Developing Web App

This document describes how to develop web-based application for LG Smart TV.

Quick Start Guide

This document is provided for beginners who want to start web app developing using LG IDE. With simple sample application, we explain how to setup the project and implement the codes.

App Development Guide

This document provides description on the NetCast platform architecture, supported media formats, protocols, minimum LG NetCast Platform capabilities and the HTML5/CSS specifications. Also, performance considerations are provided which web application developers should know.

Supported HTML5/CSS3 Standard for the NetCast Platform and Emulator

This document describes HTML5 and CSS3 supported by the NetCast platform and LG Smart TV Emulator.

Sample Tutorials: LG WebAPI Tutorials

This document includes sample tutorials for LG WebAPI Tutorial codes.

Contents

Quick Start Guide

This document is provided for beginners who want to start web app developing using LG IDE. With simple sample application, we explain how to setup the project and implement the codes.

The following samples and tutorials are provided in [DISCOVER > Legacy Platform (NetCast) > Tools & Samples] menu in LG Developer website.

• Web_Quick Start Sample App (Tic Tac Toe)
This sample app and tutorial describe how developers develop and test web applications using the Sound API and background music in the application.

• Web_Quick Start Sample App (ImageViewer)
This sample app and tutorial describe how developers develop and test web applications using the Image Viewer Framework API.

• Web_Quick Start Sample App (MediaPluginVideoPlayer)
This sample app and tutorial describe how developers develop and test web applications using the Media Plugin Video Player Framework API.

• Web_Quick Start Sample App (HTML5VideoPlayer)
This sample app and tutorial describe how developers develop and test web applications using the HTML5 Video Player Framework API.

App Development Guide

This document has been prepared to be used by developers as the technical specification for the LG Smart TV.

In this document, the platform of LG Smart TV is referred to as the “NetCast Platform”. This document defines the platform architecture, supported media formats, protocols and minimum LG NetCast Platform capabilities.

This document specifies the NetCast Platform for applications running on an LG Smart TV which can receive broadcasting media (terrestrial, cable and satellite network based) and IP based media.

This document specifies the web based application programming environment for the NetCast Platform.

The NetCast Platform is built on the Linux operating system.

This chapter descibes the following sections.

- LG Web Application Overview
- Specifications
- Tips for Web Application Development

LG Web Application Overview

Web application is an application that is accessed over a network such as the internet or an intranet. LG Smart TV can run web application through built-in web browser engine. Built-in LG web browser engine supports recent web technologies HTML5, Ajax, JSON, XML, and etc. Developers can make creative applications based on those web technologies.

Although developing LG web application is similar in many ways to developing general web application, it is not totally the same. As LG web application is executed by web browser engine, not web browser, there is no web browser’s UI components such as URL input field, scroll bar, system button, and popup. Developers should be aware of the following list:

- UI screen must adopt LG screen resolution size. It means that every element of web application should be displayed in 1280 x 720 pixel screen resolution size. Separate the elements into several pages if displaying elements in a screen is not available since scrolling is not supported.
- The smallest font must be readable and legible to users 3.5 meter (10 foot) away from the TV screen.
- Follow the “overscan area” rule because the overscan area may not be visible on some devices such as LG media devices.
- Application must be fully navigable using 4 direction and remote keys: Up/Down/Left/Right + OK + Back
- Implement the ‘Back’ button in application to move back to the previous NetCast menu.

For more detailed information, refer to the “UI Guidelines” in **Developing > Designing** section in this Library.

Web applications can be deployed in two types, hosted and packaged Web application, to LG Smart TV.

- Hosted (URL-based) web application: Developers can make a web application using just icon and URL. Hosted web application is suitable for the applications containing server-based web service.

- Packaged (downloadable) web application: Suitable for the applications containing small fixed resources such as icon, html, JavaScript, and media. (Supported from NetCast 2.0)

Note

Refer to **Developing > Using SDK** section in this Library for details on LG web application development, packaging, and deployment.

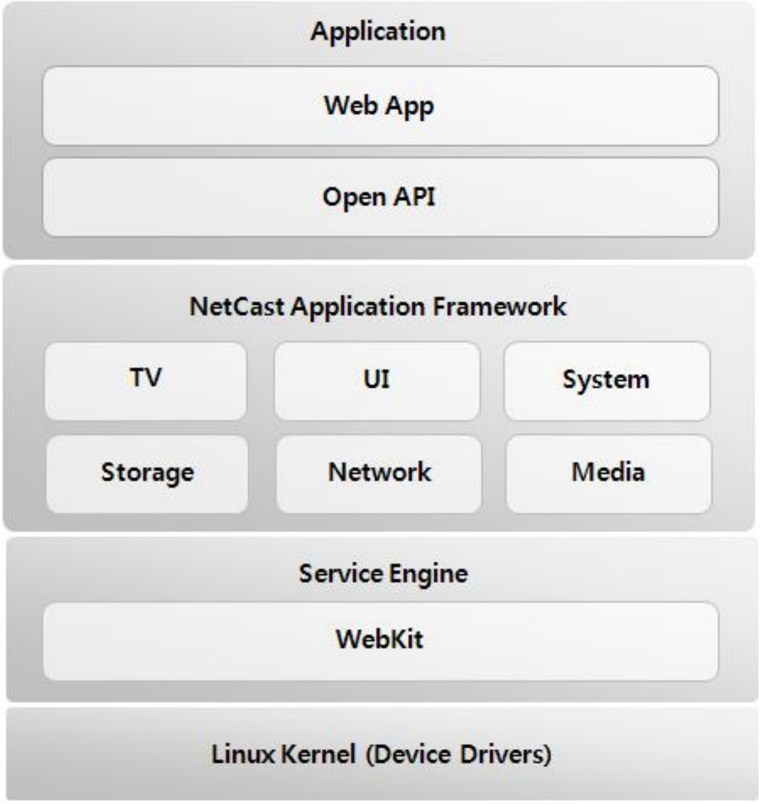
NetCast Platform Overview

The NetCast Platform uses the LG Browser, which is based on WebKit. This is not a restricted embedded web browser but provides full features of web technology.

The platform provides rich media playback functionalities including play, pause, stop and scaling. Various streaming protocols including HTTP and MMSH are supported by the media player plugin module. It also has a DRM manager to manage digital rights and a streaming manager to control the stream. The NetCast Platform supports WM-DRM 10

PD (Not supported in NetCast 3.0), PlayReady (Not supported in NetCast 2.0), Widevine, and Verimatrix (Supported in NetCast 3.0/4.0/4.5) as its DRM solutions. NetCast Platform also supports adaptive streaming solutions. NetCast Platform supports Widevine and HLS (Http Live Streaming) as its adaptive streaming solutions from NetCast 2.0.

The following figure illustrates the architecture of the NetCast Platform system. The NetCast Platform runs on the Linux operating system.



[Figure] System Block Diagram of the NetCast Platform

Specifications

This chapter describes features of LG Browser in addition to generic Web technologies as follows. With proper using of the feature, developer can create interactive LG Smart TV web applications.

- Video Output
- Web Engine
- Recommendations for DRM/Streaming Protocols
- Protocols
- CODEC & Container
- DRM
- OSD Resolution
- Subtitle
- Images
- Cascading Style Sheets
- Streaming Method
- Play List (ASX)
- Local File Play

Video Output

LG Electronics Television sets do not have video output for Web content except to the main panel display. Therefore, there is no need to provide any copy protection data with Web content as such content is only ever rendered to the screen. i.e., HDCP and Macrovision are not required for the NetCast Platform.

Web Engine

The NetCast Platform implements the LG browser engine, which is based on WebKit 537.1+ as an application programming environment for content providers and content aggregators. The supported features are listed in the following table.

[Table] Supported features of LG Web browser engine

Specification	LG Web browser engine
HTTP, HTTPS	Yes
HTML 4.01	Yes
HTML 5.0 [*]	Partly
XHTML 1.0/1.1	Yes
XML	Yes
XSLT	Yes
XPath	Yes
CSS 2.1 [*]	Yes
CSS 3 [*]	Partly
CSS 2D/3D Transforms, Transitions, Animation [*]	Partly
CSS TV Profile 1.0	Yes
Canvas [*]	Partly
SVG [*]	Partly
DOM 1 and 2	Yes
DOM 3	Partly
JavaScript (version > 1.6)	Yes
XMLHttpRequest (AJAX)	Yes

Specification	LG Web browser engine
JSON	Yes
Java	No

* Partly: a level of satisfying some of specification (60~90%).

[*] Refer to Supported HTML5/CSS3 Standard for the NetCast Platform and Emulator for detailed information.

Cookie and Cache Support of LG Web Browser Engine

The LG web browser engine supports cookies and provides a cache for enhanced performance. The cache is used only when the web browser engine is running and no more used if the application is terminated.

userAgent String[1]

The LG web browser engine supports a userAgent String as shown below.

```
Mozilla/5.0 ( operating system information ) AppleWebKit/WebKit version; LG Browser/LG
Browser version(capabilities; LGE; modelName; softwareVersion; hardwareVersion;);
LG_NetCastVersion
```

A JavaScript application can get the string from the HTML DOM property, navigator.userAgent. The same string will be included in the HTTP request User-Agent header. The following shows an example of the userAgent string of LG browser engine.

<NetCast 2.0>

```
Mozilla/5.0 (DirectFB; U; Linux mips; en) AppleWebKit/531.2+ (KHTML, like Gecko,
Safari/531.2+); LG Browser/4.0.9(+mouse+3D+SCREEN+TUNER; LGE; ModelName; 03.05.04;
0x00000001;); LG NetCast.TV-2011
```

<NetCast 3.0>

```
Mozilla/5.0 (DirectFB; Linux armv7l) AppleWebKit/534.26+ (KHTML, like Gecko)
Version/5.0 Safari/534.26+ LG Browser/5.00.00(+mouse+3D+SCREEN+TUNER; LGE; ModelName;
09.00.00; 0x00000001;); LG NetCast.TV-2012
```

<NetCast 4.0>

```
Mozilla/5.0 (Unknown; Linux armv7l) AppleWebKit/537.1+ (KHTML, like Gecko)
Safari/537.1+ LG Browser/6.00.00(+mouse+3D+SCREEN+TUNER; LGE; ModelName; 03.07.01;
0x00000001;); LG NetCast.TV-2013/03.17.01 (LG, ModelName, wired)
```

The following table illustrates the fields in userAgent string and User-Agent HTTP header. The “capabilities” field identifies the capability of the device.

[Table] Fields in userAgent string and User-Agent HTTP header

Field	Description
LG Browser version	The version of LG web browser engine
capabilities	Zero or more concatenated optional feature strings (see also Table below)
modelName	String for the model name
softwareVersion	String for the version number of the firmware
hardwareVersion	String for the version number of the hardware
LG_NetCastVersion	String for the version of NetCast Platform

The following table illustrates the feature strings which can be used for “capabilities” field. See also section Device Info Plugin and API in **Developing > API** section in this Library for information on the supported features.

[Table] Optional feature string for capabilities field

Feature string	Optional feature
“+3D”	Support for 3D display feature
“+mouse”	Support for mouse feature (i.e. Magic Remote)
“+PORTAL_KEY”	Support for portal key (VK_PORTAL)
“+SCREEN”	Support for TV screen
“+TUNER”	Support for TV tuner (e.g. DVB-T/S/C, ATSC)
“+BDP”	Support for Blu-ray Player

Other User-Agent strings

The above userAgent string is used in the LG web browser engine implementation. However, there are several other software components which use the User-Agent header. The components and their functionalities are listed in the following tables.

These include LG NetCast TV-2013 and 2013 is updated every year.

[Table] Software components which use the User-Agent string, NetCast 1.0 & 2.0

Component	Functionality
HTTP-client	Delivery of DRM related information
MSDL-client	Delivery of streaming media file information
MSDL-http	Delivery of HTTP streaming media
MSDL-mms	Delivery of MMS streaming media

[Table] Software components which use the User-Agent string, NetCast 3.0 & 4.0

Component	Functionality
HTTP-client	Delivery of HTTP steaming media and file information
HLS-client	Delivery of HLS streaming media
MMS-client	Delivery of MMS streaming media
DRM-client	Delivery of DRM related information

The software components listed in the table above have a different User-Agent string. The following tables describe the User-Agent strings which are used for the different software components in NetCast Platform.

[Table] User-Agent strings used for each software components, NetCast 1.0 & 2.0

Component	Functionality
HTTP-client	Mozilla/5.0 (compatible; LG-HttpClient-v1.0.3 DLNADOC/1.50 UPnP/1.1; MSIE 8.0; Windows NT 5.1; LG_UA; AD_LOGON=LGE.NET; .NET CLR 2.0.50727; .NET CLR 3.0.04506.30; .NET CLR 3.0.04506.648; LG_UA; AD_LOGON=LGE.NET; LG NetCast.TV-2011)
MSDL-client	Mozilla/5.0 (DirectFB; U; Linux mips; en) AppleWebKit/528.5+ (KHTML, like Gecko, Safari/528.5+) LG Browser (; LG NetCast.TV-2011)
MSDL-http	MSDL (compatible; LG NetCast.TV-2011)
MSDL-mms	NSPlayer/11.0.5721.5145 (compatible; LG NetCast.TV-2011)

[Table] User-Agent strings used for each software components, NetCast 3.0 & 4.0

Component	Functionality

Component	Functionality
HTTP-client	GStreamer souphttpsrc (compatible; LG NetCast.TV-2012) libsoup/2.34.2
HLS-client	HLS Client/2.0 (compatible; LG NetCast.TV-2012)
MMS-client	NSPlayer/4.1.0.3856 (compatible; LG NetCast.TV-2012)
DRM-client	libsoup/2.34.2 (compatible; LG NetCast.TV-2012)

The Accept-Language Header

The LG web browser engine supports the 'Accept-Language' HTTP header[2]. If the web server has multiple versions of resources, in different languages, it can provide content in user's preferred language. The language code is base on the ISO 639-1 standard.

The Accept-Language header is set by the "Language" option in TV menu system. Therefore, the LG web browser engine will apply a language preference if the user changes the language option in the TV menu. The TV language management system of LG Electronics is based on ISO 639-2 using alpha-3 code space. Therefore, a conversion rule is required for linking ISO 639-1 alpha-2 code space. The NetCast Platform uses the official conversion rule between ISO 639-2 and ISO 639-2[3]. Please see Annex C for full details.

Because ISO 639-2 and ISO 639-1 have different code space size, and ISO 639-2 has a bigger code space than ISO 639-1, the full ISO 639-2 code cannot be completely converted to the ISO 639-1 code. Some codes cannot be converted into ISO 639-1. In this case, the NetCast Platform assigns the default language. Currently, the default language is English, ISO 639-1 code 'en'. In Annex C, the blank cells in ISO 639-1 column shall be default language.

The number of languages available is limited to those provided within the LGE TV environment. Therefore, not all languages in Annex C are supported.

Protocols

The NetCast Platform supports HTTP and HTTPS for its HTML file delivery and for its media (audio and video) delivery. The NetCast Platform prefers HTTP and HTTPS for HTML delivery; HTTP, HTTPS, HLS and Widevine for VoD media delivery; HLS and Widevine for linear broadcasting.

The transport protocols which are supported by the NetCast Platform can be summarized as shown below.

[Table] Transport protocols

Use	Preferred protocols	Supported protocols
HTML file transportation	HTTP, HTTPS	HTTP, HTTPS
VoD media file transportation	HTTP, HTTPS, HLS, Widevine	HTTP, HTTPS, HLS, Widevine
Linear or live broadcasting	HLS, Widevine	HLS, Widevine

CODEC & Container

The NetCast Platform supports WMV-9, MPEG-2 and H.264 codecs for its video decoding and WMA-9, MPEG-1 and HE-AAC for its audio decoding.

Annex B specifies all the supported codecs and containers by the NetCast Platform in more detail.

Note

To ensure the stablility of streaming h.264, LG recommends you to use maximum of four reference frames.

DRM

The NetCast Platform supports PlayReady, Widevine, and Verimatrix DRM for its digital rights management system.

[Table] Supported DRM formats in NetCast 3.0, 4.0, and 4.5

Format	Version
PlayReady	2.0
Widevine	4.5
Verimatrix	1.0.5

PlayReady is backwards compatible with Windows Media DRM 10 content, meaning that content encrypted with WM DRM 10 will be played on a PlayReady terminal.

PlayReady supports both pre-delivery and post-delivery methods.
PlayReady uses content formats such as *.wmv / *.wma (WMDRM), *.pyv / *.pya (PlayReady) (ASF Container)

Using WMDRM/PlayReady

Pre-Delivery method (WMDRM Only)

In pre-delivery method, drmClientInfo and setDrmLicenseInfo APIs are used. It is application server that delivers content information to license server. Methods like cookie can be used but the information is not stored in TV.

1. Application server requests client information to Smart TV.
2. Client information is received from Smart TV using drmClientInfo API and delivered to the application server.
3. Application server delivers the content information to license server with the client information received from Smart TV.
4. License server that received client and content information delivers the license of the content and client to application server.
5. Application server delivers the license that has been received from the license server to Smart TV.
6. Smart TV sets DRM license using setDrmLicenseInfo API.
7. Starts to play the content using content URL.

Post-Delivery method

In post-delivery method, no APIs are used.

1. Application server delivers content URL to Smart TV and the content is started to play.
2. Smart TV acquires content information and license server URL by extracting the header from content stream.
3. Based on the acquired content information, Smart TV delivers the license acquisition data to license server directly.
4. License server delivers the requested license to Smart TV directly.
5. Starts to play the content.

See also section “drm_type” in **Developing > API** section in this Library for detailed information about DRM solutions.

Using Verimatrix

For Verimatrix, you must call setVMconfigData method before playing the media. As the following example, you must set the service_type to 1 when calling setVMconfigData. See **setVMconfigData** method in **API > Device Info Plugin and API** section.

```
// Example of device object in HTML
<object type="application/x-netcast-info"
    id="device"
    width="0"
    height="0">
</object>

var device = document.getElementById("device");
var result = device.setVMconfigData(1, “CompanyName”, 10.1.1.1);    //service_type
must be 1
if (result = 1) {
   console.log(“Verimatrix Initialization Success!”);
} else {
   console.log(“Verimatrix Initialization Fail!”);
```

Also, you must set drmType property in media object to “verimatrix”. See **drmType** property in **API > Media Player**

Plugin and API section.

```
// Example of media object in HTML
<object type="application/x-netcast-av"
    width="1280"
    height="720"
   drmType=”verimatrix”
    id="media">
</object>
```

OSD Resolution

The NetCast Platform supports 1280x720 for its OSD resolution and 32 bpp (bit per pixel) for its OSD color depth.

Subtitle

The subtitle can be applied only when a full size video is being played.
The NetCast Platform supports SAMI (Synchronized Accessible Media Interchange), CineCanvas file format and Timed Text format (W3C Timed text markup language (TTML) 1.0) for its subtitle decoding.

Note

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

[Table] Supported subtitle file format of LG web browser engine

Format	NetCast Platform 1.0	NetCast Platform 2.0 or higher version
SAMI	O	O
CineCanvas	O	O
Timed Text	X	O

These are the Reference URLs for SAMI, CineCanvas and Timed Text.
- SAMI : **http://msdn.microsoft.com/en-us/library/ms971327.aspx**
- CineCanvas : **http://www.deluxedigital.co.uk/assets/pdf/DDL_DCI_Compliant_Subtitle_Files.pdf**
- Timed Text : **http://www.w3.org/TR/ttaf1-dfxp/**

Following is the detailed specification.

<SAMI>

[Table] Detailed specification for SAMI

Start	End [second]	Contents
<!--	-->	Comments symbol
<head>	</head>	Container for processing information and metadata for an HTML document
<body>	</body>	Container for the displayable content of an HTML document
<sync start>	</sync>	Set the start time for the caption
<p class>	</p>	Creates a paragraph
<br>		Forced line break
&nbsp;	&nbsp	Non-breaking space
&lt;	&lt	Inequality(less than)
&gt;	&gt	Inequality(greater than)
&amp;	&amp	Definition for ampersands in HTML

Start	End [second]	Contents
&quot;	&quot	Replace for double quotation marks

<CinecanvasType>

[Table] Detailed specification for Cinecanvas Type

Start	End [second]	Contents
<!--	-->	Comments symbol
<DCSubtitle>	</DCSubtitle>	Specifies the version of subtitle XML
<Subtitle>	</Subtitle>	Beginning of a particular subtitle
<TimeIn>	</TimeIn>	Time of the subtitle first appears on screen
<TimeOut>	</TimeOut>	Time of a subtitle disappears on screen
<Text>	</Text>	Actual texts are rendered on screen
<Font>	</Font>	Font to use for the subtitle
<br>		Forced line break
&nbsp;	&nbsp	Non-breaking space
&lt;	&lt	Inequality(less than)
&gt;	&gt	Inequality(greater than)
&amp;	&amp	Definition for ampersands in HTML
&quot;	&quot	Replace for double quotation marks

<Timed Text>

[Table] Detailed specification for Timed Text

Start	End [second]	Contents
<!--	-->	Comments symbol
<div>	</div>	Marking a group of consecutive elements
<p>	</p>	Represent a paragraph
<begin>	</begin>	Time of the subtitle first appears on screen
<end>	</end>	Time of a subtitle disappears on screen
<body>	</body>	Represents the main content of the document
<tt>	</tt>	Temporal beginning and ending of a document instance
<br>		Forced line break
&nbsp;	&nbsp	Non-breaking space
&lt;	&lt	Inequality(less than)
&gt;	&gt	Inequality(greater than)
&amp;	&amp	Definition for ampersands in HTML
&quot;	&quot	Replace for double quotation marks

See also section “subtitleOn” and “subtitle” in **Developing > API** section in this Library for the detailed information about subtitling.

Images

There are several things to consider for performance when using images on web applications. Using too large image or too many images on a single page will delay the page loading and rendering time. It is usually recommended to set the size of image within the resolution of the TV screen. Currently, applications on the NetCast Platform support a resolution of 1280x720, so it is recommended not to use the image bigger than 1280x720.

Note

Limit image sizes to less than 1 megapixel (about 1280x720 pixel).

The number of images that can be used on a page of web application is determined by the memory available on the NetCast Platform. If there is no free space available, the Platform generates the "outofmemory" event to have the application recognize the current state. In order to prevent the web application from generating the "outofmemory" event, the web browser engine should be able to manage the memory by switching pages.
And, if you know the size of the image to be loaded, rendering delay can be reduced by setting "width" and "height" in the <img> tag.

NetCast Platform supports image decoding. The following table lists the supported image formats.

[Table] Supported image formats

Image formats
JPEG
PNG
GIF

Cascading Style Sheets

CSS is used often when creating web applications because it can reduce repetitive tasks and offers a variety of functions.

The CSS should be optimized to improve the performance of Web application. It is not recommended to indicate an element with the relative position and size by having the consecutive parent-child structure. If CSS is in this structure, the LG web browser engine has to recalculate an absolute position and size of each element based on the parent's style whenever processing the layout of each page. Indicating the location and size of each element as an absolute location and size is more efficient because the LG web browser engine does not have to calculate them.

In addition, web application will run faster using a clear selector like "#id"or"a:link" than using relatively an obscure selector like ":link".

Streaming Method

HTTP Streaming

The NetCast Platform supports HTTP streaming, which means 'Progressive Download'. This method transfers a file over HTTP protocol and plays it when the decodable part of the file is received in buffer before the entire file is downloaded.

Http Live Streaming (HLS) - Adaptive Streaming

The NetCast Platform 2.0 or higher version supports HLS (Http Live Streaming). HLS is the adaptive streaming method that - Apple has documented as an Internet-Draft (Reference URL of HLS: **http://tools.ietf.org/html/draft-pantos-http-live-streaming-04**).

Application authors can use HLS to send audio and video over HTTP from an ordinary web server for playback. HLS supports both live-broadcasts and prerecorded content (video on demand). HLS supports multiple alternate streams at different bit rates, and the NetCast Platform can switch streams intelligently as network bandwidth changes. HLS also provides media encryption and user authentication over HTTPS, allowing publishers to protect contents.

It is recommended that application authors use SD or HD video resolutions on the NetCast Platform. Content below SD will not have adequate video quality.

Application authors are recommended not to change the resolution of a presentation, as flicker may occur.

Note

For HLS to function correctly, meet one of the following:

- The URL includes the media file extension (e.g. .m3u8, .mpd, or .mp4).
- The MIME type is explicitly defined in media object.

To find out the supported MIME types in NetCast platform, see **Annex A**.

Note

The NetCast Platform 2.0 or higher version does not support the following tags: EXT-X-PROGRAM-DATE-TIME, EXT-X-ALLOW-CACHE, EXT-X-DISCONTINUITY

LG HLS client was developed as HLS draft doc ver.04 (**http://tools.ietf.org/html/draft-pantos-http-live-streaming-04**). So some added functions after ver.04 does not work or behavior can differ from other HLS client which is developed as recent HLS doc.

If resolution of source is very low(under 320x240), video may be broken or does not play normally.
Media data for HLS must include both audio and video data. LGTV does not support HLS stream which has audio only stream. Therefore, LG strongly recommends not to use audio only stream. If HLS stream has audio only stream, “CODECS” attribute must be added in #EXT-X-STREAM-INF tag so LGTV can ignore the audio only stream. Otherwise playback error occurs while stream is playing.

For example,
#EXT-X-STREAM-INF:PROGRAM-ID=1, BANDWIDTH=59573, CODECS=”mp4a.40.2”
audio_only.m3u8

Note

To ensure the stablility of streaming h.264, LG recommends you to use maximum of four reference frames.

Play List (ASX)

The NetCast Platform supports ASX[4][5] as a play list file format. However, not all features are supported. Refer to the MSDN document for more information about ASX elements.[5]
See also section “next” and “previous” in **Developing > API > Media Player Plugin and API** section in this Library for the detailed information about Play List.

The NetCast Platform supports the following elements.

[Table] Supported elements in ASX format

Element name	Description
ASX	Defines a file as a Windows Media metafile.
COPYRIGHT	Contains the copyright information for an ASX or ENTRY element.
ENTRY	Specifies the path to a media clip.
PARAM	Specifies the value of a custom parameter associated with a playlist or an element of a playlist.
REF	Specifies a URL for a piece of digital media content.
TITLE	Contains the title of an ASX or ENTRY element.
REPEAT	Defines the number of times the media player repeats one or more ENTRY or ENTRYREF elements.
BASE	Defines a URL string appended to the front of URLs sent to the media player.

