JavaScript API for controlling IPTV devices

MAG100 and MAG200

Specification

V. 1.15

2010

Contents.

About this document.

Document revision	1.15
JavaScript API version	316
STB API version	115
MAG200 player version	0x520
MAG100 player version	0x23

Overview.

This document describes the program interface allowing controlling IPTV-device (including playing various types of video content and the event pattern of the IPTV-device) from JavaScript. The document assumes the knowledge of JavaScript.

API use

It is assumed that the functions described are used from the JavaScript context on MAG100/MAG200 supplied with Mozilla Firefox 1.5 or WebKit as the browser.

Data types and arguments format.

Hereinafter the following designations shall be used:
int – for digital types.
bool – for logical types.
string – for string types.

In this document it is understood as follows: if the argument type preceded by the keyword **out**, this parameter is used to return values from the function. It is sufficient to call the function with an empty object as this parameter from JavaScript and then receive the value from the field **value** of this object.
For example:

```
var tColor;
var x = {} ;
stb.GetTransparentColor(x);
tColor = x.value;
```

The example of use and initialization of API and IPTV-device events are described in appendix 1 and in the chapter Configuring the event system.
Any operations with IPTV-device are performed via the objects **stb** and **stbEvent**.

stb object methods calling

More than one prototype of the object stb method can be described, due to different mechanisms of returning the result of the method operation. In this case the prototype shall be preceded with the following designations:

FireFox – the prototype is used when the method is called from the Mozilla FireFox browser.
WK/FF+Wrapper –The prototype is used when the method is called from the WebKit-based browser or from the Mozilla FireFox browser via **wrapper.js**.

To call any **stb** method from any JavaScript function, add the following string in the beginning of this function:

```
netscape.security.PrivilegeManager.enablePrivilege("UniversalXPConnect")
```

This rule is valid only when calling the method from the Mozilla FireFox browser without using **wrapper.js**.

Availability of the functions on IPTV-devices.

Interface functions, their availability and specifics for each IPTV-device are shown below. The functions marked with "–" are present in API for compatibility but they do not provide full functionality.

Table 1 Compatibility of the JavaScript API functions for MAG100 and MAG200

stb object methods	MAG100	MAG200
InitPlayer	+	+
DeinitPlayer	+	+
Play	+	+
PlaySolution	+	+
Stop	+	+
Pause	+	+
Continue	+	+
SetPosTime	+	+
SetPosTimeEx	–	+(WK)
SetPosPercent	+	+
SetPosPercentEx	+	+
GetPosTime	+	+
GetPosTimeEx	–	+(WK)
GetPosPercent	+	+
GetPosPercentEx	+	+
GetMediaLen	+	+
GetMediaLenEx	–	+(WK)
SetSpeed	+	+
SetAudioPID	+	+
SetPIG	+	+
SetAlphaLevel	+	+
SetVolume	+	+
SetUserFlickerControl	+	–
SetFlicker	+	+ (distinction from MAG100)
SetDefaultFlicker	+	+ (distinction from MAG100)
SetLoop	+	+

stb object methods	MAG100	MAG200
SetVideoControl	+	+
SetVideoState	+	+
SetChromaKey	+	+
SetMode	+	+
SetWinMode	+	+
SetTopWin	+	+
SetWinAlphaLevel	+	+
SetAspect	+	+ (add. capabilities)
Rotate	+	–
SetMute	+	+
SetMicVolume	+	–
GetMicVolume	+	–
GetVolume	+	+
Step	+	–
SetupRTSP	+	+
SetViewport	+	+
IsPlaying	+	+
Version	+	+
SetupSPdif	+	+
SetSubtitles	+	+
SetSubtitlesSize	+	+
SetSubtitlesFont	+	+
SetSubtitlesOffs	+	+
GetSpeed	+	+
GetAudioPID	+	+
GetPIG	+	+
GetAlphaLevel	+	+
GetWinAlphaLevel	+	+
SetTransparentColor	+	+
GetTransparentColor	+	+
IgnoreUpdates	+	+
ExecAction	+	+
SetCASType	+	+

stb object methods	MAG100	MAG200
SetCASParam	+	+
LoadCASIniFile	+	+
SetCASDescrambling	–	+
GetAspect	+	+
StandBy	+	+
RDir	+	+
SetAudioLangs	+	+
GetAudioPIDs	+	+
GetSubtitlePIDs	+	+
ReadCFG	+	+
WriteCFG	+	+
WritePrefs	+	+
Debug	+	+
SetListFilesExt	-	+(WK)
ListDir	-	+(WK)
SetBrightness	-	+(WK)
SetSaturation	-	+(WK)
SetContrast	-	+(WK)
GetBrightness	-	+(WK)
GetSaturation	-	+(WK)
GetContrast	-	+(WK)
GetSubtitlePID	-	+(WK)
SetSubtitlePID	-	+(WK)
SetSubtitleLangs	-	+(WK)
DeleteAllCookies	-	+(WK)

WK – only for WebKit.

stb object methods.

stb.InitPlayer

void InitPlayer()

Initializes the player. Call this function before using the player. The features are described in Appendix 1. API usage.

Parameters:

None.

Returned value:

None.

stb.DeinitPlayer

void DeinitPlayer()

De-initialize the player.

Parameters:

None.

Returned value:

None.

stb.Play

void Play(string playStr)

Start playing media content as specified in **playStr.**

Parameters:

playStr – string in the form: “<solution> <URL> [atrack:<anum>] [vtrack:<vnum>]“

Parameter	Allowed value	Description
Solution	rtp, rtsp, mp3, auto, mpegps, mpegts, mp4	Media content type. Depends on the IPTV-device type. See Appendix 2 for the table of supported formats and the description of media content types
URL		Address of the content to be started for

Parameter	Allowed value	Description
		playing. Depends on the type. See more detailed information in Appendix 2.
atrack:<anum>		Sets the number(PID) of audio track. Optional parameter.
vtrack:<vnum>		Sets the number(PID) of audio track. Optional parameter

Returned value:

None.

stb.PlaySolution

void PlaySolution(string solution, string URL)

Play media content of the preset type (**solution**) from the preset **URL**.

Parameters:

Parameter	Allowed value	Description
Solution		Corresponds to the parameter **solution** from the function stb.Play
URL		Address of the content to be started for playing. Depends on the type. See more detailed information in supplement 2.

Returned value:

None.

stb.Stop

void Stop()

Stops playing.

Continue() shall begin playing from the beginning.

Parameters:

None.

Returned value:

None.

stb.Pause

void Pause()

Pauses current playback.

Continue() continues playing from the current position.

Parameters:

None.

Returned value:

None.

stb.Continue

void Continue()

Continues playing (after Pause()) or begin anew (after Stop()).

Parameters:

None.

Returned value:

None.

stb.SetPosTime

void SetPosTime(int time)

Sets the new position of playback in time

Parameters:

Parameter	Allowed value	Description
Time	time >= 0	The position in seconds from the beginning of the content where the playback should start (positioning in the content).

Returned value:

None.

stb.SetPosTimeEx

void SetPosTimeEx(int time)

Sets the current playback position in time, ms.

Parameters:

Parameter	Allowed value	Description
Time	time >= 0	Position in ms from the beginning of the content where playback should start (positioning in the content)

Returned value:

None.

stb.SetPosPercent

void SetPosPercent(int prc)

Sets the current position in percent.

Parameters:

Parameter	Allowed value	Description
Prc	0..100	The position in percent of the total duration of the content where playback should start.

Returned value:

None.

stb.SetPosPercentEx

void SetPosPercentEx(int prc)

Set the current position in percent.

Parameters:

Parameters	Returned value	Description
Prc	0..10000	Position in hundredth fractions of percent of the total duration of the content, from which the playback should start.

Returned value:

None.

stb.GetPosTime

FireFox: void GetPosTime(**out** int time);

WK/FF+Wrapper: int GetPosTime();

Gets the current position in time.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
Time		Current position in second from the beginning of content.

stb.GetPosTimeEx

FireFox: void GetPosTimeEx(**out** int time);

WK/FF+Wrapper: int GetPosTimeEx();

Gets the current position in time in ms

Parameters:

None.

Returned value:

Parameter	Returned value	Description
Time		The current position in ms from the beginning of content.

stb.GetPosPercent

FireFox: void GetPosPercent(**out** int prc);

WK/FF+Wrapper: int GetPosPercent();

Gets the current position in percent.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
Prc		The current position in percent of the whole duration of the content.

stb.GetPosPercentEx

FireFox: void GetPosPercentEx(**out** int prc);

WK/FF+Wrapper: int GetPosPercentEx();

Gets the current position in hundredth fractions of percent.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
Prc	0..10000	The current position in percent of the whole duration of content.

stb.GetMediaLen

FireFox: void GetMediaLen(**out** int len);

WK/FF+Wrapper: int GetMediaLen();

Gets the duration of the current content.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
Len		Total duration of the current content in seconds.

stb.GetMediaLenEx

FireFox: void GetMediaLenEx(**out** int len);

WK/FF+Wrapper: int GetMediaLenEx();

Gets the duration of the current content in ms.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description

Parameter	Allowed value	Description
Len		Total duration of the current content in ms.

stb.SetSpeed

void SetSpeed(int speed)

Sets the rate of playing.

