '2014.07.10 14:56:58 565 ms' 27699.477sec  'Server': Threadpool "Listeners": Thread Started: Id=6836(0x1ab4), Count=4
000123  03  0x1ef4  0xbc8  '2014.07.10 14:57:00 520 ms' 27701.432sec  'DTVLink': Starting plugin "File"...
000124  03  0x1ef4  0xbc8  '2014.07.10 14:57:02 463 ms' 27703.376sec  'DTVLink': Starting plugin "AXIS"...
000125  03  0x1ef4  0xbc8  '2014.07.10 14:57:02 463 ms' 27703.376sec  'AXIS': Starting...
000126  03  0x1ef4  0xbc8  '2014.07.10 14:57:02 464 ms' 27703.376sec  'AXIS': Started.
000127  08  0x1ef4  0x16b4  '2014.07.10 14:57:11 846 ms' 27712.758sec  'Server': Threadpool "Connections": Thread Started: Id=8156(0x1fdc), Count=1
000128  02  0x1ef4  0x199c  '2014.07.10 14:57:59 769 ms' 27760.678sec  'Server': C1: Timeout P
000129  08  0x1ef4  0x1fdc  '2014.07.10 14:58:17 850 ms' 27778.758sec  'Server': Threadpool "Connections": Thread Shutdown: Id=8156(0x1fdc), Count=0
```

Figure A.77. Example of a bitcontrol® Digital TV Link log file

A.2.25. Get OptionsDialogVisible / Get ShowConsole / Get SystrayVisible

- *Get OptionsDialogVisible*: Returns the state of the *bitcontrol® Digital TV Link Options* dialogues
- *Get ShowConsole*: Returns the state of the console
- *Get ShowConsole*: Returns the state of the systray icon in the footline of the desktop

If the *bitcontrol® Digital TV Link Options* dialogues / console / systray icon are open or visible, the result will be true. Elsewise the result will be false.

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum15">
 <![CDATA[
 DTVLink.OptionsDialogVisible
 ]]>
</Script>
```

Figure A.78. Example JScript Snippet request for Get OptionsDialogVisible (core function)