Local File Play

In packaged web application, local media files such as music and video files cannot be played.

Tips for Web Application Development

This chapter explains the useful tips for Web application development as follows. When you develop Web applications, these tips can be useful to set your environment or conditions.

- Web Browser Engine
- Font and Font Style
- Platform Identification and Its Applications
- Input Device
- Smart Text for NetCast 2.0
- Smart Text for NetCast 3.0 and 4.0
- Virtual Keyboard
- Performance Consideration
- JavaScript

Web Browser Engine

Document Scrolling

Although web page scrolling is not recommended by the NetCast Platform, due to the poor user experience, it is supported by pressing the “PR+” (Programme Up) and “PR-” (Programme Down) buttons on the IR Remote. In the Win32 emulator, users can scroll web pages by pressing the “Page Up” and “Page Down” keys on the keyboard. Scrolling a web page in general decreases the user experience. In order to prevent users from inadvertently scrolling a web page, it is recommended that application authors adopt the code example as below.

```
<body style='margin:0'>
<div style='width:1280px; height:720px; overflow:hidden;'>

<!-- place graphical objects here -->

</div>
</body>
```

Common Control Supporting

LG web application does not recommend common controls. LG Electronics recommends application authors not to use this kind of controls.

Flushing Cookie and Cache Repositories

Application authors may want to flush cookie and cache storage of the LG web browser engine. The NetCast Platform allows application authors to flush the cookie and cache repositories. Enter the TV main menu and select “Factory Reset” submenu in “Option” menu, then the repositories will be completely flushed.

Cookie and cache can be also removed by executing “Clear data” in LG internet browser setting.

Font and Font Style

The NetCast 3.0 and 4.0 support two different weights of LG Display fonts and Tiresias v8.04 font. The LG Display fonts consist of ‘LG Display-Regular’ and ‘LG Display-Light’ including a Hong Kong variant. The ‘LG Display-Regular’ is the most preferred font for NetCast 3.0 and 4.0 platforms. For all of these, only a ‘plain’ style of the font is supported (i.e. not bold or italics, etc).

Note

It is highly recommended that application authors who start new service from NetCast 3.0 and 4.0 use ‘LG Display-Regular’ and ‘LG Display-Light’. However, those who have serviced from NetCast 2.0 or below might be able to use LG Display font for continuing services in NetCast 3.0 and 4.0.

The following example shows the using of a set of LG Display font and Tiresias font. Application authors can use this for all tags which permit the ‘style’ attribute.

```
style='font-family:LG Display-Regular'
```

```
style='font-family:LG Display-Light'
```

```
style='font-family:LG Display'
```

```
style='font-family:TiresiasScreenfont'
```

The following table shows service countries which can use each font.

[Table] Service countries which can use each font

Font	Applied countries
LG Display-Regular LG Display-Light LG Display	All countries except Hong Kong, Taiwan
LG Display_HK-Regular LG Display_HK-Light LG Display_HK	Hong Kong, Taiwan
TiresiasScreenfont	Only Europe

If application authors want to apply global service, it is mandatory to use LG Display fonts ('LG Display-Regular' and 'LG Display-Light)'.

The NetCast Platform recommends four font sizes: 'title large font', 'large font', 'medium font' and 'small font'. The following table lists the recommended size of fonts. For better display of text, application authors should follow the recommendation.

[Table] Font size recommendation

Font size category	Font size (in pixel)
Title Large Font	45 px
Large Font	38 px
Medium Font	30 px
Small Font	26 px

Note

It is strongly recommended that LG Smart TV application authors specify the font family exactly. Different fonts could be applied for rendering a single paragraph, if LG Smart TV application author does not specify the exact font family. The rendering of text can be dirty when different fonts are applied in a single paragraph.

The LG Display fonts and Tiresias font are provided by LG Electronics to developers who have agreed to the following terms.

Note

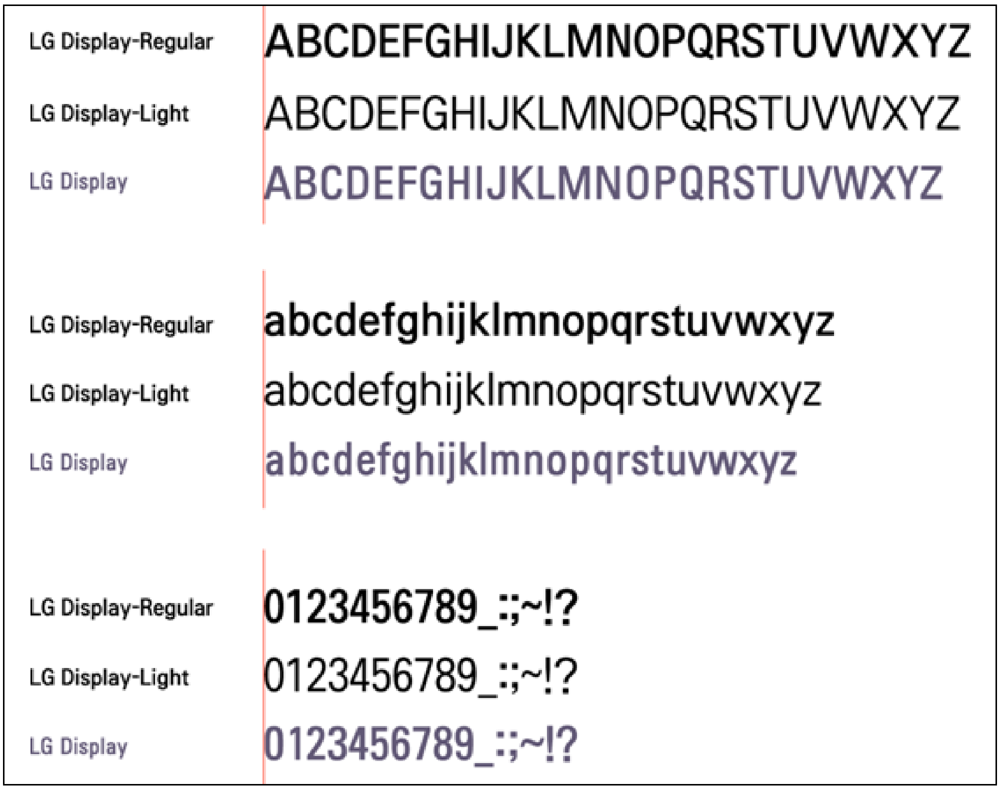
The licensee of these fonts is LG Electronics and these fonts can be only used for the development of interactive services for the Smart TV of LG Electronics. They shall not be used for other purpose without prior written authorization of LG Electronics.

Font style Supporting

NetCast 3.0 and 4.0 support two different weight of fonts for suitable display. 'LG Display-Light' is a light weight version of 'LG Display-Regular' font. The difference is shown in the following figure. Moreoever, the LG font-names of NetCast 1.0 and 2.0 are different . So, application authors who have been using the LG font from NetCast 1.0 must apply multi-font for NetCast 2.0 or 3.0/4.0. Otherwise, the rendering of text can be dirty when different fonts are applied in a sigle paragraph. Some examples of using multi-font will be given for better understanding.

Note

Application authors, who start service from NetCast 2.0 or do not want to use the LG font, do not need to use multi-font. However, application authors who service in both Netcast 2.0 and 3.0/4.0 should either use multi-font or ‘LG Display’ font only for backward compatibility.



[Figure] Comparison of thickness and width among LG Display fonts

[Table] LG font-name in NetCast 1.0

Font	Applied countries
LG Display_Eng	PAN-EU, Austrailia, Brazil
LG Display_Eng_Kor	Korea
FZHei-B01	China
LG_Big5HK_V	Hong Kong, Taiwan

[Table] LG font-name in NetCast 2.0

Font	Applied countries
LG Display	All countries except Hong Kong, Taiwan
LG Display_HK	Hong Kong, Taiwan

[Table] LG font-name in NetCast 3.0 and NetCast 4.0

Font	Applied countries
LG Display-Regular LG Display-Light LG Display	All countries except Hong Kong, Taiwan
LG Display_HK-Regular LG Display_HK-Light LG Display_HK	Hong Kong, Taiwan

The following shows the example of using multi-font.

```
style='font-family: LG Display'
```

```
style='font-family: ‘LG Display', ‘LG Display-Regular’
```

The following shows the example of using multi-font.

```
style='font-family:LG Display_Eng'
```

```
style='font-family: ‘LG Display_Eng', ‘LG Display’
```

The following shows the example of using multi-font.

```
style='font-family: LG Display_Eng_Kor'
```

```
style='font-family: ‘LG Display_Eng_Kor', ‘LG Display’
```

The following shows the example of using multi-font.

```
style='font-family: FZHei-B01'
```

```
style='font-family: ‘FZHei-B01’, ‘LG Display’
```

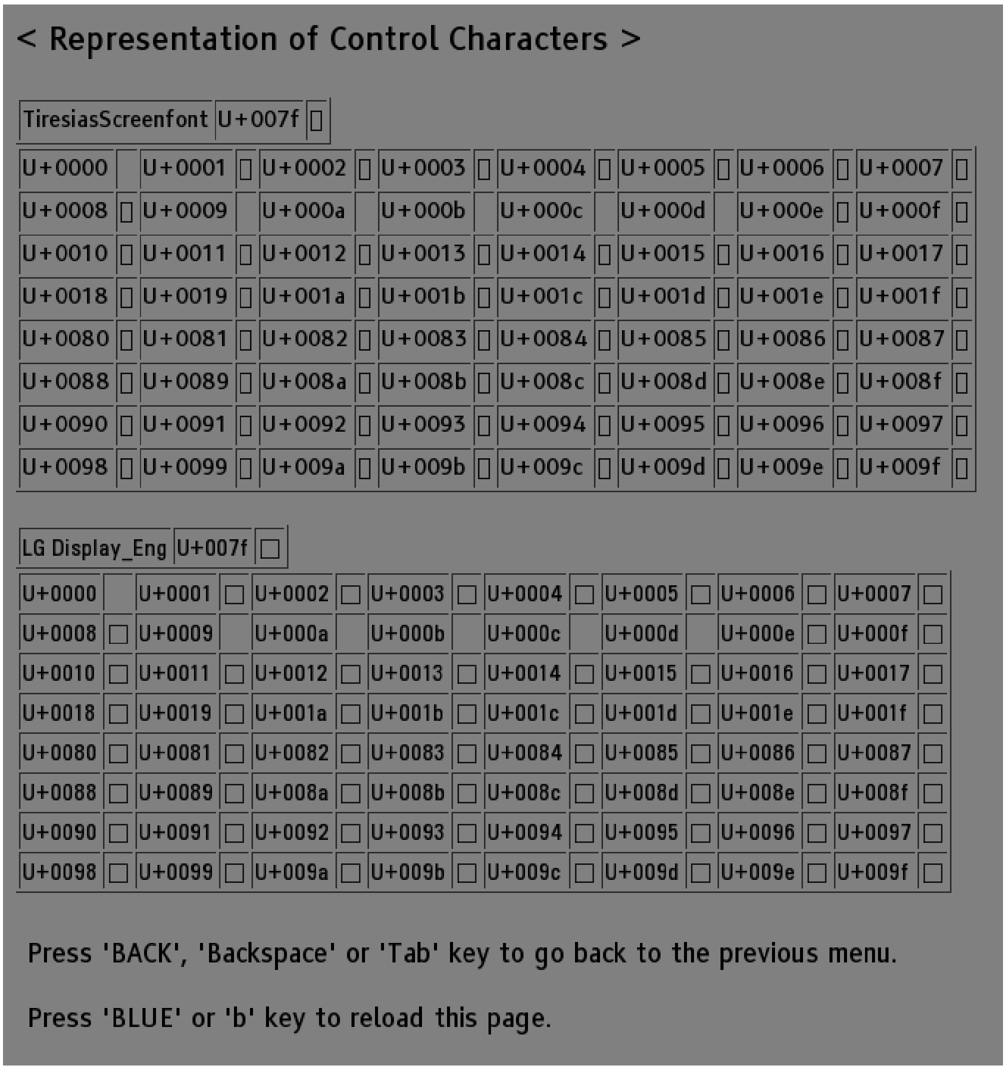
The following shows the example of using multi-font.

```
style='font-family: LG_Big5HK_V'
```

```
style='font-family: ‘LG_Big5HK_V’, ‘LG Display_HK’
```

Representation of Control Characters[6]

The NetCast Platform does not support the correct rendering of control characters. In Basic Multilingual Plane of ISO/IEC 10646 (UCS or UNICODE), code positions U+0000 to U+001F, U+007F and U+0080 to U+009F are reserved for control characters. Those characters cannot have proper glyph, and the NetCast Platform represents these characters as a square box except U+0000, U+0009, U+000A, U+000B, U+000C and U+000D. Figure below shows the representation. Therefore, it is better for application authors to avoid rendering control characters.



< Representation of Control Characters >

TiresiasScreenfont	U+007f	▯

U+0000		U+0001	▯	U+0002	▯	U+0003	▯	U+0004	▯	U+0005	▯	U+0006	▯	U+0007	▯
U+0008	▯	U+0009		U+000a		U+000b		U+000c		U+000d		U+000e	▯	U+000f	▯
U+0010	▯	U+0011	▯	U+0012	▯	U+0013	▯	U+0014	▯	U+0015	▯	U+0016	▯	U+0017	▯
U+0018	▯	U+0019	▯	U+001a	▯	U+001b	▯	U+001c	▯	U+001d	▯	U+001e	▯	U+001f	▯
U+0080	▯	U+0081	▯	U+0082	▯	U+0083	▯	U+0084	▯	U+0085	▯	U+0086	▯	U+0087	▯
U+0088	▯	U+0089	▯	U+008a	▯	U+008b	▯	U+008c	▯	U+008d	▯	U+008e	▯	U+008f	▯
U+0090	▯	U+0091	▯	U+0092	▯	U+0093	▯	U+0094	▯	U+0095	▯	U+0096	▯	U+0097	▯
U+0098	▯	U+0099	▯	U+009a	▯	U+009b	▯	U+009c	▯	U+009d	▯	U+009e	▯	U+009f	▯

LG Display_Eng	U+007f	□

U+0000		U+0001	□	U+0002	□	U+0003	□	U+0004	□	U+0005	□	U+0006	□	U+0007	□
U+0008	□	U+0009		U+000a		U+000b		U+000c		U+000d		U+000e	□	U+000f	□
U+0010	□	U+0011	□	U+0012	□	U+0013	□	U+0014	□	U+0015	□	U+0016	□	U+0017	□
U+0018	□	U+0019	□	U+001a	□	U+001b	□	U+001c	□	U+001d	□	U+001e	□	U+001f	□
U+0080	□	U+0081	□	U+0082	□	U+0083	□	U+0084	□	U+0085	□	U+0086	□	U+0087	□
U+0088	□	U+0089	□	U+008a	□	U+008b	□	U+008c	□	U+008d	□	U+008e	□	U+008f	□
U+0090	□	U+0091	□	U+0092	□	U+0093	□	U+0094	□	U+0095	□	U+0096	□	U+0097	□
U+0098	□	U+0099	□	U+009a	□	U+009b	□	U+009c	□	U+009d	□	U+009e	□	U+009f	□

Press 'BACK', 'Backspace' or 'Tab' key to go back to the previous menu.

Press 'BLUE' or 'b' key to reload this page.

[Figure] Representation of control characters in BMP of ISO/IEC 10646 UCS

Platform Identification and Its Applications

There are several ways to uniquely identify the platform. These are:

- UserAgent string,
- User-Agent http header,
- Device information APIs (including Secure Device Identification)
- HTTPS client certificate. (Client Root CA related description is described in Annex E.)

Identifying and Differentiating with the userAgent String

The following example shows how application authors can identify the LG web browser engine version using the userAgent string. The application can identify the platform environment. See userAgent String for more information about the userAgent String.

```
var userAgent = new String(navigator.userAgent);
if (userAgent.search(/LG Browser/) > -1)
{
// platform dependant code
// (in this example following code is dependent on LG web browser engine
. . .
}
```

The following example shows how application authors can differentiate LGE devices using the userAgent string. The application can identify the platform environment. See userAgent String for more information about the userAgent

String.

```
var userAgent = new String(navigator.userAgent);
if (userAgent. search(/LG NetCast.TV/) > -1)
{
// platform dependant code
// (in this example following code is dependent on LGTV device
} else if (userAgent.search(/LG NetCast.Media/) > -1)
{
// platform dependant code
// (in this example following code is dependent on LGMedia device
}
```

Identifying and Differentiating with Device Information APIs

The following example shows how application authors can differentiate an application using the device information API. The application can identify the platform environment. See section “Device Info Plugin and API” in **Developing > API** section in this Library for more information about the device information APIs.

```
<object data=““ id=“deviceInfo“ width=0 height=0
type=“application/x-netcast-info“>
</object>
<script>
var DeviceInfo = new String(deviceInfo.manufacturer);
if (DeviceInfo.search(/LGE/) > -1)
{
// platform dependant code
// (in this example following code is dependent on LGE device
. . .
}
</script>
```

Input Device

Traditional IR Remote

All LG Smart TV models support the traditional IR Remote.

Magic Remote

Models that support the Magic Remote (a wand-like point and click IR Remote device) support all the features of the traditional IR Remote.

The Magic Remote is similar to a mouse in a computer system. The easiest way to understand the Magic Remote is to consider it as a mouse which has only left mouse button for click or double click and wheel scroll input. Figure below illustrates the Magic Remote.

Pressing the “OK” button generates “click” and “double click” events.

The other buttons are “power”, “Home”, “4-arrow key”, “volume up”, “volume down”, “channel up”, “channel down” and “mute”. In NetCast 3.0, “back”, “3D”, and “My Apps” buttons and wheel are added. In NetCast 4.0, “Voice Recognition” button is added.

“power”, “Home”, “Voice Recognition”, “volume up”, “volume down”, “3D” and “My Apps” buttons are not available for LG Smart TV application; they are used by the native TV application.

The “channel up” button generates “page up” (VK_PAGE_UP) and the “channel down” button generates “page down” (VK_PAGE_DOWN) in LG Smart TV applications.

The “4-arrow key” buttons generate navigation functions “Up”(VK_UP), “Down”(VK_DOWN), “Left”(VK_LEFT), “Right”(VK_RIGHT) in LG Smart TV applications.

The “back” button generates “back” (VK_BACK) in LG Smart TV applications.

Figure below illustrates the pointer which is rendered in the TV display screen when the Magic Remote is activated. Users can select one of several pointer shapes via the TV menu (Option menu).

Users activate the Magic Remote by pressing the “OK” button on the Magic Remote of NetCast 2.0 or by shaking the Magic Remote of NetCast 3.0 or higher and deactivate it by pressing any button on the traditional IR Remote. Whenever a device change occurs, a “mouseon” or “mouseoff” event is generated. From NetCast 3.0, the Magic Remote gets deactivated when the halt of the mouse movement continues for 3 seconds.

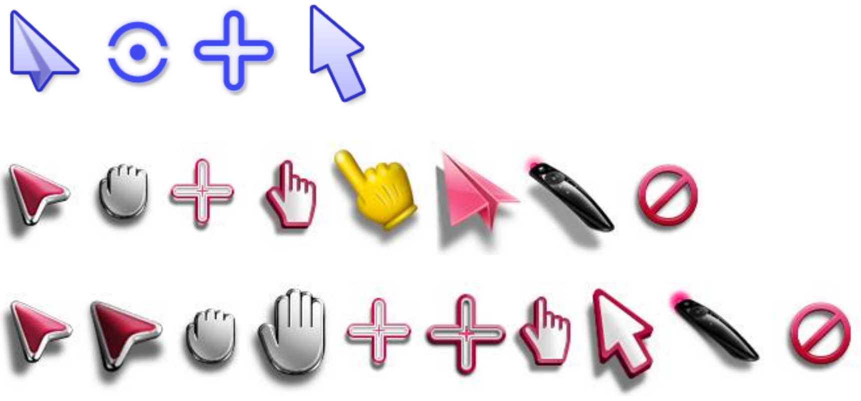
When users press any of the “4-arrow key” buttons on the Magic_Remote, the pointer will disapperar. (However it is still activated but without being displayed (sleep mode) in NetCast 2.0).

The “OK” button on the Magic Remote generates the “OK” keycode.

The pointers will re-appear when user shakes the Magic Remote.

Refer to **Developing > API** section in this Library to learn how to use the following events and methods.
- “mouseon” and “mouseoff”
- “window.NetCastGetMouseOnOff() : How to get the mouse on or off status.
- “window.NetCastMouseOff(time)”: How to deactivate the Magic Remote pointer.
- “supportMouse”: How to check whether the TV supports Magic Remote or not.

[Figure] Magic Remote (Left: NetCast 2.0, Middle: NetCast 3.0, Right: NetCast 4.0)



[Figure] The Pointers of Magic Remote (Top: NetCast 2.0, Middle: NetCast 3.0, Bottom: NetCast 4.0)

Traditional IR Remote Key Events

Users can control an application using a traditional IR Remote. The following table lists the buttons and key events, which are available for application authors to build their applications.

[Table] Supported image formats

Group	Key	Symbol	Mnemonic	Keyboard	Key code
Numeric	0 - 9	0 - 9	VK_0 - VK_9	0 - 9	48-57
Arrow	UP	▲	VK_UP	↑	38
	DOWN	▼	VK_DOWN	↓	40
	LEFT	◀	VK_LEFT	←	37
	RIGHT	▶	VK_RIGHT	→	39
Page Control	PAGE UP	↑ PAGE	VK_PAGE_UP	PgUp	33
	PAGE DOWN	PAGE ↓	VK_PAGE_DOWN	PgDn	34
Selection	OK/SELECT	⦿ OK	VK_ENTER	Enter	13
	BACK/RETURN	BACK	VK_BACK	none	461
Colour	RED	●	VK_RED	none	403
	GREEN	●	VK_GREEN	none	404
	YELLOW	●	VK_YELLOW	none	405
	BLUE	●	VK_BLUE	none	406
Playback	PLAY	▶	VK_PLAY	none	415
	STOP	■	VK_STOP	none	413
	PAUSE	‖	VK_PAUSE	none	19
	REWIND	◀◀	VK_REWIND	none	412
	FAST FORWARD	▶▶	VK_FAST_FWD	none	417
Service	INFO	INFO	VK_INFO	none	457

Basically, the key codes of the NetCast Platform follow those of CE-HTML, CEA-2014[7]. However the NetCast Platform does not support all key codes in CE-HTML. The full list of supported key codes are provided in the table above.

The ‘EXIT’ key is reserved for exiting the Web service function and is used by the native NetCast Platform. For this reason, the ‘EXIT’ key is not allowed to be used by application authors

LG Electronics provide a virtual screen IR Remote for LG Smart TV Emulator. Application authors can use this emulator to develop their applications. Note that the PC emulator supports general keyboard key codes rather than IR Remote key codes.

Key Repeat Function

Some key events are expected to be generated repeatedly while a button is held down. The native TV software supports the key repeat function for the 'channel up/down', ‘page up/down’, 'volume up/down' and 'arrow', up/down/left/right, keys. However, only page up/down keys, arrow keys and rewind/fast forward keys are allowed in a web-based application. The following table shows the repeatable keys provided by the NetCast Platform.

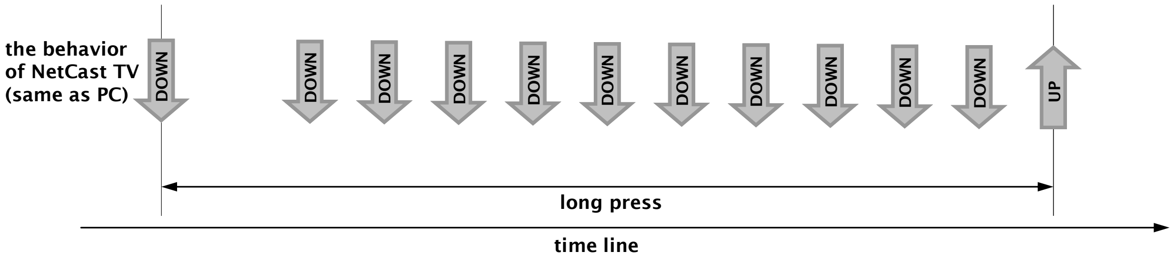
[Table] Repeatable keys

Group	Key	Symbol	Mnemonic	Keyboard	Key code
Arrow	UP	▲	VK_UP	↑	38
	DOWN	▼	VK_DOWN	↓	40
	LEFT	◀	VK_LEFT	←	37
	RIGHT	▶	VK_RIGHT	→	39
Playback	REWIND	◀◀	VK_REWIND	none	412
	FAST FORWARD	▶▶	VK_FAST_FWD	none	417
Page Control	PAGE UP	↑ PAGE	VK_PAGE_UP	PgUp	33
	PAGE DOWN	PAGE ↓	VK_PAGE_DOWN	PgDn	34

LG TV sets generate key events every 108 milliseconds. The theoretical maximum frequency is about 9.26 times per second. However, there are software overheads in processing the key events. The overall result of frequency is lower than the theoretical value of 9.26 times per second. The NetCast Platform does not guarantee the frequency of key repeat and it also depends on application's processing load. An LG Smart TV application author can measure the frequency using a simple JavaScript application and use the measured value for further development of the application.

The key repeat function behavior of NetCast Platform

The NetCast Platform provides the same behavior of the key repeat function with the Webkit based PC web browser to aid LG Smart TV application authoring. The following figure illustrates the behavior of the key repeat function in NetCast Platform. A long key press produces multiple continous key down events (onkeydown) and one key release event (onkeyup) at the end of pressing.



[Figure] The behavior of key repeat function

Key Event Handling

LG Smart TV application author can handle the key events by modifying the following example. See Traditional IR Remote Key Events for more information about available key codes.

```
function processKeyDown(e) {
    var keycode;
    if(window.event) { // IE
        keycode = e.keyCode;
    } else if(e.which) { // Netscape/Firefox/Opera
        keycode = e.which;
    } switch(keycode)
    {
        case VK_UP: doup(); break;
        . . .
    }
}
<body onkeydown="processKeyDown(event); return false">
. . .
</body>
```

Mouse Events

The following table lists and illustrates the available mouse events which are generated by the Magic Remote.