Parameters:

Parameter	Allowed value	Description
Speed	-8..8	Sets new playback speed: 1 - normal 2 - 2x 3 - 4x 4 - 8x 5 - 16x 6 - 1/2 7 - 1/4 8 – 12x -1 – reverse -2 - reverse 2x -3 - reverse 4x -4 - reverse 8x -5 - reverse 16x -8 - reverse 12x

Returned value:

None.

stb.SetAudioPID

void SetAudioPID(int pid)

Sets track number (PID) for audio.

Parameters:

Parameter	Allowed value	Description
Pid		Sets the number or PID of the audio track to be played in the current content. If such track is absent the sound will be disabled.

Returned value:

None.

stb.SetSubtitlePID

void SetSubtitlePID(int pid)

Sets the number of track (PID) for subtitles.

Parameters:

Parameter	Allowed value	Description
Pid		Set the number or PID for the subtitles track to be played in the current content. Is such track is absent subtitles will be disabled.

Returned value:

None.

stb.SetPIG

void SetPIG(int state,int scale,int x,int y)

Sets position and mode of video window.

Parameters:

Parameter	Allowed value	Description
State	0..1	If state=1 – show the video on full screen. If state=0 – show the video in the specified rectangle.
Scale		The scale of the video window. The present multiplier of the video window size equals to **scale/256.**

Parameter	Allowed value	Description
X		Horizontal offset of the upper left corner of the video window from the screen edge
Y		Vertical offset of the upper left corner of the video window from the screen edge

Returned value:

None.

stb.SetAlphaLevel

void SetAlphaLevel(int alpha)

Sets alpha transparency of the video window.

Parameters:

Parameter	Allowed value	Description
Alpha	0..255	Transpareny of the video window: 0 – completely transparent; 255 – completely opaque.

Returned value:

None.

stb.SetVolume

void SetVolume(int volume)

Sets volume level.

Parameters:

Parameter	Allowed value	Description
Volume	0..100	Volume level: 0 – no sound; 100 – maximal level.

Returned value:

None.

stb.SetUserFlickerControl

void SetUserFlickerControl(int mode)

Sets the control mode of Flicker-filter.

Platform**: MAG100**

Parameters:

Parameter	Allowed value	Description
Mode	0..1	Control mode of flicker-filter: 0 – API user controls flicker-filter himself (see. stb.SetFlicker and stb.SetDefaultFlicker); 1 – The player automatically switches on flicker-filter during pauses and stops and switches it off during playing.

Returned value:

None.

stb.SetFlicker

void SetFlicker(int state, int flk, int shp)

Sets Flicker-filter parameters.

Platforms: **MAG100,MAG200(**see. note**)**

Parameters:

Parameter	Allowed value	Description
State	0..1	Flicker filter on/off 0 – switch off the flicker-filter; 1 – switch on the flicker-filter.
Flk	0..15	Flicker level.
Shp	0..15	Sharpness level.

Returned value:

None.

Note:

Flicker filter on MAG200 is applicable only for graphic window, therefore it is advised to set it only once during loading and not to switch it off

flk and **shp** parameters are ignored for MAG 200

stb.SetDefaultFlicker

void SetDefaultFlicker(int state)

Turns on/off flicker-filter with the default parameters.

Platfoms: MAG100,MAG200(see. note**)**

Parameters:

Parameter	Allowed value	Description
state	0..1	Flicker-filter on/off: 0 – switch off the Flicker-filter; 1 – switch on the Flicker-filter. In this case default values for sharpness and flicker are set.

Returned value:

None.

Note:

Flicker filter on MAG200 is applicable only for graphic window, this is why it is recommended to set its only once and keep it switched

stb.SetLoop

void SetLoop(int loop)

Sets or cancels repeated playing.

Parameters:

Parameter	Allowed value	Description
Loop	0..1	0 – switch off repeated playing on the content; 1 – switch off repeated playing on the content.

Returned value:

None.

stb.SetVideoControl

void SetVideoControl (int mode)

Sets the video window control mode:

Parameters:

Parameter	Allowed value	Description
Mode	0..1	Control mode: 0 – the device automatically switches on the video window at the beginning of playing and switches it off when stops; 1 – API user uses stb.SetVideoState for instructing whether to show the video window or not.

Returned value:

None.

stb.SetVideoState

void SetVideoState (int state)

Switch on or switch off the video window.

Parameters:

Parameter	Allowed value	Description
State	0..1	Allow/prohibit video dicplay: 0 – video window is not dosplayed; 1 – video window is displayed if the stream is present.

Returned value:

None.

Notes:

Valid only if user control had been allowed with **stb.SetVideoControl**.

stb.SetChromaKey

void SetChromaKey(int key,int mask)

Set the preset colour and mask for using as ChromaKey (the transparency of any colour on the whole window).

Parameters:

Parameter	Allowed value	Description
key	0..0xffffff	Sets the colour in RGB. If ythe colour of a window pixel coincides with this colour after masking, the pixel is considered transparent.
mask	0..0xffffff	Set the mask for **key**. If the mask is equal to 0xffffff, the colour set by the parameter key is considered transparent.

Returned value:

None.

Notes:

Any changes on the screen shall be visible only subject to switching on the regime ChromaKey by the functions **stb.SetMode or stb.SetWinMode**.

stb.SetMode

void SetMode(int mode)

Switch on (mode=1) or switch off (mode=0) the mode ChromaKey for the video window.

Parameters:

Parameter	Allowed values	Description
Mode	0..1	ChromaKey mode for the video window: 0 – off; 1 – on. The parameters set by stb.SetChromaKey ostb.SetTransparentColor shall be valid if the on-mode is used.

Returned value:

None

stb.SetWinMode

void SetWinMode (int winNum, int mode)

Switch on or switch off the ChromaKey mode for the preset window

Parameters:

Parameter	Allowed value	Description
winNum	0..1	The number of the window for which this function is used: 0 – graphic window; 1 – video window.
Mode	0..1	ChromaKey mode for video window: 0 – off; 1 – on. The parameters set by stb.SetChromaKey or stb.SetTransparentColor shall be active in the on-mode

Returned value:

None.

stb.SetTopWin

void SetTopWin(int winNum)

Set the preset window over others.

Parameters:

Parameter	Allowed value	Description
winNum	0..1	Number of the window for which this function is used: 0 – graphic window; 1 – video window.

Returned value:

None.

stb.SetWinAlphaLevel

void SetWinAlphaLevel(int winNum, int alpha)

Set alpha transparency of the preset window.

Parameters:

Parameter	Allowed value	Description
winNum	0..1	Number of the window for which this function is used: 0 – graphic window; 1 – video window.
alpha	0..255	Transparency of the preset window: 0 – completely transparent; 255 – completely opaque

Returned value:

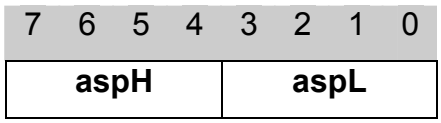
None.

stb.SetAspect

void SetAspect(int aspect)

Set video picture format.

Parameters:

Parameter	Allowed value	Description
Aspect		Sets the video picture format. Consists of 2 tetrads:  **aspH is ignored for MAG 100.**
aspL	0..3	Sets the aspect ratio: 0 – automatic; 1 – 20:9; 2 –16:9; 3 – 4:3.
aspH	0..3	Sets confersion of video format: 0 – as it is, video is stretched for the whole screen; 1 – Letter box mode, video is proportionally enlarged to the size of the screen along the larger edge; 2 – Pan&Scan mode, video is

Parameter	Allowed value	Description
		proportionally enlarged to the screen size along the lesser edge; 3 – combined mode, intermediate between Letter Box Box and Pan&Scan. 4 – enlarged mode; 5 – optimal mode. Only for MAG200

Returned value:

None.

Notes:

MAG100 ignores **aspH** .

MAG200 uses **aspL** only in windows mode, while aslH only in full screen mode, see. stb.SetPIG

stb.Rotate

void Rotate(int angle)

Rotate video.

Platform: MAG100

Parameters:

Parameter	Allowed value	Description
Angle	0, 90, 180, 270	Rotates the video window contents by the preset angle relative to the initial position.

Returned value:

None.

stb.SetMute

void SetMute(int mute)

Switch off or on the sound restoring the volume level.

Parameters:

Parameter	Allowed value	Description
Mute	0..1	Switches on/switches off the sound: 0 – on; 1 – off.
After the cycle of switching off/on with this function is completed the volume level remains unchanged.		

Returned value:

None.

stb.SetMicVolume

void SetMicVolume(int micvol)

Set the microphone volume level.

