LG Electronics

LG Web Open API Reference Guide

Version 2.5.4 – April 2012

LGDEV-021

Home Entertainment Company
LG Electronics, Inc.

About This Document

Revision History

Document Version	Date	Comment
2.5.4	April 23, 2012	- Section 1.1 Overview of LG Web Open API is added. - Window.NetCastEnableChannelKey is added in NetCast API. - Some APIs and events are added in Broadcast API. - Section 2.6 Voice Recognition is added.
2.5.3	February 24, 2012	Annex A Differences in Media Devices is updated
2.5.2	February 14, 2012	Description in section 2.3 is modified.
2.5.1	February 6, 2012	Example code of 'autostart' function is updated. (Section 2.3.5)
2.5.0	January 2, 2012	- Section 1.1 Coverage of this Document is updated. - Section 2.5 Broadcast Plugin and API is added. - window.NetCastReturn is removed. - window.NetCastEnableNumberKey is added. - readyState and onReadyStateChange are removed. - Descriptions and sample code of some APIs are updated.
2.4.0	September 23, 2011	"LG Browser Open API Technical Specification" is separated into "LG Web Open API Reference Guide" and "LG Web Application Development Guide". - outofmemory is added in section 2.2. - audioLanguage is added in section 2.3.5. - preferredSubtitleLanguage, preferredAudioLanguage, preferredSubtitleStatus, tvLanguage2, and tvContry2 are added in section 2.4.3.
~ 2.3.0	July 18, 2011	This document was originally LG NetCast Platform_LG Browser Open API Technical Specification.

Purpose

This guide explains features, methods, and how-to-use information of the LG Web Open API required to create the web application for LG Smart TV.

Reference Documents

Refer to the following documents:
- LG Web Application Development Guide

Conventions

Codes

Source code and examples are indicated in the `grey Courier New` font.

Note, Caution

Note and caution are used to emphasize information.
The following samples describe when each is used.

NOTE

Contains information about something that is helpful to you.

CAUTION

Contains important information about something that you should know.

Abbreviation

The following table defines the abbreviations used in this document.

Abbreviation	Description
DHCP	Dynamic Host Configuration Protocol
MRCU	Magic Remote Control Unit
RCU	Remote Control Unit

Glossary

The following table describes the terms used in this document.

Term	Description
NetCast 1.0	Platform that runs on LG Smart TV released in 2010.
NetCast 2.0	Platform that runs on LG Smart TV released in 2011.
NetCast 3.0	Platform that runs on LG Smart TV released in 2012.

Note for Media Products

Note the special comments for media products in Annex A. Also, “Media Device NetCast API” is described in section 2.7.

Contents

1 Introduction 8

1.1 Overview of LG Web Open API 9

1.2 Coverage of this Document 10

2 Application Programming Interface 15

2.1 NetCast API 16

2.1.1 NetCast API (Methods) 16
window.NetCastExit 16
window.NetCastBack 16
window.NetCastSetPageLoadingIcon 17
window.NetCastSetDefaultAspectRatio 18
window.NetCastLaunchQMENU 20
window.NetCastLaunchRATIO 20
window.NetCastMouseOff 21
window.NetCastGetMouseOnOff 22
window.NetCastGetUsedMemorySize 23
window.NetCastEnableNumberKey 23
window.NetCastEnableChannelKey 24

2.2 Proprietary Events 25

2.2.1 Events 25
mouseon 25
mouseoff 26
outofmemory 27

2.3 Media Player Plugin and API 28

2.3.1 Media Object 28
mode3D 28
preBufferingTime 29
oneshot_url 29
subtitleOn / subtitle 30
drm_type 30

2.3.2 Audio Object 32

2.3.3 Media Type Resolving in Media Player Plugin 32

2.3.4 Media Player API (Methods) 33
play 33
stop 34
next 34
previous 35
seek 35
mediaPlayInfo 36
setWidevineXXX 37

2.3.5 Media Player API (Properties) 38
version 38
type 39
data 39
width / height 39
playTime 40
playPosition 41
playState 41
error 42
autoStart 43
isScannable 43

speed 44
bufferingProgress 44
subtitleOn 45
subtitle 45
mode3D 46
audioLanguage 46
2.3.6 Media Player API (Events) 47
onPlayStateChange 47
onBuffering 48
onError 49
2.4 Device Info Plugin and API 50
2.4.1 Device Info Object 50
2.4.2 Device Info API (Methods) 50
setDrmLicenseInfo 50
2.4.3 Device Info API (Properties) 51
version 51
manufacturer 51
modelName 52
serialNumber 52
swVersion 52
hwVersion 53
osdResolution 53
networkType 54
net_macAddress 54
drmClientInfo 54
net_dhcp 55
net_isConnected 55
net_hasIP 56
net_ipAddress 56
net_netmask 56
net_gateway 57
net_dns1 57
net_dns2 57
supportMouse 58
support3D 58
support3DMode.from_2d_to_3d 59
support3DMode.side_by_side 59
support3DMode.side_by_side_rl 59
support3DMode.top_bottom 60
support3DMode.checker_bd 60
preferredSubtitleLanguage 61
preferredAudioLanguage 61
preferredSubtitleStatus 61
tvLanguage2 62
tvCountry2 62
2.5 Broadcast Plugin and API 63
2.5.1 Broadcast Object 63
2.5.2 Broadcast API (Methods) 63
channelUp / channelDown 63
getCurrentChannelName 64
getCurrentProgram 64
getCurrentChannelNumber 65
isChannelMapEmpty 65
isTunerInput 66
isDvb 66
2.5.3 Broadcast API (Events) 66
onchannelchange 66
onnosignal 67

2.6 Voice Recognition Plugin and API 68
2.6.1 Voice Recognition Object 68
2.6.2 Voice Recognition API (Property) 68
isInitialized 68
isEnable 68
dictation 69
2.6.3 Voice Recognition API (Methods) 69
startRecognition 69
2.6.4 Voice Recognition API (event) 70
onrecognizevoice 70
onbuttonenable 70
2.7 Media Device NetCast API 71
2.7.1 Media Device API (Methods) 71
window.NetCastSetScreenSaver 71

Annex A Differences in Media Devices 72

Tables

[Table 1] List of supporting APIs 10
[Table 2] Explanation of aspect ratio control mode 19
[Table 3] Available values for “mode3D” property 28
[Table 4] Avaliable values for “drm_type” property 30
[Table 5] Variables of mediaPlayInfo()’s return object 36
[Table 6] Credential Information list 38
[Table 7] The enumerated value of play state 41
[Table 8] The error code 42
[Table 9] Available values for “mode3D” property 46
[Table 10] Mapping table between the resolutions and enumerated return values 53
[Table 11] Mapping table between network types and enumerated return values 54

Figures

[Figure 1] Service Block Diagram of LG Web Open API 9
[Figure 2] Screen shot of browser’s page loading icon in NetCast 2.0 18
[Figure 3] Screen shot of browser’s page loading icon in NetCast 3.0 18
[Figure 4] Media Data Transmitting Algorithm of Media Playback 19
[Figure 5] Decision tree for media type resolving in the media player plugin 32
[Figure 6] Graphical explanation of buffer related variables 36
[Figure 7] Graphical implementation example of streaming speed level meter 37

1 Introduction

This chapter provides the coverage of this document.

1.1 Overview of LG Web Open API
1.2 Coverage of this Document

1.1 Overview of LG Web Open API

LG Web Open API provides APIs for developers to create web applications on LG Smart TV. The following figure illustrates the LG Web Open API service blocks.

[Figure 1] Service Block Diagram of LG Web Open API

Window Extended (NetCast)

NetCast supports the proprietary browser APIs for developers to use.

Media Player

Media Player Plugin and API contain CE-HTML and CEA-2014 compliant media player plugin.

Device

Device Info Plugin and API are for obtaining device information in the application.

Broadcast

Broadcast Plugin and API are for displaying TV broadcast images in the application. Also, provide useful APIs for developing applications in connection with broadcast TV information

Advertisement

Advertisement APIs are for integrating banner or video advertisements into the application running on LG Smart TV.

Voice Recognition

Voice Recognition Plugin and API are for using voice recognition function of Magic Remote Control in the application.

1.2 Coverage of this Document

Availability and functionality range of APIs might be different depending on the version. Refer to the description of each API in this document.

To refer to Advertisement APIs, see the **"LG Advertisement API Reference Guide"** document, which can be downloaded from Developer Lounge (http://developer.lgappstv.com). (Partners of LGE can download the library from http://netcastdev.lge.com/.)

[Table 1] List of supporting APIs

Category	LG Web Open API	NetCast 1.0	NetCast 2.0	NetCast 3.0
NetCast API	window.NetCastExit	O	O	O
	window.NetCastBack	O	O	O
	window.NetCastSetPageLoadingIcon	O	O	O
	window.NetCastSetDefaultAspectRatio	O	O	O
	window.NetCastLaunchQMENU	O	O	O
	window.NetCastLaunchRATIO	O	O	O
	window.NetCastMouseOff	O	O	O
	window.NetCastGetMouseOnOff	O	O	O
	window.NetCastGetUsedMemorySize	X	O	O
	window.NetCastEnableNumberKey	X	X	O
	window.NetCastEnableChannelKey	X	X	O
Proprietary Events	mouseon Event	O	O	O
	mouseoff Event	O	O	O
	outofmemory	X	O	O
Media Player Plugin (Methods)	play(1) - play	O	O	O
	play(0) - pause	O	O	O
	stop	O	O	O
	next	O	O	O

Category	LG Web Open API	NetCast 1.0	NetCast 2.0	NetCast 3.0
	previous	O	O	O
	width	O	O	O
	height	O	O	O
	seek	O	O	O
	play	O	O	O
	mediaPlayInfo	O	O	O
	setWidevineDrmURL	*	O	O
	setWidevineDeviceID	*	O	O
	setWidevineStreamID	*	O	O
	setWidevineClientIP	*	O	O
	setWidevineUserData	*	O	O
	setWidevineDrmAckURL	*	O	O
	setWidevineHeartbeatURL	*	O	O
	setWidevineHeartbeatPeriod	*	O	O
	setWidevineDeviceType	*	O	O
Media Player Plugin (Properties)	version	O	O	O
	type	O	O	O
	data	O	O	O
	width	O	O	O
	height	O	O	O
	playTime	O	O	O
	playPosition	O	O	O