The response is a boolean value that means the option is enabled (true) or disabled (false). In this case it is true, that means the *bitcontrol® Digital TV Link Options* dialogues were open.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum15">true</Result>
```

Figure A.79. Example JScript Snippet server response for Get OptionsDialogVisible (core function)

A.2.26. Show OptionsDialog /Show Console / Show SystrayIcon

- *Show OptionsDialog*: Opens or closes the *bitcontrol® Digital TV Link Options* dialogues.
- *Show Console*: Opens or closes the console.
- *Show SystrayIcon*: Opens or closes the sysray icon.

For this the value *true* (open) or *true* (close) can be set.

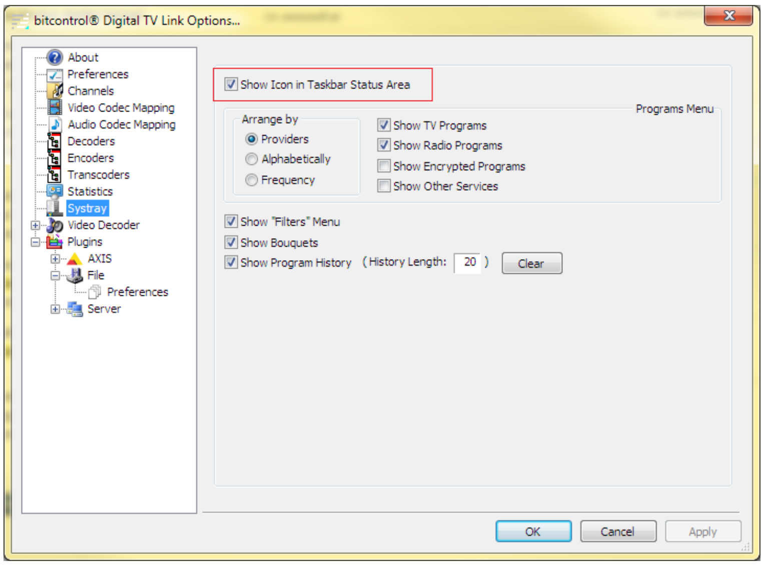
```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum15">
 <![CDATA[
 DTVLink.OptionsDialogVisible = true;
 ]]>
</Script>
```

Figure A.80. Example JScript Snippet request for Show OptionsDialog (core function)

The example request above opens the *bitcontrol® Digital TV Link Options* dialogues that were not open before.

Note

The option for setting the systray icon visible / invisible can be found in *Systray* window of the *bitcontrol® Digital TV Link Options* dialogues (see Figure A.81, “ Systray icon visible / invisible option in Systray window of the bitcontrol® Digital TV Link Options dialogues ” [46]).


bitcontrol® Digital TV Link Options...

- About
- Preferences
- Channels
- Video Codec Mapping
- Audio Codec Mapping
- Decoders
- Encoders
- Transcoders
- Statistics
- Systray
- Video Decoder
- Plugins
  - AXIS
  - File
    - Preferences
  - Server

[✓] Show Icon in Taskbar Status Area

Programs Menu

Arrange by: (●) Providers ( ) Alphabetically ( ) Frequency

[✓] Show TV Programs
[✓] Show Radio Programs
[ ] Show Encrypted Programs
[ ] Show Other Services

[✓] Show "Filters" Menu
[✓] Show Bouquets
[✓] Show Program History (History Length: 20)

Figure A.81. Systray icon visible / invisible option in Systray window of the bitcontrol® Digital TV Link Options dialogues

A.2.27. Get All Channels

Returns the channels of all started plug-ins and their properties

The next code shows the Jscript Snippit for *Get All Channels*. Expressions that are marked with // will not be returned.

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum0">
 <![CDATA[
 for (var i in DTVLink.Plugins) {
 var plugin = DTVLink.Plugins[i];
 if ((plugin.Plugin != null) && (plugin.Plugin.Channels != undefined)) {
  var receiver = plugin.Plugin;
  var channels = receiver.Channels;
  var xmlreceiver = XMLResponse.ownerDocument.createElement('Receiver');
  XMLResponse.appendChild(xmlreceiver);
  xmlreceiver.setAttribute('Name', plugin.Name);
  xmlreceiver.setAttribute('Channels', channels.Count);
  for (var i in channels) {
   var channel = channels[i];
   var xml = XMLResponse.ownerDocument.createElement('Channel');
   xmlreceiver.appendChild(xml);
   xml.setAttribute('Id', channel.Id);
   xml.setAttribute('Name', channel.Name);
   xml.setAttribute('Provider', channel.Provider);
   xml.setAttribute('Type', channel.Type);
   xml.setAttribute('Network', channel.Network);
   xml.setAttribute('Bouquets', channel.Bouquets);
   xml.setAttribute('URL', channel.URL);
   //xml.setAttribute('SID', channel.SID);
   //xml.setAttribute('PMT_PID', channel.PMT_PID);
   //xml.setAttribute('PCR_PID', channel.PCR_PID);
   //xml.setAttribute('Video_PID', channel.Video_PID);
   //xml.setAttribute('Audio_PID', channel.Audio_PID);
   //xml.setAttribute('FreeCA', channel.FreeCA);
   //xml.setAttribute('LNB', channel.LNB);
   //xml.setAttribute('Kanal', channel.Kanal);
   //xml.setAttribute('FrequencyKHz', channel.FrequencyKHz);
   //xml.setAttribute('SymbolRate', channel.SymbolRate);
   //xml.setAttribute('Polarity', channel.Polarity);
   //xml.setAttribute('FEC', channel.FEC);
   //xml.setAttribute('FECMethod', channel.FECMethod);
   //xml.setAttribute('Modulation', channel.Modulation);
   //xml.setAttribute('Bandwidth', channel.Bandwidth);
   //xml.setAttribute('GuardInterval', channel.GuardInterval);
   //xml.setAttribute('TransmissionMode', channel.TransmissionMode);
   //xml.setAttribute('SpectralInversion', channel.SpectralInversion);
   //xml.setAttribute('ModulationSystem', channel.ModulationSystem);
   //xml.setAttribute('RollOff', channel.RollOff);
   //xml.setAttribute('Pilot', channel.Pilot);
   }
  }
 }
 ]]>
</Script>
```