[Table] Available mouse events

Event	Generating condition
onclick	The Magic Remote clicks on an object
ondblclick	The Magic Remote double-clicks on an object
onmousedown	“OK” button on the Magic Remote is pressed
onmouseup	“OK” button on the Magic Remote is released
onmousemove	The Magic Remote is moved
onmouseout	The Magic Remote is moved off an element
onmouseover	The Magic Remote is moved over an element
onmousewheel	The wheel of the Magic Remote is rotated (From NetCast 3.0)

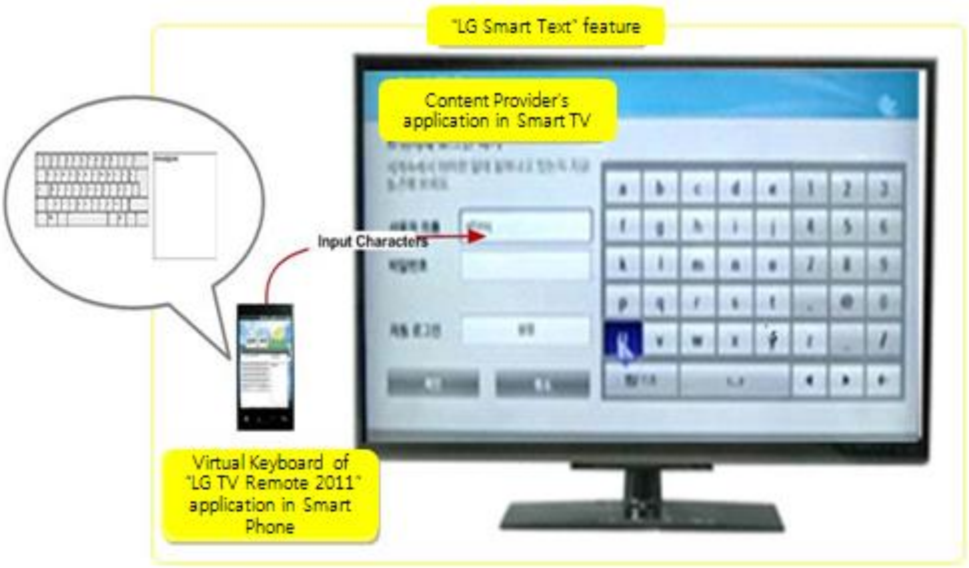
Smart Text for NetCast 2.0

“LG Smart Text” function establishes the connection between a smartphone (Android phone and iPhone) and the TV using the text input feature of a smartphone application, “LG TV Remote 2011”.

LG web appliction of NetCast 2.0 supports the “LG TV Remote 2011” program (smartphone application) to control TV. With the virtual keyboard of the “LG TV Remote 2011” program, the “LG Smart Text” function allows users to enter characters into a text field of content provider’s application running on TV.

Using LG Smart Text

The following shows the “LG Smart Text” function running with the “LG TV Remote 2011” application.


"LG Smart Text" feature
Content Provider's application in Smart TV
Input Characters
Virtual Keyboard of "LG TV Remote 2011" application in Smart Phone

[Figure] Using LG Smart Text (NetCast 2.0)

At the content provider’s application in Smart TV that needs input texts, users can input characters with their smartphone’s virtual keyboard.

System Requirements

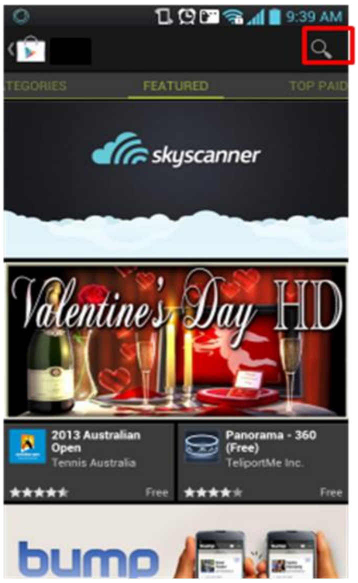
[Table] System Requirements for Smart Text (NetCast 2.0)

Type	Phone	TV
Hardware	- iPhone - Android phone (any version)	LG Smart TV 2011 - NetCast 2.0
Software	LG TV Remote 2011 program (A free app that can be downloaded to a smartphone. Visit the Apple's App Store or Android Market on your smartphone and search this app name to download.)	No additional software program is required.

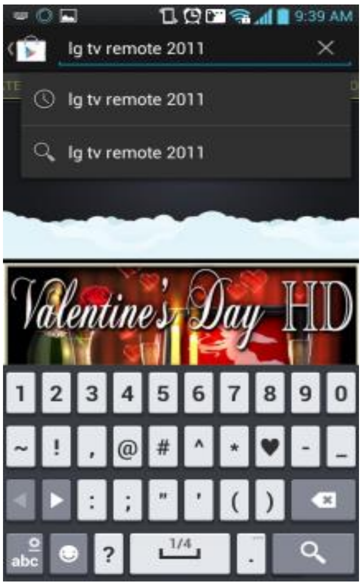
Installing LG TV Remote 2011 Program into Android Phone

This LG TV Remote 2011 program will work on both Android and iOS. The following process describes how to install the "LG TV Remote 2011" on your Android smartphone. It behaves the same for iPhone.

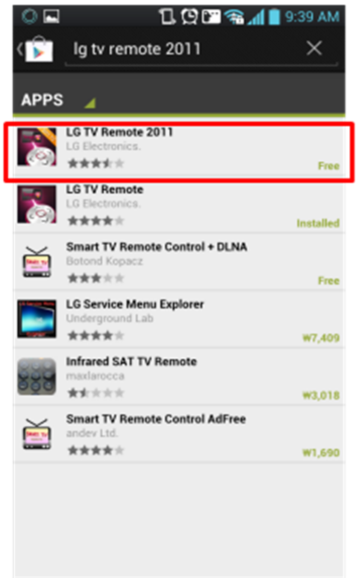
Step1. Visit the Android Market of your smartphone and select 🔍. Then, search and find the app you want.



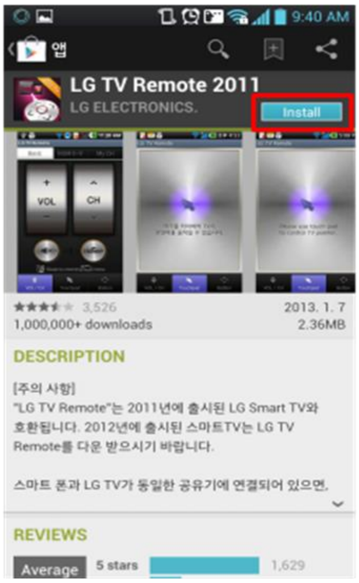
Step2. A virtual keyboard appears. To search "LG TV Remote 2011", type the app name.


lg tv remote 2011

Step3. After search, you can see the search result as shown in the figure.



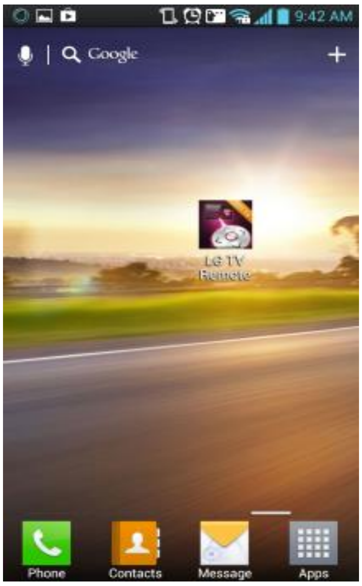
Step4. Select “LG TV Remote 2011” and click [Install]. Then, the app will be downloaded to your phone.



[주의 사항]
"LG TV Remote"는 2011년에 출시된 LG Smart TV와 호환됩니다. 2012년에 출시된 스마트TV는 LG TV Remote를 다운 받으시기 바랍니다.

스마트 폰과 LG TV가 동일한 공유기에 연결되어 있으면,

Step5. After download is complete, the following icon appears.



Pairing LG TV Remote 2011 Program with TV

The following describes how to pair “LG TV Remote 2011” program on your smartphone with LG TV.

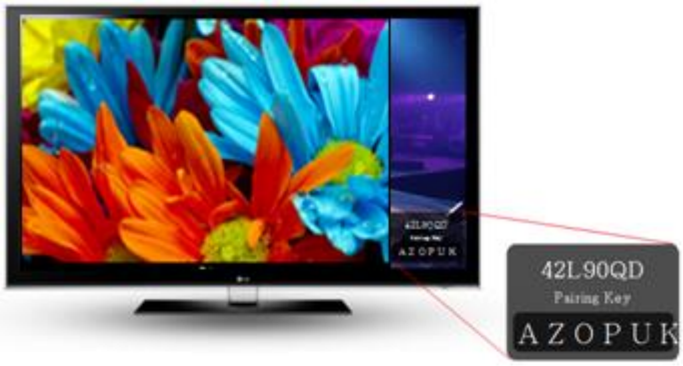
Pairing means making a wireless connection between the "LG TV Remote 2011" program and LG Smart TV. Once you pair them, the TV searches and finds "LG TV Remote 2011" every time it is turned on.

Note

The following process is same for both iPhone and Android. The User Interface may be slightly different depending on the operating system.

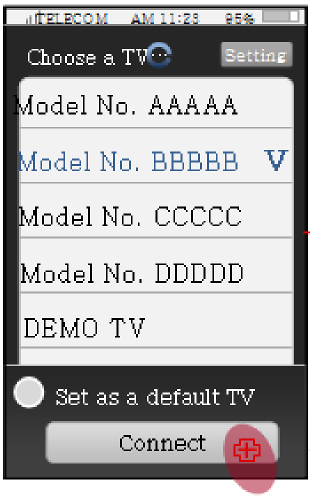
Step1. Run “LG TV Remote 2011” on your smartphone and turn on your TV. At the bottom right of the TV screen, you can see the pairing key.

A pairing key consists of six random alphabet characters (A~Z.) When pairing is successfully performed or canceled, it disappears. After 1 minute, it disappears, too.



Step2. Your smartphone shows a list of connected TV sets. In the list, choose one and click “Connect” button.

The connected TV list means the list of TV sets that your phone is ready to connected.
If you check the “Set as a default TV” radio button, the selected one becomes your default TV to connect. Therefore, you do not need to scroll and select the TV whenever you use “LG TV Remote2011”. The default TV is always marked with V or O.


TELECOM AM 11:23 95%

Choose a TV Setting

Model No. AAAAA

Model No. BBBBB V

Model No. CCCCC

Model No. DDDDD

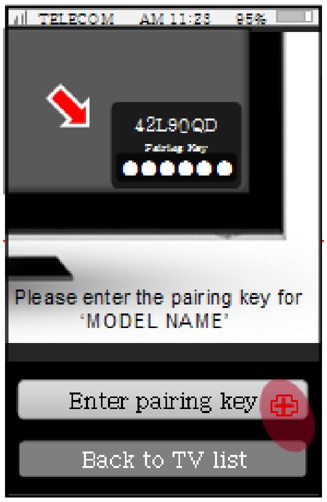
DEMO TV

○ Set as a default TV

Connect

Step3. After selecting the “Connect” button, this page appears on your smartphone. Select “Enter pairing key.” If you want to go back to the previous TV list, select “Back to TV list.”

In the figure below, “MODEL NAME” shows the actual TV model name that you are using (For example, 42L90QD)



Please enter the pairing key for 'MODEL NAME'

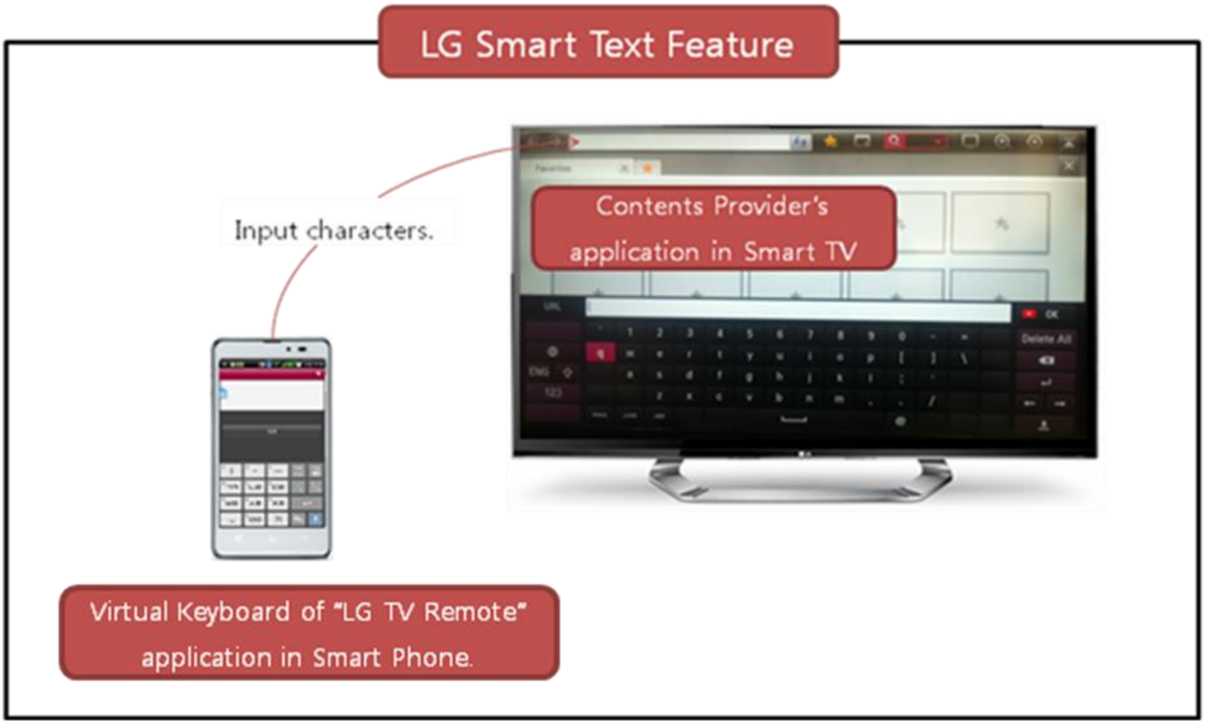
Smart Text for NetCast 3.0 and 4.0

“LG Smart Text” function establishes the connection between a smartphone (Android phone and iPhone) and the TV using the text input feature of a smartphone application, “LG TV Remote”.

LG web application of NetCast 3.0/4.0 supports the “ LG TV Remote” program (smartphone application) to control TV. With the virtual keyboard of the “LG TV Remote” program, the “LG Smart Text” function allows users to enter characters into a text field of content provider’s application running on TV.

Using LG Smart Text

The following shows the “LG Smart Text” function running with the “LG TV Remote” application.


LG Smart Text Feature
Input characters.
Contents Provider’s application in Smart TV
Virtual Keyboard of “LG TV Remote” application in Smart Phone

[Figure] Using LG Smart Text (NetCast 3.0 and 4.0)

At the content provider’s application in Smart TV that needs input texts, users can input characters with their smartphone’s virtual keyboard.

System Requirements

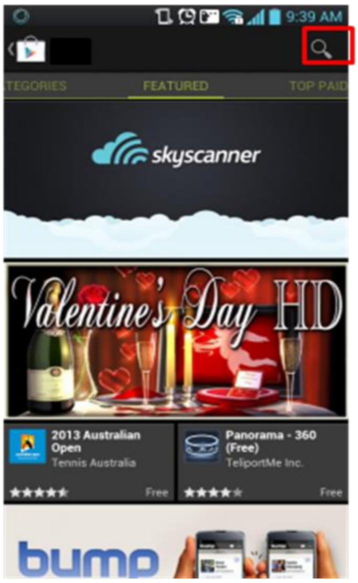
[Table] System Requirements for Smart Text (NetCast 3.0 and 4.0)

Type	Phone	TV
Hardware	-iPhone/iPad (4.1 and above) -Android phone/pad (any version)	LG Smart TV 2012 / LG Smart TV 2013 - NetCast 3.0 / NetCast 4.0
Software	- LG TV Remote (A free app that can be downloaded to a smartphone. Visit the Apple's App Store or Android Market on your smartphone and search this app name to download.)	No additional software program is required.

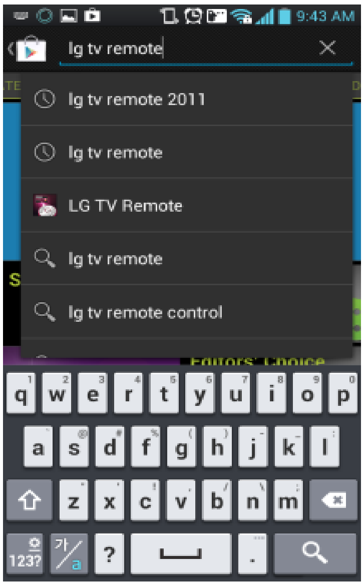
Installing LG TV Remote Program into Android smartphone

This LG TV Remote program will work on both Android and iOS. The following process describes how to install the "LG TV Remote" on your Android smartphone. It behaves the same for iPhone.

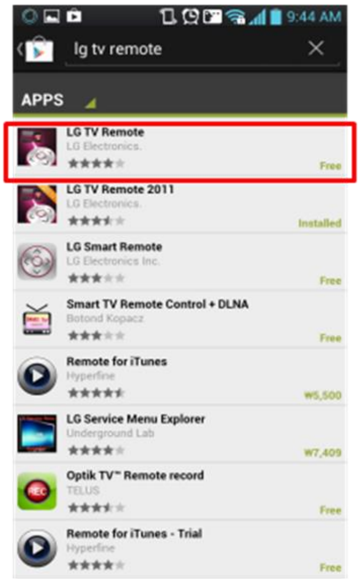
Step1. Visit the Android Market of your smartphone and select 🔍.
Then, search and find the app you want.



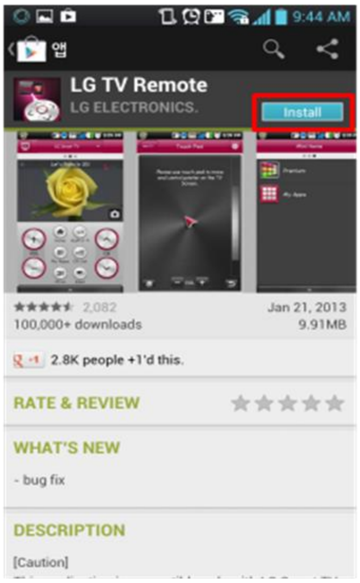
Step2. A virtual keyboard appears.
To search "LG TV Remote", type the app name.



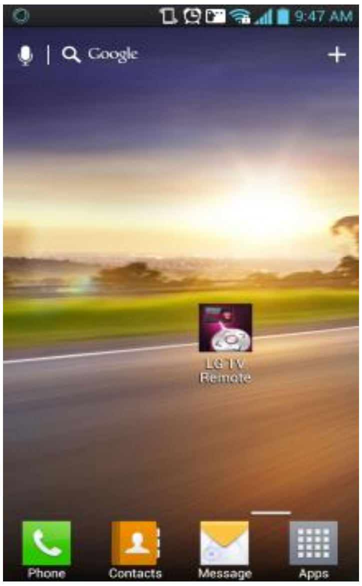
Step3. After search, you can see the search result as shown in the figure.



Step4. Select “LG TV Remote” and click [Install].
Then, the app will be downloaded to your phone.



Step5. After download is complete, the following icon appears.



Pairing LG TV Remote Program with TV

The following describes how to pair “LG TV Remote” program on your smartphone with LG TV.

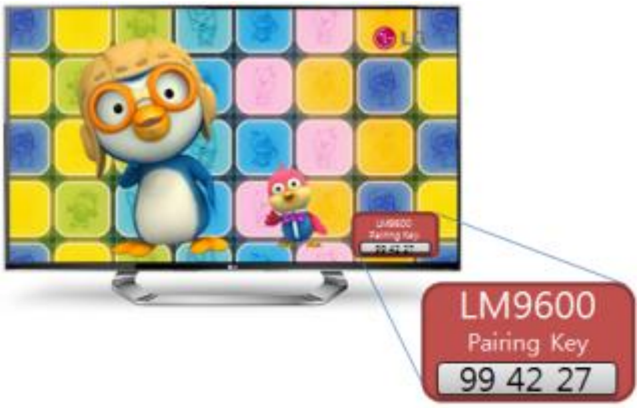
Pairing means making a wireless connection between the "LG TV Remote" program and LG Smart TV.

Note

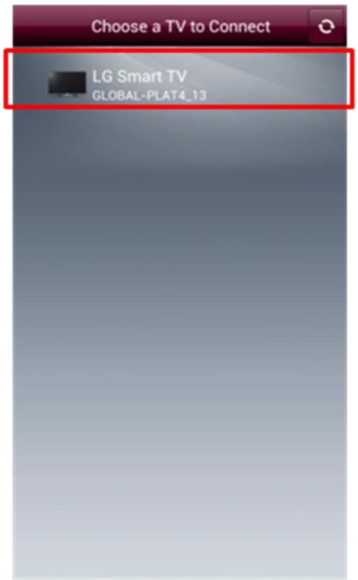
The following process is same for both iPhone and Android. The User Interface may be slightly different depending on the operating system.

Step1. Run “LG TV Remote” on your smartphone and turn on your TV. At the bottom right of the TV screen, you can see the pairing key.

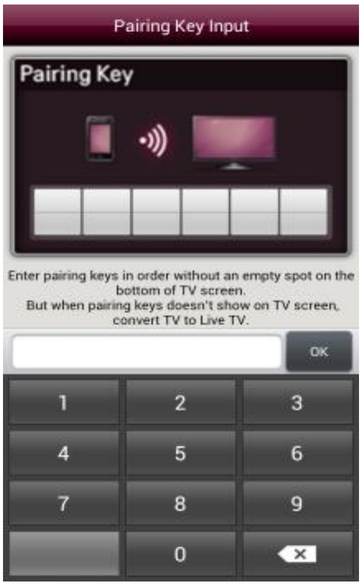
A pairing key consists of six random numeric characters (0~9). When pairing is successfully performed or canceled, it disappears. After 1 minute, it disappears, too.



Step2. Your smartphone shows a list of connected TV sets. In the list, choose one and click “Connect” button. The connected TV list means the list of TV sets that your phone is ready to connect.



Step3. After selecting a TV set, this page appears on your smartphone. Enter pairing key and select “Submit” button. If you want to go back to the previous TV list, select “Back” button.



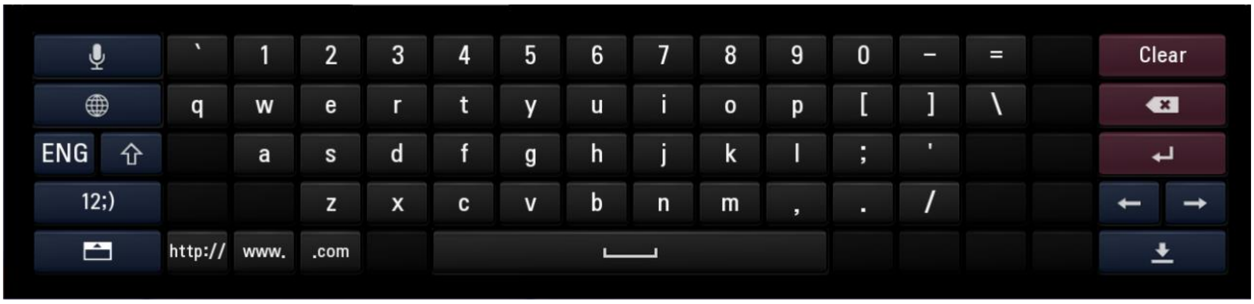
Enter pairing keys in order without an empty spot on the bottom of TV screen.
But when pairing keys doesn't show on TV screen, convert TV to Live TV.

Virtual Keyboard

Virtual Keyboard is designed to appear as below when a text input element is focused. Developers can easily use virtual keyboard by including virtual keyboard library.

USB keyboard input and Voice Recognition are supported since NetCast 3.0 (LG Smart TV 2012).

To download the Virtual Keyboard Library, go to [DISCOVER > Legacy Platform (NetCast) > Tools & Samples] menu in LG Developer (**http://webostv.developer.lge.com**) website.



Using Virtual Keyboard Library

To use the virtual keyboard, add the following script in the head of html file. In order to use virtual keyboard in iframes, it should be added in the html file connected to the iframe.
(Relative path of the jsLgVKeyboard folder should be modified according to the real path.)