Category	LG Web Open API	NetCast 1.0	NetCast 2.0	NetCast 3.0
	playState	O	O	O
	error	O	O	O
	autoStart	O	O	O
	isScannable	O	O	O
	speed	O	O	O
	bufferingProgress	O	O	O
	subtitleOn	O	O	O
	subtitle	O	O	O
	mode3D	X	O	O
	audioLanguage	X	O	O
Media Player Plugin (Events)	onPlayStateChange	O	O	O
	onBuffering	O	O	O
	onError	O	O	O
Device Info Plugin (Methods)	setDrmLicenseInfo	O	O	O
Device Info Plugin (Properties)	version	O	O	O
	manufacturer	O	O	O
	modelName	O	O	O
	serialNumber	O	O	O
	swVersion	O	O	O
	hwVersion	O	O	O
	osdResolution	O	O	O
	networkType	O	O	O

Category	LG Web Open API	NetCast 1.0	NetCast 2.0	NetCast 3.0
	net_macAddress	O	O	O
	drmClientInfo	O	O	O
	net_dhcp	O	O	O
	net_isConnected	O	O	O
	net_hasIP	O	O	O
	net_ipAddress	O	O	O
	net_netmask	O	O	O
	net_gateway	O	O	O
	net_dns1	O	O	O
	net_dns2	O	O	O
	supportMouse	O	O	O
	support3D	O	O	O
	support3DMode.from_2d_to_3d	X	O	O
	support3DMode.side_by_side	X	O	O
	support3DMode.side_by_side_rl	X	O	O
	support3DMode.top_bottom	X	O	O
	support3DMode.checker_bd	X	O	O
	preferredSubtitleLanguage	X	O	O
	preferredAudioLanguage	X	O	O
	preferredSubtitleStatus	X	O	O
	tvLanguage2	X	O	O
	tvContry2	X	O	O

Category	LG Web Open API	NetCast 1.0	NetCast 2.0	NetCast 3.0
Broadcast Plugin (Methods)	channelUp	X	X	O
	channelDown	X	X	O
	getCurrentChannelName	X	X	O
	getCurrentProgram	X	X	O
	getCurrentChannelNumber	X	X	O
	isChannelMapEmpty	X	X	O
	isTunerInput	X	X	O
	isDvb	X	X	O
Broadcast Plugin (Events)	onchannelchange	X	X	O
	onnosignal	X	X	O
Voice Recognition Plugin (Property)	isInitialized	X	X	O
	isEnabled	X	X	O
	dictation	X	X	O
Voice Recognition Plugin (Methods)	startRecognition	X	X	O
Voice Recognition Plugin (Events)	onrecognizevoice	X	X	O
	onbuttonenable	X	X	O

[*] : This API is only supported on LG Smart TV in Austrailia.

2 Application Programming Interface

This chapter describes the data structure and APIs provided by LG Web Open API. Each API is described in five categories: Description, Syntax, Parameter, Return Value, and Example.

2.1 NetCast API

2.2 Proprietary Events

2.3 Media Player Plugin and API

2.4 Device Info Plugin and API

2.5 Broadcast Plugin and API

2.6 Voice Recognition Plugin and API

2.7 Media Device NetCast API

2.1 NetCast API

Following sections describe the proprietary browser APIs for developers to use.

2.1.1 NetCast API (Methods)

window.NetCastExit

Description

The NetCast Platform provides a proprietary API, 'window.NetCastExit()', to implement the exit function to AV. A JavaScript application can use this API for users to exit or quit the application to AV.

Syntax

```
window.NetCastExit();
```

Parameters

None

Return Value

None

Example

```
function processExit()
{
    window.NetCastExit();
}
```

window.NetCastBack

Description

The NetCast Platform provides a proprietary API, 'window.NetCastBack()', to implement the back function to the NetCast portal menu. A JavaScript application can use this API for users to move back to the NetCast portal menu. It is highly recommended for developers to implement the window.NetCastBack() API since a user experience of "historical back function" is widely used in LG Smart TV system.

Note

This API works on v3.0.23 or higher version of LG Browser.

Syntax

```
window.NetCastBack();
```

Parameters

None

Return Value

None

Example

```
function processBack()
{
     if(window.NetCastBack) {
            window.NetCastBack();
     }
}
```

window.NetCastSetPageLoadingIcon

Description

It is recommended that developers provide a “loading” icon so that users are provided with an indication of the latency of data downloading from a server. Developers can implement this feature using JavaScript, however, it may not be possible do this, as there would not be any JavaScript running while a HTML page is loading. The NetCast Platform supports a proprietary API, ‘window.NetCastPageLoadingIcon()’, to provide the browser’s own page loading animation function. Developers can use this API during HTML page loading.

Note

This function will be applied while the next page is loaded.

The following figures are example screen shots of a browser’s page loading icon.

Syntax

```
window.NetCastSetPageLoadingIcon(control);
```

Parameters

`control`	[in] ‘enabled’ or ’disabled’

Return Value

None

Example

```
function enablePageLoadingIcon() {
     window.NetCastSetPageLoadingIcon('enabled');
}
```

```
function disablePageLoadingIcon() {
     window.NetCastSetPageLoadingIcon('disabled');
}
```

[Figure 2] Screen shot of browser's page loading icon in NetCast 2.0

[Figure 3] Screen shot of browser's page loading icon in NetCast 3.0

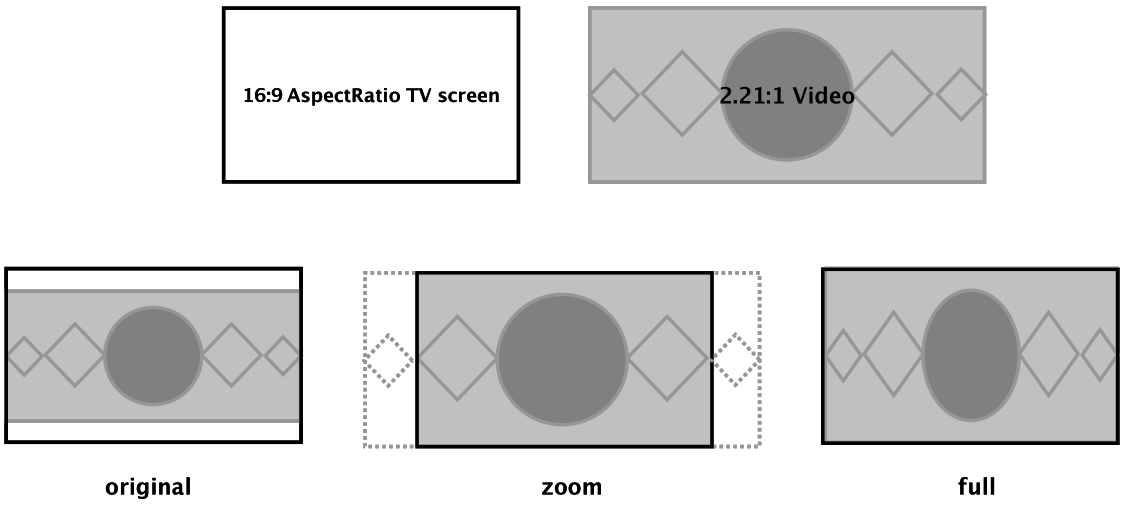
window.NetCastSetDefaultAspectRatio

Description

Developers may want to set the default aspect ratio for users to view full screen video with the correct aspect ratio. The NetCast Platform allows developers to set the default aspect ratio by using the 'window.NetCastSetDefaultAspectRatio()' API. The setting only applies if the video runs in full screen mode, 1280 x 720. This API takes a string type of argument. The list of arguments and their behaviors are listed in the following table and figure.

Note

This API call applies only once for a whole application life cycle. Second and subsequent calls will be ignored by the LG Browser automatically. Therefore, it is recommended to locate the API call at the time of launching the application. If the application is launched again after exiting, the API will be enabled again.



[Figure 4] Media Data Transmitting Algorithm of Media Playback

Syntax

```
window.NetCastSetDefaultAspectRatio(control);
```

Parameters

`control` [in] Aspect ratio control mode

[Table 2] Explanation of aspect ratio control mode

Aspect ratio control mode	Meaning
original	view original video image with original correct aspect ratio (TV screen may not be filled with video image)
zoom	fill the full TV screen with original aspect ratio video (there may be some cropping of the original video image)
full	fill the full TV screen with video (aspect ratio may be distorted, but with no loss of original video image)

Return Value

None

Example

```
function setDefaultAspectRatio()
{
    window.NetCastSetDefaultAspectRatio('original');
}
</script>
<body onload='setDefaultAspectRatio(); return false;'>
    ...
</body>
```

window.NetCastLaunchQMENU

Description

The NetCast Platform provides a 'QMENU' (Quick Menu for Audio and Video Adjustment) for users to setup the aspect ratio for full screen video, picture quality adjustment and audio adjustment. The QMENU can be launched by users when playing video in full screen mode by pressing the 'QMENU' button on the remote control. This only works in full screen video mode.

Note, it is possible for users to operate a LG Smart TV application using the Magic-RCU, the pointing device of the NetCast Platform. There is no 'QMENU' button on the Magic-RCU, Therefore, it is strongly recommended that the LG Smart TV developer implements a graphical user interface to launch the QMENU over a full screen video.

The NetCast Platform thus provides a proprietary API, 'window.NetCastLaunchQMENU()' to enable this feature. If this API is called then the LG Smart TV will overlay the QMENU on the full screen video. Developers can launch the QMENU over a full screen video using this API.

Note

This API works on v3.0.23 or higher version of LG Browser.