Platform: MAG100

Parameters:

Parameter	Allowed value	Description
Micvol	0..100	Set the microphone volume level: 0 – minimal volume; 100 – maximal volume.

Returned value:

None.

stb.GetMicVolume

FireFox: void GetMicVolume(**out** int micvol);

WK/FF+Wrapper: int GetMicVolume();

Receive the current microphone volume level.

Platform : MAG100

Parameters

None

Returned volume:

Parameter	Allowed volume	Description
Micvol	0..100	Returns the current microphone volume level.

stb.GetVolume

FireFox: void GetVolume(**out** int vol);

WK/FF+Wrapper: int GetVolume();

Receive the volume level.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
vol	0..100	Returns the current volume level.

stb.Step

void Step()

Display one next frame of video content.

Platform: MAG100

Parameters:

None.

Returned value:

None.

stb.SetupRTSP

void SetupRTSP(int type, int flags)

Set-client to STB.

Parameters:

Parameter	Allowed value	Description
Type	0..3	Supported RTSP-server type: 0 – RTSP server based on VLC; 1 – BitBand RTSP server; 2 – Kasenna RTSP server; 3 – ARRIS (C-COR) RTSP server; 4 – Live555 RTSP server. The servers ARRIS and Live555 are

Parameter	Allowed value	Description
		supported only for MAG200.
flags	0..0x3f	Control flags: 1 – switch on the keep-alive mode; 2 – determination of the stream end by the field **x-notice** in the message ANNOUNCE from the server 4 – determination of te stream end by the field x-notice in the answer to GET_PARAMETER; 8 – determination of the stream end after a period of time of the video stream from the server absence; 16 (0x10) – determination of the stream end by the field according to the field rtptime sent in the RTP heading of the package (Only for the mode of sending video under RTP); 32 (0x20) – Use UDP transport to send video.

Returned value:

Нет.

stb.SetViewport

void SetViewport(int xsize, int ysize, int x, int y)

Set the location and size of the video window.

Parameters:

Parameter	Allowed value	Description
xsize	Depends on the screen resolution.	Horizontal size of the video window (width).
ysize		Vertical size of the video window (height).
x	Must not exceed the screen width in sum with cxsize	Left upper corner of the video window horizontal offset from the screen edge.
y	Must not exceed the scxreen width in sum with ysize .	Left upper cornet of the video window vertical offset from the screen edge.

Returned value:

None.

stb.IsPlaying

FireFox: void IsPlaying(**out** bool bPlaying);

WK/FF+Wrapper: bool IsPlaying()

Receive the current state of display:

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
bPlaying	true, false	Current state of display: false – currently the content is not displayed; true – currently the content is displayedв .

stb.Version

FireFox: void Version(**out** string version);

WK/FF+Wrapper: string Version();

Receive API version

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
version		The string in ten form opf: “JS API version: <JS_API version>; STB API version: <STB_API version>; Player Engine version: <Player version>”. **JS_API version** – this API version number; **STB_API version** – player API version **Player version** – version of the player used in API in API in HEX code виде.
Example: “JS API version: 301; STB API version: 104; Player Engine version: 0x23”		

stb.SetupSPdif

void SetupSPdif(int flags);

Set the mode of sound output through SPdif

Parameters:

Parameter	Allowed value	Description
flags	0..2	Output mode through SPdif: 0 – the sound is supplied only to analogue output. 1 – sound is supplied to analogue output and through SPdif in tne format 2- channel PCM 2 – sound is supplied to SPdif without decoding(AC3 …), if supported by codec, othjerwise through SPdif in te format of 2-channel PCM .

Returned value:

None.

stb.SetSubtitles

void SetSubtitles(bool enable);

Subtitle on/off.

Parameters:

Parameter	Allowed value	Description
Enable	true, false	true – subtitles on; false – subtitles off.

Returned value:

None.

Notes:

For MAG100 subtitles are displayed in full screen mode.

stb.SetSubtitlesSize

void SetSubtitlesSize(int size);

Set the size of text subtitles – size in pixels.

Platforms: MAG100, MAG200.

Parameters:

Parameter	Allowed value	Description
size		Set the size of text subtitles.

Returned value:

None.

stb.SetSubtitlesFont

void SetSubtitlesFont(string font);

Set the font for displaying text subtitles.

Platforms: MAG100, MAG200.

Parameters:

Parameter	Allowed value	Description
font	URL-адрес	URL addressing the font file in the root file system. For example: “/home/default/arial.ttf”

Returned value:

None.

stb.SetSubtitlesOffs

void SetSubtitlesOffs(int offs);

Set the offset for displaying text subtitles.

Platforms: MAG100, MAG200.

Parameters:

Parameter	Allowed value	Description
offs		Horizontal offset of subtitles.

Returned value:

None.

stb.GetSpeed

FireFox: void GetSpeed(**out** int speed);

WK/FF+Wrapper: int GetSpeed();

Receive the current speed of display

Parameters:

Parameter	Allowed value	Description
speed	-8..8	Current speed of display: 1 - normal 2 - 2x 3 - 4x 4 - 8x 5 - 16x 6 - 1/2 7 - 1/4 8 – 12x

Parameter	Allowed value	Description
		0 – stop or pause -1 – reverse -2 - reverse 2x -3 - обратное воспроизведение 4x -4 - reverse 8x -5 - reverse 16x -8 – reverse 12x

Returned value.

None.

stb.GetAudioPID

FireFox: void GetAudioPID(**out** int pid);

WK/FF+Wrapper: int GetAudioPID();

Receive the number (PID) of the current audio track.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
pid	0..0x1fff	Current audio track number.

Notes:

The list of all audio tracks determined by the player can be received with stb.GetAudioPIDs.

stb.GetSubtitlePID

FireFox: void GetAudioPID(**out** int pid);

WK/FF+Wrapper: int GetAudioPID();

Receive the number (PID) of the current subtitles track.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
pid	0..0x1fff	Current subtitles track number.

Notes:

The list of all subtitles track determined by the player can be received with stb.GetSubtitlePIDs.

stb.GetPIG

FireFox: void GetPIG(**out** bool isWindowed);

WK/FF+Wrapper: bool GetPIG();

Receive the video window state:

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
isWindowed	true, false	The resupt specifies whether full screen mode is set for the video window: true – the content is displayed in a reduced window; false – the content is displayed in a full screen mode.

stb.GetAlphaLevel

FireFox: void GetAlphaLevel(**out** int alpha);

WK/FF+Wrapper: int GetAlphaLevel();

Receive the video window alpha transparency level.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
alpha	0..255	Returne the current value of alpha transparency for the video window.

stb.GetWinAlphaLevel

FireFox: void GetWinAlphaLevel(int winNum, **out** int alpha);

WK/FF+Wrapper: int GetWinAlphaLevel(int winNum);

Receive the level of alpha transparency for the set window

Parameters:

Parameter	Allowed value	Description
winNum	0..1	Number of the window for which this function is used: 0 – graphic window; 1 – video window.

Returned value:

Parameter	Allowed value	Description
Alpha	0..255	Returns the current value of alpha transparency for video window.

stb.SetTransparentColor

void SetTransparentColor(int color);

Sets the colour considered transparent at the moment:

`Parameters`:

Parameter	Allowed value	Description
Color	0..0xffffff	Colour in RGB format that can be considered transparent.

Returned value:

None.

Notes:

The function is a special case of stb.SetChromaKey.

Any changes on the screen are visible only provided the ChromaKey mode is switched on by functions **stb.SetMode or stb.SetWinMode**.

stb.GetTransparentColor

FireFox: void GetTransparentColor(**out** int color);

WK/FF+Wrapper: int GetTransparentColor();

Returns the colour considered transparent at the moment:

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
Color	0..0xffffff	The colour in RGB format considered transparent at the moment

stb.IgnoreUpdates

void IgnoreUpdates(bool bIgnore);

Blocks or unblocks the screen browser upgrade:

Parameters:

Parameter	Allowed value	Description
bIgnore	true, false	**true** – after this call the graphic window stops upgrading till the next call with the parameter **false**; **false** – after this call the graphic window resumes upgrading – passing to normal mode.

Returned value:

None.

stb.ExecAction

void ExecAction(string action);

Perform the script /home/default/action.sh with the parameters set.

Parameters:

Parameter	Allowed value	Description
Action		String contains parameters for the script /home/default/action.sh. Examle: stb.ExecAction('param 23 s') calls sh command from the shell "/home/default/action.sh param 23 s"

Returned value:

None.

stb.SetCASType

void SetCASType(int CAS_type);

Set default access server type after each start of the portal.

Platforms: MAG100,MAG200

Parameters:

Parameters	Allowed velue	Description
Type	0,1	0 – not set; 1 – Verimatrix.

Returned value:

None.

Notes:

Set default server type once after each start of the portal.

stb.SetCASParam

void SetCASParam(string serverAddr, int serverPort, string CompanyName, int opID, int errorLevel);

Set CAS server parameters:

Platforms: MAG100,MAG200.