The result shows that there are two plug-ins started (*AXIS* plug-in line 3 and *File* plug-in line 10). Each plug-in has one active channel (also line 3 and line 10). In addition to these information, the available channel properties are displayed too.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum0">
    <Receiver Name="AXIS" Channels="1">
        <Channel Id="0"
        Name="AXIS M3011 - 00408C9492B7"
        Provider="" Type="1"
        Network="unicast" Bouquets=""
        URL="rtsp://192.168.1.28/axis-media/media.amp?videocodec=mpeg4"/>
    </Receiver>
    <Receiver Name="File" Channels="1">
        <Channel Id="2"
        Name="FIFASPOT_LVB_3gmp4_352x288.mp4"
        Provider=""
        Type="1" Network=""
        Bouquets=""
        URL="H:\video_samples\MPEG-4 3GP\FIFASPOT_LVB_3gmp4_352x288.mp4"/>
    </Receiver>
</Result>
```

Figure A.82. JScript Snippet server response for Show All Channels (core function)

A.2.28. Statistics

Returns statistical values of the channel the is currently in use

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum3">
 <![CDATA[
 var stat = DTVLink.Statistics;
 var xmlstat = XMLResponse.ownerDocument.createElement('Statistics');
 var xmlstatvideo = XMLResponse.ownerDocument.createElement('Video');
 var xmlstataudio = XMLResponse.ownerDocument.createElement('Audio');
 XMLResponse.appendChild(xmlstat);
 xmlstat.appendChild(xmlstatvideo);
 xmlstat.appendChild(xmlstataudio);
 xmlstatvideo.setAttribute('FCC', stat.VideoFCC);
 xmlstatvideo.setAttribute('ReceivedBytes', stat.VideoReceivedBytes);
 xmlstatvideo.setAttribute('ReceivedPackets', stat.VideoReceivedPackets);
 xmlstatvideo.setAttribute('BufferFillBytes', stat.VideoBufferFillBytes);
 xmlstatvideo.setAttribute('BufferFillTimeSec', stat.VideoBufferFillTimeSec);
 xmlstatvideo.setAttribute('StreamTimestampSec',stat.VideoStreamTimestampSec);
 xmlstataudio.setAttribute('FCC', stat.AudioFCC);
 xmlstataudio.setAttribute('ReceivedBytes', stat.AudioReceivedBytes);
 xmlstataudio.setAttribute('ReceivedPackets', stat.AudioReceivedPackets);
 xmlstataudio.setAttribute('BufferFillBytes', stat.AudioBufferFillBytes);
 xmlstataudio.setAttribute('BufferFillTimeSec', stat.AudioBufferFillTimeSec);
 xmlstataudio.setAttribute('StreamTimestampSec', stat.AudioStreamTimestampSec);
 xmlstat.setAttribute('Substreams', stat.SubstreamsCount);
 for (i=0; i<stat.SubstreamsCount; i++) {
  var xmlstatsubstream = XMLResponse.ownerDocument.createElement('Substream');
  xmlstat.appendChild(xmlstatsubstream);
  xmlstatsubstream.setAttribute('Type', stat.GetSubstreamType(i));
  xmlstatsubstream.setAttribute('FCC', stat.GetSubstreamFCC(i));
  xmlstatsubstream.setAttribute('Id', stat.GetSubstreamId(i));
  xmlstatsubstream.setAttribute('Description', stat.GetSubstreamDescription(i));
  xmlstatsubstream.setAttribute('ReceivedBytes', stat.GetSubstreamReceivedBytes(i));
  xmlstatsubstream.setAttribute('ReceivedSamples',
  stat.GetSubstreamReceivedSamples(i));
  xmlstatsubstream.setAttribute('PacketizedSamples',
  stat.GetSubstreamPacketizedSamples(i));
  xmlstatsubstream.setAttribute('LastOriginalTimestampSec',
  tat.GetSubstreamLastOriginalTimestampSec(i));
  xmlstatsubstream.setAttribute('LastOriginalDTSSec',
  stat.GetSubstreamLastOriginalDTSSec(i));
  xmlstatsubstream.setAttribute('LastOriginalPTSSec',
  stat.GetSubstreamLastOriginalPTSSec(i));
  xmlstatsubstream.setAttribute('TimestampAverageJitterSec',
  stat.GetSubstreamTimestampAverageJitterSec(i));
 }
 ]]>
</Script>
```

Figure A.83. JScript Snippet request for Statistics (core function)

The result is a screenshot of available statistical values of the channel the is currently in use.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum3">
    <Statistics Substreams="">
        <Video FCC="MP4V"
        ReceivedBytes="27963913"
        ReceivedPackets="31570"
        BufferFillBytes="1.84467440737095E+19"
        BufferFillTimeSec="0"
        StreamTimestampSec="5.92"/>
        <Audio FCC=" AAC"
        ReceivedBytes="237332"
        ReceivedPackets="39374"
        BufferFillBytes="0"
        BufferFillTimeSec="0"
        StreamTimestampSec="15994.1045403"/>
    </Statistics>
</Result>
```

Figure A.84. JScript Snippet server response for Statistics (core function)

A.2.29. Digital TV Link collections

A.2.29.1. Enumerate Satelittes / Enumertate Cables / Enumerate Terrestrials

Returns the available transponders

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum15">
 <![CDATA[
 XMLResponse.setAttribute('Terrestrials', DTVLink.Terrestrials.Count);
 for (var i in DTVLink.Terrestrials) {
 var terrestrial = DTVLink.Terrestrials[i];
 var xmlterrestrial = XMLResponse.ownerDocument.createElement('Terrestrial');
 XMLResponse.appendChild(xmlterrestrial);
 xmlterrestrial.setAttribute('Transponders', terrestrial.Transponders.Count);
 xmlterrestrial.setAttribute('Id', terrestrial.Id);
 xmlterrestrial.setAttribute('Name', terrestrial.Name);
 }
 ]]>
</Script>
```