Syntax

```
window.NetCastLaunchQMENU();
```

Parameters

None

Return Value

None

Example

```
function launchQMENU()
{
     if(window.NetCastLaunchQMENU) {
            window.NetCastLaunchQMENU();
     }
}
```

window.NetCastLaunchRATIO

Description

The NetCast Platform provides a 'RATIO' (Aspect Ratio Control) menu for users to setup the aspect ratio of a full screen video. The RATIO menu can be launched by a user while playing a video in a full screen mode by pressing the 'RATIO' button on the remote control. This only works in full screen video mode.

Note, it is possible for users to operate a LG Smart TV application using the Magic-RCU, the pointing device of NetCast Platform. There is no 'RATIO' button on the Magic-RCU, therefore, it is strongly recommended that the LG Smart TV developer implements a graphical user interface to launch the RATIO menu over a full screen video.

The NetCast Platform thus provides a proprietary API 'window.NetCastLaunchRATIO()' to enable this feature. If this API is called then the LG Smart TV will overlay the RATIO menu on the full

screen video. Developers can launch the RATIO over a full screen video using this API.

Note

This API works on v3.0.23 or higher version of LG Browser.

Syntax

```
window.NetCastLaunchRATIO();
```

Parameters

None

Return Value

None

Example

```
function launchRATIO()
{
    if(window.NetCastLaunchRATIO) {
          window.NetCastLaunchRATIO();
    }
}
```

window.NetCastMouseOff

Description

This API can be used by an LG Smart TV developer to deactivate the Magic-RCU and its pointer.

Refer to section 3.4 Input Device and 2.2.2 userAgent String in **"LG Web Application Development Guide"**, and supportMouse, mouseon, mouseoff, and window.NetCastGetMouseOnOff() for more information about the Magic-RCU and its status related event and API.

Developers can deactivate the Magic-RCU and its pointer using this API. In the following example, the "time" parameter is the time value after which the deactivation is applied. This parameter is processed to "second" level accuracy. For example, the Magic-RCU pointer would disappear 5 seconds after calling "window.NetCastMouseOff(5);".

Note

This API works on v3.0.23 or higher version of LG Browser.

In NetCast 3.0, this function is not supported and the mouse gets deactivated when the halt of the mouse movement continues for 3 seconds.

Syntax

```
window.NetCastMouseOff(time);
```

Parameters

time	[in] Time value after which the deactivation is applied (in second)

Return Value

None

Example

```
function mouseOff(time)
{
     if(window.NetCastMouseOff){
           window.NetCastMouseOff(time);
     }
}
```

window.NetCastGetMouseOnOff

Description

This API can be used by an LG Smart TV developer to get the on or off status of the Magic-RCU.

Refer to section 3.4 Input Device and 2.2.2 userAgent String in **"LG Web Application Development Guide"**, and supportMouse, mouseon, mouseoff, and window.NetCastGetMouseOff(time) for more information about the Magic-RCU and its status related event and API.

Note

This API works on v3.0.23 or higher version of LG Browser.

Syntax

```
window.NetCastGetMouseOnOff();
```

Parameters

None

Return Value

Return value is "on" or "off", the mouse status. (String type)

Example

```
var mouseOnOffStatus;

function getMouseOnOff()
{
     if(window.NetCastGetMouseOnOff) {
          mouseOnOffStatus = window.NetCastGetMouseOnOff();
     }
}
```

window.NetCastGetUsedMemorySize

Description

This API can be used by an LG Smart TV developer to get the total memory size used by the Browser process.

Note

This API works on v4.1.2 or higher version of LG Browser.

Syntax

```
window.NetCastGetUsedMemorySize();
```

Parameters

None

Return Value

Returns the memory size used in the application. (Int type)

Example

```
var usedMemorySize;

function getUsedMemorySize()
{
    if(window.NetCastGetUsedMemorySize) {
        usedMemorySize = window.NetCastGetUsedMemorySize();
    }
}
```

window.NetCastEnableNumberKey

Description

The NetCast Platform provides a proprietary API, 'window.NetCastEnableNumberKey' to change a channel with number keys from RCU. Contrary to the normal operation, developers can use this API for DTV to process number keys for TV.
If the value of input parameter is TRUE, a number key is transferred to the application by default.
If the value of input parameter is FALSE, a number key is transferred to DTV, not application, for changing the channel.

Syntax

```
window.NetCastEnableNumberKey(boolean);
```

Parameters

boolean	[in] true or false

Return Value

None

Example

```
function enableChannelChangeByNumberKey() {
    window.NetCastEnableNumberKey(false);
}
```

```
function registerNumberKey() {
    window.NetCastEnableNumberKey(true);
}
```

window.NetCastEnableChannelKey

Description

The NetCast Platform provides a proprietary API, 'window.NetCastEnableChannelKey' to change a channel with channel Up/Down keys from RCU. Contrary to the normal operation, developers can use this API for DTV to process channel Up/Down keys for TV.
If the value of input parameter is TRUE, a channel Up or Down key is transferred to the application by default. If the value of input parameter is FALSE, the key is transferred to DTV, not application, for changing the channel.

Syntax

```
window.NetCastEnableChannelKey(boolean);
```

Parameters

boolean	[in] true or false

Return Value

None

Example

```
function enableChannelChangeByChannelKey() {
    window.NetCastEnableChannelKey(false);
}
```

```
function registerChannelKey() {
    window.NetCastEnableChannelKey(true);
}
```

2.2 Proprietary Events

This section describes proprietary browser events available to an LG Smart TV developer.

2.2.1 Events

mouseon

Description

This event is generated when the Magic-RCU is activated. Refer to the section 3.4 Input Device in **"LG Web Application Development Guide"**, and 2.4.3 Device Info API (Properties) and supportMouse for more detailed information about the Magic-RCU and input devices.

The following examples illustrate how developers can register the "mouseon" event handler in three ways.

Note

This API works on v3.0.23 or higher version of LG Browser.

Example

```
// Registering "mouseon" event (in body tag)
<body
    onkeydown='processKeyDown(event)'
    ondragstart='return false'
    onselectstart='return false'
    onmouseon='mouseon_handler()'>
. . .
</body>
```

```
// Registering "mouseon" event (window property)
<script language='javascript'>
    if(window.onmouseon) {
    window.onmouseon = mouseon_handler;
    }
</script>
```

```
// Registering "mouseon" event (DOM Level 2 event)
<script language='javascript'>
    if(window.onmouseon) {
    window.addEventListener('mouseon', mouseon_handler, true);
    }
</script>
```

See Also

window.NetCastMouseOff
window.NetCastGetMouseOnOff

mouseoff

Description

This event is generated when the Magic-RCU is deactivated. Refer to the section 3.4 Input Device and 2.2.2 userAgent String in **"LG Web Application Development Guide"**, and supportMouse for more detailed information about the Magic-RCU and input devices.

The following examples illustrate how developers can register the "mouseoff" event handler in three ways.

Note

This API works on v3.0.23 or higher version of LG Browser.

Example

```
// Registering "mouseoff" event (in body tag)
<body
    onkeydown='processKeyDown(event)'
    ondragstart='return false'
    onselectstart='return false'
    onmouseoff='mouseoff_handler()'>
</body>
```

```
// Registering "mouseoff" event (window property)
<script language='javascript'>
    if(window.onmouseoff) {
    window.onmouseoff = mouseoff_handler;
    }
</script>
```

```
// Registering "mouseoff" event (DOM Level 2 event)
<script language='javascript'>
    if(window.onmouseoff) {
         window.addEventListener('mouseoff', mouseoff_handler, true);
    }
</script>
```

See Also

window.NetCastMouseOff
window.NetCastGetMouseOnOff

outofmemory

Description

LG Browser gets force shut down when the TV system memory goes under 40 MB.
The 'outofmemory' event gives the remaining memory size to developer before the browser gets force shut down. This event occurs three times (at about 70 MB (40 MB + 30 MB), 60 MB (40 MB + 20 MB), 50 MB (40 MB + 10 MB)). If you use 'event.available' property, you can get the current available memory size.

The following examples illustrates how developer can register the 'outofmemory' event handler in three ways.

Note

This API works on v3.0.23 or higher version of LG Browser.

Example

```
// Registering "outofmemory" event (in body tag)
<body
    onkeydown='processKeyDown(event)'
    ondragstart='return false'
    onselectstart='return false'
    onoutofmemory ='outofmemory _handler()'>
. . .
</body>
```

```
// Registering "outofmemory" event (window property)
<script language='javascript'>
    if(window.onoutofmemory) {
          window.onoutofmemory = outofmemory_handler;
    }
</script>
```

```
// Registering "outofmemory" event (DOM Level 2 event)
<script language='javascript'>
    if(window.onoutofmemory) {
         window.addEventListener(outofmemory, outofmemory_handler,
true);
    }
</script>
```

2.3 Media Player Plugin and API

The NetCast Platform supports CE-HTML and CEA-2014 compliant media player plugin. The following sub-sections describe the API of CE-HTML compliant media player plugin. The supported API list is a subset of CE-HTML, and it is not the whole set of media player APIs as provided by CE-HTML.

2.3.1 Media Object

An example MIME type for the media object is video/x-ms-wmv for NetCast Platform media player plugin. The media object supports the data URL, width, height, id, preBufferingTime, oneshot_url, subtitle, subtitleOn and drm_type properties.

The NetCast Platform supports only one instance of the media object at any one time, so developers must not attempt to use more than one media object simultaneously.

Refer to Annex A Complete List of Supported MIME Types in **"LG Web Application Development Guide"**. See also section 2.3.3 Media Type Resolving in Media Player Plugin for the media type resolving rule.

The following sample code is an example of using the wmv media object in HTML.

```
// Example of media object in HTML
<object type="video/x-ms-wmv"
      data="http://192.168.1.50/example.wmv"
      width="1280"
      height="720"
      id="media">
</object>
```

mode3D

Description

The NetCast Platform supports 3D formats such as 2D-to-3D, Side-by-Side, Top-and-Bottom, and Checker Board. It provides mode3D write-only property for developers to set a specific 3D format in 3D mode. The NetCast Platform will automatically display 3D video with defined 3D format if this property is set properly. The default value of this property is off and NetCast Platform displays original 2D video if this property is not defined.
Note that Netcast 3.0 supports 3D rendering for any size of video screen. However, NetCast 2.0 supports it only for full screen mode. You can check if the TV supports 3D by using support3D in Device API.