Parameters:

Parameter	Allowed value	Description
serverAddr		CAS server URL.
serverPort		CAS server port.
companyName		Name of the company under which this operator is registered on CAS server.
opID	1..255	Operator identifier used by STB. If opID is equal to -1, the value is not updated.
errorLevel	0..5	Level of error. 0 – minimal level.

Parameter	Allowed value	Description
		If error Level equals to -1, it is not updated.

Returned value

None.

Notes:

Call of the function becomes effective only if made before stb.SetCASType.

stb.LoadCASIniFile

void LoadCASIniFile(string iniFileName);

Load CAS settings from the set file.

Platforms: MAG100,MAG200.

Parameters:

Parameter	Allowed value	Description
iniFileName		URL of the settings file in the root file systemф.

Returned value:

None.

Notes:

See instruction on adjusting CAS Verimatrix in the supplement.

The call of the function becomes effective only if made before stb.SetCASType.

stb.SetCASDescrambling

void SetCASDescrambling(int isSoftware);

Set hard or soft mode of descrambling.

PLatforms: MAG100,MAG200.

Parameters:

Parameter	Allowed value	DESCRIPTION
isSoftware	0,1	0 –use hard descrambling. 1 – use soft descrambling. In the absence of this call soft descrambling is used.

Parameter	Allowed value	DESCRIPTION
		Only soft descrambling can be used for MAG100.

Returned value:

None.

Notes:

At present the use of the function is expedient only for CAS **Verimatrix**. Depending on the mode set, the player can descramble only the streams scrambled by the following algorithm.:

Soft mode: RC4, AES;

Hard mode: AES, DVB-CSA.

This mode is set only once after the start of the portal.

The call of the function becomes effective only if it is called before stb.SetCASType.

stb.GetAspect

FireFox: void GetAspect(**out** int aspect);

WK/FF+Wrapper: int GetAspect();

Return the current video content format.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
aspect		Returns the current format of video content. Consists of 2 tetrads: 7 6 5 4 3 2 1 0 **aspH** | **aspL** **ForMAG100 aspH is always equal to 0.**
aspL	0..3	Sets aspect ratio: 0 – automatically; 1 – 20:9; 2 –16:9;

Parameter	Allowed value	Description
		3 – 4:3.
aspH	0..3	Sets video format conversion: 0 – as it is., the video is stretched to the whole screen; 1 – Letter Box mode, the video is proportionally enlarged to the screen size along the larger edge; 2 – Pan&Scan mode, the video is proportionally enlarged to the screen size along the shorter edge; 3 – combined mode intermediate between Letter Box and Pan&Scan. 4 – enlarged mode; 5 – optimal mode. Only for MAG200

stb.StandBy

void StandBy(bool bStandby);

Enter or exit StandBy mode .

Parameters:

Parameter	Allowed value	Description
bStandby	true, false	true – enter Standby mode; false – exit from Standby mode.

Returned value:

None.

Notes:

When entering StandBy mode the following operations take place:

1. All video outputs switch off.
2. Content display, if it was on, stops.

stb.RDir

FireFox: void RDir(string par, **out** string result);

WK/FF+Wrapper: string RDir(string par);

Perform script **/home/default/rdir.cgi** with set parameters and return the standard output of this script.

Parameters:

Parameter	Allowed value	Description
par	Any string	The string contains parameters with which the script is started /home/default/rdi.cgi.

Returned value:

Parameters	Allowed value	Description
result		Standard output received when performing the script /home/default/rdi.cgi with parameters set.

Notes:

The **rdir.cgi** supplied with the root file system has several commands preset**:**

stb.RDir("SerialNumber",x) – **x** returns serial number of this device to **x**.

stb.RDir("MACAddress",x) - receive MAC address

stb.RDir("IPAddress",x) - receive IP addressадрес

stb.RDir("HardwareVersion",x) – receive hardware version

stb.RDir("Vendor",x) – receive the name of STB manufacturer

stb.RDir("Model ",x) – receive the name of STB pattern

stb.RDir("ImageVersion",x) – receive the version of the software flash

imagestb.RDir("ImageDescription",x) – receive the information on the image of the software flash

stb.RDir("ImageDate",x) – receive the date of creation of the flash software image.

stb.RDir("getenv v_name",x) – receive the value of environment variable with the name **v_name**. See detailed description of operations with environment variables in supplement 11.

stb.RDir("setenv v_name value") – set environment variable with the name **v_name** to the value **value**. See detailed description of operations with environment variables in supplement 11.

SetAudioLangs

void SetAudioLangs(string priLang, string secLang);

Set languages of audio tracks to be automatically selected when receiving the information on the channel.

Parameters:

Parameter	Allowed value	Description
priLang	3 –symbol tags according to ISO 639, For example: “rus” или “eng”	If the information of several audio tracks is present the player selects the track preset by the language priLang. If such track is not found, the track with the language secLang is selected. If this one is not found either the first track from the list is selected.
secLang		

Returned value:

None.

stb.SetSubtitleLangs

void SetSubtitleLangs(string priLang, string secLang);

Set the languages of subtitles tracks to be automatically selected when receiving the information on the channel.

Parameters:

Parameter	Allowed value	Description
priLang	3 –symbol tags according to ISO 639, For example: “rus” или “eng”	If the information of several audio tracks is present the player selects the track preset by the language priLang. If such track is not found, the track with the language secLang is selected. If this one is not found either the first track from the list is selected.
secLang		

Returned value:

None.

stb.GetAudioPIDs

FireFox: void GetAudioPIDs(**out** string pidsList);

WK/FF+Wrapper: string GetAudioPIDs();

The function returns the list of audio tracks in the stream with the description of the language.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
pidsList		List of the audio tracks found in the following format: **[{pid:<***PID1***>, lang:[<***lang1_1***>, <***lang2_1***>]}, … , {pid:<***PIDn***>, lang:[** <*lang1_n*>**, <***lang2_n***>]}]**
PIDn		*PID of audio track with the number n*.
lang1_n lang2_n	3-symbol tags according to ISO 639	First two descriptions of languages in audio track with the number *n*.

Example: the result in the form:

[{pid:114, lang:["rus", "ru"]}, {pid:115, lang:["eng", ""]}]

Means that 2 audio streams were found in the stream: Russian having PID=114 and English having PID=115;

Notes:

This stream can be easily converted into a structure array by calling the function **eval()**.

This function must be called after the event having the code 2 occurs (see description of events)

stb.GetSubtitlePIDs

FireFox: void GetSubtitlePIDs(**out** string pidsList);

WK/FF+Wrapper: string GetSubtitlePIDs();

The function returns the list of subtitles track in the stream with the description of the language.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
pidsList		List of subtitles tracks found in the following format: **[{pid:<***PID1***>, lang:[<***lang1_1***>, <***lang2_1***>]}, … , {pid:<***PIDn***>, lang:[** <*lang1_n*>**, <***lang2_n***>]}]**
PIDn		*PID of subtitle track with the number n*.
lang1_n lang2_n	3-symbol tags according to ISO 639	First two descriptions of languages in subtitle track with the number *n*.

Example: the result in the form:

[{pid:114, lang:["rus", "ru"]}, {pid:115, lang:["eng", ""]}]

means that 2 subtitle streams were found in the stream: Russian having PID=114 and English having PID=115;

Notes:

This string can be easily converted into a structure array by calling the function **eval()**. This function must be called after the event having the code 2 occurs (see description of events)

stb.ReadCFG

FireFox: void ReadCFG(**out** string result);

WK/FF+Wrapper: string ReadCFG();

Read the file of portal settings /etc/stb_params.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
result		Returns the contents of the file /etc/stb_params.

stb.WriteCFG

void WriteCFG(string cfg);

Read the file of portal settings /etc/stb_params.

Parameters:

Parameter	Allowed value	Description
Cfg		The data to be stored in the file /etc/stb_params.

Returned value:

None.

Notes:

It must be kept in mind that the values PORTAL_IP, PORTAL_1, PORTAL_2 are used in the starting portal stored in /home/web of the root file system, therefore it is desirable to receive source values of these parameters via stb.ReadCFG before making the call and add them to the string cfg.

stb.WritePrefs

void WritePrefs(string prefs);

Save the string as the browser set up (prefs.js).

Parameters:

Parameter	Allowed value	Description
prefs		Data to be saved in the file of browser settings.

Returned value:

None.

Notes:

This function is not browser specific and it is used to set the right of access to the portal. This is performed in starting portal saved at /home/web of the root file system and it is recommended to avoid using it anywhere else.

stb.Debug

void Debug(string debugString);

Show the contents of the string **debugString** in the stream of standard output in the format:

DEBUG: <time> **debugString**

Parameters:

Parameter	Allowed value	Description
debugString		This string is shown in the stream of standard output.

Returned value:

None.

stb.SetListFilesExt

void SetListFilesExt (string fileExts);

Set the list of file extensions for returning to the function.

Parameters:

Parameter	Allowed value	Description
fileExts		List of files extensions followed by a space. For example: “.mkv .mov .mpg”

Returned value:

None.