In a normal or parent page

```
<link type="text/css" rel="Stylesheet" href="../jsLgVKeyboard/LgVKeyboard.css" />
<script id="mainVKScript" type="text/javascript"
src="../jsLgVKeyboard/LgVKeyboard.js"></script>
```

In iframe page

```
<script type="text/javascript"
src="../jsLgVKeyboard/LgVKeyboardIframe.js"></script>
```

As Virtual Keyboard uses keydown and keyup events, use the lgKb.onKeyDown and lgKb.onKeyUp methods as below to use keydown event in the application instead of setting handler of window.onkeydown event.

```
<script>
    function onKeyDown(event) {
        var keyCode = event.keyCode;
        switch(keyCode) {
            case VK_LEFT :
```

```
            ...
            break;
            case VK_RIGHT :
            ...
            break;
            ...
        }
    }

    function onKeyUp(event)    {
        var keyCode = event.keyCode;
        switch(keyCode) {
            case VK_LEFT :
            ...
            break;
            case VK_RIGHT :
            ...
            break;
            ...
        }
    }

    lgKb.onKeyDown = onKeyDown;
    lgKb.onKeyUp = onKeyUp;

</script>
```

Virtual Keyboard library calls lgKb.onKeyboardShow or lgKb.onKeyboardHide methods after Virtual Keyboard appear or disappear. Override the lgKb.onKeyboardShow and lgKb.onKeyboardHide methods as below for additional task.

```
<script>
    function onKeyboardShow(event)    {
        ...
    }

    function onKeyboardHide(event)    {
        ...
    }

    lgKb.onKeyboardShow = onKeyboardShow;
    lgKb.onKeyboardHide = onKeyboardHide;

</script>
```

- Do not modify the folder structure and file name of jsLgVKeyboard.
- The script must include the following declaration: **id = “mainVKScript”**
- A web page should be set character set as UTF-8 to use virtual keyboard as following:

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
```

- Virtual keyboard is enabled in the following tag/types:

Tag	Type
INPUT	TEXT
INPUT	PASSWORD
TEXTAREA	-

Note

Beware of cross-domain problem when using iframe. If an application uses multiple html files those are loaded from different domains, cross-domain problem occurs and virtual keyboard cannot be used in iframe.

Supported Language

Virtual keyboard supports 42 languages, and 16 languages are supported for voice recognition of them. Refer to **Developing > API > Voice Recognition Plugin and API** section in this Library for detailed information.

Performance Consideration

This section describes the performance considerations for web application developers.

- Application screen size "1280 x 720" is recommended. If the width is bigger than the height, unvisible area may occur.

- With the use of authoring tools, you can make Web applications that do not have complex structure and unnecessary codes. When using an Web authoring tool, you must check if it supports the standard markup and CSS layout. Also, check if grammers are correctly applied even after you finished the development. W3C validator can be used to find some mistakes in the content (**http://validator.w3.org/**)

- Since TV has lower performance than PC, optimization must be done with JavaScript functions. A time-consuming task should be separated into several tasks so that it can be implemented with several functions.

- If animation and transition features are implemented without considering performance using UI Framework library which is based on JavaScript and CSS, performance issues can be caused.

- Although the Transparent function with the usage of CSS filter is useful for making fancy UI, it may cause performence issue due to the huge operations.

- If you minimize the usage of JavaScript functions in the initial page, the execution time will be reduced. Hence, we recommend to make the 'onload' function simple.

- For Web application's optimization, we recommend to analyze and optimize the contents to the page speed. (**https://developers.google.com/speed/pagespeed/**)

- To close the application, call window.NetCastBack() function.

- The speed of Web application will be significantly different depending on the size of each page and how complex the Web application is. If a Web application consists of one page implemented with multiple content blocks including hidden blocks, this may cause performance issues. In this case, the application has to be divided into multiple images, and each page has to be consists of limited number of content blocks to improve the performance.

- It is recommended not to use a redirect page because it deteriorates the performance by increasing the loading time.

Note

Keep pagesize small and limit the number of elements in a page.

JavaScript

General

When implementing code, it is important to optimize the code as much as possible because the NetCast Platform has lower computing performance than general PC platforms. In general, it is better to avoid using complex expressions in order to optimize the speed of Web application. Nested loops should not be used as much as possible. Compressing the web application is not always necessary, but removing unnecessary tags and CSS codes and shortening the name of functions and variables will help to improve performance.

One of the most frequent mistakes made by inexperienced developers is trying to use the element within the <body> tag in the Javascript to initialize the web page before the all <body> tags loaded. This will cause an error because the element cannot be used in the Javascript. Therefore, when initializing a Web page using the element included in the <body> tag, you need to use the OnLoad event handler that is generated only after the web page is fully loaded.

The following is a code sample of how you should not call your Javascript functions:

```
<script type="text/javascript">
    function testFunction()
    {
        document.getElementById('testDiv').innerHTML='Test text';
    }
    testFunction();
</script>
</head>
<body>
    ...
    <div id="testDiv">Initial text</div>
</body>
```

A correct code using the body-onload is provided as follows:

```
<script type="text/javascript">
    function testFunction()
    {
       document.getElementById('testDiv').innerHTML='Test text';
    }
</script>
</head>
<body onload="testFunction();">
    ...
    <div id="testDiv">Initial text</div>
</body>
```

Note

You must use the OnLoad event handler of the <body> tag to initialize web pages.

Timers

Timer is often used to implement animation effects using Javascript. Using a delay of 100 msec or less on the NetCast Platform is insignificant.

Also, using multiple timers on a single page is not desirable. If you do, the NetCast Platform does not work properly.

Note

When using a timer on the NetCast Platform, the delay time should be set to longer than 100 msec.

XMLHttpRequest

The following is an excerpt for using the XMLHttpRequest object:

```
    xmlHttp = new XMLHttpRequest();
...
    xmlHttp.onreadystatechange = handleStateChange;
    xmlHttp.open("GET", "http://testserver/testdata.xml", true);
    xmlHttp.send(null);

    function handleStateChange() {
        ...
    }
...
```

To increase performance, one can use JavaScript Object Notation (JSON) instead. An example of a JSON data structure is provided as follows:

```
{
   "tagName": "testObject",
   "tagProperties": {
      "data": "testData"
   },
   "tagContent": [
      "test content 1"
      "test content 2"
   ]
}
```

The NetCast platform has support for native JSON parsing using JSON.parse().

Eval

The eval function is very fast. However, it can compile and execute any JavaScript program, so there can be security issues. The use of eval is indicated when the source is trusted and competent. It is much safer to use a JSON parser. In web applications over XMLHttpRequest, communication is permitted only to the same origin that provide that page, so it is trusted. But it might not be competent. If the server is not rigorous in its JSON encoding, or if it does not scrupulously validate all of its inputs, then it could deliver invalid JSON text that could be carrying dangerous script. The eval function would execute the script, unleashing its malice.
To defend against this, a JSON parser should be used. A JSON parser will recognize only JSON text, rejecting all scripts. In NetCast platform, JSON parsers are also much faster than eval.

Popups

Popup is not supported because only one window is available for a Web application on the NetCast Platform. To implement the popup window effect, use the <div> tag, instead of window.open.

Note

The NetCast Platform does not support popups for a Web application.

Annex A Complete List of Supported MIME Types

The following table contains all MIME types which are currently supported by the NetCast platform.

Container	MIME type
general supported media	application/x-netcast-av
Device Information	application/x-netcast-info
WMV (ASF)	video/x-ms-asf, video/x-ms-wmv
WMA	audio/x-ms-wma
MP3	audio/mpeg, audio/mp3
MP4	video/mp4, video/mpeg
m3u8	application/vnd.apple.mpegurl

Annex B Complete List of Supported Codecs and Containers

Common File Extension	Media Container	A/V	Codec	Four CC	Profile @ Level	Max Bitrate	Max Resolution	Note
*mp 4	MP 4	Video	H.264/AVC	H.264 AVC1	Baseline@L1.2 / L3.0 Main Profile @ Auto level Main Profile @ L3.0 / L4.1 High Profile @ L3.2 / L4.1 (e.g. 720p60, 1080i60, 1080p30)	700 kbps(minimum) ~ 4000 kbps	1920*1080	We encode at a Constant BitRate (CBR) for delivery via streaming as this smooths out any need to buffer
		Audio	AAC		Low Complexity Profile			AAC-LC and AAC-HE
*.wmv *.asf	ASF	Video	VC-1 Advanced Profile	WVC 1	Advanced Profile @ L1 / L3	6000 kbps	1280*720 @ 30 fps	
			VC-1	WMV 3	Main Profile @ Low / Main Level	3000 kbps	1280*720 @ 30 fps	
		Audio	WMA 9 Standard	WMA 2		32 ~ 192 kbps, 48kHz		
			WMA 9 PRO	WMA 3		48 ~ 384 kbps, 48 kHz		Cannot support - over 48 KHz sampling rate - over 6 channel lossless - WMA10 Pro LBR v2, v3 (M0b profile)
*.asf	ASX	Audio	WMA 9 Standard	WMA 2		32 ~ 192 kbps, 48kHz		
			WMA 9 PRO			48 ~ 384 kbps, 48 kHz		Cannot support WMA10 Pro LBR v2, v3 (M0b profile)
*.wvm	Widevine proprietary	Video	H.264	H.264	Main Profile @ L3.0 / L3.1 / L4.0	Adaptive streaming: 600kbps ~ 8300kbps	720x406 @ 25 fps 1280x720 @ 25 fps 1920x1080 @ 25 fps	Widevine package
		Audio	HE-AAC			128 kbps		
*.avi	AVI	Video	Divx4, Divx5			1250 kbps	480x360 @ 25fps	
		Audio	mp3			128 kbps / 44 Khz		
*.ts	TS	Video	H.264			3000 kbps	1920x1080p	
		Audio	AAC					

Common File Extension	Media Container	A/V	Codec	Four CC	Profile @ Level	Max Bitrate	Max Resolution	Note
*.mp3	MP3	Audio	MPEG1 Layer III			320 kbps		
*.mp3	ASX	Audio	MP3			320 kps		
*.wma	ASF	Audio	WMA 9 Standard	WMA 2		32 ~ 192 kbps, 48 kHz		
			WMA 9 PRO	WMA 3		48 ~ 384 kbps, 48 kHz		Cannot support - over 48 KHz sampling rate - over 6 channel lossless - WMA10 Pro LBR v2, v3 (M0b profile)

Video and audio data should be interleaved in time base when they are encoded in the file.

Annex C ISO 639 Code Conversion Table[3][8]

ISO 639-2	ISO 639-1	Name of language
aar	aa	Afar
abk	ab	Abkhazian
ace		Achinese
ach		Acoli
ad		Audio Description (France HD Forum)
ada		Adangme
ady		Adyghe; Adygei
afa		Afro-Asiatic (Other)
afh		Afrihili
afr	af	Afrikaans
ain		Ainu
aka	ak	Akan
akk		Akkadian
alb	sq	Albanian (ISO 639-2/B)
ale		Aleut
alg		Algonquian languages
alt		Southern Altai
amh	am	Amharic
ang		English, Old (ca. 450-1100)
anp		Angika
apa		Apache languages
ara	ar	Arabic
arc		Official Aramaic (700-300 BCE); Imperial Aramaic (700-300 BCE)

ISO 639-2	ISO 639-1	Name of language
arg	an	Aragonese
arm	hy	Armenian (ISO 639-2/B)
arn		Araucanian; Mapuche
arp		Arapaho
art		Artificial (Other)
arw		Arawak
asm	as	Assamese
ast		Asturian; Bable; Leonese; Asturleonese
ath		Athapascan languages
aus		Australian languages
ava	av	Avaric
ave	ae	Avestan
awa		Awadhi
aym	ay	Aymara
aze	az	Azerbaijani
bad		Banda
bai		Bamileke languages
bak	ba	Bashkir
bal		Baluchi
bam	bm	Bambara
ban		Balinese
baq	eu	Basque (ISO 639-2/B)
bas		Basa
bat		Baltic (Other)
bej		Beja; Bedawiyet
bel	be	Belarusian
bem		Bemba
ben	bn	Bengali
ber		Berber (Other)
bho		Bhojpuri
bih	bh	Bihari
bik		Bikol
bin		Bini; Edo
bis	bi	Bislama
bla		Siksika
bnt		Bantu (Other)
bod	bo	Tibetan (ISO 639-2/T)
bos	bs	Bosnian (not from ISO 639-2 1998, but from Wekipedia)
bra		Braj

ISO 639-2	ISO 639-1	Name of language
bre	br	Breton
btk		Batak (Indonesia)
bua		Buriat
bug		Buginese
bul	bg	Bulgarian
bur	my	Burmese (ISO 639-2/B)
byn		Blin; Bilin
cad		Caddo
cai		Central American Indian (Other)
car		Galibi; Carib
cat	ca	Catalan; Valencian
cau		Caucasian (Other)
cdo		Min Dong Chinese (from ISO 639-3)
ceb		Cebuano
cel		Celtic languages
ces	cs	Czech (ISO 639-2/T)
cha	ch	Chamorro
chb		Chibcha
che	ce	Chechen
chg		Chagatai
chi	zh	Chinese (ISO 639-2/B) : means Traditional Chinese in HK Subtitle (by Labelling Scheme Meeting 20080617.doc)
chk		Chuukese
chm		Mari
chn		Chinook jargon
cho		Choctaw
chp		Chipewyan; Dene Suline
chr		Cherokee
chs		Simplified Chinese in HK Subtitle (by Labelling Scheme Meeting 20080617.doc) : Eventhough it conflicts with Chumash in ISO 639-3, Chumash is extinct.
chu	cu	Church Slavic; Old Slavonic; Church Slavonic; Old Bulgarian; Old Church Slavonic
chv	cv	Chuvash
chy		Cheyenne
cmc		Chamic languages
cmn	zh-cn	Mandarin Chinese (from ISO 639-3)
cop		Coptic
cor	kw	Cornish
cos	co	Corsican
cpe		Creoles and pidgins, English based (Other)
cpf		Creoles and pidgins, French-based (Other)

ISO 639-2	ISO 639-1	Name of language
cpp		Creoles and pidgins, Portuguese-based (Other)
cre	cr	Cree
crh		Crimean Tatar; Crimean Turkish
crp		Creoles and pidgins (Other)
csb		Kashubian
cus		Cushitic (Other)
cym	cy	Welsh (ISO 639-2/T)
cze	cs	Czech (ISO 639-2/B)
dak		Dakota
dan	da	Danish
dar		Dargwa
day		Dayak
del		Delaware
den		Slave (Athapascan)
deu	de	German (ISO 639-2/T)
dgr		Dogrib
din		Dinka
div	dv	Divehi; Dhivehi; Maldivian
doi		Dogri
dra		Dravidian (Other)
dsb		Lower Sorbian
dua		Duala
dum		Dutch, Middle (ca. 1050-1350)
dut	nl	Dutch; Flemish (ISO 639-2/B)
dyu		Dyula
dzo	dz	Dzongkha
efi		Efik
egy		Egyptian (Ancient)
eka		Ekajuk
ell	el	Greek, Modern (post 1453) (ISO 639-2/T)
elx		Elamite
eng	en-gb, en-us, en-au	English, ‘en-us’ is for USA and Taiwan, ‘en-au’ is for Australia, ‘en-gb’ is for other nations
enm		English, Middle (1100-1500)
epo	eo	Esperanto
esp	es	Spanish; Castilian (ISO 639-2/T)
est	et	Estonian
eus	eu	Basque (ISO 639-2/T)
ewe	ee	Ewe

ISO 639-2	ISO 639-1	Name of language
ewo		Ewondo
fan		Fang
fao	fo	Faroese
fas	fa	Persian (ISO 639-2/T)
fat		Fanti
fij	fj	Fijian
fil		Filipino; Pilipino
fin	fi	Finnish
fiu		Finno-Ugrian (Other)
fon		Fon
fra	fr, fr-ca	French (ISO 639-2/T), ‘fr-ca’ is for Canada
fre	fr, fr-ca	French (ISO 639-2/B), ‘fr-ca’ is for Canada
frm		French, Middle (ca. 1400-1600)
fro		French, Old (842-ca. 1400)
frr		Northern Frisian
frs		Eastern Frisian
fry	fy	Western Frisian
ful	ff	Fulah
fur		Friulian
gaa		Ga
gae		Gaelic (Scots) (not from ISO 639, versions below 5.0 of DBook specified this)
gay		Gayo
gba		Gbaya
gdh		Gaelic (Scots) (not from ISO 639-2 1998, maybe it was from old ver. of ISO 639)
gem		Germanic (Other)
geo	ka	Georgian (ISO 639-2/B)
ger	de	German (ISO 639-2/B)
gez		Geez
gil		Gilbertese
gla	gd	Gaelic; Scottish Gaelic
gle	ga	Irish
glg	gl	Galician
glv	gv	Manx
gmh		German, Middle High (ca. 1050-1500)
goh		German, Old High (ca. 750-1050)
gon		Gondi
gor		Gorontalo
got		Gothic

ISO 639-2	ISO 639-1	Name of language
grb		Grebo
grc		Greek, Ancient (to 1453)
gre	el	Greek, Modern (post 1453) (ISO 639-2/B)
grn	gn	Guarani
gsw		Swiss German; Alemannic; Alsatian
guj	gu	Gujarati
gwi		Gwich'in
hai		Haida
hat	ht	Haitian; Haitian Creole
hau	ha	Hausa
haw		Hawaiian
heb	he	Hebrew
her	hz	Herero
hil		Hiligaynon
him		Himachali languages; Western Pahari languages
hin	hi	Hindi
hit		Hittite
hmn		Hmong
hmo	ho	Hiri Motu
hrv	hr	Croatian (ISO 639-2/T)
hsb		Upper Sorbian
hun	hu	Hungarian
hup		Hupa
hye	hy	Armenian (ISO 639-2/T)
iba		Iban
ibo	ig	Igbo
ice	is	Icelandic (ISO 639-2/B)
ido	io	Ido
iii	ii	Sichuan Yi; Nuosu
ijo		Ijo
iku	iu	Inuktitut
ile	ie	Interlingue; Occidental
ilo		Iloko
ina	ia	Interlingua (International Auxiliary Language Association)
inc		Indic (Other)
ind	id	Indonesian
ine		Indo-European (Other)
inh		Ingush
ipk	ik	Inupiaq

ISO 639-2	ISO 639-1	Name of language
ira		Iranian (Other)
iro		Iroquoian languages
isl	is	Icelandic (ISO 639-2/T)
ita	it	Italian
jav	jv	Javanese (ISO 639-2/B)
jbo		Lojban
jaw		Javanese (ISO 639-2/T)
jpn	ja	Japanese
jpr		Judeo-Persian
jrb		Judeo-Arabic
kaa		Kara-Kalpak
kab		Kabyle
kac		Kachin; Jingpho
kal	kl	Kalaallisut; Greenlandic
kam		Kamba
kan	kn	Kannada
kar		Karen
kas	ks	Kashmiri
kat	ka	Georgian (ISO 639-2/T)
kau	kr	Kanuri
kaw		Kawi
kaz	kk	Kazakh
kbd		Kabardian
kha		Khasi
khi		Khoisan (Other)
khm	km	Khmer
kho		Khotanese; Sakan
kik	ki	Kikuyu; Gikuyu
kin	rw	Kinyarwanda
kir	ky	Kirghiz; Kyrgyz
kmb		Kimbundu
kok		Konkani
kom	kv	Komi
kon	kg	Kongo
kor	ko	Korean
kos		Kosraean
kpe		Kpelle
krc		Karachay-Balkar
krl		Karelian

ISO 639-2	ISO 639-1	Name of language
kro		Kru
kru		Kurukh
kua	kj	Kuanyama; Kwanyama
kum		Kumyk
kur	ku	Kurdish
kut		Kutenai
lad		Ladino
lah		Lahnda
lam		Lamba
lao	lo	Lao
lat	la	Latin
lav	lv	Latvian
lez		Lezghian
lim	li	Limburgan; Limburger; Limburgish
lin	ln	Lingala
lit	lt	Lithuanian
lol		Mongo
loz		Lozi
ltz	lb	Luxembourgish; Letzeburgesch
lua		Luba-Lulua
lub	lu	Luba-Katanga
lug	lg	Ganda
lui		Luiseno
lun		Lunda
luo		Luo (Kenya and Tanzania)
lus		Lushai
mac	mk	Macedonian (ISO 639-2/B)
mad		Madurese
mag		Magahi
mah	mh	Marshall
mai		Maithili
mak		Makasar
mal	ml	Malayalam
man		Mandingo
mao	mi	Maori
map		Austronesian (Other)
mar	mr	Marathi
mas		Masai
may	ms	Malay (ISO 639-2/B)

ISO 639-2	ISO 639-1	Name of language
mdf		Moksha
mdr		Mandar
men		Mende
mga		Irish, Middle (900-1200)
mic		Mi'kmaq; Micmac
min		Minangkabau
mis		Miscellaneous languages
mkd	mk	Macedonian (ISO 639-2/T)
mkh		Mon-Khmer (Other)
mlg	mg	Malagasy
mlt	mt	Maltese
mnc		Manchu
mni		Manipuri
mno		Manobo languages
moh		Mohawk
mol		Moldavian
mon	mn	Mongolian
mos		Mossi
mri	mi	Maori (ISO 639-2/T)
msa	ms	Malay (ISO 639-2/T)
mul		Multiple languages
mun		Munda languages
mus		Creek
mwl		Mirandese
mwr		Marwari
mya	my	Burmese (ISO 639-2/T)
myn		Mayan languages
myv		Erzya
nah		Nahuatl
nai		North American Indian
nap		Neapolitan
nan	zh-tw	Min Nan Chinese (from ISO 639-3)
nau	na	Nauru
nav	nv	Navajo; Navaho
nbl	nr	Ndebele, South; South Ndebele
nde	nd	Ndebele, North; North Ndebele
ndo	ng	Ndonga
nds		Low German; Low Saxon; German, Low; Saxon, Low
nep	ne	Nepali

ISO 639-2	ISO 639-1	Name of language
new		Nepal Bhasa; Newari
nia		Nias
nic		Niger-Kordofanian (Other)
niu		Niuean
nld	nl	Dutch; Flemish (ISO 639-2/T)
nno	nn	Norwegian Nynorsk; Nynorsk, Norwegian
nob	nb	Bokmal, Norwegian; Norwegian Bokmal
nog		Nogai
non		Norse, Old
nor	no	Norwegian
nqo		N'Ko
nso		Pedi; Sepedi; Northern Sotho
nub		Nubian languages
nwc		Classical Newari; Old Newari; Classical Nepal Bhasa
nya	ny	Chichewa; Chewa; Nyanja
nym		Nyamwezi
nyn		Nyankole
nyo		Nyoro
nzi		Nzima
oci	oc	Occitan (post 1500)
oji	oj	Ojibwa
ori	or	Oriya
orm	om	Oromo
osa		Osage
oss	os	Ossetian; Ossetic
ota		Turkish, Ottoman (1500-1928)
oto		Otomian languages
paa		Papuan (Other)
pag		Pangasinan
pal		Pahlavi
pam		Pampanga; Kapampangan
pan	pa	Panjabi; Punjabi
pap		Papiamento
pau		Palauan
peo		Persian, Old (ca. 600-400 B.C.)
per	fa	Persian (ISO 639-2/B)
phi		Philippine (Other)
phn		Phoenician
pli	pi	Pali

ISO 639-2	ISO 639-1	Name of language
pol	pl	Polish
pon		Pohnpeian
por	pt, pt-br	Portuguese, ‘pt-br’ is for Brazil
pra		Prakrit languages
pro		Provencal, Old (to 1500);Occitan, Old (to 1500)
pus	ps	Pushto; Pashto
qaa		original audio (not from ISO 639, from 'EN 300 468 Annex F')
qad		Audio Description (France HD Forum)
que	qu	Quechua
raj		Rajasthani
rap		Rapanui
rar		Rarotongan; Cook Islands Maori
roa		Romance (Other)
roh	rm	Raeto-Romance
rom		Romany
ron	ro	Romanian; Moldavian; Moldovan (ISO 639-2/T)
rum	ro	Romanian; Moldavian; Moldovan (ISO 639-2/B)
run	rn	Rundi
rup		Aromanian; Arumanian; Macedo-Romanian
rus	ru	Russian
sad		Sandawe
sag	sg	Sango
sah		Yakut
sai		South American Indian (Other)
sal		Salishan languages
sam		Samaritan Aramaic
san	sa	Sanskrit
sas		Sasak
sat		Santali
scn		Sicilian
scc		Serbian (ISO 639-2/B)
sco		Scots
scr		Croatian (ISO 639-2/B)
sel		Selkup
sem		Semitic (Other)
sga		Irish, Old (to 900)
shn		Shan
sid		Sidamo
sin	si	Sinhala; Sinhalese

ISO 639-2	ISO 639-1	Name of language
sio		Siouan languages
sit		Sino-Tibetan (Other)
sla		Slavic (Other)
slk	sk	Slovak (ISO 639-2/T)
slo	sk	Slovak (ISO 639-2/B)
slv	sl	Slovenian
sma		Southern Sami
sme	se	Northern Sami
smi		Sami languages
smj		Lule Sami
smn		Inari Sami
smo	sm	Samoan
sms		Skolt Sami
sna	sn	Shona
snd	sd	Sindhi
snk		Soninke
sog		Sogdian
som	so	Somali
son		Songhai
sot	st	Sotho, Southern
spa	es	Spanish; Castilian (ISO 639-2/B)
sqi	sq	Albanian (ISO 639-2/T)
srd	sc	Sardinian
srn		Sranan Tongo
srp	sr	Serbian (ISO 639-2/T)
srr		Serer
ssa		Nilo-Saharan (Other)
ssw	ss	Swati
suk		Sukuma
sun	su	Sundanese
sus		Susu
sux		Sumerian
swa	sw	Swahili
swe	sv	Swedish
syc		Classical Syriac
syr		Syriac
tah	ty	Tahitian
tai		Tai (Other)
tam	ta	Tamil

ISO 639-2	ISO 639-1	Name of language
tat	tt	Tatar
tel	te	Telugu
tem		Timne
ter		Tereno
tet		Tetum
tgk	tg	Tajik
tgl	tl	Tagalog
tha	th	Thai
tib	bo	Tibetan (ISO 639-2/B)
tig		Tigre
tir	ti	Tigrinya
tiv		Tiv
tkl		Tokelau
tlh		Klingon; tlhIngan-Hol
tli		Tlingit
tmh		Tamashek
tog		Tonga (Nyasa)
ton	to	Tonga (Tonga Islands)
tpi		Tok Pisin
tsi		Tsimshian
tsn	tn	Tswana
tso	ts	Tsonga
tuk	tk	Turkmen
tum		Tumbuka
tup		Tupi languages
tur	tr	Turkish
tut		Altaic (Other)
tvl		Tuvalu
twi	tw	Twi
tyv		Tuvinian
uga		Ugaritic
uig	ug	Uighur; Uyghur
ukr	uk	Ukrainian
umb		Umbundu
und		Undetermined
urd	ur	Urdu
uzb	uz	Uzbek
vai		Vai
val		Valencian (used Spain)

ISO 639-2	ISO 639-1	Name of language
ven	ve	Venda
vie	vi	Vietnamese
vol	vo	Volapuk
vot		Votic
wak		Wakashan languages
wal		Wolaitta; Wolaytta
war		Waray
was		Washo
wel	cy	Welsh (ISO 639-2/B)
wen		Sorbian languages
wln	wa	Walloon
wol	wo	Wolof
xal		Kalmyk; Oirat
xho	xh	Xhosa
yao		Yao
yap		Yapese
yid	yi	Yiddish
yor	yo	Yoruba
ypk		Yupik languages
yue	zh-hk	Yue Chinese (Cantonese) (from ISO 639-3)
zap		Zapotec
zbl		Blissymbols; Blissymbolics; Bliss
zen		Zenaga
zha	za	Zhuang; Chuang
zho	zh	Chinese (ISO 639-2/T)
znd		Zande
zul	zu	Zulu
zun		Zuni
zxx		No linguistic content; Not applicable
zza		Zaza; Dimili; Dimli; Kirdki; Kirmanjki; Zazaki

Annex D Differences in Media Devices

Chapter of this Document	Constraints and difference of Media Devices compared with TV

<table>
<tr><th>Chapter of this Document</th><th>Constraints and difference of Media Devices compared with TV</th></tr>
<tr><td>Video Outputf</td><td>Some of LG Media products have HDMI, Composite, Component and Scart as output connection types.

- HDMI:
The video content is outputted on a HDCP-protected HDMI output interface only. If the TV does not support HDCP, the video contents can only be outputted on the analog video outputs.

- Analog Outputs:
Analog video outputs are protected with CGMS-A. (CopyNever)</td></tr>
<tr><td>Web Engine</td><td>Support only in AV tag from NetCast 3.0</td></tr>
<tr><td>userAgent String</td><td>A. All LG_NetCastVersion:
Change LG NetCast.TV to LG NetCast.Media for Media products.

B. [Table] Optional feature string for capabilities field:
Add Feature String (+NO_NUM), Optional Feature (Support for non-numeric IR Remote).

C. Applications only need to use LG Smart TV for defining TV products and LG Media for defining Media products.</td></tr>
<tr><td>Protocols</td><td>A. Media uses below protocols.
<table>
<tr><th>Use</th><th>Preferred protocols</th><th>Supported protocols</th></tr>
<tr><td>HTML file transportation</td><td>HTTP, HTTPS</td><td>HTTPS, HTTPS</td></tr>
<tr><td>VoD media file transportation</td><td>HTTP, HTTPS, HLS, Widevine, MMSH/MMST</td><td>HTTP, HTTPS, HLS, Widevine, MMSH/MMST</td></tr>
<tr><td>Linear or live broadcasting</td><td>HLS, Widevine</td><td>HLS, Widevine</td></tr>
</table></td></tr>
<tr><td>DRM</td><td>Media products do not support Adobe Access 3.0 DRM.</td></tr>
<tr><td>Subtitle</td><td>A. Media products support below subtitle formats.
<table>
<tr><th>Format</th><th>2011 platform/SP520</th><th>2012 / 2013 platform</th></tr>
<tr><td>SAMI</td><td>O</td><td>O</td></tr>
<tr><td>CineCanvas</td><td>O</td><td>X</td></tr>
<tr><td>Timed Text</td><td>O</td><td>X</td></tr>
<tr><td>SRT</td><td>O</td><td>O</td></tr>
<tr><td>SUB</td><td>O</td><td>O</td></tr>
<tr><td>SSA</td><td>X</td><td>O</td></tr>
</table>
B. Reference url add
- SRT : http://matroska.org/technical/specs/subtitles/srt.html
- SSA : http://matroska.org/technical/specs/subtitles/ssa.html

C. Detailed specification for SRT
<table>
<tr><th>Start</th><th>End</th><th>Contents</th></tr>
<tr><td>--></td><td></td><td>Expression time interval to display</td></tr>
<tr><td>&nbsp</td><td></td><td>Non-breaking space</td></tr>
</table>
D. Detailed specification for SSA
<table>
<tr><th>Start</th><th>End</th><th>Contents</th></tr>
<tr><td>[Script Info]</td><td></td><td>Subtitle file header information</td></tr>
<tr><td>[V4 Styles]</td><td></td><td>Style information</td></tr>
<tr><td>[Events]</td><td></td><td>Represents the main content of the document</td></tr>
<tr><td>&nbsp</td><td></td><td>Non-breaking space</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Chapter of this Document</th><th>Constraints and difference of Media Devices compared with TV</th></tr>
<tr><td></td><td>E. Detailed specification for SUB
<table>
<tr><th>Start</th><th>End</th><th>Contents</th></tr>
<tr><td>{</td><td>}</td><td>Set the start and last time for the caption</td></tr>
<tr><td>&nbsp</td><td></td><td>Non-breaking space</td></tr>
</table></td></tr>
<tr><td>Images</td><td>A. JPEG/PNG
i. Less than 4,000 x 3,000 x 24 bit/pixel
ii. Less than 3,000 x 3,000 x 32 bit/pixel

B. GIF
Media products do not support GIF format.</td></tr>
<tr><td>HTTP Streaming</td><td>Media products do not support Progressive download but HTTP/HTTPS streaming.</td></tr>
<tr><td>Document Scrolling</td><td>"PR+" (Programme Up) and "PR-" (Programme Down) buttons are not available on Media products.</td></tr>
<tr><td>Font and Font Style</td><td>Media products have only "LG Display", "LG Display_HK" and "TiresiasScreenfont"' font.</td></tr>
<tr><td>Input Device</td><td>Traditional IR Remote, Magic Remote and HID (USB Keyboard) are available.</td></tr>
<tr><td>Traditional IR Remote Key Events</td><td>There are some unavailable keys on Media products which are VK_INFO, VK_PAGE_UP and VK_PAGE_DOWN. Moreover, some Media products do not support numeric key.</td></tr>
<tr><td>Key Repeat Function</td><td>There are some unavailable keys on Media products which are VK_INFO, VK_PAGE_UP and VK_PAGE_DOWN</td></tr>
<tr><td>Smart Text</td><td>Media products supports this function only in NetCast 3.0.</td></tr>
<tr><td>Complete List of Supported MIME Types</td><td>Media products also support below MIME
- AVI : video/x-msvideo</td></tr>
<tr><td>Complete List of Supported Codecs and Containers</td><td>- *.mp4
Add codec (MPEG4 P2), FourCC (MP4V) and Max Resolution (1280x720@60FPS).
- *.asf (- extension when Media container is ASX)
Change common File Extension from *.asf to *.asf, *.wmv, *.wma
- *.mov
Media product does not support this.
- *.ts
Change Video Max Bitrate to 800~4500Kbps
- *.mp3
Change Media container from ID3V1, ID3V2 to mp3.
- *.wav
Change Media Container from ASF to WAV
- WMA PRO
Media product don't support WMA PRO</td></tr>
</table>

Annex E HTTPS Server and Client Certificated

The NetCast Platform supports HTTPS server certificate and client certificate.

1. Server certificate
A. The NetCast platform has trusted CAs to be able to verify Secure Socket Layer (SSL) server certificates. The list of

public CAs can be downloaded from [DISCOVER > Legacy Platform (NetCast) > Technical Notes > Web_List of Public CAs and LG Root CA] in this site.

B. The NetCast platform supports the hostname matching with the Subject Alternative Name (SAN) extension and wild card certificates.

2. Client certificate
A. If the server requests the client certificate during the Secure Socket Layer (SSL) connection, the LG Browser sends client certificate. However, the streaming module of media player does not support the client certificate.
The client certificate is a standard X.509 v3 certificate and has been issued by LG custom CA. The LG root CA of TV and media can be downloaded with this document.

Annex F References

The following referenced documents are required for the application of this specification. For dated or versioned references, only the edition cited applies. For nonspecific references, the latest edition of the referenced document (including any amendments) applies.

Index	Reference & Version	Title
[1]	ETSI TS 102 796 V1.1.1 (2010-06)	Hybrid Broadcast Broadband TV
[2]	IETF RFC 2616 (http://tools.ietf.org/html/rfc2616)	Hypertext Transfer Protocol -- HTTP/1.1
[3]	http://www.loc.gov/standards/iso639-2/php/code_list.php	ISO 639-2 Language Code List
[4]	http://en.wikipedia.org/wiki/Advanced_Stream_Redirector	Advanced Stream Redirector (ASX)
[5]	http://msdn.microsoft.com/en-us/library/dd564668%28VS.85%29.aspx	Windows Media Metafile Elements Reference (ASX)
[6]	ISO/IEC 10646	Universal Multiple-Octet Coded Character Set (UCS)
[7]	CEA-2014	Web-based Protocol and Framework for Remote User Interface on UPnP™ Networks and the Internet (Web4CE) : CE-HTML
[8]	ISO 639	Codes for the representation of names of languages

Supported HTML5/CSS3 Standard for the NetCast Platform and Emulator

This document describes HTML5 and CSS3 supported by the NetCast platform and Emulator.
The table below provides a brief description of each element or API.

Topics	Description
HTML5 Video and Audio	This topic describes how to play a video or audio file on the web without plugins. It also lists the methods, attributes and events for Video and Audio. For the detail support information on the NetCast platform and Emulator, refer to the table on each **Video** and **Audio** section.
HTML5 Communication	This topic describes how to share data between multiple programs in an asynchronous manner. It also lists the methods, attributes and events for Cross-Document Messaging, Channel Messaging, MessageEvent, Web Sockets and Server-sent Events. For the detail support information on the NetCast platform and Emulator, refer to the table on each **Cross-Document Messaging**, **Channel Messaging**, **MessageEvent**, **Web Sockets** and **Server-sent Events** section.
HTML5 Canvas	This topic describes how to create graphics using javascript. It also lists the methods and attributes for Canvas. For the detail support information on the NetCast platform and Emulator, refer to the table on each **Canvas** section.
HTML5 Inline SVG, WebGL	This topic describes how to define vector-based graphics in XML format. It also lists the elements for Inline SVG and WebGL. For the detail support information on the NetCast platform and Emulator, refer to the table on each **Inline SVG** and **WebGL** section.
HTML5 Offline Web Applications	This topic describes how to operate web applications on offline state. It also lists the methods, attributes and events for Offline Web Applications. For the detail support information on the NetCast platform and Emulator, refer to the table on each **Offline Web Applications** section.
HTML5 Web Storage	This topic describes how to save the data on the disk of clients. It also lists the methods and attributes for Storage, Indexed Database and WebSQL Database. For the detail support information on the NetCast platform and Emulator, refer to the table on each **Storage**, **Indexed Database** and **WebSQL Database** section.
HTML5 User Interactions	This topic describes Drag and Drop and History Management. For the detail support information on the NetCast platform and Emulator, refer to the table on each **Drag and Drop** and **History Management** section.
HTML5 Web Workers	This topic describes how to use the multi-task in the web browser. It also lists the methods, attributes and events for Workers and Shared workers. For the detail support information on the NetCast platform and Emulator, refer to the table on each **Workers** and **Shared workers** section.
CSS3	This topic describes which properties of CSS3 are supported by the NetCast platform and LG Emulator. It also provides information on each Dynamic CSS and Static CSS. For the detail support information on the NetCast platform and Emulator, refer to the table on each **Dynamic CSS** and **Static CSS** section.

Introduction

This document describes the elements and properties of HTML5 and CSS3 which are currently supported by the NetCast platform and LG Smart TV Emulator.

The NetCast platform uses the LG Browser, which is based on Webkit as an application development environment. The NetCast platform is also compatible with Safari. See the table below for details.

	NetCast 3.0	NetCast 4.0
WebKit Version	534.26+	537.1+
Safari Compatible Version	5.1	5.1.7

Status Definition

This document specifies the technical features of the platform, using the terms defined in the table below.

Status	Definition
O	Fully Supported. All devices must support this feature in order to comply with this specification.
X	Not Supported.

HTML5 Standard

This section describes the elements of HTML5 which are currently supported by the NetCast platform and LG Smart TV Emulator.

- HTML5 Video
- HTML5 Audio
- HTML5 Communication API
- HTML5 Canvas
- HTML5 Inline SVG
- HTML5 Offline Web Applications API
- HTML5 Web Storage API
- HTML5 User Interaction API
- HTML5 Web Workers API

HTML5 Video[1],[2]

The <video> element is used to play videos and movies on the web without plugins. This element has methods, attributes, and events.

Note

- In NetCast 3.0, if the video clip is changed during playback in autoplay mode, it may not play.
- In NetCast 3.0, the <video> element can not play the video file which is in file system (local storage or external storage such as USB memory stick). Use media plugin object to play the video file from file system.

Video Methods

Method	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emu 20
canPlayType()	MIME type without a codecs parameter will not return ‘probably’.	O	O	O	O
load()		O	O	O	O
play()		O	O	O	O
pause()		O	O	O	O

Video Attributes

Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
autoplay			O	O	O	O
controls		Control UI needs to be implemented in the application.	O	O	X	X
width/height	480x360		O	O	O	O
	640x360		O	O	O	O
	720x406		O	O	O	O
	720x486		O	O	O	O
	1280x720		O	O	O	O
	1920x1080		O	O	O	O
	fullscreen		O	O	O	O
loop		no repeat	O	O	O	O
		repeat once	O	O	O	O
		repeat ten times	O	O	X	O
		repeat infinitely	O	O	O	O
preload	auto		O	O	O	O
	metadata		O	O	O	O
	none		O	O	O	O
src	http		O	O	O	O
	file		X	X	X	X
	ftp		X	X	X	X
	href		O	O	O	O
media			X	O	O	O
type	H.264	e.g.) MIME Type: video/mp4	O	O	O	O
	MPEG-4	e.g.) MIME Type: video/mp4	X	X	X	X
	Ogg	e.g.) MIME Type: video/ogg	X	X	X	X
extend type	avi		X	O	X	O
	asf		O	O	O	O
	wmv		X	X	X	X
	mov		O	O	O	O
	mp4		O	O	O	O
	ts		X	O	O	O
	swf		X	X	X	X
	flv		X	X	O	O
poster			O	O	O	O
error			O	O	O	O
error.code			O	X	X	X
currentSrc			O	O	O	O
networkState			O	O	X	X
preload			O	O	O	O
buffered			O	O	O	O

Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
duration			O	O	O	O
currentTime			O	O	X	O
initialTime			O	O	O	O
readyState			O	O	O	O
paused			O	O	O	O
ended			O	O	O	O
defaultPlaybackRate			X	X	X	X
playbackRate			X	X	X	X
played			O	O	O	O
seeking			O	O	O	O
seekable			O	O	O	O
volume			X	X	X	X
muted			X	X	X	X

Video Events

Event	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
loadstart		X	O	O	O
progress		O	O	O	O
suspend		X	O	X	X
abort		X	O	O	O
error		X	X	X	X
emptied		X	O	O	O
stalled		X	O	X	X
play		O	O	O	O
pause		O	O	O	O
loadedmetadata		O	O	O	O
loadeddata		O	O	O	O
waiting		X	O	O	O
playing		O	O	O	O
canplay		O	O	O	O
canplaythrough		O	O	O	O
seeking		X	O	O	O
seeked		O	O	O	O
timeupdate		O	O	O	O
ended		O	O	O	O
ratechange		O	X	O	X
durationchange		O	O	X	O
volumechange		X	X	X	X

HTML5 Audio[3]

The <audio> element is used to play audio on the web without plugins. This element has methods, attributes, and events.

Note

When playing music on overseas websites with a slow network, there may be no reaction on web pages or any linked pages. For NetCast 4.0, music can be played but there is no reaction on linked pages.

Audio Methods

Method	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emu 20
canPlayType()	MIME type without a codecs parameter will not return ‘probably’.	O	O	O	O
load()		O	O	O	O
play()		O	O	O	O
pause()		O	O	O	O

Audio Attributes

Attribute	Value	Remark	NetCast 3.0	NetCas t 4.0	Emulator 2012	Emulator 2013
autoplay			O	O	O	O
controls		Control UI needs to be implemented in the application.	O	O	X	X
loop		no repeat	O	O	O	O
		repeat once	O	O	O	O
		repeat ten times	O	X	X	O
		repeat infinitely	O	X	O	O
preload	auto		O	O	O	O
	metadata		O	O	O	O
	none		O	O	O	O
src	http		O	O	O	O
	file		X	X	X	X
	ftp		X	X	X	X
	href		O	O	O	O
media			X	O	X	O
type	PCM		O	O	X	O
	MP3		O	O	O	O
	AAC		X	X	O	O
	Ogg		X	X	X	X
	WebM		X	X	X	X
error			O	O	O	O
error.code			O	X	X	X
currentSrc			O	O	O	O
networkState			O	O	X	X

Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
preload			O	O	O	O
buffered			O	O	O	O
duration			O	O	O	O
currentTime			O	O	O	O
initialTime			O	O	O	O
readyState			O	O	O	O
paused			O	O	O	O
ended			O	O	O	O
defaultPlaybackRate			X	X	X	X
playbackRate			X	X	X	X
played			O	O	O	O
seeking			X	O	O	O
seekable			O	O	O	O
volume			X	X	X	X
muted			X	X	X	X

Audio Events

Event	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
loadStart		X	O	O	O
progress		O	O	O	O
suspend		X	O	X	O
abort		X	O	O	O
error		X	X	X	X
emptied		X	O	O	O
stalled		X	O	X	O
play		O	O	O	O
pause		O	O	O	O
loadedmetadata		O	O	O	O
loadeddata		X	O	O	O
waiting		X	O	O	O
playing		O	O	O	O
canplay		O	O	O	O
canplaythrough		O	O	O	O
seeking		X	O	O	O
seeked		O	O	O	O
timeupdate		O	O	O	O
ended		O	O	O	O
ratechange		O	X	X	X

Event	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
durationchage		O	O	O	O
volumechange		X	X	X	X

HTML5 Communication API[4],[5]

Communication API is used to share data between multiple programs in an asynchronous manner. This manner is also used for Web Workers and Server-sent Events.
Cross Document Messaging enables secure communication across the web pages.
Channel Messaging uses two ports for many-to-many message communication. Two-way ports are used at each end for communication. So if one port sends messages, then it is delivered to another port, and vice versa.
WebSockets is used to exchange data between web servers and browsers in real-time. Once a WebSocket connection is established, the client and server can send messages to each other at any time.

MessageEvent Attributes

Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
data			O	O	O	O
origin			O	O	O	O
lastEventId			O	X	O	X
source			O	O	O	O
ports			O	O	O	O

Cross-Document Messaging Methods

Method	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
postMessage( )	message, targetOrigin and ports can be used as arguments (ports argument is optional). Enter the URL of the domain as the targetOrigin argument which sent the messages.	O	O	O	O

Channel Messaging Methods

Method	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
messageChannel()		O	O	O	O
postMessage()	message and ports can be used as arguments (ports arguments is optional).	O	O	O	O
start()		O	O	O	O
close()		O	O	O	O

Channel Messaging Attributes

Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
port1			O	O	O	O
port2			O	O	O	O

WebSockets Attributes and Events[6]

Attribute/ Event	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
readyState			O	O	O	O

Attribute/ Event	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
bufferedAmount			O	O	O	O
open			O	O	O	O
message			O	O	O	O
error			X	X	X	X
close			O	O	O	O

Server-sent Events Attributes and Events[7]

Attribute/ Event	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
readyState			O	X	O	X
open			O	X	O	X
message			O	X	O	X
error			O	O	O	O

HTML5 Canvas[8],[9]

The HTML5 Canvas element is used to create graphics using JavaScript. Text, videos or animations can be inserted in the canvas as well as two-dimensional shapes such as circles, squares, lines and images.

Colors, Styles, Transformations, and Pixel Manipulation Methods

Method	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
createPattern()	fillStyle	O	O	X	O
createLinearGradient()	fillStyle	O	O	O	O
createRadialGradient()	fillStyle	O	O	O	O
createPattern()	strokeStyle	O	O	O	O
createLinearGradient()	strokeStyle	O	O	O	O
createRadialGradient()	strokeStyle	O	O	O	O
rotate()		O	O	O	O
scale()		O	O	O	O
translate()		O	O	O	O
setTransform()		O	O	O	O
getImageData()		O	O	O	O
putImageData()		O	O	O	O
arc()	Include all methods and attributes for arc	O	O	O	O
bezierCurveTo()	Include all methods and attributes for bezier curve	O	O	O	O
clip()	Include all methods and attributes for clip	O	O	O	O
fillRect()	Include all methods and attributes for rectangle	O	O	O	O
quadraticCurveTo()	Include all methods and attributes for quadratic curve	O	O	O	O

Colors, Styles, Composing, Shadows, Line Styles, and Shapes Attributes

Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
fillStyle	color		O	O	O	O
	rgba		O	O	O	O
strokeStyle	color		O	O	O	O
	rgba		O	O	O	O
globalAlpha			O	O	O	O
shadowOffset X			O	O	O	O
shadowOffset Y			O	O	O	O
shadowBlur			O	O	O	O
shadowColor			O	O	O	O
lineWidth			O	O	O	O
lineCap	round		O	O	O	O
	square		O	O	O	O
	butt		O	O	O	O
lineJoin	bevel		O	O	O	O
	round		O	O	O	O
	miter		O	O	O	O
miterLimit			O	O	O	O

Interactivity, Animation, and Image[10]

Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
Mouse	Mouse	Include all methods and attributes for Mouse	O	O	O	O
Animation	Animation	Include all methods and attributes for Animation	O	O	O	O
Images/video	drawImage()		O	O	O	O
	Video	Include all methods and attributes for Video	X	X	X	X

Canvas Text Methods

Method	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
createPattern()	fillStyle	O	O	O	O
createLinearGradient()	fillStyle	X	O	X	O
createRadialGradient()	fillStyle	O	O	O	O
createPattern()	strokeStyle	O	O	O	O
createLinearGradient()	strokeStyle	X	O	X	O
createRadialGradient()	strokeStyle	O	O	O	O
fillText()		O	O	O	O
strokeText()		O	O	O	O
rotate()		O	O	O	O
scale()		O	O	O	O

Method	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
translate()		O	O	O	O
setTransform()		O	O	O	O

Canvas Text Attributes

Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
@font-family	font-family	Provides ten fonts: Graublau Sans Web, Fertigo Pro, Tallys, Diavlo, Fontin, Fontin Sans, Pykes Peak Zero, Kaffeesatz, Vollkorn, Tagesschrift	O	O	O	O
font		fillText	O	O	O	O
textAlign		fillStyle	O	O	O	O
textBaseline			O	O	O	O
fillStyle	color		O	O	O	O
	rgba		O	O	O	O
strokeStyle	color		O	O	O	O
	rgba		O	O	O	O
shadowOffset X			O	O	O	O
shadowOffset Y			O	O	O	O
shadowBlur			O	O	O	O
shadowColor			O	O	O	O
animation		Include all methods and attributes for text animation	O	O	O	O

HTML5 Inline SVG[11]

SVG is used to define vector-based graphics in XML format for the web.
Two-dimensional graphics can be drawn using graphic tags like <rect> and <line>. It also helps to change graphics dynamically using JavaScript or to decorate graphics using CSS. SVG is better than Canvas for creating clear, vivid graphics.
WebGL is used to define three dimensional graphics. But NetCast and Emulator do not support this feature now.

Gradients and Patterns

Element	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
linearGradient	Include all methods and attributes for linearGradient element	O	O	O	O
radialGradient	Include all methods and attributes for radialGradient element	O	O	O	O
pattern	Include all methods and attributes for pattern element	O	O	O	O

Filter Effects and Basic Shapes

Element	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
feGaussianBlur	Include all methods and attributes for feGaussianBlur element	X	X	X	X
feBlend	Include all methods and attributes for feBlend element	X	X	X	X

Element	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
feColorMatrix	Include all methods and attributes for feColorMatrix element	X	X	X	X
feComponentTransfer	Include all methods and attributes for feComponentTransfer element	X	X	X	X
feOffset	Include all methods and attributes for feOffset element	X	X	X	X
rect	Include all methods and attributes for rect (rectangle) element	O	O	O	O
Rectangles with opacity	Include all attributes for rect element using fill-opacity and stroke-opacity.	O	O	O	O
Rectangles with rounded corners	Include all attributes for rect element using rx and ry.	O	O	O	O
circle	Include all methods and attributes for circle element	O	O	O	O
ellipse	Include all methods and attributes for ellipse element	O	O	O	O
line	Include all methods and attributes for line element	O	O	O	O
polygon	Include all methods and attributes for polygon element	O	O	O	O
polyline	Include all methods and attributes for polyline element	O	O	O	O
path	Possible to draw spirals	O	O	O	O

SVG Clipping Paths and Animation

Element	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
clipPath		O	O	O	O
mask		X	X	X	X
g	Include all methods and attributes for g element	O	O	O	O
animateMotion	Include all methods and attributes for animateMotion element	O	O	O	O
animateTransform	Include all methods and attributes for animateTransform element	O	O	O	O
rect	Include all methods and attributes for rect element	O	O	O	X
animate	Include all methods and attributes for animate element	O	O	O	X
animateColor	Include all methods and attributes for animateColor element	O	O	O	X

WebGL

Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
		Activates 3D Canvas graphics	X	X	X	X

HTML5 Offline Web Applications API[12]

The Offline Web Application provides cache for working applications, even when they are offline. The application cache saves current web pages and links related to the current page to use the application without an internet connection. Therefore, the application can be used offline. Application cache also loads cached resources so the application runs quickly and the browser just receives changes from the server. This helps to reduce the burden.

ApplicationCache Methods and Attributes

Method/ Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
update()			O	O	O	O

Method/ Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
swapCache ()		The cache is changed in the background, but the screen does not refresh automatically so the reload() method should be used.	O	O	O	X
status		There are six possible status: UNCACHED, IDLE, CHECKING, DOWNLOADING, UPDATEREADY, OBSOLETE	O	O	O	O

ApplicationCache Events

Event	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
checking		O	O	O	O
noupdate		O	O	O	O
downloading		O	O	O	O
progress		O	O	O	O
cached		O	O	O	O
updateready		O	O	O	O
obsolete		O	O	O	O
error		O	O	O	O

HTML5 Web Storage API[13]

Web Storage is used to save a small amount of data into a client's disk storage.

Storage Attributes

Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
sessionStorage			O	O	O	O
localStorage			O	O	O	O

Indexed Database Methods

Method	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
open()		X	X	X	X
createObjectStore()		X	X	X	X
transaction()		X	X	X	X
deleteObjectStore()		X	X	X	X
openCursor(range, direction)		X	X	X	X
put(value, key)		X	X	X	X
add(value, key)		X	X	X	X
get(key)		X	X	X	X
delete(key)		X	X	X	X
continue(key)		X	X	X	X

WebSQL Database Methods

Method	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
openDatabase()		O	O	O	O
transaction()		O	O	O	O
executeSql()		O	O	O	O

HTML5 User Interaction API[14],[15]

User Interaction provides two key features: Drag and Drop and History Management.
Drag and Drop is used to move any object into another place, and Browser History is used for access to a browser's history.

Drag&Drop Attributes

Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
draggable			X	X	X	X
dropzone			X	X	X	X

Drag&Drop Events

Event	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
drag		X	X	X	X
dragstart		X	X	X	X
dragenter		X	X	X	X
dragover		X	X	X	X
dragleave		X	X	X	X
drop		X	X	X	X
dragend		X	X	X	X

History Management Methods

Method	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
go()		O	O	X	O
back()		O	O	O	O
forward()		O	O	X	O
pushState()		O	O	O	O
replaceState()		O	O	O	O

History Management Attributes

Attribute	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
length			O	O	X	O
state			O	O	X	O

HTML5 Web Workers API[16]

Web Workers is used to support multi-tasking in the web browser. It runs JavaScript codes in parallel in the

background to optimize performance.
There are dedicated workers and shared workers in Web Workers. For dedicated workers, there is a consistent one-to-one match between each worker object and each background process. For shared workers, the background processes are shared by a large number of worker objects.

Workers

Method	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
Worker()		O	O	O	O
terminate()		O	O	O	O

Event	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
onmessage			O	O	O	O

Shared workers

Method	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
SharedWorker()		O	O	O	O

Event	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
onmessage			X	O	O	O

CSS3 Standard

This section describes the properties of CSS3 which are currently supported by the NetCast platform and LG Smart TV Emulator.

- Dynamic CSS
- Statics CSS

Dynamic CSS

Transforms[17]

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
matrix	matrix(n,n,n,n,n,n)		O	O	O	O
	matrix3d(n,n,n,n,n,n,n,n,n,n,n,n,n,n,n,n)		X	X	X	X
translate	translate(x,y)		O	O	O	O
	translateZ(z)		X	X	X	X
	translate3d(x,y,z)		X	X	X	X
scale	scale(x,y)		O	O	O	O
	scale3d(x,y,z)		O	O	X	X
rotate	rotate(angle)		O	O	O	O
	rotateX(angle)		O	O	O	O
	rotateY(angle)		O	O	O	O
	rotateZ(angle)		O	O	O	O
	rotate3d(x,y,z		O	O	O	O

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
	,angle)					
skew	skew(x-angle,y-angle)		O	O	O	O
transform-style	preserve-3d		X	X	X	X
	flat		O	O	O	O
transform-origin	x-axis		O	O	O	O
	y-axis		O	O	O	O
	x-axis y-axis z-axis		X	X	X	X
perspective	number		X	X	X	X
perspective-origin	x-axis y-axis		X	X	X	X
backface-visibility	visible|hidden		X	X	X	X

Transitions[18]

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
transition-property	all		O	O	O	O
transition-duration	time (s)		O	O	O	O
transition-timing-function	linear|cubic-bezier(n,n,n,n)		O	O	O	O
	ease|cubic-bezier(n,n,n,n)		O	O	O	O
	ease-in|cubic-bezier(n,n,n,n)		O	O	O	O
	ease-out|cubic-bezier(n.n.n,n)		O	O	O	O
	ease-in-out|cubic-bezier(n,n,n,n)		O	O	O	O
transition-delay	time (s)		O	O	O	O

Animation[19]

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
@keyframes	animationname keyframes-selector {css-styles;}		O	O	O	O
animation-name	keyframename|none		O	O	O	O
animation-duration	time (s)		O	O	O	O
animation-timing-function	linear|cubic-bezier(n,n,n,n)		X	O	X	O
	ease|cubic-		X	O	X	O

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
	bezier(n,n,n,n)					
	ease-in|cubic-bezier(n,n,n,n)		X	O	X	O
	ease-out|cubic-bezier(n,n,n,n)		X	O	X	O
	ease-in-out|cubic-bezier(n,n,n,n)		X	O	X	O
animation-delay	time (s)		O	O	O	O
animation-iteration-count	number|infinite		O	O	O	O
animation-direction	normal		O	O	O	O
	alternate		O	O	O	O
animation-play-state	running|paused		O	O	O	O

Static CSS

Background and Borders[20]

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
background-size	length (px)		O	O	O	O
	percentage (%)		O	O	O	O
	cover		O	O	O	O
	contain		O	O	O	O
background-clip	padding-box		O	O	O	O
	border-box		O	O	O	O
	content-box		O	O	O	O
background-origin	padding-box		O	O	O	O
	border-box		O	O	O	O
	content-box		O	O	O	O
multiple-backgrounds			O	O	O	O
box-shadow	h-shadow		O	O	O	O
	v-shadow		O	O	O	O
	blur		O	O	O	O
	spread		O	O	O	O
	color		O	O	O	O
	inset		O	O	O	O
border-bottom-left-radius			O	O	O	O

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
border-bottom-right-radius			O	O	O	O
border-top-left-radius			O	O	O	O
border-top-right-radius			O	O	O	O
border-radius			O	O	O	O
border-image			O	O	O	O

Box[21]

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
overflow-x	visible		O	O	O	O
	hidden		O	O	O	O
	scroll		O	O	O	O
	auto		O	O	O	O
overflow-y	visible		O	O	O	O
	hidden		O	O	O	O
	scroll		O	O	O	O
	auto		O	O	O	O

Color[22]

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
HSL|RGB			O	O	O	O
HSLA|RGB			O	O	O	O
RGB & opacity|RGB			O	O	O	O
RGB & opacity|RGBA			O	O	O	O
RGBA|RGB			O	O	O	O

Columns and Multi Column Layout[23]

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
column-width			O	O	O	O
column-count			O	O	O	O
column-gap			O	O	O	O
column-rule			O	O	O	O
column-rule-color	color		O	O	O	O
column-rule-style	none		O	O	O	O
	hidden		O	O	O	O

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
	dotted		O	O	O	O
	dashed		O	O	O	O
	solid		O	O	O	O
	double		O	O	O	O
	groove		O	O	O	O
	ridge		O	O	O	O
	inset		O	O	O	O
	outset		O	O	O	O
column-rule-width	thin		O	O	O	O
	medium		O	O	O	O
	thick		O	O	O	O
	length		O	O	O	O
column-span	1		O	O	O	O
	all		O	O	O	O

Flexible Box[24],[25]

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
box-align	start		O	O	O	O
	end		O	O	O	O
	center		O	O	O	O
	stretch		O	O	O	O
box-direction	normal		O	O	O	O
	reverse		O	O	O	O
box-flex	value		O	O	O	O
box-ordinal-group	integer		O	O	O	O
box-orient	horizontal		O	O	O	O
	vertical		O	O	O	O
	inline-axis		O	O	O	O
	block-axis		O	O	O	O
box-pack	start		O	O	O	O
	end		O	O	O	O
	center		O	O	O	O
	justify		O	O	O	O

Text[26],[27],[28]

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
text-shadow	basic		O	O	O	O
	blur effect		O	O	O	O

Property	Value	Remark	NetCast 3.0	NetCast 4.0	Emulator 2012	Emulator 2013
	shadow on text		O	O	O	O
	neon glow		O	O	O	O
	multiple shadow		O	O	O	O
	quick and dirty letterpress		O	O	O	O
	hard shadow		O	O	O	O
	double shadow		O	O	O	O
	down and distant		O	O	O	O
	close and heavy		O	O	O	O
	3D text		O	O	O	O
	true inset text		O	O	O	O
	Glowing		O	O	O	O
	Multiple Light Sources		O	O	O	O
	Soft Emboss		O	O	O	O
text-overflow	clip		O	O	O	O
	ellipsis		O	O	O	O
word-break	normal		O	O	O	O
	break-all		O	O	O	O
word-wrap	normal		O	O	O	O
	break-word		O	O	O	O

Annex A References

The following referenced websites are required for the application of this specification.

Index	Reference	Title
[1]	http://www.w3.org/TR/2012/WD-html5-20120329/media-elements.html#media-element-attributes	W3C Working Draft 29 March 2012 – Media
[2]	http://www.w3.org/TR/2012/WD-html5-20120329/the-video-element.html#the-video-element	W3C Working Draft 29 March 2012 – Video
[3]	http://www.w3.org/TR/2012/WD-html5-20120329/the-audio-element.html#the-audio-element	W3C Working Draft 29 March 2012 – Audio
[4]	http://www.whatwg.org/specs/web-apps/current-work/multipage/comms.html#comms	WHATWG Living Standard Last Update 17 January 2013 - Communication
[5]	http://www.w3.org/TR/2012/CR-webmessaging-20120501/	W3C Candidate Recommendation 01 May 2012 – Web Messaging
[6]	http://www.w3.org/TR/2012/WD-websockets-20120809/	W3C Working Draft 09 August 2012 – WebSocket API
[7]	http://www.w3.org/TR/2012/WD-eventsource-20120426/	W3C Working Draft 26 April 2012 – Server-Sent Events
[8]	http://www.w3.org/TR/2012/WD-html5-20120329/the-canvas-element.html	W3C Working Draft 29 March 2012 - Canvas
[9]	http://www.whatwg.org/specs/web-apps/current-work/multipage/the-canvas-element.html	WHATWG Living Standard Last Update 17 January 2013 - Canvas

Index	Reference	Title
[10]	http://www.w3.org/TR/SVG11/interact.html	W3C Recommendation 16 August 2011 - Scalable Vector Graphics 1.1 Interactivity
[11]	http://www.w3.org/TR/SVG11/ http://www.w3.org/TR/SVG11/struct.html	W3C Recommendation 16 August 2011 – Scalable Vector Graphics 1.1 Structure
[12]	http://www.whatwg.org/specs/web-apps/current-work/multipage/offline.html	WHATWG Living Standard Last Update 17 January 2013 – Offline Web Applications
[13]	http://www.whatwg.org/specs/web-apps/current-work/multipage/webstorage.html	WHATWG Living Standard Last Update 17 January 2013 – Web Storage
[14]	http://www.whatwg.org/specs/web-apps/current-work/multipage/dnd.html	WHATWG Living Standard Last Update 17 January 2013 – Drag and Drop
[15]	http://www.whatwg.org/specs/web-apps/current-work/multipage/history.html#the-history-interface	WHATWG Living Standard Last Update 17 January 2013 – History Management
[16]	http://www.whatwg.org/specs/web-apps/current-work/multipage/workers.html#workers	WHATWG Living Standard Last Update 17 January 2013 – Web Workers
[17]	http://www.w3.org/TR/2012/WD-css3-transforms-20120403/	W3C Working Draft 3 April 2012 – CSS Transforms
[18]	http://www.w3.org/TR/2012/WD-css3-transitions-20120403/	W3C Working Draft 3 April 2012 – CSS Transitions
[19]	http://www.w3.org/TR/2012/WD-css3-animations-20120403/	W3C Working Draft 3 April 2012 – CSS Animations
[20]	http://www.w3.org/TR/2012/CR-css3-background-20120417/	W3C Candidate Recommendation 17 April 2012 – CSS Backgrounds and Borders Module Level 3
[21]	http://dev.w3.org/csswg/css3-box/	W3C Editor’s Draft 1 November 2012 – CSS Basic Box Model
[22]	http://www.w3.org/TR/2011/REC-css3-color-20110607/	W3C Recommendation 07 June 20 - CSS Color Module Level 3
[23]	http://www.w3.org/TR/2011/CR-css3-multicol-20110412/	W3C Candidate Recommendation 12 April 2011 – CSS Multi-column Layout Module
[24]	http://www.html5rocks.com/en/tutorials/flexbox/quick/	HTML5 Rocks Tutorial 5 October 2010 – Quick hits with the flexible box model
[25]	http://www.w3schools.com/cssref/#flexbox	w3schools CSS Flexible Box Properties
[26]	http://www.w3.org/TR/2013/WD-css-text-decor-3-20130103/#text-shadow-property	W3C Working Draft 13 January 2013 – CSS Text Decoration Module Level 3
[27]	http://www.w3.org/TR/2012/WD-css3-ui-20120117/#text-overflow	W3C Working Draft 17 January 2012 – Basic User Interface Module Level 3 (CSS3 UI)
[28]	http://www.w3.org/TR/css3-text/#word-break http://www.w3.org/TR/css3-text/#word-wrap	W3C Working Draft 13 November 2012 – CSS Text Module Level 3

Sample Tutorials: LG WebAPI Tutorials

LG WebAPI samples are provided to help developers create web application using LG Web APIs easily. Several sample codes related to various topics are provided as follows.

- Activating or Deactivating Browser Page Loading Icon
- Activating or Deactivating Magic Remote
- Controlling Voice Recognition
- Displaying Subtitle in Media Player
- Executing Windows Media Player 1
- Executing Windows Media Player 2
- Executing Windows Meida Player 3
- Generating Outofmemory Event
- Getting and Displaying Device Information
- Handling Key Inputs Using JavaScript Events
- Handling Key Repeat Using JavaScript Events
- Implementing Functionality of Back and Exit Keys
- Implementing Functionality of Multi Audio
- Playing Media with ASX File
- Playing Video in Full Screen Mode

You can download “Web_LG API Sample Applications” sample codes [DISCOVER > Legacy Platform (NetCast) > Tools & Samples] menu in LG Developer (**http://webostv.developer.lge.com**) website.

Note

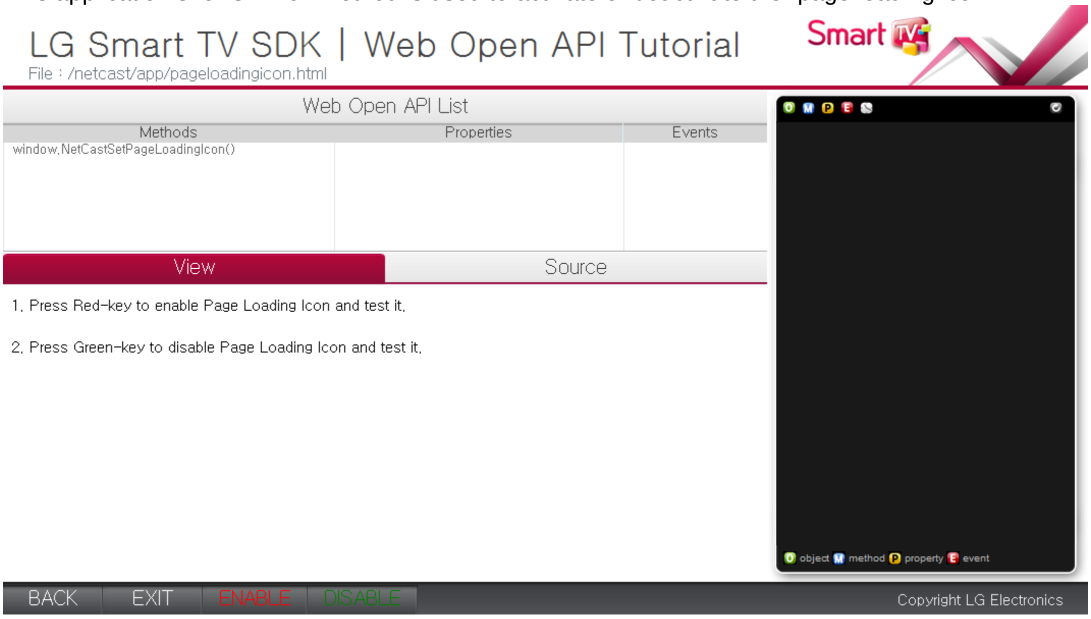
Refer to **Developing > API** section in this Library for detailed information on how to use Web APIs.

Activating or Deactivating Browser Page Loading Icon

This section describes how to activate and deactivate the browser page loading icon in web application.

- Initializing the Page
- Inputting Keys
- Handling Keys
- Handling Events
- Counting Time
- Source Code of pageloadingicon.html

It is recommended that application authors provide a “page loading icon” so that users are provided with an indication of the latency of data downloading from a server.
This application is designed to show how to activate or deactivate the “page loading icon” using the Web Open API of LG Smart TV.
This application shows which method is used to activate or deactivate the “page loading icon”.



LG Smart TV SDK | Web Open API Tutorial

File : /netcast/app/pageloadingicon.html

Web Open API List

Methods	Properties	Events
window.NetCastSetPageLoadingIcon()		

1. Press Red-key to enable Page Loading Icon and test it.

2. Press Green-key to disable Page Loading Icon and test it.

Needed APIs
This application uses following Web Open API:

API Class	Function Name	Description
Method	window.NetCastSetPageLoadingIcon()	This API can be used by an LG Smart TV application author to activate or deactivate the browser's own page loading animation.
Property	N/A	N/A
Event	N/A	N/A

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Initializing the Page

Uspse the **initPage** function to set the basic functions of the application.

04: Records the last visited page when running the application.
07: Initializes the page.
08: Gets the source code of the page using the XMLHttpRequest object.
09: Sets the page ID.
10: Initializes the Log function.
13-16: Registers an event handler which will executed when the corresponding button is pressed.