[Table 3] Available values for "mode3D" property

Variable	Meaning
off	Original 2D Format
from_2d_to_3d	2D-to-3D Conversion Format
side_by_side	Side-by-Side Frame Compatible 3D (Left / Right)
side_by_side_rl	Side-by-Side Frame Compatible 3D (Right / Left)
top_bottom	Top-and-Bottom (or Over-Under) Frame Compatible 3D

Variable	Meaning
checker_bd	Checker board Frame Compatible 3D only available for HD format (1080p @ 30Hz)

Example

```
<object type="video/x-ms-wmv"
    data="http://192.168.1.50/3dexample.wmv"
    mode3D="side_by_side”
    width="1280" height="720"
    id="media">
</object>
```

preBufferingTime

Description

Since NetCast 2.0, the preBufferingTime property is supported. Developers can adjust buffering time through preBufferingTime before playback. The unit of this property is an integer number of seconds. In the following example, developer is requesting that the media content starts playing after performing 5 seconds buffering time through calling “preBufferingTime = 5”.

Example

```
<object type="video/x-ms-wmv"
    data="http://192.168.1.50/example.wmv"
    preBufferingTime = 5
    width="1280"
    height="720"
    id="media">
</object>
```

oneshot_url

Description

Developers may want to use a one shot URL to help prevent unwarranted content downloading. A “one shot URL” means a URL which is not available again after having been accessed once. The NetCast Platform supports a property, oneshot_url, for developers to play content linked to the one shot URL.

Note

To avoid multiple accesses to one shot URL, the NetCast Platform does not perform content type checking by reading the head of content file. Therefore, developers have to describe the exact MIME type. For example, “application/x-netcast-av” should be avoided because it does not describe the exact MIME type. ASX file should be avoided for one shot URL.

Example

```
<object type="video/x-ms-wmv"
    data="http://192.168.1.50/example.wmv"
    oneshot_url=true
```

```
    width="1280"
    height="720"
    id="media">
</object>
```

subtitleOn / subtitle

Description

The NetCast Platform supports subtitle decoding. Since NetCast 2.0, SAMI (Synchronized Accessible Media Interchange), CineCanvas and Timed Text subtitle formats are supported. The NetCast Platform supports the subtitle and subtileOn properties.

The subtitle must be applied when a full size video is being played.

Note

The NetCast Platform does not support multiplexing multiple subtitle tracks in one file.

Example

```
<object type="video/x-ms-wmv"
    data="http://192.168.1.50/example.wmv"
    subtitleOn=true
    subtitle="http://192.168.1.50/example.smi"
    width="1280"
    height="720"
    id="media">
</object>
```

drm_type

Description

The NetCast Platform supports WM-DRM 10 PD, PlayReady (supported only in NetCast 3.0), and Widevine DRM solutions. The NetCast Platform supports a property, drm_type, for developers to set the DRM type. The default value of this property is “wm-drm” and NetCast Platform will use the WM-DRM solution if this property is not defined.

[Table 4] Avaliable values for “drm_type” property

Values	Description
wm-drm	WM-DRM 10 PD or PlayReady (default value)
widevine	Widevine DRM and its adaptive/live streaming

Example

```
<object type="video/x-ms-wmv"
    data="http://192.168.1.50/example.wmv"
    drm_type="widevine"
    width="1280"
```

```
    height="720"
    id="media">
</object>
```

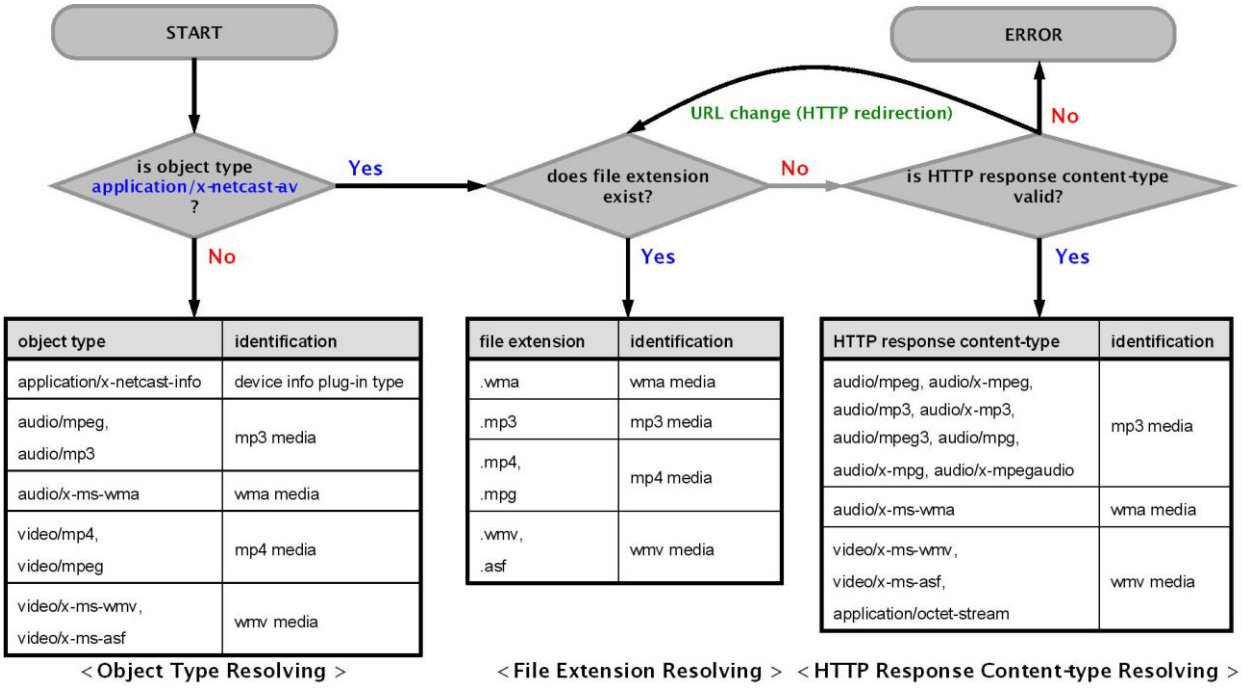
2.3.2 Audio Object

As opposed to a video object, an audio object cannot have width and height. However, the NetCast Platform supports only one media player plugin and media object for both video and audio objects, so developers must specify the width and height for audio object as 0. This kind of example can be used for “radio like” services.

```
// Example of audio object in HTML
<object type="audio/x-ms-wma"
      data="http://192.168.1.50/example.wma"
      width="0"
      height="0"
      id="media">
</object>
```

2.3.3 Media Type Resolving in Media Player Plugin

The following figure describes the decision tree for resolving media types on the NetCast Platform media player plugin. Refer to Annex A Complete List of Supported MIME Types in **“LG Web Application Development Guide”**.



[Figure 5] Decision tree for media type resolving in the media player plugin

2.3.4 Media Player API (Methods)

In this version of specification, the NetCast Platform does not support mute/unmute APIs. Users can mute and unmute audio using the mute function in the TV native system.

play

Description

Developers can play media at normal speed using media.play(speed) API.

Developers can also implement the trick mode play using media.play(speed) API.

Note

In mms streaming, the speed parameter is transparently transmitted to the server without any conversion. Therefore, developers are responsible for matching the speed parameter between the JavaScript application and the server.

Syntax

```
media.play(speed);
```

Parameters

`speed` [in] The range of allowed values of speed is from -30.0 to 30.0.

1 : Normal play speed. Default value is 1.
0 : Pause

Return Value

None

Example

```
// Example of 'play'
var media = document.getElementById("media");
media.play(1);
```

```
// Example of 'trick mode play'
var media = document.getElementById("media");
media.play(-1.0);
```

```
// Example of 'pause'
var media = document.getElementById("media");
media.play(0);
```

See Also

isScannable
speed

stop

Description

Developers can stop media using media.stop() API.

Syntax

```
media.stop();
```

Parameters

None

Return Value

None

Example

```
var media = document.getElementById("media");
media.stop();
```

next

Description

The NetCast Platform supports playlist, and developers can play the next media by calling media.next() API.

Refer to section 2.2.2 userAgent String in **"LG Web Application Development Guide"**.

Syntax

```
media.next();
```

Parameters

None

Return Value

None

Example

```
var media = document.getElementById("media");
media.next();
```

previous

Description

The NetCast Platform supports playlist, and developers can play previous media by calling media.previous() API.

Refer to section 2.2.2 userAgent String in **"LG Web Application Development Guide"**.

Syntax

```
media.previous();
```

Parameters

None

Return Value

None

Example

```
var media = document.getElementById("media");
media.previous();
```

seek

Description

This seek API will be supported for HTTP streaming only if the server supports the HTTP range header and MMS.
Developers can set the time position of playback using media.seek(position) API, and the position value has millisecond precision.

Syntax

```
media.seek(position);
```

Parameters

`position` [in] Position value. It must have millisecond precision.

Return Value

None

Example

```
var media = document.getElementById("media");
media.seek(30000);
```

mediaPlayInfo

Description

This API is for getting media playback related information.

Syntax

```
media.mediaPlayInfo();
```

Parameters

None

Return Value

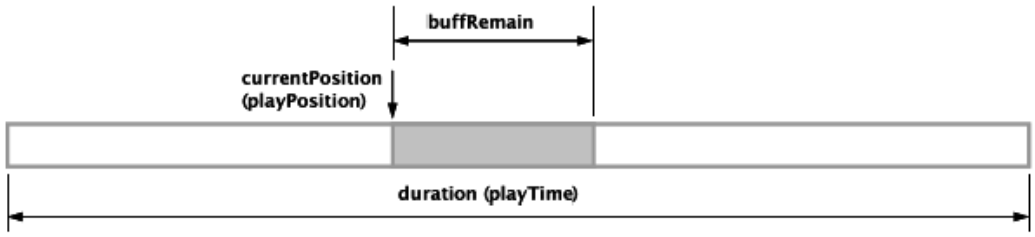
This API returns an object which contains values for duration, current position, remaining amount of buffer, instant bitrate and target bitrate. Property names and meanings are listed in the following table. The duration variable has same meaning and value with the property named as playTime. The currentPosition has same meaning and value as the property named as playPosition.