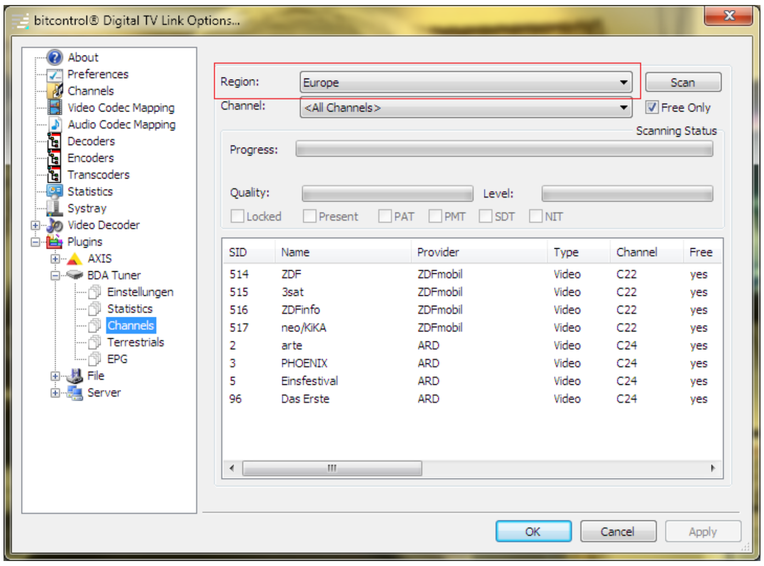
Figure A.85. Example JScript Snippet request for Get Enumerate Terrestrials (core function)