Notes:

This function is realized only for the browser based on WebKit.

stb.ListDir

string ListDir (string dirName);

Returns the list of directories and files having the extension set with SetListFilesExt, located in the directory **dirName**.

Parameters:

Parameter	Allowed value	Description
dirName		Route to the directory the contents whereof must be received.

Returned value:

The string in the following form is returned:

```
var dirs = [
"dir1/",
…
"dirn/",
""
]
var files = [
{"name" : "fileName1", "size" :size1},
…
{"name" : "fileNamem", "size" :sizem},
{}
]
```

Where dirn – the name of n-sub-directory,

fileNamem and sizem – name and size of m-file.

Notes:

This function is realized only for the browser based on WebKit. For browsers based on FireFox such function can be realized using the function RDir with the parameter “rdir”.

stb.SetBrightness

void SetBrightness (int bri);

Set the brightness of video output in SD mode.

Parameters:

Parameter	Allowed value	Description
Bri	1..254	Brightness in the SD mode.

Returned value:

None.

Notes:

This function is realized only for the browsers based on WebKit.

stb.SetSaturation

void SetSaturation (int sat);

Set the saturation of video output in SD mode.

Parameters:

Parameter	Allowed value	Description
Sat	1..254	Saturation of video output in SD mode.

Returned value:

None.

Notes:

This function is realized only for the browser based on WebKit.

stb.SetContrast

void SetContrast (int con);

Set contrast of video output in SD mode.

Parameters:

Parameter	Allowed value	Description
Con	-128..127	Video output contrast in SD mode

Returned value:

None.

Notes:

This function is realized only for the browser based on WebKit.

stb.GetBrightness

int GetBrightness ();

Receive current brightness of video output in SD.

Parameters:

None.

Returned value:

Parameter	Allowed value	Description
Bri	1..254	Brightness of video output in SD mode

Notes:

This function is realized only for the browser based on WebKit.

stb.GetSaturation

int GetSaturation ();

Receive current saturation of video output in SD mode

Parameters:

None.

Возвращаемое значение:

Parameter	Allowed value	Description
Sat	1..254	Saturation of video output in SD mode

Notes:

This function is realized only for the browser based on WebKit.

stb.GetContrast

void GetContrast (int con);

Receive current contrast of video output in SD mode

Parameters:

None

Returned value

Parameter	Allowed value	Description
Con	-128..127	Contrast of video output in SD mode

Notes:

This function is realized only for the browser based on WebKit.

stb.DeleteAllCookies

void DeleteAllCookies ();

Delete all cookie saved by the browser.

Parameters:

None.

Возвращаемое значение:

None.

Notes:

This function is realized only for the browser based on WebKit.

Event model in JavaScript.

Event model in JavaScript assumes the possibility for API user to receive the events indicating some changes of the player playback state.

Configuring the event system

To configure event system proceed as follows:

1. Include the script event.js in the portal:

<script language="JavaScript" src="event.js"></script>

This script can be taken from /home/web/ directory of the root file system nfs-image.

Notes:

The contents of this script **must** be in the global scope.

For the browser based on WebKit the declaration of the stbEvent object in the global scope is sufficient instead of including this script:

```
var stbEvent=
{
   onEvent : function(data){},
   event : 0
}
```

2. after the initialization of the player (see appendix 1) call **initEvents()** function

3. the function to be called when getting the event must be set through **stbEvent** object:

stbEvent.onEvent = EventCallback,

where EventCallback – is the function used for processing the events in the portal with the event code as the parameter.

For example:

```
function EventCallback(event) {_debug('event '+event)}
```

4. The code of the last event is also stored in the **stbEvent.event**.

List of the events used

The following events are defined:

Event code события	Description
1	The player reached the end of the media content or detected a discontinuity of the stream.
2	Information on audio and video tracks of the media content is received.
4	Video and/or audio playback has begun.
5	Error when opening the content: content not found on the server or connection with the server was rejected.
0x81	When playing RTP-stream the numbering of RTP-packets was broken.

Appendix 1. API usage.

stb object initialization.

First of all the main object **stb** must be created. Proceed as follows:

1. Declare object **stb**:
   var stb;
2. Initialize **stb** in the page initialization function with the following lines:

```
netscape.security.PrivilegeManager.enablePrivilege("UniversalXPConnect")
const cid = "@mydomain.com/XPCOMSample/MyComponent;1"
stb = Components.classes[cid].createInstance()
stb = stb.QueryInterface(Components.interfaces.IMyComponent)
```

These lines created **stb** object but the player, so any media content cannot be played back at this stage and developer can use just a few set of function such as stb.RDir. Because only one player instance can be initialized simultaneously this mode is used only for auxiliary pages such as /home/web/index.html.

The following functions can be called in this mode:

stb.Version,
stb.ExecAction,
stb.RDir,
stb.ReadCFG,
stb.WriteCFG,
stb.WritePrefs,
stb.InitPlayer.

Player initialization

For using all API functions initialize the player with stb.InitPlayer function. Only one player can be initialized during a certain period of time. To initialize another player (for example on another page) first call stb.DeinitPlayer for the player that had been already initialized.

Specifics of JavaScript API >= 308 versions.

Beginning from the version JavaScript API 308 the initialization scheme described above can be used or the strings:

```
const cid = "@mydomain.com/XPCOMSample/MyComponent;1"
stb = Components.classes[cid].createInstance()
stb = stb.QueryInterface(Components.interfaces.IMyComponent) in item 2)
```

can be replaced by:

```
stb = gSTB
```

Besides, beginning from version 308 the possibility of repeated call of stb.InitPlayer appeared; in this situation the player will be initialized when this function is called for the first time, and it will be de-initialized after the exit from the browser.

Player initialization (Version JavaScript API >= 308).

To use all API functions initialize the player with stb.InitPlayer function. Only one player can be initialized at a time. To initialize another player (for example on another page) first call stb.DeinitPlayer for the player that had been already initialized

Wrapper.js

For those programmers who do not want to call the function

```
netscape.security.PrivilegeManager.enablePrivilege("UniversalXPConnect")
```

in every function that uses **stb** object, an auxiliary script wrapper.js, was written, which allows calling the methods of object stb from any place of JS code without setting privileges.

To include this script:

1. include this script at the very beginning
   <script language="JavaScript" src="wrapper.js"></script>
2. comment out the strings:
   var stb
   stb=gSTB

   const cid = "@mydomain.com/XPCOMSample/MyComponent;1"
   stb = Components.classes[cid].createInstance()
   stb = stb.QueryInterface(Components.interfaces.IMyComponent)
   if they are present in the main script of the page.

Thereafter **stb** object appears in the global scope of the script, which allows calling **stb** object methods without setting privileges.

Event system initialization

This item is described in detail in the Event model in JavaScript chapter.

API usage example.

The page below shows minimal HTML code of the page, which is simply loaded and starts playing rtp stream and stops playing or restarts playing the stream with the buttons **stop** and **continue** correspondingly.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
	<head>
		<meta http-equiv="Content-Type" content="text/html; charset=windows-1251">
		<title></title>
		<script language="JavaScript" src="event.js"></script>
		<script>
			var stb
			function init(){
				netscape.security.PrivilegeManager.enablePrivilege("UniversalXPConnect")
				const cid = "@mydomain.com/XPCOMSample/MyComponent;1"
				stb = Components.classes[cid].createInstance()
				stb = stb.QueryInterface(Components.interfaces.IMyComponent)
				stb.InitPlayer()
				stb.Play('rtp rtp://224.10.0.123:1234')
			}
			function getkeydown(e) {
				netscape.security.PrivilegeManager.enablePrivilege("UniversalXPConnect")
				ec = e.keyCode
				ew = e.which
				es = e.shiftKey
				pat = /^(\S+)_(\S+)/
				switch (ew){
					case 114: // Play
					{
						stb.Play('rtp rtp://224.10.0.123:1234')
						break;
```