[Table 5] Variables of mediaPlayInfo()'s return object

Variable	Meaning	Unit
duration	Duration of media file (same as playTime property).	millisecond
currentPosition	Current play position (same as playPosition property).	millisecond
bufRemain	Remaining amount of buffer (if the buffer reaches the end of stream then the value will be -1)	millisecond
bitrateInstant	Instant stream input bitrate	bit per second
bitrateTarget	Target bitrate for stream playback	bit per second

Developers can use this information for drawing a progress bar and its associated buffering status. Moreover, developers can use bitrateInstant and bitrateTarget for drawing a streaming speed level meter. **The implementation of a streaming speed level meter is strongly required by the NetCast Platform specification, in order that the user can better understand the state of his network connection.**

The following figure graphically illustrates the buffer related variables.



[Figure 6] Graphical explanation of buffer related variables

The following figure is a reference graphical implementation of the streaming speed level meter.

[Figure 7] Graphical implementation example of streaming speed level meter

Example

```
// Example of 'get playback time position'
var playInfo = document. media.mediaPlayInfo();
duration = playInfo.duration;
currentPosition = playInfo.currentPosition;
bufBegin = playInfo.bufBegin;
bufEnd = playInfo.bufEnd;
bufRemain = playInfo.bufRemain;
bitrateInstant = playInfo.bitrateInstant;
bitrateTarget = playInfo.bitrateTarget;
```

See Also

playTime
playPosition

setWidevineXXX

Description

Since NetCast 2.0, setWidevineXXX APIs for setting Credential Information are provided as shown in the following table. All these values are set at Runtime.
Developers must send the credential information through these APIs. If the value is not set through the API, the default value will be empty (null) or zero (0). All setWidevineXXX API arguments are string types.

Developers should call these APIs before playing media content (i.e. before the play() API is called).

Syntax

```
media.setWidevineDrmURL(credentialInfo);
media.setWidevineDeviceID(credentialInfo);
media.setWidevineStreamID(credentialInfo);
media.setWidevineClientIP(credentialInfo);
media.setWidevineUserData(credentialInfo);
media.setWidevineDrmAckURL(credentialInfo);
media.setWidevineHeartbeatURL(credentialInfo);
media.setWidevineHeartbeatPeriod(credentialInfo);
media.setWidevineDeviceType(credentialInfo);
```

Parameters

`credentialInfo`	[in] Credential information

[Table 6] Credential Information list

List	Meaning
DrmServerURL	URL for DRM server
DeviceID	Unique player device ID
StreamID	Unique stream ID
ClientIP	IP address of client
UserData	Additional optional user data
DrmAckServerURL	URL for server that receives entitlement confirmations
HeartbeatURL	URL to receive client heartbeats
HeartbeatPeriod	Duration between consecutive heartbeats in seconds
DeviceType	Device type (default value : 0)

Return Value

None

Example

```
// Example of 'set Widevine Credential information'
var media = document.getElementById("media");
result = media.setWidevineDrmURL(https://drmser.cgi);
result = media.setWidevineStreamID("123");
result = media.setWidevineDeviceID("abcd");
```

2.3.5 Media Player API (Properties)

version

Description

The NetCast Platform provides a version read-only property in the Media Player plugin object. Developers can get version of Media Player Plugin using this property. It will return the Media Player Plugin version information as a string type.

Note

This API works on v3.0.13 or higher version of LG Browser.

Syntax

```
media.version;
```

Example

```
var media = document.getElementById("media");
mediaVersion = media.version;
```

type

Description

The NetCast Platform provides a type read-write property in the Media Player plugin object. Developers can get media type (MIME type) using this property. It will return the MIME type information as a string type.

Syntax

```
media.type;
```

Example

```
var media = document.getElementById("media");
mimeType = media.type;
```

data

Description

The NetCast Platform provides a data read-write property in the Media Player plugin object. Developers can get media URL information using this property. It will return the media URLinformation as a string type.

Syntax

```
media.data;
```

Example

```
var media = document.getElementById("media");
mediaURL = media.data;
```

width / height

Description

The NetCast Platform provides ‘width’ and ‘height’ read-write properties in the Media Player plugin object. It will return the width and height information of the media object as string types.

Developers can set the size of video using ‘media.width = value; media.height = value;’ API.

Developers can also make a full-screen video by specifying the media object size.

Syntax

```
media.width;
media.height;
```

Example

```
// Example of 'get media size information'
var media = document.getElementById("media");
width = media.width;
height = media.height;
```

```
// Example of 'set size'
var media = document.getElementById("media");
media.width = 1280;
media.height = 720;
```

```
// Example of 'setting video full screen'
<body style='margin:0'>
<script language="javascript">
. . .
var media = document.getElementById("media");
media.width = 1280;
media.height = 720;
. . .
</script>

. . .
</body>
```

playTime

Description

The NetCast Platform provides playtime read-only property in the Media Player plugin object. Developers can get play time using this property. It will return the duration of the currently playing media item as a string type in milliseconds.

Syntax

```
media.playTime;
```

Example

```
var media = document.getElementById("media");
playTime = media.playTime;
```

playPosition

Description

The NetCast Platform provides playPosition read-only property in the Media Player plugin object. Developers can get play position using this property. It will return the play position of the currently playing media item as a string type in milliseconds.

Syntax

```
media.playPosition;
```

Example

```
var media = document.getElementById("media");
playPosition = media.playPosition;
```

playState

Description

The NetCast Platform provides playState read-only property in the Media Player plugin object. Developers can get play state using this property. It will return the play state of the currently playing media item as an enumerated number. See the following table for the mapping rule between the resolutions and enumerated return values.

[Table 7] The enumerated value of play state

Play State	Enumerated Return Value
0	Stopped
1	Playing
2	Paused
3	Connecting
4	Buffering
5	Finished
6	Error

Syntax

```
media.playState;
```

Example

```
var media = document.getElementById("media");
playState = media.playState;
```

error

Description

The NetCast Platform provides error read-only property in the Media Player plugin object. Developers can get error code using this property. Developers can get the error code using the API if the Media Player plugin meets an error while the current media file is playing.

[Table 8] The error code

Error code	Description
0	A/V format not supported
1	Cannot connect to server or connection lost
2	Unidentified error
1000	File is not found
1001	Invalid protocol
1002	DRM failure
1003	Play list is empty
1004	Unrecognized play list
1005	Invalid ASX format
1006	Error in downloading play list
1007	Out of memory
1008	Invalid URL list format
1009	Not playable in play list
1100	Unidentified WM-DRM error
1101	Incorrect license in local license store
1102	Fail in receiving correct license from server
1103	Stored license is expired

Syntax

```
media.error;
```

Example

```
var media = document.getElementById("media");
errorCode = media.error;
```

See Also

onError

autoStart

Description

The NetCast Platform provides autoStart read-write property in the Media Player plugin object. Developers can get and set autoStart property. Developers should set the value to true if the media file playout is to be started automatically.

Syntax

```
media.autoStart;
```

Example

```
var media = document.getElementById("media");
autoStart = media.autoStart; // read
```

```
// Example of 'set autoStart of media'
<object type=" application/x-netcast-av"
    data="http://192.168.1.50/example.wmv"
    width="1280"
    height="720"
    autoStart = "true"
    id="media">
</object>  //write
```

isScannable

Description

The NetCast Platform provides isScannable read-only property in the Media Player plugin object. Developers can get 'isScannable' property. If the value of this property is true, the current media can be scanned (fastforward or rewind). If a media file has not been opened, the value of this property will be false. The media can be scanned only if media is indexed and delivered via the MMS protocol.

Syntax

```
media.isScannable;
```

Example

```
var media = document.getElementById("media");
isScanable = media.isScannable;
```

See Also

play (Trick mode play)

speed

Description

The NetCast Platform provides ‘speed’ read-only property in the Media Player plugin object. Developers can get ‘speed’ property. The value of this property is the relative playback speed of the media file currently being played. Its value is 0 if the playState is not 1 (playing). The speed can differ from 1 only if the media is indexed and delivered via the MMS protocol.

Syntax

```
media.speed;
```

Example

```
var media = document.getElementById("media");
speed = media.speed;
```

See Also

isScannable
playState

bufferingProgress

Description

The NetCast Platform provides ‘bufferingProgress’ read-only property in the Media Player plugin object. Developers can get ‘bufferingProgress’ property. The value of this property is the percentage of buffering complete. Each time playback stops and restarts, this property may decrease or increase. It does not vary if playback is paused. This property returns a valid value only after a media file has been opened and decoding starts.

Note

The value of this property is not very accurate. It is just informative. Therefore, developers should not use the value of this property for logical decisions in an application. For example, an application should not enable and disable a loading message using the value of this property. In this example, it would be better for developers to use the onBuffering event instead of this property. See onBuffering.

Syntax

```
media.bufferingProgress;
```

Example

```
var media = document.getElementById("media");
bufferingProgress = media.bufferingProgress;
```

See Also

onBuffering
mediaPlayInfo

subtitleOn

Description

The NetCast Platform provides ‘subtitleOn’ read-write property in the Media Player plugin object. Developers can get and set ‘subtitleOn’ property. The value of this property is the status of subtitle decoder and is a Boolean type. LG Smart TV applications can turn the subtitle decoder on and off by setting this property with “true” or “false” values respectively.

The subtitle must be applied when a full size video is being played.