The result shows that one transponder was found.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum1" Terrestrials="1">
<Terrestrial Transponders="57" Id="0" Name="Europe"/>
</Result>
```

Figure A.86. Example JScript Snippet server response for Get Enumerate Terrestrials (core function)

The *bitcontrol® Digital TV Link Options* dialogues contain a corresponding field for the example. It can be found in the *BDA* plug-in preferences window (see red marked aera in Figure A.87, “ Transponder in BDA plug-in preferences window ” [51]).



Region: Europe

Channel: <All Channels>

[✓] Free Only

SID	Name	Provider	Type	Channel	Free
514	ZDF	ZDFmobil	Video	C22	yes
515	3sat	ZDFmobil	Video	C22	yes
516	ZDFinfo	ZDFmobil	Video	C22	yes
517	neo/KiKA	ZDFmobil	Video	C22	yes
2	arte	ARD	Video	C24	yes
3	PHOENIX	ARD	Video	C24	yes
5	Einsfestival	ARD	Video	C24	yes
96	Das Erste	ARD	Video	C24	yes

Figure A.87. Transponder in BDA plug-in preferences window

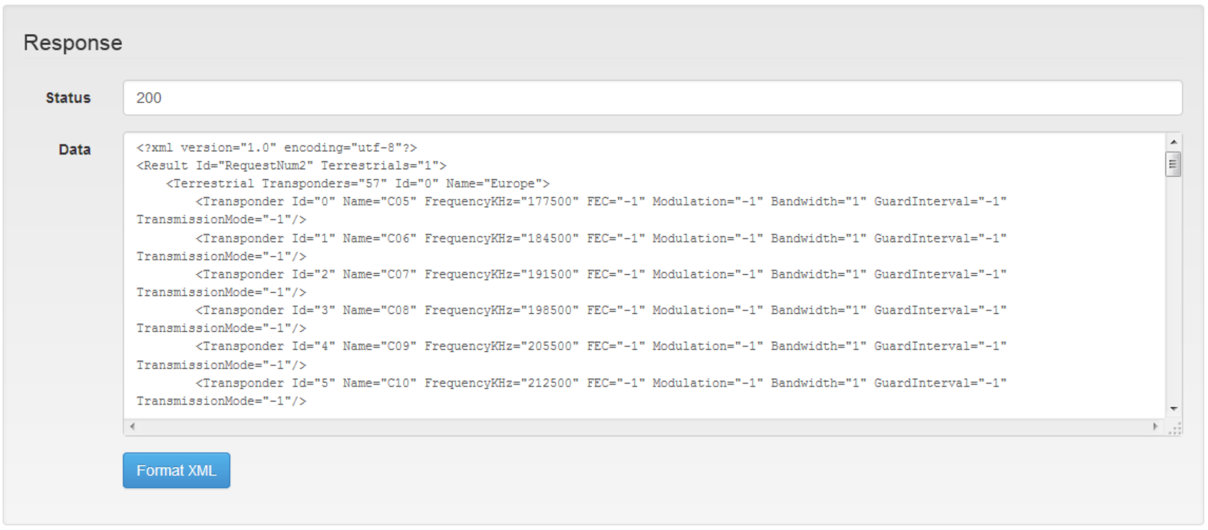
A.2.29.2. Enumerate Satelittes (Extra) / Enumertate Cables (Extra) / Enumerate Terrestrials (Extra)

Returns the available transponders and their channels

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum15">
 <![CDATA[
 XMLResponse.setAttribute('Terrestrials', DTVLink.Terrestrials.Count);
 for (var i in DTVLink.Terrestrials) {
 var terrestrial = DTVLink.Terrestrials[i];
 var xmlterrestrial = XMLResponse.ownerDocument.createElement('Terrestrial');
 XMLResponse.appendChild(xmlterrestrial);
 xmlterrestrial.setAttribute('Transponders', terrestrial.Transponders.Count);
 xmlterrestrial.setAttribute('Id', terrestrial.Id);
 xmlterrestrial.setAttribute('Name', terrestrial.Name);
 // Enum Transponders...
 for (var t in terrestrial.Transponders) {
  var transponder = terrestrial.Transponders[t];
  var xmltransponder = XMLResponse.ownerDocument.createElement('Transponder');
  xmlterrestrial.appendChild(xmltransponder);
  xmltransponder.setAttribute('Id', transponder.Id);
  xmltransponder.setAttribute('Name', transponder.Name);
  xmltransponder.setAttribute('FrequencyKHz', transponder.FrequencyKHz);
  xmltransponder.setAttribute('FEC', transponder.FEC);
  xmltransponder.setAttribute('Modulation', transponder.Modulation);
  xmltransponder.setAttribute('Bandwidth', transponder.Bandwidth);
  xmltransponder.setAttribute('GuardInterval', transponder.GuardInterval);
  xmltransponder.setAttribute('TransmissionMode', transponder.TransmissionMode);
  }
 }
 ]]>
</Script>
```

Figure A.88. Example JScript Snippet request for Get Enumerate Terrestrials (core function)

The next picture shows the server response window for JScript Snippit *Get Enumerate Terrestrials Extra*. In this case there is one Transponder named *Region* available. The transponder has 69 channels. So 69 channel entries and their properties are visible too.



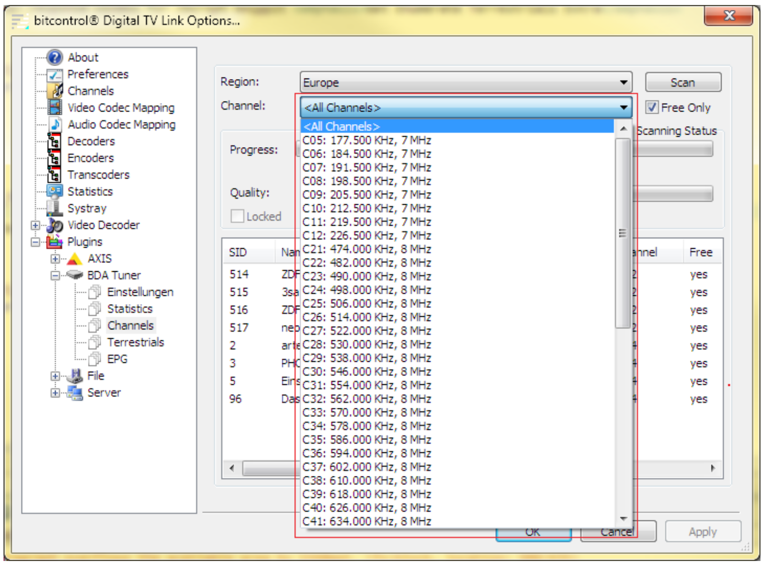
Status: 200

Data:

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum2" Terrestrials="1">
    <Terrestrial Transponders="57" Id="0" Name="Europe">
        <Transponder Id="0" Name="C05" FrequencyKHz="177500" FEC="-1" Modulation="-1" Bandwidth="1" GuardInterval="-1"
TransmissionMode="-1"/>
        <Transponder Id="1" Name="C06" FrequencyKHz="184500" FEC="-1" Modulation="-1" Bandwidth="1" GuardInterval="-1"
TransmissionMode="-1"/>
        <Transponder Id="2" Name="C07" FrequencyKHz="191500" FEC="-1" Modulation="-1" Bandwidth="1" GuardInterval="-1"
TransmissionMode="-1"/>
        <Transponder Id="3" Name="C08" FrequencyKHz="198500" FEC="-1" Modulation="-1" Bandwidth="1" GuardInterval="-1"
TransmissionMode="-1"/>
        <Transponder Id="4" Name="C09" FrequencyKHz="205500" FEC="-1" Modulation="-1" Bandwidth="1" GuardInterval="-1"
TransmissionMode="-1"/>
        <Transponder Id="5" Name="C10" FrequencyKHz="212500" FEC="-1" Modulation="-1" Bandwidth="1" GuardInterval="-1"
TransmissionMode="-1"/>
```

Figure A.89. Part of the server response window for JScript Snippit Get Enumerate Terrestrials Extra

Picture Figure A.90, “ Transponder channels in BDA plug-in preferences window [53] shows the corresponding area in in *BDA* plug-in preferences window (*bitcontrol® Digital TV Link Options* dialogues).



Region: Europe

Channel: <All Channels>

[✓] Free Only

```
<All Channels>
C05: 177.500 KHz, 7 MHz
C06: 184.500 KHz, 7 MHz
C07: 191.500 KHz, 7 MHz
C08: 198.500 KHz, 7 MHz
C09: 205.500 KHz, 7 MHz
C10: 212.500 KHz, 7 MHz
C11: 219.500 KHz, 7 MHz
C12: 226.500 KHz, 7 MHz
C21: 474.000 KHz, 8 MHz
C22: 482.000 KHz, 8 MHz
C23: 490.000 KHz, 8 MHz
C24: 498.000 KHz, 8 MHz
C25: 506.000 KHz, 8 MHz
C26: 514.000 KHz, 8 MHz
C27: 522.000 KHz, 8 MHz
C28: 530.000 KHz, 8 MHz
C29: 538.000 KHz, 8 MHz
C30: 546.000 KHz, 8 MHz
C31: 554.000 KHz, 8 MHz
C32: 562.000 KHz, 8 MHz
C33: 570.000 KHz, 8 MHz
C34: 578.000 KHz, 8 MHz
C35: 586.000 KHz, 8 MHz
C36: 594.000 KHz, 8 MHz
C37: 602.000 KHz, 8 MHz
C38: 610.000 KHz, 8 MHz
C39: 618.000 KHz, 8 MHz
C40: 626.000 KHz, 8 MHz
C41: 634.000 KHz, 8 MHz
```

SID	Nan	nnel	Free
514	ZDF	2	yes
515	3sa	2	yes
516	ZDF	2	yes
517	neo	2	yes
2	arte	4	yes
3	PHO	4	yes
5	Eins	4	yes
96	Das	4	yes

Figure A.90. Transponder channels in BDA plug-in preferences window

A.2.30. Plug-ins

A.2.30.1. Enumerate Plugins

Returns the state of the plug-ins

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum15">
 <![CDATA[
 XMLResponse.setAttribute('Plugins', DTVLink.Plugins.Count);
 for (var i in DTVLink.Plugins) {
 var plugin = DTVLink.Plugins[i];
 var xmlplugin = XMLResponse.ownerDocument.createElement('Plugin');
 XMLResponse.appendChild(xmlplugin);
 xmlplugin.setAttribute('Name', plugin.Name);
 xmlplugin.setAttribute('State', plugin.State);
 xmlplugin.setAttribute('Class', plugin.Class);
 xmlplugin.setAttribute('Priority', plugin.Priority);
 xmlplugin.setAttribute('StartupType', plugin.StartupType);
 }
 ]]>
</Script>
```

Figure A.91. Example JScript Snippet request for Enumerate Plugins (core function)

The following properties and their states will be read ou by the JScript Snippit:

- *Name*: Name of the plug-in (needn't to be the type pf the plug-in)
- *State*: State of the plug-in (0 = not loaded; 1 = loaded but not started; 2 = started)
- *Class*: Type of the plug-in
- *Priority*: Execution priority of the plug-in (high number means high priority and prefered execution; decreasing number means decreasing priority of execution)
- *StartupType*: Type of plug-in start (0 = Disabled; 1 = Auto; 2 = Manual)

The result of the request (see next code) shows all available plug-ins, their properties and states.

```
<?xml version="1.0" encoding="utf-8"?>
<Result Id="RequestNum0" Plugins="13">
    <Plugin Name="AXIS" State="1" Class="AXIS" Priority="110"
    StartupType="2"/>
    <Plugin Name="BDA Tuner" State="1" Class="BDA" Priority="100"
    StartupType="2"/>
    <Plugin Name="BoschVIPX" State="0" Class="BoschVIPX" Priority="110"
    StartupType="0"/>
    <Plugin Name="Capture" State="0" Class="Capture" Priority="100"
    StartupType="0"/>
    <Plugin Name="DBox2" State="0" Class="DBox2" Priority="100"
    StartupType="0"/>
    <Plugin Name="DreamBox" State="0" Class="DreamBox" Priority="100"
    StartupType="0"/>
    <Plugin Name="File" State="1" Class="ReceiverFile" Priority="110"
     StartupType="2"/>
    <Plugin Name="NetPublisher" State="0" Class="NetPublisher" Priority="100"
    StartupType="0"/>
    <Plugin Name="Recorder" State="0" Class="Recorder" Priority="50"
    StartupType="0"/>
    <Plugin Name="ReelBox" State="0" Class="ReelBox" Priority="100"
    StartupType="0"/>
    <Plugin Name="RTSP Receiver" State="0" Class="RTSPReceiver" Priority="110"
    StartupType="0"/>
    <Plugin Name="Server" State="2" Class="MultistreamServer" Priority="200"
    StartupType="2"/>
    <Plugin Name="TraficonVIPT" State="0" Class="TraficonVIPT" Priority="110"
    StartupType="0"/>
</Result>
```

Figure A.92. Example JScript Snippet server response for Enumerate Plugins (core function)

A.2.30.2. Create Plugin Instance

This JScript Snippit allows the creation of an own plug-in. For this, the default code (see Figure A.95, “ JScript Snippit Create Plugin Instance code with modifications in request window[57]) can be modified. The properties mentioned in Section A.2.30.1, “Enumerate Plugins”[53] can be changed for the new plug-in.

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum15">
 <![CDATA[
 // Create Plugin Context object
 var plugin = DTVLink.CreatePlugin();

 // Setup Class
 plugin.Class = 'ReceiverFile'; // Alternative:
 'Bitcontrol.DTVLink.Plugin.ReceiverFile'
 or '{AA6CEC10-2408-4E69-845F-28F30755B00C}'

 // Add context to Digital TV Link plugins list
 DTVLink.Plugins['My Plugin'] = plugin;
 // Alternative:
 // plugin.Name = 'My Plugin';
 // DTVLink.Plugins.Add(plugin);

 // Load Plugin
 plugin.State = 1;

 // Set plugin params
 plugin.Plugin.Mode = 2;

 // Start plugin
 plugin.State = 2;
 ]]>
</Script>
```

Figure A.93. Example JScript Snippet request for Create Plugin Instance (core function)

Example modification:

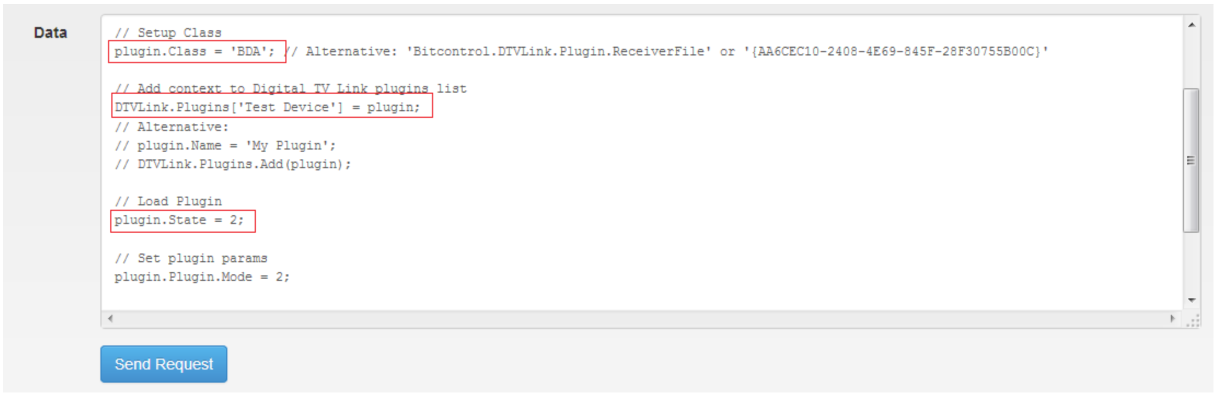
- Plug-in class: BDA
- Plug-in name: Test Device
- Plug-in state: started (2)
- Rest default

The next lines show the example modifications in the code.