```
01 : function initPage()
02 : {
03 :   //save page as last visited page
04 :   setLastVisitPage();
05 :
06 :   //common initialize function
07 :   commonInitialize();
08 :   requestSourceCode();
09 :   setPageID("Page Loading Icon");
10 :   jsLog.initLG();
11 :
12 :   //add onclick event handler
13 :   addEventHandler(document.getElementById("btn_back"),"click", onClickHandler);
14 :   addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
15 :   addEventHandler(document.getElementById("btn_green"),"click",onClickHandler);
16 :   addEventHandler(document.getElementById("btn_exit"),"click", onClickHandler);
17 :
18 : }
```

Inputting Keys

The **onUserInput** function is called by the onClickHandler function; it receives a key value as the userInput parameter from onClickHandler and creates the corresponding function for each key value to operate the key.

06: When the Back key is pressed, this code is executed.
07: When the Red key is pressed, this code is executed.
08: When the Green key is pressed, this code is executed.

```
01 : //onUserInput function should be implemented
02 : function onUserInput(userInput)
03 : {
04 :   switch(userInput)
05 :   {
06 :      case VK_BACK : window.location.replace("../menu_netcast.html"); break;
07 :      case VK_RED : case 82 : processRedKey(); break;
08 :      case VK_GREEN : case 71 : processGreenKey(); break;
09 :   }
10 : }
```

Handling Keys

The following functions are executed when each key of onUserInput() is pressed.

processRedKey

This function is called when the Red (Enable) key is pressed.
05: Enables the page loading icon.
08: Shows on the screen that the page loading icon is enabled.
09: Maintains the text printed from code 10 through an alert until you press the [OK] button.
10: Reloads the page. When the page is reloaded, you can see the "Loading Icon" appearing on the screen.

processGreenKey

This function is called when the Green (Disable) key is pressed.
18: Calls the setCountDown() function.
19: Disables the page loading icon.
21: Shows on the screen that the page loading icon is disabled.

```
01 : function processRedKey()
02 : {
03 :   if(isThisLGEBrowser())
04 :   {
05 :      window.NetCastSetPageLoadingIcon('enabled');
06 :      jsLog.lgmethod('window.NetCastSetPageLoadingIcon()');
07 :
08 :      setInnerTextById("description", "Loading Icon is enabled!");
09 :      alert("Enable!");
10 :      window.location.reload();
11 :   }
12 : }
13 :
14 : function processGreenKey()
15 : {
16 :   if(isThisLGEBrowser())
17 :   {
18 :      setCountDown();
19 :      window.NetCastSetPageLoadingIcon('disabled');
20 :      jsLog.lgmethod('window.NetCastSetPageLoadingIcon()');
21 :      setInnerTextById("description", "Loading Icon is disabled!");
22 :   }
23 : }
```

Handling Events

The **setCountDown** function is called when the Green key is pressed.

07: Sets timerCount to 5 so it counts the remaining time of 5 seconds until the loading icon is disabled.
08: Calls the showLeftTime function every 1 second to display the remaining time until the loading icon is disabled. Set the timer.
09: Shows the text on the screen.

```
01 : var timer;
02 : var timeCount = 5;
03 :
04 : //"GreenKey" click handler
05 : function setCountDown()
06 : {
07 :   timeCount = 5;
08 :   timer = setInterval(showLeftTime, 1000);
09 :   setInnerTextById("description", "Loading Icon will be off after 5 second");
10 : }
```

Counting Time

The following sample code counts and displays the remaining time until the “loading icon” is set to “disable”.

The **showLeftTime** function counts the remaining time until the loading icon is disabled.
04: Decreases timeCount by 1.
05-06: If timeCount is greater than 0, If timeCount is greater than 0, the message "This page will be reloaded after + “timeCount” + second" will be displayed on the id = "description" area.
09: If timeCount is 0, the clearInterval is called the setInterval function stops the specified timer.
10: Reloads the current page. When the page is reloaded, you can see the "Loading Icon" not appearing on the screen.

```
01 : //show left time
02 : function showLeftTime()
03 : {
04 :    timeCount--;
05 :    if(timeCount > 0)
06 :        setInnerTextById("description", "Loading Icon will be off after " + 
timeCount + " second");
07 :    else
08 :    {
09 :        clearInterval(timer);
10 :        window.location.reload();
11 :    }
12 : }
```

Source Code of pageloadingicon.html

Source code of pageloadingicon.html is as follows:

```
<!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=UTF-8">
<title>App Template of API Unit Sample App</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/common.js"></script>
<script language="javascript" src="../../js/menu.js"></script>

<script type="text/javascript" src="../../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../../blackbirdjs/blackbird.css" />

<script>

    //initialize page
    function initPage()
    {
        //save page as last visited page
        setLastVisitPage();

        //common initialize function
        commonInitialize();
        requestSourceCode();
        setPageID("Page Loading Icon");
        jsLog.initLG();

        //add onclick event handler
        addEventHandler(document.getElementById("btn_back"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_red"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_green"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_exit"), "click", onClickHandler);
    }

    //onUserInput function should be implemented
    function onUserInput(userInput)
    {
        switch(userInput)
        {
            case VK_BACK : window.location.replace("../menu_netcast.html"); break;
            case VK_RED : case 82 :processRedKey();   break;
            case VK_GREEN : case 71 : processGreenKey(); break;
        }
    }

    function processRedKey()
    {
        if(isThisLGEBrowser())
        {
            window.NetCastSetPageLoadingIcon('enabled');
            jsLog.lgmethod('window.NetCastSetPageLoadingIcon()');

            setInnerTextById("description", "Loading Icon is enabled!");
            alert("Enable!");
            window.location.reload();
        }

    }

    function processGreenKey()
    {
        if(isThisLGEBrowser())
        {
            setCountDown();
            window.NetCastSetPageLoadingIcon('disabled');
            jsLog.lgmethod('window.NetCastSetPageLoadingIcon()');
            setInnerTextById("description", "Loading Icon is disabled!");
        }
    }

    var timer;
    var timeCount = 5;

    //"GreenKey" click handler
    function setCountDown()
    {
        timeCount = 5;
        timer = setInterval(showLeftTime, 1000);
        setInnerTextById("description", "Loading Icon will be off after 5 second");
    }

    //show left time
    function showLeftTime()
    {
        timeCount--;
        if(timeCount > 0)
            setInnerTextById("description", "This page will be reloaded after " +
timeCount + " second");
        else
        {
            clearInterval(timer);
```

```
          window.location.reload();
       }
   }

</script>
</head>

<body ondragstart='return false' onselectstart='return false'
onload="javascript:initPage();">

<!-- title -->
<div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>

<!-- navigation -->
<div class='SuiteNavigation'>
   <div style="float:left;">File : /netcast/app/pageloadingicon.html</div>
</div>

<div class='SuiteTitleLine'> </div>

<!-- test contents -->
<div id='content_body' class='ContentArea'>
   <div class='ApiListTitleArea'>Web Open API List</div>
   <div class='ApiListArea'>
       <div class='MethodTitleArea'>
          Methods
          <div class='MethodListArea'>
             window.NetCastSetPageLoadingIcon()<br>
          </div>
       </div>
       <div class='PropertyTitleArea'>
          Properties
          <div class='PropertyListArea'>

          </div>
       </div>
       <div class='EventTitleArea'>
         Events
         <div class='EventListArea'>

         </div>
       </div>
   </div>

   <div class='ViewTitleArea'>
       <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
       <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
   </div>

   <div id='view'>
       <div class='ViewArea'>
          <table border="0" cellpadding="0" cellspacing="0">
             <tr height="50px">
                <td width=500px align=left>1. Press Red-key to enable Page Loading
Icon and test it.</td>
                <td width=200px align="left"><div id="eachTestGuide"></div></td>
                <td width=100px align="left"></td>
             </tr>
```

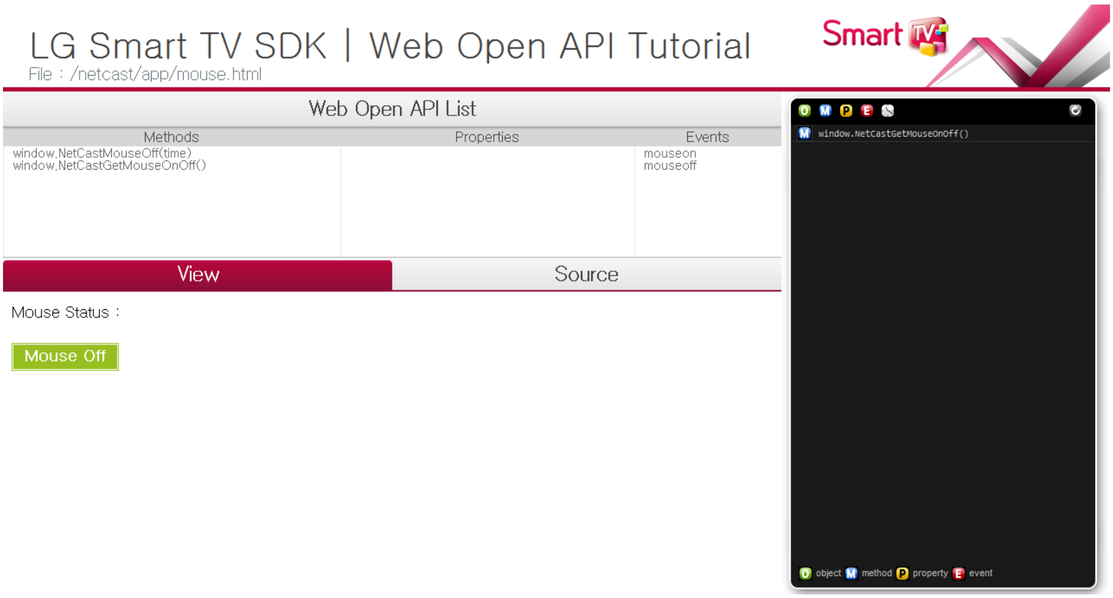
```
            <tr height="50px">
               <td width=500px align=left>2. Press Green-key to disable Page Loading Icon and test it.</td>
               <td width=200px align="left"><div id="eachTestGuide"></div></td>
               <td width=100px align="left"></td>
            </tr>
         </table>
         <table border="0" cellpadding="0" cellspacing="0">
            <tr height="50px">
               <td align=center><div id="description"></div></td>
            </tr>
         </table>
      </div>
   </div>

   <div style="visibility: hidden" id='codeview'>
      <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
   </div>
</div>

<!-- button and copyright -->
<div class='SuiteButtonArea'>

   <!-- button -->
   <div id='btn_back' class='buttonDescription'>BACK</div>

   <!-- exit button -->
   <div id='btn_exit' class='buttonDescription'>EXIT</div>

   <!-- red key description -->
   <div id='btn_red' class='buttonDescription redColor'>ENABLE</div>

   <!-- green key description -->
   <div id='btn_green' class='buttonDescription greenColor'>DISABLE</div>

   <!-- copyright -->
   <div class='copyright'>Copyright LG Electronics</div>
</div>

</body>
</html>
```

Activating or Deactivating Magic Remote

This section describes how to activate and deactivate the magic remote in web applcation.

- Initializing the Page
- Inputting Keys
- Handling Events
- Counting Time
- Adding Mouse Event Handler
- Source Code of mouse.html

This application is designed to show how to deactivate the Magic Remote using the Web Open API of LG Smart TV. This application shows which method is used to deactivate the Magic Remote and how Magic Remote handles activation/deactivation events.



LG Smart TV SDK | Web Open API Tutorial

File : /netcast/app/mouse.html

Web Open API List

Methods	Properties	Events

window.NetCastMouseOff(time)
window.NetCastGetMouseOnOff()
		mouseon
		mouseoff

View | Source

Mouse Status :

Mouse Off

M window.NetCastGetMouseOnOff()

O object M method P property E event

Needed APIs
This application uses following Web Open API:

API Class	Function Name	Description
Method	Window.NetCastMouseOff(time)	This API can be used by an LG Smart TV application author to deactivate the Magic Remote and its pointer.
	Window.NetCastGetMouseOnOff()	This API can be used by an LG Smart TV application author to get the on or off status of the Magic Remote. Its return value is “on” or “off”.
Property	N/A	N/A
Event	mouseon	This event is generated when the Magic Remote is activated.
	mouseoff	This event is generated when the Magic Remote is deactivated.

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

04: Records the last visited page when running the application.
07: Initializes the page.
08 Gets the source code of the page using the XMLHttpRequest object.
09: Sets the page ID.
10: Initializes the Log function.
13-17: Adds an event handler which will executed when the corresponding button is pressed.

19: Calls the updateMouseStatus() function.