Syntax

```
media.subtitleOn;
```

Example

```
var media = document.getElementById("media");

//get subtitleOn property
subtitleOn = media.subtitleOn;

//set subtitleOn property
media.subtitleOn = newSubtitleOn;
```

subtitle

Description

The NetCast Platform provides ‘subtitle’ read-write property in the Media Player plugin object. Developers can get and set ‘subtitle’ property. The value of this property is the URL of the subtitle file. The media player retrieves the subtitle file before decoding the media file. LG Smart TV applications set this property every time a new subtitle file is required.

The subtitle must be applied when a full size video is being played.

Note

This function is available since NetCast 2.0 or higher.

Characters other than ASCII are recommended to be encoded with UTF-8. (ISO8859-* or UTF-16/32 may not work normally.)

Syntax

```
media.subtitle;
```

Example

```
var media = document.getElementById("media");

//get subtitle property
subtitleURL = media.subtitle;

//set subtitle property
media.subtitle = newSubtitleURL;
```

mode3D

Description

The NetCast Platform provides 'mode3D' read-only property in the Device Information / Media Player plugin object. Developers can get a 3D mode. It will return the current 3D format for 3D mode. Developers should get the value when TVs are presently in 3D mode. See the following table for available values about 3D format.

[Table 9] Available values for "mode3D" property

Variable	Meaning
off	Original 2D Format
from_2d_to_3d	2D-to-3D Conversion Format
side_by_side	Side-by-Side Frame Compatible 3D (Left / Right)
side_by_side_rl	Side-by-Side Frame Compatible 3D (Right / Left)
top_bottom	Top-and-Bottom (or Over-Under) Frame Compatible 3D
checker_bd	Checker board Frame Compatible 3D only available for HD format (1080p @ 30Hz)

Syntax

```
media.mode3D;
```

Example

```
var media = document.getElementById("media");
triMode= media.mode3D;
```

audioLanguage

Description

The NetCast Platform provides 'audioLanguage' write-only property that is used for selecting audio language when media content has the multi-audio language.

Note

The value of this property shall be exactly matched with the language code of audio track in media container. Matching of the two language codes is developer's own responsibility. NetCast Platform does only search the matched audio track in media container.

Note

When video is paused, this API does not work normally.

Note

After the audio is changed, audio mute may occur for a few seconds due to the synchronization of video and audio.

Syntax

```
media.audioLanguage;
```

Example

```
// Example of 'set audio language information of media'
<object type="video/x-ms-wmv"
    data="http://192.168.1.50/example.wmv"
    width="1280"
    height="720"
    audioLanguage="en"
    id="media">
</object>
```

```
// Example of 'set audio language information of media'
<script>
    var media = document.getElementById("media");

    // set audioLanguage property
    media.audioLanguage = newAudioLanguage
</script>
```

2.3.6 Media Player API (Events)

onPlayStateChange

Description

The NetCast Platform provides an onPlayStateChange event in the Media Player plugin object. Developers can receive play state change event. Developer can receive a play state change event when the play state of currently playing media item is changed.

To refer to the values of the playState property, see playState.

Syntax

```
media.onPlayStateChange = processPlayStateChangeFunction;
```

Example

```
<script language='javascript'>
function processPlayStateChangeFunction()
{
    . . .
    // read and process playState property
    . . .
}
</script>

<object type="video/x-ms-wmv"
```

```
    data="http://192.168.1.50/example.wmv"
    width="1280"
    height="720"
    id="media">
</object>
<script>
    var media = document.getElementById('media');
    media.onPlayStateChange = processPlayStateChangeFunction;
</script>
```

See Also

playState

onBuffering

Description

The NetCast Platform provides an onBuffering event in the Media Player plugin object. Developers can receive buffering event. Developers can receive a buffering event when the media player begins and ends buffering. A Boolean type parameter specifies whether data buffering has started or finished. A value of true indicates that the data buffering has started. Buffering also occurs whenever playback stops and then restarts (either from calls to play() and stop()) methods or when network congestion occurs during playing streamed media.

Syntax

```
media.onBuffering = processBufferingFunction;
```

Example

```
<script language='javascript'>
function processBufferingFunction(isStarted)
{
    . . .
    // process buffering
    . . .
}
</script>

<object type="video/x-ms-wmv"
    data="http://192.168.1.50/example.wmv"
    width="1280"
    height="720"
    id="media">
</object>
<script>
    var media = document.getElementById('media');
    media.onBuffering = processBufferingFunction;
</script>
```

See Also

bufferingProgress

onError

Description

The NetCast Platform provides an onError event in the Media Player plugin object. Developers can receive error event. Developers can receive an error event when the media player encounters an error while playing.

Syntax

```
media.onError = processErrorFunction;
```

Example

```
<script language='javascript'>
function processErrorFunction ()
{
    . . .
    // read and process error codes
    . . .
}
</script>

<object type="video/x-ms-wmv"
    data="http://192.168.1.50/example.wmv"
    width="1280"
    height="720"
    id="media">
</object>
<script>
    var media = document.getElementById('media');
    media.onError = processErrorFunction;
</script>
```

2.4 Device Info Plugin and API

The NetCast Platform supports device information plugin. The following sub-sections describe the API of the device information plugin. See also section 2.2.2 userAgent String in **"LG Web Application Development Guide"** for supporting feature information.

2.4.1 Device Info Object

An example MIME type of the device information object is application/x-netcast-info for the NetCast Platform device information plugin. The device information object supports a single propery called 'id'.

The NetCast Platform supports only one instance of the device information object at the same time so developers must not use more than one device information object.

```
// Example of device object in HTML
<object type="application/x-netcast-info"
     id="device">
</object>
```

2.4.2 Device Info API (Methods)

setDrmLicenseInfo

Description

The NetCast Platform provides a 'setDrmLicenseInfo()' API in the Device Info plugin object. Developers can set DRM license information. It will return '1' if the call is processed without error or '0' if there is an error in processing the call or license setup is not completed. Therefore, applications should wait until they receive a '1' before proceeding to the next step.

Note

This API is only applicable for the pre-delivery method of WMDRM.

Syntax

```
device.setDrmLicenseInfo(licenseInfo);
```

Parameters

`licenseInfo`	[in] License information

Return Value

Returns '1' if the call is processed without error or '0' if there is an error in processing the call or license setup is not completed.

Example

```
var device = document.getElementById("device");
callSuccess = device.setDrmLicenseInfo("_license_info_string_");
```

See Also

drmClientInfo

2.4.3 Device Info API (Properties)

version

Description

The NetCast Platform provides a 'version' read-only property in the Device Info Plugin object. Developers can get version of Device Info Plugin. It will return the string type of Device Info Plugin version information as a string type.

Note

This API works on v3.0.13 or higher version of LG Browser.

Syntax

```
device.version;
```

Example

```
var device = document.getElementById("device");
deviceVersion = device.version;
```

manufacturer

Description

The NetCast Platform provides a 'manufacturer' read-only property in the Device Information plugin object. Developers can get manufacturer ID using this property. It will return "LGE" as a string.

Syntax

```
device.manufacturer;
```

Example

```
var device = document.getElementById("device");
manufacturerId = device.manufacturer;
```

modelName

Description

The NetCast Platform provides a ‘modelName’ read-only property in the Device Information plugin object. Developers can get model name using this property. It will return the model name as a string type.

Syntax

```
device.modelName;
```

Example

```
var device = document.getElementById("device");
modelName = device.modelName;
```

serialNumber

Description

The NetCast Platform provides a ‘serialNumber’ read-only property in the Device Information plugin object. Developers can get serial number using this property. It will return the serial number of the product as a string type.

Syntax

```
device.serialNumber;
```

Example

```
var device = document.getElementById("device");
serialNumber = device.serialNumber;
```

swVersion

Description

The NetCast Platform provides a ‘swVersion’ read-only property in the Device Information plugin object. Developers can get software version using this property. It will return the software version of the product as a string type.

Syntax

```
device.swVersion;
```

Example

```
var device = document.getElementById("device");
swVersion = device.swVersion;
```

hwVersion

Description

The NetCast Platform provides a ‘hwVersion’ read-only property in the Device Information plugin object. Developers can get hardware version using this property. It will return the hardware version of the product as a string type.

Syntax

```
device.hwVersion;
```

Example

```
var device = document.getElementById("device");
hwVersion = device.hwVersion;
```

osdResolution

Description

The NetCast Platform provides a ‘osdResolution’ read-only property in the Device Information plugin object. Developers can get OSD resolution which LG Browser uses for rendering web content using this property. It will return the OSD resolution as an enumerated number. The following table shows the mapping rules between the resolutions and enumerated return values.

[Table 10] Mapping table between the resolutions and enumerated return values

OSD resolution	Enumerated return value
640x480	0
720x576	1
1280x720	2
1920x1080	3
1366x768	4

Syntax

```
device.osdResolution;
```

Example

```
var device = document.getElementById("device");
osdResolution = device.osdResolution;
```

networkType

Description

The NetCast Platform provides a 'networkType' read-only property in the Device Information plugin object. Developers can get network type using this property. It will return the network type as an enumerated number. The following table shows the mapping rules between network types and enumerated return values.

[Table 11] Mapping table between network types and enumerated return values

Network type	Numeric type of return value
Wired network	0
Wireless network	1

Syntax

```
device.networkType;
```

Example

```
var device = document.getElementById("device");
networkType = device.networkType;
```

net_macAddress

Description

The NetCast Platform provides a 'net_macAddress' read-only property in the Device Information plugin object. Developers can get MAC address using this property. It will return MAC address as a string type.

Syntax

```
device.net_macAddress;
```

Example

```
var device = document.getElementById("device");
macAddress = device.net_macAddress;
```

drmClientInfo

Description

The NetCast Platform provides a 'drmClientInfo' read-only property in the Device Information plugin object. Developers can get DRM client information using this property. It will return the DRM client information as a string type.

Note

This API is only applicable for the pre-delivery method of WMDRM.

Syntax

```
device.drmClientInfo;
```

Example

```
var device = document.getElementById("device");
drmClientInfo = device.drmClientInfo;
```

See Also

setDrmLicenseInfo

net_dhcp

Description

The NetCast Platform provides a 'net_dhcp' read-only property in the Device Information plugin object. Developers can get DHCP information using this property. It returns true if the system uses DHCP to access the network.