```
<Script Language="JScript" GlobalContext="true" Expression="true"
Response="XMLResponse" Id="RequestNum15">
 <![CDATA[
 // Create Plugin Context object
 var plugin = DTVLink.CreatePlugin();

 // Setup Class
 plugin.Class = 'BDA'; // Alternative:
 'Bitcontrol.DTVLink.Plugin.ReceiverFile'
 or '{AA6CEC10-2408-4E69-845F-28F30755B00C}'

 // Add context to Digital TV Link plugins list
 DTVLink.Plugins['Test Device'] = plugin;
 // Alternative:
 // plugin.Name = 'My Plugin';
 // DTVLink.Plugins.Add(plugin);

 // Load Plugin
 plugin.State = 2;

 // Set plugin params
 plugin.Plugin.Mode = 2;

 // Start plugin
 plugin.State = 2;
 ]]>
</Script>
```

Figure A.94. Example JScript Snippet request for Create Plugin Instance with modifications (core function)

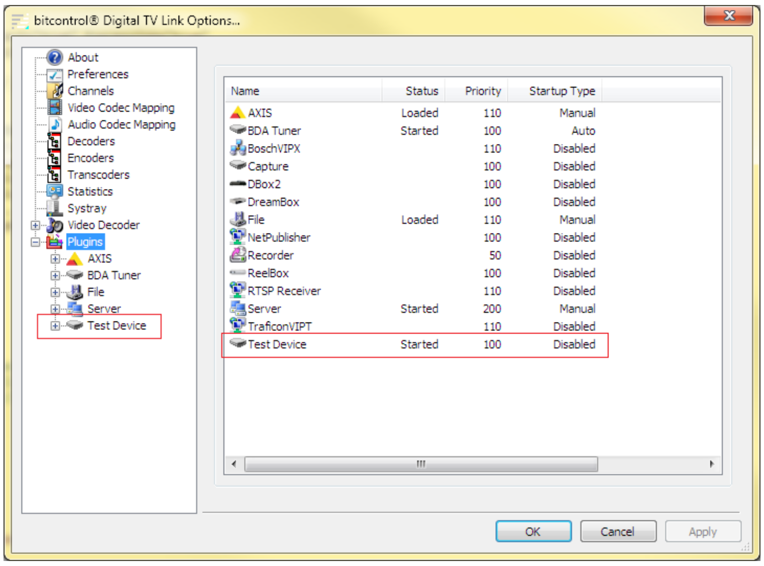
Picture Figure A.95, “ JScript Snippit Create Plugin Instance code with modifications in request window ” [57] shows the code example again. The modifications are marked red.



```
// Setup Class
plugin.Class = 'BDA'; // Alternative: 'Bitcontrol.DTVLink.Plugin.ReceiverFile' or '{AA6CEC10-2408-4E69-845F-28F30755B00C}'

// Add context to Digital TV Link plugins list
DTVLink.Plugins['Test Device'] = plugin;
// Alternative:
// plugin.Name = 'My Plugin';
// DTVLink.Plugins.Add(plugin);

// Load Plugin
plugin.State = 2;

// Set plugin params
plugin.Plugin.Mode = 2;
```

Figure A.95. JScript Snippit Create Plugin Instance code with modifications in request window

The result of the request can be found in the plug-in list (*bitcontrol® Digital TV Link Options* dialogues). The added plug-in is visible there (red marked). It has the state *started* by default.



Name	Status	Priority	Startup Type
AXIS	Loaded	110	Manual
BDA Tuner	Started	100	Auto
BoschVIPX		110	Disabled
Capture		100	Disabled
DBox2		100	Disabled
DreamBox		100	Disabled
File	Loaded	110	Manual
NetPublisher		100	Disabled
Recorder		50	Disabled
ReelBox		100	Disabled
RTSP Receiver		110	Disabled
Server	Started	200	Manual
TraficonVIPT		110	Disabled
Test Device	Started	100	Disabled

Figure A.96. Added BDA plug-in in plug-in list