```
                                        }
                                        case 115: // Stop
                                        {
                                                stb.Stop()
                                                break;
                                        }
                                }
                        }
                </script>
        <body onload="init()" onKeyPress="getkeydown(event)">
        </body>
</html>
```

Besides, the text page, which can be used for checking and seeing the operation of all API functions, is contained in the root file system for MAG200 in the folder /home/web/ ..

Appendix 2. Video content formats and examples of use.

This appendix describes the types of the content played back and their use.

Playback can be started by two functions: stb.Play and stb.PlaySolution. The parameters of stb.PlaySolution function are included in the complex parameter **playStr** of the function stb.Play, therefore further description shall be based on the example stb.Play function and on the parameters of this function.

stb.Play function parameters format.

playStr has the following format:

"solution URL [atrack:num] [vtrack:num] [strack:num] [position:time]", where

solution

The type of media content, which determines the content format, for example media container type and/or the method of broadcasting.

Type (**solution**)	MAG100	MAG200	Description
auto	+	+	Automatic detection of the type of content, container, codec by the givern URL. **Distinctions:** **MAG100**: when using automatic defining URLs that begin with rtp://, udp://, rtsp:// cannot be used, i.e.. automatic definition is used only with files. **MAG200**: correctly accepts the URLs beginning with rtp://, udp://, rtsp://.
"" (empty)	+	+	Similar to **auto**.
rtp	+	+	Play the string in the format MPEG2TS. If **URL** begins with rtp:// , the RTP stream shall be played, if it begins with udp:// , the UDP stream shall be played. **Distinctions:** **MAG100** automatically connects decoder for MPEG2 Video and MPEG Audio. **MAG200** set required codecs if addional

Type (**solution**)	MAG100	MAG200	Description
			information of stream is present, for example, H.264, AC-3 и т.д.
rtsp	+	+	Play the content from RTSP-server. **Distinctions:** **MAG100** automatically connects decoder for MPEG2 Video and MPEG Audio. **MAG200** set required codecs if additional information on the stream is present, for example, H.264, AC-3 ,etc.
rtpac3	+	+	Play the stream in the format MPEG2TS using decoders MPEG2 Video and AC-3 Audio
rtsp_ac3	+	+	Play content from RTSP-server using decoders MPEG2 Video and AC-3 Audio
rtpmpeg4	+	–	Play the stream in the format MPEG2TS using decoders MPEG4p2 Video and MPEG Audio
rtpmpeg4_aac	+	–	Play the stream in the format MPEG2TS using decoders MPEG4p2 Video and AAC Audio
mpegts	+	+	Play the file in the format MPEG2TS. **Distinctions:** **MAG100** automatically connects decoder for MPEG2 Video and MPEG Audio. **MAG200** sets required codecs if additional information on the stream is present, for example H.264, AC-3 , etc.
mpegps	+	+	Play file in the format MPEG2 Program Stream. **Distinctions:** **MAG100** automatically connects decoder for MPEG2 Video and MPEG Audio. **MAG200** sets required codecs if additional information on the stream is present, for example, AC-3 , etc
file	–	+	Play file at the URL set with automatic determining

Type (**solution**)	MAG100	MAG200	Description
			content type, container and codec.
mp4	+	–	Play file in MP4 format using codecs MPEG4p2 Video and AAC Audio
mp4_mpa	+	–	Play file in MP4 format using codecs MPEG4p2 Video and MPEG Audio
fm	+	+	Play audio from MPEG-TS of the (udp://, rtp://)
ffmpeg	–	+	MAG200 allows playing files in various formats: avi, mkv, mpg, mp4, mov, wmv, AC-3, mp3.

Notes

In contrast to MAG100, MAG200 can determine and change codecs during a playback, for example, when audio tracks compressed by different codecs are present.

URL

Specifies where the content is stored:

URL format	Description
/path	Local address of the file in the root file system, for example, /media/1.mp3
rtp://addr:port	RTP-stream received from multicast or unicast address ***addr*** and the port ***port***.
udp://addr:port	UDP-stream received from multicast or unicast address ***addr*** and the port ***port***.
rtsp://addr:port/path	The content stored in the RTSP-server at the address ***addr*** and port ***port*** with the relative route ***path***

atrack, vtrack и strack

Optional parameters setting the numbers of audio, video tracks and subtitle tracks (PID-s for MPEG2TS) of the content to be played.

position

Optional parameter setting the time **time** in seconds, from which the content is to be played.

Examples:

“mpegps /media/1.mpg” – plays the file Mpeg2 Program Stream файл /media/1.mpg.

“mpegts /media/1.mpg” – plays the file Mpeg2 Transport Stream файл /media/1.mpg.

“mp4 /media/1.mp4” – plays the file /media/1.mp4 in the format MP4.

“rtp 224.10.0.30:5004” – plays Mpeg2 in the format Transport Stream from the preset multicast address (224.10.0.30) and port (5004) using IGMP protocol for multicast broadcast.

“auto /media/file1” – an attempt to automatically determine the file format and play it.

“rtpmpeg4 224.10.0.31:5004” – plays Mpeg4 in the format Transport Stream from the preset multicast address (224.10.0.31) and port (5004) with Mpeg2 Audio using IGMP protocol for multicast broadcast.

“rtpmpeg4_aac 224.10.0.32:5004 atrack:930 vtrack:920” – plays Mpeg4 video in the format Transport Stream from the preset multicast address (224.10.0.32) and port (5004) with AAC audio using IGMP protocol for multicast broadcast. In this situation the stream with PID=920 is automatically selected as the video track and the stream with PID=930 – as the audio track, irrespective of the presence of information on the tracks in the stream.

“rtsp rtsp://192.168.1.32:554/video/media003.mpg” – plays the content /video/media003.mpg, located on RTSP-server with the address 192.168.1.32 and port 554.

Appendix 3. Verimatrix CAS usage and settings

For using Verimatrix CAS proceed as follows:

1. Store the rootcert.pem certificate to the /flash/ folder.
2. Set the correct time, for example, from ntp server.
3. Set initial parameters of the CAS server using one of the two methods:
    a. With LoadCASIniFile function, when the parameters are automatically taken from the specified file.
    b. With SetCASParam function.
    c. For MAG200 set descrambling mode using stb.SetCASDescrambling function.
4. Set the type of the CAS server after setting initial parameters.

Appendix 4. Specifics of JS API when using the browser based on WebKit.

Initialization.

To initialize **stb** object proceed as follows:

1. Declare the object **stb**:
   var stb;
2. Initialize **stb** in the page initialization function with the following string:
   stb = gSTB;

This method obviously corresponds with the new method used for Mozilla Firefox, as described above.

Also make sure that the lines in the file event.js

```
observerService = Components.classes["@mozilla.org/observer-
service;1"].getService(Components.interfaces.nsIObserverService);
observerService.addObserver(myObserver, "TeletecSTB", false);
```

are replaced with the following lines:

```
try
{
  observerService = Components.classes["@mozilla.org/observer-
service;1"].getService(Components.interfaces.nsIObserverService);
  observerService.addObserver(myObserver, "TeletecSTB", false);
}catch(e)
{}
```

(it has already been done for the default nfs image.)

Wrapper.js

JS API for WebKit provides to the user (without the necessity of connecting **wrapper.js**) with the same interface as wrapper.js, i.e.:

1. No need to continuously call
   netscape.security.PrivilegeManager.enablePrivilege("UniversalXPConnect")
2. If the function returns the value, it can be received in a usual manner, for example :

```
var tColor = stb.GetTransparentColor();
```

In this case the required prototype of the method is denoted using **WK/FF+Wrapper** string according to the Object stb methods calling chapter.

Cookie

When setting a cookie, as opposite to the code of Mozilla Firefox

```
function set_coookie(str)
{
    document.cookie = str
}
```

Add path=/ :

```
function set_coookie(str)
{
    document.cookie = str+'; path=/;'
}
```

If the correct expiry date of the cookie (”expiry=”), is specified, the browser shall save the cookie into /flash directory. That is, the cookie shall be valid till the term expires or stb.DeleteAllCookies function is called.

Use of alpha-transparency.

To create transparent or semi-transparent applications based on WebKit browser add the following attribute to BODY:

background-color: none;

For setting transparency use **opacity** attribute or set the colour to **transparent**.

Alpha-transparency shall be functioning only if the following mode is specified in /etc/directfbrc:

```
pixelformat=ARGB
depth=32
bg-color=0
#bg-none
```

If alpha-transparency is not required (when ChromaKey is sufficient), 16-bit mode can be set by changing the mode in /etc/directfbrc to:

pixelformat=RGB16

depth=16

bg-none

#bg-color=0

The performance of graphic sub-system will be increased in this mode, while the memory load will be reduced.

Appendix 5. Remote control key codes in JavaScript.

Remote control key codes sent to JavaScript are completely determined by the settings of the program irxevent. Mozilla Firefox uses the file/etc/lirc/lircrc, while WebKit uses the file /etc/lirc/lircrc.wk.

The table of key codes for MAG100/MAG200 (release version <= 0.1.4)

Table below shows the key codes received by the events processor of JavaScript:

Table 2 Remote control keys codes (version <= 0.1.4)

Remote key control key	keyCode, dec(hex)	which, dec(hex)	Flags
exit	27 (0x1b)	0	
ok	13 (0x0d)	0	
right	39 (0x27)	0	
left	37 (0x25)	0	
up	38 (0x26)	0	
down	40 (0x28)	0	
PageUp	33 (0x21)	0	
PageDown	34 (0x22)	0	
menu	122 (0x7a)	0	
back	8 (0x08)	0	
refresh	116 (0x74)	0	
red	112 (0x70)	0	
green	113 (0x71)	0	
yellow	114 (0x72)	0	
blue	115 (0x73)	0	
channel+	9 (0x09)	0	
channel–	9 (0x09)	0	SHIFT
service	120 (0x78)	0	
tv	121 (0x79)	0	
phone	119 (0x77)	0	
web	123 (0x7b)	0	
frame	117 (0x75)	0	

Remote key control key	keyCode, dec(hex)	which, dec(hex)	Flags
vol+	0	43 (0x2b)	
vol–	0	45 (0x2d)	
rew	0	98 (0x62)	CTRL, ALT *
ffwd	0	102 (0x66)	CTRL, ALT *
stop	0	115 (0x73)	CTRL, ALT *
play/pause	0	114 (0x72)	CTRL, ALT *
rec	0	119 (0x77)	CTRL, ALT *
mic	0	32 (0x20)	CTRL, ALT *
mute	0	96 (0x60)	CTRL, ALT
power	0	117 (0x75)	CTRL, ALT
info	0	121 (0x79)	CTRL, ALT *
Empty	0	107 (0x6b)	CTRL, ALT *
1	0	49 (0x31)	
2	0	50 (0x32)	
3	0	51 (0x33)	
4	0	52 (0x34)	
5	0	53 (0x35)	
6	0	54 (0x36)	
7	0	55 (0x37)	
8	0	56 (0x38)	
9	0	57 (0x39)	
0	0	48 (0x30)	
The event generated when connecting USB Flash Drive	0	112 (0x70)	CTRL, ALT *
The event generated when disconnecting USB Flash Drive	0	113 (0x71)	CTRL, ALT *

The table of keys codes for MAG200 (RELEASE VERSION > 0.1.4)

The table for event processor onKeyPress

The table below shows the keys codes received by the event processor of JavaScript onKeyPress for the browsers Mozilla Firefox and WebKit:

Table 3 Remote control keys codes for the processor onKeyPress

Remote control key	Browser	keyCode, dec(hex)	which, dec(hex)	Flags
Exit	FF	27 (0x1b)	0	
	WK	27 (0x1b)	27 (0x1b)	
Ok	**FF**	**13 (0x0d)**	**0**	
	WK	**13 (0x0d)**	**13 (0x0d)**	
right	FF	39 (0x27)	0	
	WK	39 (0x27)	39 (0x27)	
left	FF	37 (0x25)	0	
	WK	37 (0x25)	37 (0x25)	
Up	FF	38 (0x26)	0	
	WK	38 (0x26)	38 (0x26)	
down	FF	40 (0x28)	0	
	WK	40 (0x28)	40 (0x28)	
PageUp	FF	33 (0x21)	0	
	WK	33 (0x21)	33 (0x21)	
PageDown	FF	34 (0x22)	0	
	WK	34 (0x22)	34 (0x22)	
menu	FF	122 (0x7a)	0	CTRL
	WK	122 (0x7a)	122 (0x7a)	CTRL
back	FF	8 (0x08)	0	
	WK	8 (0x08)	8 (0x08)	
refresh	FF	116 (0x74)	0	CTRL
	WK	116 (0x74)	116 (0x74)	CTRL
red	FF	112 (0x70)	0	CTRL
	WK	112 (0x70)	112 (0x70)	CTRL
green	FF	113 (0x71)	0	CTRL

Remote control key	Browser	keyCode, dec(hex)	which, dec(hex)	Flags
	WK	113 (0x71)	113 (0x71)	CTRL
yellow	FF	114 (0x72)	0	CTRL
	WK	114 (0x72)	114 (0x72)	CTRL
blue	FF	115 (0x73)	0	CTRL
	WK	115 (0x73)	115 (0x73)	CTRL
channel+	FF	9 (0x09)	0	
	WK	9 (0x09)	9 (0x09)	
channel–	FF	9 (0x09)	0	SHIFT
	WK	9 (0x09)	9 (0x09)	SHIFT
service	FF	120 (0x78)	0	CTRL
	WK	120 (0x78)	120 (0x78)	CTRL
Tv	FF	121 (0x79)	0	CTRL
	WK	121 (0x79)	121 (0x79)	CTRL
phone	FF	119 (0x77)	0	CTRL
	WK	119 (0x77)	119 (0x77)	CTRL
web	FF	123 (0x7b)	0	CTRL
	WK	123 (0x7b)	123 (0x7b)	CTRL
frame	FF	117 (0x75)	0	CTRL
	WK	117 (0x75)	117 (0x75)	CTRL
vol+	FF	0	43 (0x2b)	
	WK	43 (0x2b)	43 (0x2b)	
vol–	FF	0	45 (0x2d)	
	WK	45 (0x2d)	45 (0x2d)	
rew	FF	0	98 (0x62)	ALT
	WK	98 (0x62)	98 (0x62)	ALT
ffwd	FF	0	102 (0x66)	ALT
	WK	102 (0x66)	102 (0x66)	ALT
stop	FF	0	115 (0x73)	ALT
	WK	115 (0x73)	115 (0x73)	ALT
play/pause	FF	0	114 (0x72)	ALT
	WK	114 (0x72)	114 (0x72)	ALT
rec	FF	0	119 (0x77)	ALT
	WK	119 (0x77)	119 (0x77)	ALT

Remote control key	Browser	keyCode, dec(hex)	which, dec(hex)	Flags
mic	FF	0	32 (0x20)	ALT
	WK	32 (0x20)	32 (0x20)	ALT
mute	FF	0	96 (0x60)	ALT
	WK	96 (0x60)	96 (0x60)	ALT
power	FF	0	117 (0x75)	ALT
	WK	117 (0x75)	117 (0x75)	ALT
info	FF	0	121 (0x79)	ALT
	WK	121 (0x79)	121 (0x79)	ALT
Empty	FF	0	108 (0x6c)	ALT
	WK	108 (0x6c)	108 (0x6c)	ALT
1	FF	0	49 (0x31)	
	WK	49 (0x31)	49 (0x31)	
2	FF	0	50 (0x32)	
	WK	50 (0x32)	50 (0x32)	
3	FF	0	51 (0x33)	
	WK	51 (0x33)	51 (0x33)	
4	FF	0	52 (0x34)	
	WK	52 (0x34)	52 (0x34)	
5	FF	0	53 (0x35)	
	WK	53 (0x35)	53 (0x35)	
6	FF	0	54 (0x36)	
	WK	54 (0x36)	54 (0x36)	
7	FF	0	55 (0x37)	
	WK	55 (0x37)	55 (0x37)	
8	FF	0	56 (0x38)	
	WK	56 (0x38)	56 (0x38)	
9	FF	0	57 (0x39)	
	WK	57 (0x39)	57 (0x39)	
0	FF	0	48 (0x30)	
	WK	48 (0x30)	48 (0x30)	
Event generated when connecting USB Flash Drive	FF	0	112 (0x70)	ALT
	WK	112 (0x70)	112 (0x70)	ALT

Remote control key	Browser	keyCode, dec(hex)	which, dec(hex)	Flags
Event generated when disconnecting USB Flash Drive	FF	0	113 (0x71)	ALT
	WK	113 (0x71)	113 (0x71)	ALT

Where *keyCode* – is the *keyCode* field of the event received by the processor, and *which* – is the *which field of the event received by the processor*.

FF – means Mozilla Firefox, and WK – WebKit.

Remark 1. As compared to the previous version the code of "OK" key for the browser based on WebKit was changed.

Remark 2. With the purpose of making the processor independent of the browser it is recommended to add the following code at the beginning of the processor:

var code = e.keyCode | e.which;

and further to analyze the meaning of the **code** as the key code taking into account the modifiers specified in the table.

Remark 3. As compared to the previous releases for all keys having the modifier CTRL+ALT it was replaced with ALT for Mozilla Firefox.

The table for event processor onKeyDown and onKeyUp

The table below lists the keys codes received by the event processor JavaScript onKeyDown and onKeyUp for browsers Mozilla Firefox and WebKit:

Table 4. Remote control keys codes for processors onKeyDown и onKeyUp

Remote control key	keyCode, dec(hex)	which, dec(hex)	Flags
exit	27 (0x1b)	27 (0x1b)	
ok	13 (0x0d)	13 (0x0d)	
right	39 (0x27)	39 (0x27)	
left	37 (0x25)	37 (0x25)	
up	38 (0x26)	38 (0x26)	
down	40 (0x28)	40 (0x28)	
PageUp	33 (0x21)	33 (0x21)	
PageDown	34 (0x22)	34 (0x22)	
menu	122 (0x7a)	122 (0x7a)	CTRL
back	8 (0x08)	8 (0x08)	
refresh	116 (0x74)	116 (0x74)	CTRL
red	112 (0x70)	112 (0x70)	CTRL
green	113 (0x71)	113 (0x71)	CTRL
yellow	114 (0x72)	114 (0x72)	CTRL
blue	115 (0x73)	115 (0x73)	CTRL
channel+	9 (0x09)	9 (0x09)	
channel–	9 (0x09)	9 (0x09)	SHIFT
service	120 (0x78)	120 (0x78)	CTRL
tv	121 (0x79)	121 (0x79)	CTRL
phone	119 (0x77)	119 (0x77)	CTRL
web	123 (0x7b)	123 (0x7b)	CTRL
frame	117 (0x75)	117 (0x75)	CTRL
vol+	107(0x6b)	107(0x6b)	
vol–	109(0x6d)	109(0x6d)	
rew	66 (0x42)	66 (0x42)	ALT
ffwd	70 (0x46)	70 (0x46)	ALT

Remote control key	keyCode, dec(hex)	which, dec(hex)	Flags
stop	83(0x53)	83(0x53)	ALT
play/pause	82(0x52)	82(0x52)	ALT
rec	87(0x57)	87(0x57)	ALT
mic	32 (0x20)	32 (0x20)	ALT
mute	192(0xC0)	192 (0xC0)	ALT
power	85 (0x55)	85 (0x55)	ALT
info	89 (0x59)	89 (0x59)	ALT
Empty	76 (0x4c)	76 (0x4c)	ALT
1	49 (0x31)	49 (0x31)	
2	50 (0x32)	50 (0x32)	
3	51 (0x33)	51 (0x33)	
4	52 (0x34)	52 (0x34)	
5	53 (0x35)	53 (0x35)	
6	54 (0x36)	54 (0x36)	
7	55 (0x37)	55 (0x37)	
8	56 (0x38)	56 (0x38)	
9	57 (0x39)	57 (0x39)	
0	48 (0x30)	48 (0x30)	
Event generated when connecting USB Flash Drive	80 (0x50)	80 (0x50)	ALT
Event generated when disconnecting USB Flash Drive	81 (0x51)	81 (0x51)	ALT

Remark 1. Event processing with onKeyDown is much simpler than event processing with onKeyPress, because remote control keys codes are not duplicated in onKeyDown, excluding the keys channel+ и channel-.

Appendix 6. MAG200 front panel indication control

For controlling the indicator and LED on the front panel call the function stb.ExecAction as follows:

stb.ExecAction(“front_panel param”) ,

where param – is the parameter string for the script setFpanel.sh, which performs output to the front panel. The parameters of this script are described in the document “Operator guide_MAG200.pdf”.

Appendix 7. Use of keys on the MAG200 front panel

Pressing the keys on the front panel generates the event of pressing the key on the keyboard. Utilities **fpxevent** for **FireFox** and **fpqevent** for **WebKit** are used for this purpose. The events of pressing are translated according to configuration files /etc/lirc/lircrc wk for FireFox and /etc/lirc/lircrc.wk for WebKit. These utilities are described in more detail in the documents “Operator guide _MAG200.pdf”.

Appendix 8. Switching video output modes.

Setting video output mode.

Switching video output mode is performed by calling the method **stb.ExecAction** in the following form:

stb.ExecAction(“tvsystem mode”),

where mode can accept the following values:

PAL

576p-50

720p-50

1080i-50

NTSC

576p-60

720p-60

1080i-60

For example, stb.ExecAction(“tvsystem PAL”) sets the video output in the mode PAL(576i).

Remark. The changes shall come into force after the device is restarted.

Receiving the current mode of video output

Use the function stb.RDir in the form:

var mode = stb.RDir('vmode') for receiving te current mode of te video output.

Where **mode** can take the following values:

576i – PAL mode

576p – 576p mode

720p – 720p mode

1080i – 1080i mode

Remark. In this case the current operating mode shall be returned, that is, it can be changed only after reloading.

Appendix 9. Control of the size and position of the browser window on the basis of WebKit.

If necessary, the size of the browser window can be reduced and its position on the screen changed. To do so call the following functions:

window.moveTo(x, y) – offsets the window to the position with the coordinates **x** and **y**.

window.resizeTo(width, height) – sets the window width in the value **width**, and the height - in the value **height**.

Appendix 10. Setting graphic resolution of the browser based on the WebKit.

Setting resolution

Graphic resolution can be set using the function stb.ExecAction in the following manner:

stb.ExecAction('graphicres **mode**'),

where **mode** can take the following values:

tvsystem**_res** – graphic resolution agrees with the resolution of the video output (aspect 1:1)

720 – graphic resolution 720x576, with hard scaling of this resolution to the whole screen in the modes 1080i and 720p.

1280 – graphic resolution 1280x720, with hard scaling of this resolution to the whole screen in the mode 1080i.

1920 – graphic resolution 1920x1080.

Remark. If the resolution of the video output is less than the graphic resolution set, the graphic resolution shall be considered equal to the video output resolution.

Remark. The changes shall come into force after the device is restarted.

Receiving current graphic resolution

Current graphic resolution can be received using the function stb.RDir in the following way:

var gres = stb.RDir('gmode'),

where **gres** shall take the values: **tvsystem_res, 720, 1280, 1920** as described in the previous item.

Graphic resolution can be also received using **screen.width** и **screen.height** and:

screen.width – horizontal rersolution.

screen.height – vertical resolution.

Remark. In this case the current operating mode shall be returned because it can be changed only after reloading.

Appendix 11. Operation with environment variables.

Javascript API for MAG 200 device allows receiving and setting special environment variables stored in ROM. Hereinafter they shall be defined as “environment variables”.

Attention. Environment variables are stored in ROM possessing a large but limited number of re-recordings, therefore, it is ultimately recommended not to save the parameters that often change, for example, at each start of the device.

Setting and getting environment variables.

For getting an environment variable use the function RDir with the parameter getenv (see. stb.RDir)

To set the environment variable use the function RDir with the parameter setenv (see. stb.RDir)

If several variables must be set, it is recommended to set all these variables in one call. Simply insert the line «"|"» (3 symbols), between the pairs “name-value”, i.e. call RDir in the following form:

stb.RDir(‘setenv name_1 val_1 "|" name_2 val_2 "|" … "|" name_n val_n’),

where n – the number of variables, name_n – the name of variable under number n, val_n – the value to be set to the variable with the name name_n.

For example:

stb.RDir(‘setenv mcip_conf 224.50.7.50 "|" mcip_img_conf 111.1.2.3 "|" portal2 http://some_portal.com’)

set variables mcip_conf, mcip_img_conf and portal2 to the values 224.50.7.50, 111.1.2.3 and http://some_portal.com corerspondingly.

Example of Javascript code for such call is shown below:

```
.....
var CONCAT = ' "|" ',
```

```
str    = '',
ipaddr = '',    // variable 1
mcip   = '',    // variable 2
mcport = '';    // variable 3
str += 'ipaddr_conf ' + ipaddr;
str += CONCAT + 'mcip_conf ' + mcip;
str += CONCAT + 'mcport_conf ' + mcport;
batchSetEnvValues(str);
    ........
// Packet setting of environment variables in ROM
function batchSetEnvValues(str){
  stb.RDir ('setenv ' + str);
}
    ........
```

Environment variables used in standard program

To ensure correct operation of the standard program use the following environment variables:

ipaddr_conf – static IP address. If this variable is not set the device receives IP address, netmask, gateway, DNS and NTP automatically (via DHCP) when starting.

netmask – subnetwork mask.

gatewayip – IP address of the default gateway.

dnsip – IP address of DNS server.

ntpurl – URL of NTP server.

mcip_conf – multicast address, from which bootstrap is received.

mcport_conf – port number from which bootstrap is received

mcip_img_conf – multicast address from which the image for updating is received (imageupdate).

mcport_img_conf – port number from which the image for updating is received (imageupdate).

mcip_mng_conf – multicast address of the managing channel.

mcport_mng_conf – port number of the managing channel.

portal1 – URL of portal 1.

portal2 – URL of portal 2.

volume – default volume level.

language – language index of user interface. 0 – English, 1 – Russian.

upnp_conf – Start(true) or not start(false) UPnP client.

use_portal_dhcp – use(true) or not use(false) the variable valueportal_dhcp as starting portal if the variables portal1 and portal2 are not set.

portal_dhcp – URL of the portal set by the operator using protocol DHCP.

Change history.

Version 1.14

The functions added:

stb.SetPosTimeEx,

stb.GetPosTimeEx,

stb.GetMediaLenEx.

Information on the functions stb.SetSubtitlesSize, stb.SetSubtitlesFont and stb.SetSubtitlesOffs was updated in the table “Accessibility of function on IPTV devices”. Description of new parameters for the function stb.RDir. was added .

Supplement 11. Operation with environment variables was added.

Version 1.13

Function stb.DeleteAllCookies was added.

Information in the section Cookie is updated.

Version 1.12

Description of the function stb.GetAspect. was updated

Description of the function stb.SetCASDescrambling. was corrected.

Version 1.11

Added:

Supplement 10. Setting graphic resolution of the browser based on WebKit.

Getting current video output mode

Version 1.10

Functions added:

SetSubtitleLangs,

GetSubtitlePID,

SetSubtitlePID,

SetBrightness,
SetSaturation,
SetContrast,
GetBrightness,
GetSaturation,
GetContrast.

Information on setting event system was added to the item “Setting event system”.

Information on Standby mode was added. See .StandBy.

Appendix 9. Control of the size and position of the browser window on the basis of WebKit. was added.

Error in the description of the function stb.SetViewport was corrected.