Syntax

```
device.net_dhcp;
```

Example

```
var device = document.getElementById("device");
useDHCP = device.net_dhcp;
```

net_isConnected

Description

The NetCast Platform provides a 'net_isConnected' read-only property in the Device Information plugin object. Developers can get network connection information using this property. It returns true when the system is connected to the network. "Connected" means the status when an ethernet cable is conneted and Internet is available.

Syntax

```
device.net_isConnected;
```

Example

```
var device = document.getElementById("device");
isConnected = device.net_isConnected;
```

net_hasIP

Description

The NetCast Platform provides a 'net_hasIP' read-only property in the Device Information plugin object. Developers can get IP information using this property. It returns true if the system has valid IP address. It returns false if the IP address has been set by DHCP.

Syntax

```
device.net_hasIP;
```

Example

```
var device = document.getElementById("device");
hasIP = device.net_hasIP;
```

net_ipAddress

Description

The NetCast Platform provides a 'net_ipAddress' read-only property in the Device Information plugin object. Developers can get IP address using this property. It returns true if the system has valid IP address. It returns the IP address of the system as a string type.

Syntax

```
device.net_ipAddress;
```

Example

```
var device = document.getElementById("device");
ipAddress = device.net_ipAddress;
```

net_netmask

Description

The NetCast Platform provides a 'net_netmask' read-only property in the Device Information plugin object. Developers can get netmask using this property. It returns the netmask value of the system as a string type.

Syntax

```
device.net_netmask;
```

Example

```
var device = document.getElementById("device");
netmask = device.net_netmask;
```

net_gateway

Description

The NetCast Platform provides a 'net_gateway' read-only property in the Device Information plugin object. Developers can get gateway address using this property. It returns the gateway address value of the system as a string type.

Syntax

```
device.net_gateway;
```

Example

```
var device = document.getElementById("device");
netmask = device.net_gateway;
```

net_dns1

Description

The NetCast Platform provides a 'net_dns1' read-only property in the Device Information plugin object. Developers can get DNS1 address using this property. It returns the DNS1 address value of the system as a string type.

Syntax

```
device.net_dns1;
```

Example

```
var device = document.getElementById("device");
dns1 = device.net_dns1;
```

net_dns2

Description

The NetCast Platform provides a 'net_dns2' read-only property in the Device Information plugin object. Developers can get DNS2 address using this property. It returns the DNS2 address value of the system as a string type.

Syntax

```
device.net_dns2;
```

Example

```
var device = document.getElementById("device");
dnss = device.net_dns2;
```

supportMouse

Description

Some LG Smart TVs support a pointing remote control unit, called the Magic-RCU (Magic Remote Control Unit). Developers can check whether or not supports the Magic-RCU in TV by using the 'supportMouse' read-only property in the Device Information plugin object. It returns true if LG Smart TV supports Magic-RCU, otherwise, it returns false.

See also section 3.4 Input Device in **"LG Web Application Development Guide"** for detail information about Magic-RCU.
See also section 2.2.2 userAgent String in **"LG Web Application Development Guide"** for optional feature supporting.

Note

This API works on v3.0.23 or higher version of LG Browser.

Syntax

```
device.supportMouse;
```

Example

```
var device = document.getElementById("device");
if(device.supportMouse) {
    supportMouse = device.supportMouse;
}
```

support3D

Description

Some LG Smart TVs support 3D technology. For 3D videos, it is required that developers to check whether the TV has a 3D rendering ability. The NetCast Platform provides a 'support3D' read-only property in the Device Information plugin object. It returns true if LG Smart TV supports 3D rendering ability, otherwise, it returns false.

See also section 2.2.2 userAgent String in **"LG Web Application Development"** for optional feature support.

Syntax

```
device.support3D;
```

Example

```
var device = document.getElementById("device");
if(device.support3D) {
    // can do something for 3D video
}
```

support3DMode.from_2d_to_3d

Description

The NetCast Platform provides ‘support3DMode.from_2d_to_3d’ read-only property in the Device Information plugin object. Developers can check compatibility of 2D-to-3D format using this property. It will return ‘true’ if NetCast Platform supports a 2D-to-3D format for 3D mode. Otherwise, it returns ‘false’.

Syntax

```
device.support3DMode.from_2d_to_3d;
```

Example

```
var device = document.getElementById("device");
if(device.support3DMode.from_2d_to_3d){
    // set the 3D mode to 2D-to-3D
}
```

support3DMode.side_by_side

Description

The NetCast Platform provides ‘support3DMode.side_by_side’ read-only property in the Device Information plugin object. Developers can check compatibility of Side-by-Side (L/R) format. It will return ‘true’ if NetCast Platform supports a Side-by-Side format (Left / Right) for 3D mode. Otherwise it returns ‘false’.

Syntax

```
device.support3DMode.side_by_side;
```

Example

```
var device = document.getElementById("device");
if(device.support3DMode.side_by_side){
    // set the 3D mode to Side-by-Side(L/R)
}
```

support3DMode.side_by_side_rl

Description

The NetCast Platform provides ‘support3DMode.side_by_side_rl' read-only property in the Device Information plugin object. Developers can check compatibility of Side-by-Side (R/L) format. It will return ‘true’ if NetCast Platform supports a Side-by-Side format (Right /Left) for 3D mode. Otherwise it returns ‘false’.

Syntax

```
device.support3DMode.side_by_side_rl;
```

Example

```
var device = document.getElementById("device");
if(device.support3DMode.side_by_side_rl){
    // set the 3D mode to Side-by-Side(R/L)
}
```

support3DMode.top_bottom

Description

The NetCast Platform provides ‘support3DMode.top_bottom' read-only property in the Device Information plugin object. Developers can check compatibility of Top-and-Bottom format. It will return ‘true’ if NetCast Platform supports a Top-and-Bottom format for 3D mode. Otherwise it returns ‘false’.

Syntax

```
device.support3DMode.top_bottom;
```

Example

```
var device = document.getElementById("device");
if(device.support3DMode.top_bottom) {
    // set the 3D mode to Top-and-Bottom
}
```

support3DMode.checker_bd

Description

The NetCast Platform provides ‘support3DMode.checker_bd' read-only property in the Device Information plugin object. Developers can check compatibility of Checker Board format. It will return ‘true’ if NetCast Platform supports a Checker Board format for 3D mode. Otherwise it returns ‘false’. Please be noted that the Checker Board format will work properly only for 1080 p/30 Hz of HD resolution.

Syntax

```
device.support3DMode.checker_bd;
```

Example

```
var device = document.getElementById("device");
if(device.support3DMode.checker_bd) {
    // set the 3D mode to Checker Board
}
```

preferredSubtitleLanguage

Description

This read-only property returns the value of subtitle, which has been set through TV setting menu. The return value is based on ISO 639-2 language code.

Note

In some region, the TV setting menu does not have subtitle language setting menu. Therefore, the property may not have the value.

Syntax

```
device.preferredSubtitleLanguage;
```

Example

```
var device = document.getElementById("device");
preferredSubtitleLanguage = device.preferredSubtitleLanguage;
```

preferredAudioLanguage

Description

This read-only property returns the value of audio language, which has been set through TV setting menu by user. The return value is based on ISO 639-2 language code.

Note

In some region, the TV setting menu does not have audio language setting menu. Therefore, the property may not have the value.

Syntax

```
device.preferredAudioLanguage;
```

Example

```
var device = document.getElementById("device");
preferedAudioLanguage = device.preferredAudioLanguage;
```

preferredSubtitleStatus

Description

This read-only property returns the on/off status of subtitle, which has been set through TV setting menu.

Note

In some region, the TV setting menu does not have subtitle on/off setting menu. Therefore, the property may not have the value.

Syntax

```
device.preferredSubtitleStatus;
```

Example

```
var device = document.getElementById("device");
isSubtitleOn = device.preferredSubtitleStatus;
```

tvLanguage2

Description

This read-only property returns the value of language, which has been set through TV setting menu by user. The value will be returned in maximum 2 bytes (e.g. en) and is based on ISO 639-1.

Syntax

```
device.tvLanguage2;
```

Example

```
var device = document.getElementById("device");
tvLanguage = device.tvLanguage2;
```

tvCountry2

Description

This read-only property returns the value of country, which has been set through TV setting menu by user. The value will be returned in maximum 2 bytes (e.g. en) and is based on ISO 3166.

Syntax

```
device.tvCountry2;
```

Example

```
var device = document.getElementById("device");
tvCountry= device.tvCountry2;
```

2.5 Broadcast Plugin and API

Broadcast plugin is used for displaying TV broadcast images in the application. The plugin provides APIs for changing channels.

Broadcast Plugin and API is supported since NetCast 3.0.

2.5.1 Broadcast Object

The MIME type of Broadcast object is ‘application/x-netcast-broadcast’. Properties of broadcast object are ‘id’, ‘width’, and ‘height’.

The following sample code is an example of broadcast object in HTML.

```
<object type="application/x-netcast-broadcast"
        id="broadcast"
        width="500"
        height="400">
</object>
```

2.5.2 Broadcast API (Methods)

channelUp / channelDown

Description

This API is used for changing channels.

> **Note**
>
> To change a channel with number keys of remote control unit, use window.NetCastEnableNumberKey(false) to make number keys transferred to DTV, not application.

Syntax

```
broadcast.channelUp();
broadcast.channelDown();
```

Parameters

None

Return Value

None

Example

```
Function processKeyDown(e)
{
    switch(e.keyCode)
    {
           case VK_PAGE_UP:
                  broadcast.channelUp();
                  break;
```

```
            case VK_PAGE_DOWN:
                  broadcast.channelDown();
                  break;
            ...
      }
}
```

getCurrentChannelName

Description

This API is used for obtaining the name of current channel.

Syntax

```
broadcast.getCurrentChannelName();
```

Parameters

None

Return Value

Return value is the name of current channel. (String type)

Example

```
var channelName = broadcast.getCurrentChannelName();
```

getCurrentProgram

Description

This method is used to view the title, start time, end time and description of the program broadcasted on the current channel.