```
01 : function initPage()
02 : {
03 :   //save page as last visited page
04 :   setLastVisitPage();
05 :
06 :   //common initialize function
07 :   commonInitialize();
08 :   requestSourceCode();
09 :   setPageID("Mouse");
10 :   jsLog.initLG();
11 :
12 :   //add onclick event handler
13 :   addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
14 :   addEventHandler(document.getElementById("btn_exit"),"click",onClickHandler);
15 :
16 :   //add event handler for test
17 :   addEventHandler(document.getElementById("start"),"click", onClickMouseOff);
18 :
19 :   updateMouseStatus();
20 : }
```

Inputting Keys

The **onUserInput** function is called by the onClickHandler function; it receives a key value as the userInput parameter from onClickHandler and creates the corresponding function for each key value to operate the key.

06: When the Back key is pressed, this code is executed.

```
01 : //onUserInput function should be implemented
02 : function onUserInput(userInput)
03 : {
04 :   switch(userInput)
05 :   {
06 :      case VK_BACK : window.location.replace("../menu_netcast.html"); break;
07 :   }
08 : }
```

Handling Events

The following functions are for event handlers added by the initPage function.

mouseon_handler

The event handler called when the mouse is turned on.
04: updateMouseStatus is called. It defines functions which will be executed when the corresponding button is pressed.

mouseoff_handler

The event handler called when the mouse is turned off.
10: updateMouseStatus is called. It defines functions which will be executed when the corresponding button is pressed.

updateMouseStatus

Defines the action that will be taken depending on the mouse status.
16: Stores the mouse on or off status in the status variable.
17: Displays the value stored in the status variable.
19-23: If the mouse status is on and device is LG Browser4.0, Mouse off button is enabled and the message is printed.
24-27: If the mouse status is on and device is LG Browser5.0, the message is printed.
28-32: If the mouse status is off, Mouse off button is disabled and the message is printed.

checkNetCastVersion

Checks the LG Browser version and decides registration of mouseoff event.
35: Stores version information in the nBrowserVersion.
38-42: If the device is LG Browser4.0, mouseoff event is registered and the message is printed.
43-47: If the device is LG Browser5.0, Mouse off button is disabled and the message is printed.

onClickMouseOff

The event handler called when the Mouse off button is pressed.
53: Calls the window.NetCastMouseOff(time) API to turn off the mouse. To have the mouse turned off in 5 seconds, the parameter of 5 is entered.
54: Sets timeCount to 5 so it counts the remaining time of 5 seconds until the mouse is turned off.
55: Calls the showLeftTime function every 1 second to display the remaining time until the mouse is turned off. Set the timer.

```
01 : function mouseon_handler()
02 : {
03 :   jsLog.lgevent('mouseon');
04 :   updateMouseStatus();
05 : }
06 :
07 : function mouseoff_handler()
08 : {
09 :   jsLog.lgevent('mouseoff');
10 :   updateMouseStatus();
11 : }
12 :
13 : function updateMouseStatus()
14 : {
15 :   jsLog.lgmethod('window.NetCastGetMouseOnOff()');
16 :   status = window.NetCastGetMouseOnOff();
17 :   setInnerTextById("mouse_status", status);
18 :
19 :   if(status == 'on' && nBrowserVersion == 4)
20 :   {
21 :      document.getElementById("mouse_off").style.visibility = "visible";
22 :      setInnerTextById("description", "");
23 :   }
24 :   else if(status == 'on' && nBrowserVersion >= 5)
25 :   {
26 :      setInnerTextById("description", "");
27 :   }
28 :   else
29 :   {
30 :      document.getElementById("mouse_off").style.visibility = "hidden";
31 :      setInnerTextById("description", "Press the OK button of mouse to use
mouse");
32 :   }
33 : }
34 :
35 : var nBrowserVersion = getBrowserVersion();
36 : function checkNetCastVersion()
37 : {
38 :   if(nBrowserVersion == 4) // NetCast 2.0
39 :   {
40 :      addEventHandler(document.getElementById("mouse_off"), "click",
onClickMouseOff);
41 :      setInnerTextById("APIdescription", "In the LG DTV Emulator 2011, mouseoff
event is not supported.");
42 :   }
43 :   else if(nBrowserVersion >= 5) // NetCast 3.0
```

```
  {
     document.getElementById('mouse_off').style.visibility = "hidden";
     setInnerTextById("APIdescription", "In NetCast 3.0, NetCastMouseOff(time) API is not supported and the mouse gets deactivated when the halt of the mouse movement continues for 3 seconds only on TV set.");
  }
}
//"Mouse Off" click handler
function onClickMouseOff()
{
  jsLog.lgmethod('window.NetCastMouseOff(time)');
  window.NetCastMouseOff(5);
  timeCount = 5;
  timer = setInterval(showLeftTime, 1000);
  setInnerTextById("description", "Mouse will be off after 5 second");
}
```

Counting Time

The **showLeftTime** function counts the remaining time until the mouse is turned off.
The following sample code counts and displays the remaining time until the mouse is set to off by the onClickMouseOff function.

07: Decreases timeCount by 1.
08-09: If timeCount is greater than 0, the message "Mouse will be off after + “timeCount” + second" will be displayed on the id = "description" area.
10-11: If timeCount is 0, the clearInterval is called the setInterval function stops the specified timer.

```
01 : //show left time
02 : var timer;
03 : var timeCount = 5;
04 :
05 : function showLeftTime()
06 : {
07 :   timeCount--;
08 :   if(timeCount > 0)
09 :      setInnerTextById("description", "Mouse will be off after " +  timeCount + " second");
10 :   else
11 :      clearInterval(timer);
12 : }
```

Adding Mouse Event Handler

The following sample code adds the event handlers of mouseon and mouseoff.

```
  window.onmouseon = mouseon_handler;
  window.onmouseoff = mouseoff_handler;
```

Source Code of mouse.html

Source code of mouse.html is as follows:

```
<!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=UTF-8">
<title>App Template of API Unit Sample App</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/common.js"></script>
```

```
<script language="javascript" src="../../js/menu.js"></script>

<script type="text/javascript" src="../../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../../blackbirdjs/blackbird.css" />

<script>

    //initialize page
    function initPage()
    {
        //save page as last visited page
        setLastVisitPage();

        //common initialize function
        commonInitialize();
        requestSourceCode();
        setPageID("Mouse");
        jsLog.initLG();

        //add onclick event handler
        addEventHandler(document.getElementById("btn_back"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_exit"), "click", onClickHandler);

        //add event handler for test
        checkNetCastVersion();
        // addEventHandler(document.getElementById("mouse_off"), "click",
onClickMouseOff);

        updateMouseStatus();
    }

    //onUserInput function should be implemented
    function onUserInput(userInput)
    {
        switch(userInput)
        {
            case VK_BACK : window.location.replace("../menu_netcast.html"); break;
        }
    }

    function mouseon_handler()
    {
        jsLog.lgevent('mouseon');
        updateMouseStatus();
    }

    function mouseoff_handler()
    {
        jsLog.lgevent('mouseoff');
        updateMouseStatus();
    }

    function updateMouseStatus()
    {
        jsLog.lgmethod('window.NetCastGetMouseOnOff()');
        status = window.NetCastGetMouseOnOff();
        setInnerTextById("mouse_status", status);

        if(status == 'on' && nBrowserVersion == 4)
        {
            document.getElementById("mouse_off").style.visibility = "visible";
            setInnerTextById("description", "");
        }
        else if(status == 'on' && nBrowserVersion >= 5)
        {
            setInnerTextById("description", "");
        }
        else
        {
            document.getElementById("mouse_off").style.visibility = "hidden";
            setInnerTextById("description", "Press the OK button of mouse to use
mouse");
        }
    }

    var nBrowserVersion = getBrowserVersion();
    function checkNetCastVersion()
    {
        if(nBrowserVersion == 4) // NetCast 2.0
        {
            addEventHandler(document.getElementById("mouse_off"), "click",
onClickMouseOff);
            setInnerTextById("APIdescription", "In the LG DTV Emulator 2011, mouseoff
event is not supported.");
        }
        else if(nBrowserVersion >= 5) // NetCast 3.0
        {
            document.getElementById('mouse_off').style.visibility = "hidden";
            setInnerTextById("APIdescription", "In NetCast 3.0, NetCastMouseOff(time)
API is not supported and the mouse gets deactivated when the halt of the mouse
movement continues for 3 seconds only on TV set.");
        }
    }

    //"Mouse Off" click handler
    function onClickMouseOff()
    {
        jsLog.lgmethod('window.NetCastMouseOff(time)');
        window.NetCastMouseOff(5);
        timeCount = 5;
        timer = setInterval(showLeftTime, 1000);
        setInnerTextById("description", "Mouse will be off after 5 second");
    }

    //show left time
    var timer;
    var timeCount = 5;
    function showLeftTime()
    {
        timeCount--;
        if(timeCount > 0)
            setInnerTextById("description", "Mouse will be off after " + timeCount + "
second");
        else
            clearInterval(timer);
    }

    window.onmouseoff = mouseoff_handler;
    window.onmouseon = mouseon_handler;
```

```
</script>
</head>

<body ondragstart='return false' onselectstart='return false'
onload="javascript:initPage();">

<!-- title -->
<div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>

<!-- navigation -->
<div class='SuiteNavigation'>
   <div style="float:left;">File : netcast/app/mouse.html</div>
</div>

<div class='SuiteTitleLine'> </div>

<!-- test contents -->
<div id='content_body' class='ContentArea'>
   <div class='ApiListTitleArea'>Web Open API List</div>
   <div class='ApiListArea'>
      <div class='MethodTitleArea'>
         Methods
         <div class='MethodListArea'>
            window.NetCastMouseOff(time)<br>
            window.NetCastGetMouseOnOff()
         </div>
      </div>
      <div class='PropertyTitleArea'>
         Properties
         <div class='PropertyListArea'>

         </div>
      </div>
      <div class='EventTitleArea'>
         Events
         <div class='EventListArea'>
            mouseon<br>
            mouseoff<br>
         </div>
      </div>
   </div>

   <div class='ViewTitleArea'>
      <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
      <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
   </div>
   <div id='view'>
   <div class='ViewArea'>
      <table border="0" cellpadding="0" cellspacing="0">
         <tr height="50px">
            <td width=200px align=left><div>Mouse Status :</div></td>
            <td width=200px align="left"><div id="mouse_status"></div></td>
            <td width=100px align="left"></td>
         </tr>
         <tr height="50px">
            <td width=200px align=left><div id="mouse_off"
class="executeButton">Mouse Off</div></td>
            <td width=200px align="left"><div></div></td>
```

```
                <td width=100px align="left"></td>
            </tr>
        </table>
        <table border="0" cellpadding="0" cellspacing="0">
            <tr height="50px">
                <td align=left><div id="description" class="blueColor"></div></td>
            </tr>
            <tr height="50px">
                <td align=left><div id="APIdescription" ></div></td>
           </tr>
        </table>
    </div>
    </div>
    <div style="visibility: hidden" id='codeview'>
        <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
    </div>

</div>

<!-- button and copyright -->
<div class='SuiteButtonArea'>

    <!-- back key description -->
    <div id='btn_back' class='buttonDescription'>BACK</div>

    <!-- exit key description -->
    <div id='btn_exit' class='buttonDescription'>EXIT</div>

    <!-- copyright -->
    <div class='copyright'>Copyright LG Electronics</div>
</div>

</body>
</html>
```

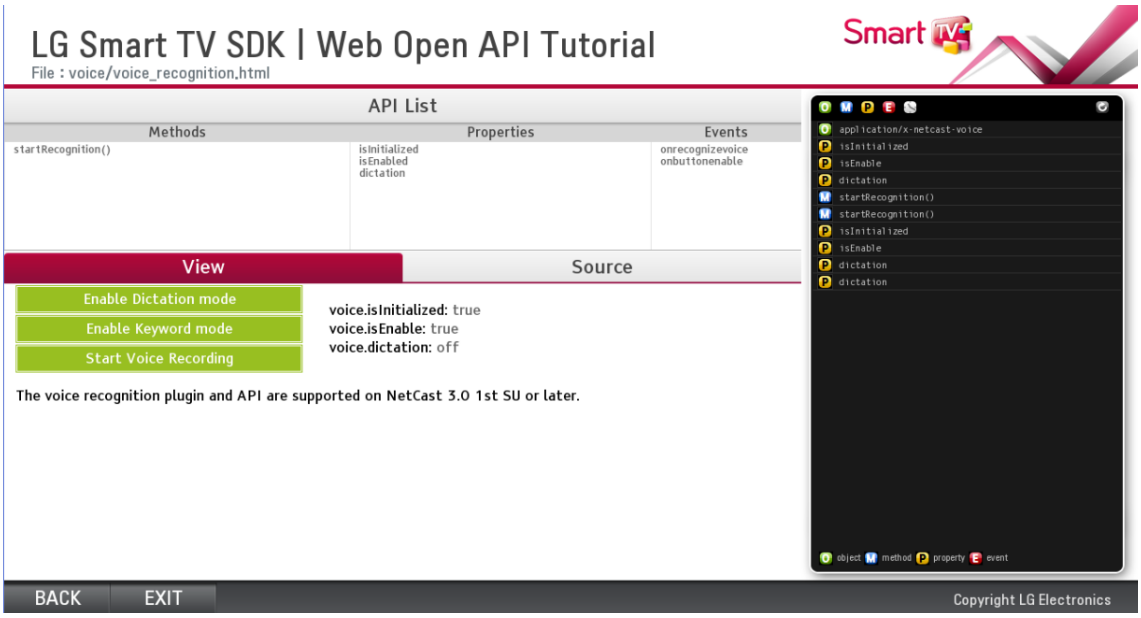
Controlling Voice Recognition

This section describes how to create the web application using LG Web Open APIs.

- Creating Web Applications

The sample application is designed to show which method, properties, and events of LG Web Open API are used to use the voice recognition function of the Magic Remote on web applications in LG Smart TV.

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. Now, it supports Dutch, English, French, German, Italian, Korean, Norwegian, Russian, Spanish, Swedish and UK English. Also, occurred and used events are shown, when status of voice recognition object is changed.



LG Smart TV SDK | Web Open API Tutorial

File : voice/voice_recognition.html

API List

Methods	Properties	Events
startRecognition()	isInitialized isEnabled dictation	onrecognizevoice onbuttonenable

voice.isInitialized: true
voice.isEnable: true
voice.dictation: off

The voice recognition plugin and API are supported on NetCast 3.0 1st SU or later.

```
O application/x-netcast-voice
P isInitialized
P isEnable
P dictation
M startRecognition()
M startRecognition()
P isInitialized
P isEnable
P dictation
P dictation
```

Needed APIs
This application uses following Web Open API:

API Class	Function Name	Description
Method	startRecognition	Call native UI of the voice recognition function and receive the result as an event.
Property	isInitialized	Determine whether the voice recognition function is initialized and returns the value as Boolean.
	isEnable	Determine whether the Magic Remote is paired (including its type) and whether the voice recognition function is enabled, and returns the value as Boolean.
	dictation	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.
Event	onrecognizevoice	Receive the voice recognition result from the TV.
	onbuttonenable	Enable or disable the voice recognition button

For more information on these functions, refer to **Developing > API > Voice Recognition API** section in this Library.

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Creating Web Applications

This section describes how to create web applications that controls the voice recognition of the Magic Remote using LG Web Open APIs.

- Initializing the Page
- Check the NetCast Version
- Initializing Voice Recognition
- Handling Button Click Events
- Handling Voice Events
- Displaying Text
- Inputting Keys
- Setting Voice Recognition Object
- Source Code of voice_recognition.html

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

04: Record the last visited page when running the application.
07: Initialize the page.
08: Get the source code of the page using the XMLHttpRequest object.
09: Initialize the Log function.
11: Check NetCast version.
14-15: Add event handlers which will be executed when the corresponding button is pressed.
16-17: Add event handlers related to voice.
20: Call function to initialize voice.
21: Call function to show property values.

```
01 : function initPage()
02 : {
03 :    //save page as last visited page
04 :    setLastVisitPage();
05 :
06 :    //common initialize function
07 :    commonInitialize();
08 :    requestSourceCode();
09 :    jsLog.initLG();
10 :
11 :    checkNetCastVersion();
12 :
13 :    //add onclick event handler
14 :    addEventHandler(document.getElementById("btn_back"), "click", onClickHandler);
15 :    addEventHandler(document.getElementById("btn_exit"), "click", onClickHandler);
16 :    addEventHandler(document.getElementById("dictation_mode"), "click",
onDictationModeHandler);
17 :    addEventHandler(document.getElementById("keyword_mode"), "click",
onKeywordModeHandler);
18 :
19 :    jsLog.lgobject('application/x-netcast-voice');
20 :    initVoice();
21 :    showPropResult();
22 : }
```

Check the NetCast Version

Use checkNetCastVersion function to check NetCast version as the voice recognition plugin and API are supported on NetCast 3.0 1st SU or later.

03: Declare nBrowserVersion.
04-08: If return value of nBrowserVersion is 4, then hide the visibility of all three buttons :
a. dictation_mode, b. keyword_mode, c. start_Recording
11-16: If return value of nBrowserVersion is equal or greater than 5, then display the names for supported method, property and event in API list area.

```
01 : function checkNetCastVersion()
02 : {
03 :    var nBrowserVersion = getBrowserVersion();
04 :    if(nBrowserVersion == 4) // NetCast 2.0
05 :    {
06 :       document.getElementById('dictation_mode').style.visibility = "hidden";
07 :       document.getElementById('keyword_mode').style.visibility = "hidden";
08 :       document.getElementById('start_Recording').style.visibility = "hidden";
09 :    }
10 :    else if(nBrowserVersion >= 5) // NetCast 3.0
11 :    {
12 :       setInnerTextById("APIdescription", "The voice recognition plugin and API are
supported on NetCast 3.0 1st SU or later.");
13 :       setInnerTextById("method","startRecognition()");
14 :       setInnerTextById("property","isInitialized<br> isEnabled<br>dictation");
15 :       setInnerTextById("events","onrecognizevoice<br>onbuttonenable");
16 :    }
17 : }
```

Initializing Voice Recognition

03: Declare device. Some LG Smart TVs support voice recognition of the Magic Remote.
04: Declare voice.
06: device.supportVoiceRecog: Developers can check whether or not supports voice recognition in TV by using the “supportVoiceRecog" read-only property in the Device Information plugin object. It returns true if LG Smart TV supports voice recognition, otherwise, it returns false.
voice.isInitialized: Determine whether the voice recognition function is initialized and returns the value as Boolean.
voice.isEnable: Determine whether the Magic Remote is paired (including its type) and whether the voice recognition function is enabled, and returns the value as Boolean.
13: If any one of three conditions is true, then only handle event for ‘Start Recording’ button.
15-16: Handle voice events.

```
01 : function initVoice()
02 : {
03 :    var device = document.getElementById("device");
04 :    var voice = document.getElementById('voice');
05 :
06 :    if(!(device.supportVoiceRecog || voice.isInitialized || voice.isEnable))
07 :    {
08 :       //alert("em");
09 :       document.getElementById("start_Recording").className="executeBigButtonOff";
10 :    }
11 :    else
12 :    {
13 :       addEventHandler(document.getElementById("start_Recording"), "click",
voiceHandler);
14 :    }
15 :    voice.onrecognizevoice = recognizeVoiceHandler;
16 :    voice.onbuttonenable = buttonStateHandler;
17 : }
```

Handling Button Click Events

The following functions are event handlers which were added in **initPage** function.

dictationModeHandler
This function is called when ‘dictation mode’ button is clicked.

03: Declare voice.
04: Set ‘dictation’ ON to use dictation property of voice recognition function. By default, it is in keyword mode.
05: Call function to show property values.

keywordModeHandler
This function is called when ‘keyword mode’ button is clicked.

11: Declare voice.
12: Set ‘dictation’ OFF to use keyword property of voice function.
13: Call function to show property values.

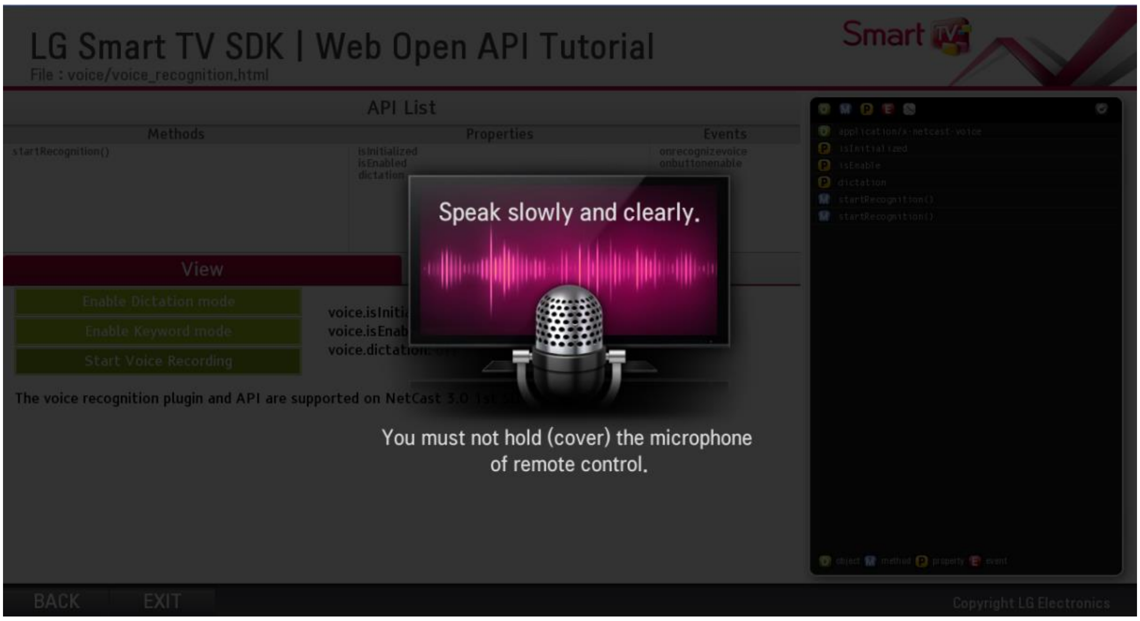
```
01 : function onDictationModeHandler()
02 : {
03 :    var voice = document.getElementById('voice');
04 :    voice.dictation = "on";
05 :    showPropResult()
06 :    jsLog.lgproperty('dictation');
07 : }
08 :
09 : function onKeywordModeHandler()
10 : {
11 :    var voice = document.getElementById('voice');
12 :    voice.dictation = "off";
13 :    showPropResult()
14 :    jsLog.lgproperty('dictation');
15 : }
```

The following functions are event handlers which were added in **initVoice** function.

voiceHandler
This function is called when ‘Start Recording’ button is clicked.

03: Declare voice.
04-07: If voice recognition function is enabled, call startRecognition function and receive the result.

```
01 : function voiceHandler()
02 : {
03 :    var voice = document.getElementById('voice');
04 :    if(voice.isEnable == true)
05 :    {
06 :       voice.startRecognition();
07 :       jsLog.lgmethod('startRecognition()');
08 :    }
09 : }
```



LG Smart TV SDK | Web Open API Tutorial
File : voice/voice_recognition.html

API List

Methods	Properties	Events
startRecognition()	isInitialized isEnabled dictation	onrecognizevoice onbuttonenable

voice.isIniti
voice.isEnab
voice.dictation:

The voice recognition plugin and API are supported on NetCast 3.0 1st

Speak slowly and clearly.

You must not hold (cover) the microphone of remote control.

application/x-netcast-voice
isInitialized
isEnable
dictation
startRecognition()
startRecognition()

Copyright LG Electronics

Handling Voice Events

The following functions are voice event handlers which were added in **initVoice** function.

recognizeVoiceHandler
This function is handler of onrecognizevoice event , in order to receive the voice recognition result from the TV.

04-11: If event occurs, display its value in log window. Otherwise display “No Event”.

buttonStateHandler
This function is handler of onbuttonenable event , in order to enable or disable the voice recognition button. Upon pairing, the event receives the availability of the voice recognition according to the Magic Remote type from the TV. If the Magic Remote 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.

19-21: If event value is true, then enable “Start Recording” button.
22: Add event handler for “Start Recording” button.
23: If event value is false, then disable “Start Recording” button.
29: Remove event handler for “Start Recording” button.