Syntax

```
broadcast.getCurrentProgram();
```

Parameters

None

Return Value

Returns an object type value.
The properties of the object are the title, start time, end time and description of the program broadcasted on the current channel and are returned as string. The time value is displayed in the format of yyyy/mm/dd hh:mm:ss.

Example

```
var currentProgram = broadcast.getCurrentProgram();

var title = currentProgram.title;
var startTime = currentProgram.startTime;
var endTime = currentProgram.endTime;
```

```
var description = currentProgram.description;
```

getCurrentChannelNumber

Description

This method is used to view the physical number, major number, minor number, program number, source ID and TSID (Transport Stream ID) of the current channel.

Syntax

```
broadcast.getCurrentChannelNumber()
```

Parameters

None

Return Value

Returns an object type value.
The properties of the object are the physical number, major number, minor number, program number, source ID and TSID of the program broadcasted on the current channel and are returned as int.

Note

If the TSID is requested upon channel change, an invalid value of 0 may be returned. In this case, call this method again to get a valid value.

Example

```
var currentChannel= broadcast.getCurrentChannelNumber();

var physicalNum   = currentChannel.physicalNum;
var majorNum      = currentChannel.majorNum;
var minorNum      = currentChannel.minorNum;
var programNum    = currentChannel.programNum;
var sourceId      = currentChannel.sourceId;
var tsId   = currentChannel.tsId;
```

isChannelMapEmpty

Description

This method is used to determine whether there is a channel map of the current TV. No channel map exists for the TV that has been reset.

Syntax

```
broadcast.isChannelMapEmpty()
```

Parameters

None

Return Value

Returns True if there is the channel map of the current TV; otherwise, False.

Example

```
var checkChannelMapEmpty =  broadcast.isChannelMapEmpty();
```

isTunerInput

Description

This method is used to determine whether the current input signal is from TV.

Syntax

```
broadcast.isTunerInput()
```

Parameters

None

Return Value

Returns True if the current input signal is from TV; otherwise, False.

Example

```
var checkTunerInput = broadcast.isTunerInput()
```

isDvb

Description

This method is used to determine whether the TV signal complies with DVB or ATSC standard.

Syntax

```
broadcast.isDvb()
```

Parameters

None

Return Value

Returns True if the TV signal complies with DVB standard; otherwise, False.

Example

```
var checkDvb = broadcast.isDvb();
```

2.5.3 Broadcast API (Events)

onchannelchange

Description

This event occurs when a TV channel is changed. The message value of this event is passed as an object. The properties of the object are the physical number, major number, minor number, program number, source ID and TSID of the program broadcasted on the current channel and are returned as int.

Syntax

```
broadcast.onchannelchange = processChannelChangeFunction;
```

Example

```
<script language='javascript'>
function processChannelChangeFunction(e)
{
    If(e) {
     // receive the channel number
          var physicalNum = e.physicalNum;
          var majorNum = e.majorNum;
          var minorNum = e.minorNum;
          var programNum = e.programNum;
          var sourceId = e.sourceId;
          var tsId     = e.tsId;
    }
. . .
}
</script>
```

onnosignal

Description

This event occurs when there is no signal of the TV tuner. The message value of this event is success.

Syntax

```
broadcast.onnosignal = processNoSignalFunction;
```

Example

```
<script language='javascript'>
function processNoSignalFunction(e)
{
    If(e) {
    // update information when TV goes no signal
    }

. . .
}
```

2.6 Voice Recognition Plugin and API

The voice recognition plugin is used to use the voice recognition function of the Magic Remote Control on web applications. The plugin provides the API for voice recognition (converting voice into text). There are two modes for the voice recognition function: dictation and keyword. (In keyword mode, the search keyword is selected from the voice recognition word list.) The language of the voice recognition function can be set under OPTION > Language > Voice Search Language.

The voice recognition plugin and API are supported on NetCast 3.0 1st SU or later. The models with the voice recognition enabled are L9 (Korea/North America) and MTK (Korea).

2.6.1 Voice Recognition Object

The MIME type of voice recognition object is ‘application/x-netcast-voice’. The properties of voice recognition object are ‘id’, ‘dictation’.
The dictation property can be used when the voice recognition function is in dictation mode. The default is the keyword mode.

The following sample code is an example of Voice Recognition object in HTML.

```
<object type="application/x-netcast-voice"
        id="voice"
        dictation=”on”>
</object>
```

2.6.2 Voice Recognition API (Property)

isInitialized

Description

This method is used to determine whether the voice recognition function is initialized and returns the value as Boolean.

Syntax

```
voice.isInitialized;
```

Example

```
var voice = document.getElementById(‘voice’);
var isInitialized = voice.isInitialized;
```

isEnable

Description

This method is used to determine whether the MRCU is paired (including its type) and whether the voice recognition function is enabled, and returns the value as Boolean.

Syntax

```
voice.isEnable;
```

Example

```
var voice = document.getElementById('voice');
var isEnable = voice.isEnable;
```

dictation

Description

This method is used to determine whether the voice recognition function is in dictation mode and returns the value as string (on/off). If the return value is "off", the function is in keyword mode. This methods can also be used to select the mode.

Syntax

```
voice.dictation;
```

Example

```
var voice = document.getElementById('voice');
var dictationMode = voice.dictation;
voice.dictation = "on"; // dictation mode
voice.dictation = "off"; // keyword mode
```

2.6.3 Voice Recognition API (Methods)

startRecognition

Description

This method is used to call native UI of the voice recognition function and receive the result as an event.

Syntax

```
voice.startRecognition();
```

Parameters

None

Return Value

None

Example

```
function startVoiceRecognition()
{
    var voice = document.getElementById('voice');
    voice.startRecognition();
}
```

2.6.4 Voice Recognition API (event)

onrecognizevoice

Description

This event is added in order to receive the voice recognition result from the TV.

Syntax

```
voice.onrecognizevoice;
```

Example

```
function initPage()
{
    var voice = document.getElementById('voice');
    voice.onrecognizevoice = function(e){
          document.write(e);
    };
}
```

onbuttonenable

Description

The event is added in order to enable or disable the voice recognition button. Upon pairing, the event receives the availability of the voice recognition according to the MRCU type from the TV. If the MRCU with the voice recognition disabled is paired with the RV, false is returned; otherwise, true. The voice recognition button is enabled or disabled based on the value returned by this event.

Syntax

```
voice.onbuttonenable;
```

Example

```
function initPage()
{
    var voice = document.getElementById('voice');
    var button = document.getElementById('button');
    voice.onbuttonenable = function(e){
          button.disabled = !e;
    };
}
```

2.7 Media Device NetCast API

This API is supported only for PDP TV and Media device, not for LCD/LED TV device.

2.7.1 Media Device API (Methods)

window.NetCastSetScreenSaver

Description

This API can be used for developers to control the function of 'Screen Saver' in LG device.
The life cycle of 'Screen Saver' operation is same as the application's life cycle. The value will be set as the default value, enable 'Screen Saver', when users exit the application. However, the developer should enable the operation explicitly in their implementation.

The operations have distinctive screen saving animation of their own. The animation starts after a defined time passes. An application cannot be shown after the operation starts, because 'Screen Saver' animation is displayed on the top layer of all OSDs.
For an example of operation, Media device's 'Screen Saver' appears when you leave the player in Stop mode for about five minutes.

If developers do not want to use the function of 'screen saver' while application is running, developers can use this API.
Developers can use this API, window.NetCastSetScreenSaver(), to disable the function of 'screen saver' while application is running.

Note

- Developers have to set 'enable' value of this API, when the application ends.
- This API works on 4.1.2 or higher version of LG Browser.

Syntax

```
window.NetCastSetScreenSaver(control)
```

Parameters

control	[in] 'enabled' or 'disabled'

Return Value

None

Example

```
// Enabling the function of Screen Saver
function enableDeviceScreenSaver()
{
    window.NetCastSetScreenSaver('enabled');
}
```

```
// Disabling the function of Screen Saver
disableDeviceScreenSaver()
{
    window.NetCastSetScreenSaver('disabled');
}
```

Annex A Differences in Media Devices

Media products partly support NetCast 3.0. Please see the detailed information below for Media products development.

Chapter of this Document	Constraints and difference of Media Devices compared with TV
1.1 Coverage	A. Media products do not provide following NetCast APIs - window.NetCastLaunchQMENU - window.NetCastLaunchRATIO B. Media products provide following Media Player Plugin (Methods) APIs - setWidevinePortalID - setWidevineStoreFront
2.1 NetCast API	A. window.NetCastSetDefaultAspectRatio : Not applicable (Media products set Aspect Ratio at Device Setup menu) B. Media specific API (window.NetCastSetScreenSaver(control)): Refer to section 2.6 Media Device NetCast API. C. window.NetCastLaunchQMENU(), window.NetCastLaunchRATIO() : Not applicable.
2.3.1 Media Object	A. mode3D : from Table3, Media products do not support variable “off” and “from_2d_to_3d”.

2.3.4 Media Player Plugin (Methods)	A. In “Set Widevine Credential Information”, the followings are added.
	i. media.setWidevinePortalID("Portal");
	ii. media.setWidevineStoreFront("StoreFront");
	
	B. In example, the followings are added.
	i. UserData (Portal), additional optional user data (Identifies the operator)
	ii. UserData (Storefront), additional optional user data (Identifies store run by operator)
	
	C.in Table 6, the followings are added:
	- List: Portal, Meaning: Identifies the operator
	- List: Storefront, Meaning: Identifies store run by operator

2.3.5 Media Player API (Properties)	A. error - ErrorCode (1200) : Verimatrix failure B. mode3D - Media products do not support variable “off” and “from_2d_to_3d”. - Media products only support “checker_bd” for HD format (1080p @ 24Hz).
2.4.2 Device Info Plugin (Methods)	A. getResponseFailMsg() is added.
2.5 Broadcast Plugin and API	A. Media devices do not use Broadcast Plugin and API.