```
01 : function recognizeVoiceHandler(e)
02 : {
03 :   console.log(">O< recognizeVoiceHandler is called.");
04 :   if(e)
05 :   {
06 :     result = e;
07 :     jsLog.lgevent('recognizeVoiceHandler : '+'result = '+ result);
08 :   }
09 :   else
10 :   {
11 :     jsLog.lgevent('recognizeVoiceHandler : '+' No Event');
12 :   }
13 : }
14 :
15 : function buttonStateHandler(e)
16 : {
17 :   if(e)
18 :   {
19 :     if(e == true)
20 :     {
21 :       console.log(">O< buttonStateHandler is called");
22 :       document.getElementById("start_Recording").className="executeBigButton";
23 :       addEventHandler(document.getElementById("start_Recording"), "click",
voiceHandler);
24 :       jsLog.lgevent('buttonStateHandler : '+' true');
25 :     }
26 :     else
27 :     {
28 :      document.getElementById("start_Recording").className="executeBigButtonOff";
29 :      removeEventHandler(document.getElementById("start_Recording"), "click",
voiceHandler);
30 :      jsLog.lgevent('buttonStateHandler : '+' false');
31 :     }
32 :   }
33 : }
```

Displaying Text

The following code displays the voice recognition object information on screen by using properties.

showPropResult
04: Display the value of ‘voice.isInitialized’ according to the return value of ‘voice.isInitialized’ property.
06: Display the value of ‘voice.isEnable’ according to the return value of ‘voice.isEnable’ property.
08: Display the value of ‘voice.dictation’ according to the return value of ‘voice.dictation’ property.

```
01 : function showPropResult() {
```

```
02 :   var msgArea = document.getElementById('displayArea');
03 :   var htmlStr="";
04 :   htmlStr+=(voice.isInitialized!=undefined) ? "<font style='color:#000000; font-
weight:bold'>voice.isInitialized:</font> "+voice.isInitialized : "";
05 :   jsLog.lgproperty('isInitialized');
06 :   htmlStr+=(voice.isEnable!=undefined)? "<br><font style='color:#000000; font-
weight:bold'>voice.isEnable:</font> "+voice.isEnable : "";
07 :   jsLog.lgproperty('isEnable');
08 :   htmlStr+=(voice.dictation!=undefined)? "<br><font style='color:#000000; font-
weight:bold'>voice.dictation:</font> "+voice.dictation : "";
09 :   jsLog.lgproperty('dictation');
10 :   msgArea.innerHTML =htmlStr;
11 : }
```

Inputting Keys

Use **onUserInput** function is called by onClickHandler function; it receives a key value as the userInput parameter from onClickHandler and creates the corresponding function for each key value to operate the key.

06: When the back key is pressed, it will open previous page.

```
01 : //onUserInput function should be implemented
02 : function onUserInput(userInput)
03 : {
04 :   switch(userInput)
05 :   {
06 :     case VK_BACK :
07 :     window.location.replace("../main_menu.html");
08 :     break;
09 :   }
10 : }
```

Setting Voice Recognition Object

The following code shows how to set voice recognition object.

02: Set data type. Refer to “LG Web Application Development Guide” for related information.
03: Set id property as ‘voice’.
04: The dictation property can be used when the voice recognition function is in dictation mode. The default is the keyword mode.

```
01 : <object
02 : type="application/x-netcast-voice"
03 : id="voice"
04 : dictation=”on”>
05 : </object>
```

Source Code of voice_recognition.html

Source code of voice_recognition.html is as follows:

```
<!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=EUC-KR">
<title>Voice recognition API Test</title>
<link rel="stylesheet" href="../css/style.css">
<script language="javascript" src="../js/common.js"></script>
<script language="javascript" src="../js/keycode.js"></script>
<script language="javascript" src="../js/menu.js"></script>
```

```
<script type="text/javascript" src="../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../blackbirdjs/blackbird.css" />

<script>

   //initialize page
   function initPage()
   {
      //save page as last visited page
      setLastVisitPage();

      //common initialize function
      commonInitialize();
      requestSourceCode();
      jsLog.initLG();

      checkNetCastVersion();

      //add onclick event handler
      addEventHandler(document.getElementById("btn_back"), "click", onClickHandler);
      addEventHandler(document.getElementById("btn_exit"), "click", onClickHandler);
      addEventHandler(document.getElementById("dictation_mode"), "click",
onDictationModeHandler);
      addEventHandler(document.getElementById("keyword_mode"), "click",
onKeywordModeHandler);

      jsLog.lgobject('application/x-netcast-voice');
      initVoice();
      showPropResult();
   }

   function showPropResult(){
      var msgArea = document.getElementById('displayArea');
      var htmlStr="";
      htmlStr+=(voice.isInitialized!=undefined) ? "<font style='color:#000000; font-
weight:bold'>voice.isInitialized:</font> "+voice.isInitialized : "";
      jsLog.lgproperty('isInitialized');
      htmlStr+=(voice.isEnable!=undefined)? "<br><font style='color:#000000; font-
weight:bold'>voice.isEnable:</font> "+voice.isEnable : "";
      jsLog.lgproperty('isEnable');
      htmlStr+=(voice.dictation!=undefined)? "<br><font style='color:#000000; font-
weight:bold'>voice.dictation:</font> "+voice.dictation : "";
      jsLog.lgproperty('dictation');
      msgArea.innerHTML =htmlStr;
   }

   //onUserInput function should be implemented
   function onUserInput(userInput)
   {
      switch(userInput)
      {
         case VK_BACK :
         window.location.replace("../main_menu.html");
         break;
      }
   }

   function initVoice()
   {
      var device = document.getElementById("device");
```

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

    if(!(device.supportVoiceRecog || voice.isInitialized || voice.isEnable))
    {
      //alert("em");
      document.getElementById("start_Recording").className="executeBigButtonOff";
    }
    else
    {
      addEventHandler(document.getElementById("start_Recording"), "click",
voiceHandler);
    }

    voice.onrecognizevoice = recognizeVoiceHandler;
    voice.onbuttonenable = buttonStateHandler;
  }

  function voiceHandler()
  {
    var voice = document.getElementById('voice');
    if(voice.isEnable == true)
    {
      voice.startRecognition();
      jsLog.lgmethod('startRecognition()');
    }
  }

  function recognizeVoiceHandler(e)
  {
    console.log(">O< recognizeVoiceHandler is called.");
    if(e)
    {
      result = e;
      jsLog.lgevent('recognizeVoiceHandler : '+'result = '+ result);
    }
    else
    {
      jsLog.lgevent('recognizeVoiceHandler : '+' No Event');
    }
  }

  function buttonStateHandler(e)
  {
    if(e)
    {
      if(e == true)
      {
        console.log(">O< buttonStateHandler is called");
        document.getElementById("start_Recording").className="executeBigButton";
        addEventHandler(document.getElementById("start_Recording"), "click",
voiceHandler);
        jsLog.lgevent('buttonStateHandler : '+' true');
      }
      else
      {
        document.getElementById("start_Recording").className="executeBigButtonOff";
        removeEventHandler(document.getElementById("start_Recording"), "click",
voiceHandler);
        jsLog.lgevent('buttonStateHandler : '+' false');
      }
    }

    function onDictationModeHandler()
    {
      var voice = document.getElementById('voice');
      voice.dictation = "on";
      showPropResult()
      jsLog.lgproperty('dictation');
    }

    function onKeywordModeHandler()
    {
      var voice = document.getElementById('voice');
      voice.dictation = "off";
      showPropResult()
      jsLog.lgproperty('dictation');
    }

    function checkNetCastVersion()
    {
      var nBrowserVersion = getBrowserVersion();
      if(nBrowserVersion == 4) // NetCast 2.0
      {
        document.getElementById('dictation_mode').style.visibility = "hidden";
        document.getElementById('keyword_mode').style.visibility = "hidden"
        document.getElementById('start_Recording').style.visibility = "hidden";
      }
      else if(nBrowserVersion >= 5) // NetCast 3.0
      {
        setInnerTextById("APIdescription", "The voice recognition plugin and API are 
supported on NetCast 3.0 1st SU or later.");
        setInnerTextById("method","startRecognition()");
        setInnerTextById("property","isInitialized<br> isEnabled<br>dictation");
        setInnerTextById("events","onrecognizevoice<br>onbuttonenable");
      }
    }
</script>
</head>

<body ondragstart='return false' onselectstart='return false' 
onload="javascript:initPage();">

<!-- title -->
<div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>

<!-- navigation -->
<div class='SuiteNavigation'>
    <div style="float:left;">File : voice/voice_recognition.html</div>
</div>

<div class='SuiteTitleLine'> </div>

<!-- test contents -->
<div id='content_body' class='ContentArea'>

    <div class='ApiListTitleArea'>API List</div>
    <div class='ApiListArea'>
       <div class='MethodTitleArea'>
          Methods
          <div class='MethodListArea' id="method"></div>
        </div>
        <div class='PropertyTitleArea'> Properties
            <div class='PropertyListArea' id="property"> </div>
        </div>
        <div class='EventTitleArea'> Events
            <div class='EventListArea'  id="events"> </div>
        </div>
    </div>

    <div class='ViewTitleArea'>
        <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
        <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
    </div>

    <div id='view'>
        <div class='ViewArea'>
            <object
                id="voice"
                type="application/x-netcast-voice"
                width="0"
                height="0"
                style="float: left"
            >
            </object>
            <object
                id="device"
                type="application/x-netcast-info"
                width="0"
                height="0"
                style="float: left"
            >
            </object>
            <table>
                <tr><td width="350"><div id="dictation_mode" class="executeBigButton"
>Enable Dictation mode</div></td>
                    <td rowspan="3"><div id="displayArea" class="displayAreaStyle">Hi
this is the area for the item</div></td>
            </tr>
            <tr><td><div id="keyword_mode" class="executeBigButton">Enable Keyword
mode</div></td>
            </tr>
            <tr><td><div id="start_Recording" class="executeBigButton">Start Voice
Recording</div></td>
            </tr>

            <tr height="50px">
            <td colspan="2" align=left><div id="APIdescription" ></div></td>
            </tr>
            </table>

        </div>
    </div>

    <div style="visibility: hidden" id='codeview'>
        <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
    </div>

</div>
```

```
<!-- button and copyright -->
<div class='SuiteButtonArea'>

   <!-- button -->
   <div id='btn_back' class='buttonDescription'>BACK</div>

   <!-- exit key description -->
   <div id='btn_exit' class='buttonDescription'>EXIT</div>

   <!-- copyright -->
   <div class='copyright'>Copyright LG Electronics</div>
</div>

</body>
</html>
```

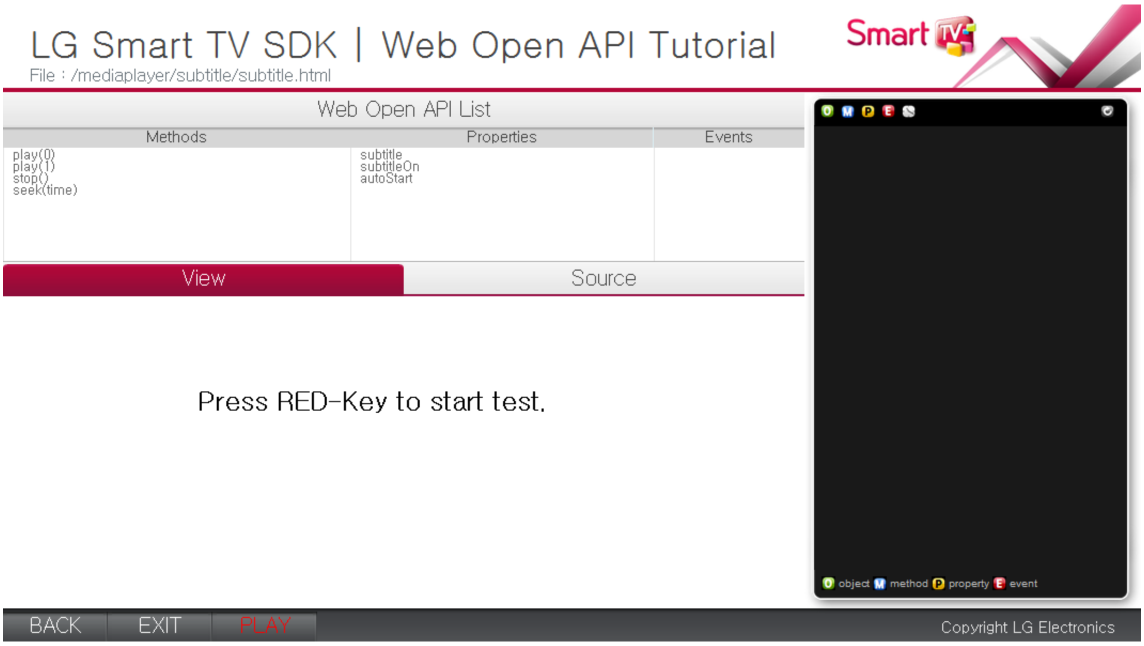
Displaying Subtitle in Media Player

This section describes how to display the subtitle in media player.

- Initializing the Page
- Inputting Keys
- Displaying Text
- Controlling Play Mode and Subtitle
- Setting Media Object
- Source Code of subtitle.html

This application is designed to show how to display subtitle while media file is playing using the Web Open API of Smart TV.

This application shows which property is used to set and control on/off status of subtitle.



LG Smart TV SDK | Web Open API Tutorial

File : /mediaplayer/subtitle/subtitle.html

Web Open API List

Methods	Properties	Events

play(0)
play(1)
stop()
seek(time)
	subtitle
	subtitleOn
	autoStart	

Press RED-Key to start test.

Copyright LG Electronics

Needed APIs
This application uses following Web Open API:

API Class	Function Name	Description
Method	play(0)	Pauses media.
	play(1)	Plays media.
	stop()	Stops media.
	seek(time)	Seeks media to specific time.
	play(0)	Pauses media.
Property	subtitle	Supports SAMI, CineCanvas ,Timed Text subtitle.
	subtitleOn	Controls subtitle display by true/false.
	autoStart	Plays media automatically.

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

04: Record the last visited page when running the application.
07: Initialize the page.
08: Get the source code of the page using the XMLHttpRequest object.
09: Set the page ID.
10: Initialize the Log function.
13-15: Add an event handler which will executed when the corresponding button is pressed.
17: Declare userAgent.
18~21: Check if the device uses LG Browser.

For information on userAgent string, refer to App Development Guide.

```
01 : function initPage()
02 : {
03 :   //save page as last visited page
04 :   setLastVisitPage();
05 :
06 :   //common initialize function
07 :   commonInitialize();
08 :   requestSourceCode();
09 :   setPageID("Subtitle");
10 :   jsLog.initLG();
11 :
12 :   //add onclick event handler
13 :   addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
14 :   addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
15 :   addEventHandler(document.getElementById("btn_exit"),"click",onClickHandler);
16 :
17 :   var userAgent = new String(navigator.userAgent);
18 :   if (userAgent != null && userAgent.search(/LG Browser/) > -1)
19 :      isLGEBrowser = true;
20 :   else
21 :      isLGEBrowser = false;
22 :   }
```

Inputting Keys

The **onUserInput** function is called by the onClickHandler function; it receives a key value as the userInput parameter from onClickHandler and creates the corresponding function for each key value to operate the key.

05: When the Back key is pressed, this code is executed.
07: When the Red key is pressed, this code is executed.

```
01 : function onUserInput(userInput)
02 : {
03 :  switch(userInput)
04 :  {
05 :    case VK_BACK : window.location.replace ("../menu_mediaPlayer.html");
06 :        break;
07 :    case VK_RED : case 82 changeTestProgress(); break;
08 :        break; 09 :   }
10 : }
```

Displaying Text

Use **setTestProgress** function to set the text of subtitle_test_description and btn_red.

```
01 : function setTestProgress( descString, buttonString)
02 : {
03 :   setInnerTextById("subtitle_test_description", descString);
04 :   setInnerTextById("btn_red", buttonString);
05 : }
```

Controlling Play Mode and Subtitle

The following code controls video play mode and subtitle settings.

The **changeTestProgress** function controls the play mode or on/off the subtitle according to the value of testStep.

01: Declare and initialize testStep.
05-06: Declare s video and sourcecode.
07: Hide sourcecode.
09-96: Using switch-case, control the play mode or subtitle settings according to the value of testStep. Also, display the text for each mode. Refer to the comments of each case.
97: Reinitialize testStep if the value gets bigger than the setup value.
98: Whenever the function is called, the value of testStep is increased by 1.

```
01 : var testStep = 1;
02 :
03 : function changeTestProgress()
04 : {
05 :   var video = document.getElementById("video");
06 :   var sourcecode = document.getElementById("sourcecode");
07 :   sourcecode.style.visibility = "hidden";
08 :
09 :   switch(testStep)
10 :   {
11 :      case 1 :
12 :          //Play video and check whether subtitle is displayed properly
13 :          if(isLGEBrowser)
14 :          {
15 :             video.play(1);
16 :             video.subtitle="/ApiTutorial/mediafile/cinecanvas-timing.1.dcs";
17 :             video.style.visibility="visible";
18 :             bodycontent.style.visibility="hidden";
19 :             subtitle_test_description.style.visibility="visible";
20 :           }
21 :           jsLog.lgobject('application/x-netcast-av');
22 :           jsLog.lgmethod('video.play(1)');
23 :           jsLog.lgproperty('subtitle');
24 :           setTestProgress( "Check whether subtitle is displayed in proper
sync.<br>Compare subtitle with time shown at the bottom.<br>Press RED-Key to pause.",
"PAUSE");
25 :           break;
26 :
27 :      case 2 :
28 :          //Pause video which is being played.
29 :          if(isLGEBrowser){video.play(0);}
30 :             jsLog.lgmethod('video.play(0)');
31 :             setTestProgress( "The video has been paused.<br>Play the video and
then check subtitle.<br>Press RED-Key to play video.", "PLAY");
32 :             break;
33 :
34 :      case 3 :
35 :          //play video and check subtitle
36 :         if(isLGEBrowser){video.play(1);}
37 :         jsLog.lgmethod('video.play(1)');
38 :         setTestProgress( "Check whether subtitle is displayed in proper
sync.<br>Compare subtitle with time shown at the bottom.<br>Press RED-Key to move to
60 sec position.", "MOVE to 60 sec position");
39 :          break;
40 :
    case 4 :
        //seek video and check subtitle
        if(isLGEBrowser){video.seek(60000);}
        jsLog.lgmethod('video.seek(time)');
        setTestProgress( "The time position has been moved to 60 sec position.<br>Check whether subtitle is displayed in proper sync.<br>Compare subtitle with time shown at the bottom.<br>Press RED-Key to stop video.", "STOP");
        break;

    case 5 :
        //Stop video which is being played.
        if(isLGEBrowser){video.stop();}
        jsLog.lgmethod('video.stop()');
        setTestProgress( "The video has been stopped.<br>Restart the video and then check subtitle.<br>Press RED-Key to restart video.", "RESTART");
        break;

    case 6 :
        //Restart video and check subtitle
       if(isLGEBrowser){video.play(1);}
       jsLog.lgmethod('video.play(1)');
       setTestProgress( "Check whether subtitle is displayed in proper sync.<br>Compare subtitle with time shown at the bottom.<br>Press RED-Key to turn subtitle off.", "SUBTITLE OFF");
        break;

    case 7 :
        //Turn subtitle off
        if(isLGEBrowser){video.subtitleOn=false;}
        jsLog.lgproperty('subtitleOn');
        setTestProgress( "The subtitle has been turned off.<br>Check that subtitle is no longer displayed.<br>Press RED-Key to turn subtitle on.", "SUBTITLE ON");
        break;

    case 8 :
        //Turn subtitle on
        if(isLGEBrowser){video.subtitleOn=true;}
        jsLog.lgproperty('subtitleOn');
        setTestProgress("The subtitle has been turned on.<br>Check whether subtitle is displayed in proper sync.<br>Compare subtitle with time shown at the bottom.<br>Press RED-Key to change subtitle.", "SUBTITLE CHANGE TO SMI");
        break;

    case 9 :
        //change subtitle to different type of files :smi
        if(isLGEBrowser)
        {
            video.subtitle="/ApiTutorial/mediafile/smi-timing.1.smi";
        }
        jsLog.lgproperty('subtitle');
        setTestProgress("The subtitle has been changed.<br>Check that subtitle of 'SMI file' is displayed in proper sync.<br>Compare subtitle with time shown at the bottom.", "SUBTITLE CHANGE TO XML");
        break;

    case 10 :
        //change subtitle to different type of files :xml
        if(isLGEBrowser)
        {
            video.subtitle="/ApiTutorial/mediafile/danish.xml";
         }
         jsLog.lgproperty('subtitle');
         setTestProgress("The subtitle has been changed.<br>Check that subtitle of 'XML file' is displayed in proper sync.<br>Compare subtitle with time shown at the bottom.<br>No more test is left on this page.", "");
         document.getElementById("btn_red").style.display="none";
         break;
      }
  if(testStep == 10) testStep = 1;
  testStep ++;
}
```

Setting Media Object

03: Set data type. Refer to App Development Guide.
04-05: Set width and height.
08-09: Set basic information of subtitle.
10-11: Set other properties of Media object.

```
01 : <object
02 :   id="video"
03 :   type="application/x-netcast-av"
04 :   width=1280
05 :   height=720
06 :   data="/ApiTutorial/mediafile/timer.mp4"
07 :   style="float: left; z-index: 1; visibility: hidden;"
08 :   subtitle="/ApiTutorial/mediafile/cinecanvas-timing.1.dcs"
09 :   subtitleOn=true
10 :   autoStart=false
11 :   PlayCount="0">
12 : </object>
```

Source Code of subtitle.html

Source code of subtitle.html is as follows:

```
<!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=UTF-8">
<title>Subtitle Test Page</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/common.js"></script>
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/menu.js"></script>
<script language="javascript" src="../../js/media.js"></script>

<script type="text/javascript" src="../../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../../blackbirdjs/blackbird.css" />
<style type="text/css">
   .OverLay { position: absolute; z-index:1; opacity: 1; filter: alpha(opacity = 100); }
   body { height: 100%; }
   html { height: 100%; }
</style>
<script>

   //initialize page
   function initPage()
```

```
{
   //save page as last visited page
   setLastVisitPage();

   //common initialize function
   commonInitialize();
   requestSourceCode();
   setPageID("Subtitle");
   jsLog.initLG();

   //add onclick event handler
   addEventHandler(document.getElementById("btn_back"), "click", onClickHandler);
   addEventHandler(document.getElementById("btn_red"), "click", onClickHandler);
   addEventHandler(document.getElementById("btn_exit"), "click", onClickHandler);

   var userAgent = new String(navigator.userAgent);
   if (userAgent != null && userAgent.search(/LG Browser/) > -1)
      isLGEBrowser = true;
   else
      isLGEBrowser = false;
}

function onUserInput(userInput)
{
   switch(userInput)
   {
      case VK_BACK : window.location.replace("../menu_mediaPlayer.html"); break;
      case VK_RED : case 82 : changeTestProgress(); break;
   }
}

function setTestProgress( descString, buttonString)
{
   setInnerTextById("subtitle_test_description", descString);
   setInnerTextById("btn_red", buttonString);
}

var testStep = 1;
function changeTestProgress()
{
   var video = document.getElementById("video");
   var sourcecode = document.getElementById("sourcecode");
   sourcecode.style.visibility = "hidden";

   switch(testStep)
   {
      case 1 :
         //Play video and check whether subtitle is displayed properly
         if(isLGEBrowser)
         {
            video.play(1);
            video.subtitle="/ApiTutorial/mediafile/cinecanvas-timing.1.dcs";
            video.style.visibility="visible";
            bodycontent.style.visibility="hidden";
            subtitle_test_description.style.visibility="visible";
         }
         jsLog.lgobject('application/x-netcast-av');
         jsLog.lgmethod('video.play(1)');
         jsLog.lgproperty('subtitle');
         setTestProgress( "Check whether subtitle is displayed in proper
sync.<br>Compare subtitle with time shown at the bottom.<br>Press RED-Key to pause.",
"PAUSE");
             break;
         case 2 :
            //Pause video which is being played.
            if(isLGEBrowser){video.play(0);}
            jsLog.lgmethod('video.play(0)');
            setTestProgress( "The video has been paused.<br>Play the video and then
check subtitle.<br>Press RED-Key to play video.", "PLAY");
            break;
         case 3 :
            //play video and check subtitle
            if(isLGEBrowser){video.play(1);}
            jsLog.lgmethod('video.play(1)');
            setTestProgress( "Check whether subtitle is displayed in proper
sync.<br>Compare subtitle with time shown at the bottom.<br>Press RED-Key to move to
60 sec position.", "MOVE to 60 sec position");
            break;
         case 4 :
            //seek video and check subtitle
            if(isLGEBrowser){video.seek(60000);}
            jsLog.lgmethod('video.seek(time)');
            setTestProgress( "The time position has been moved to 60 sec
position.<br>Check whether subtitle is displayed in proper sync.<br>Compare subtitle
with time shown at the bottom.<br>Press RED-Key to stop video.", "STOP");
            break;
         case 5 :
            //Stop video which is being played.
            if(isLGEBrowser){video.stop();}
            jsLog.lgmethod('video.stop()');
            setTestProgress( "The video has been stopped.<br>Restart the video and
then check subtitle.<br>Press RED-Key to restart video.", "RESTART");
            break;
         case 6 :
            //Restart video and check subtitle
            if(isLGEBrowser){video.play(1);}
            jsLog.lgmethod('video.play(1)');
            setTestProgress( "Check whether subtitle is displayed in proper
sync.<br>Compare subtitle with time shown at the bottom.<br>Press RED-Key to turn
subtitle off.", "SUBTITLE OFF");
            break;
         case 7 :
            //Turn subtitle off
            if(isLGEBrowser){video.subtitleOn=false;}
            jsLog.lgproperty('subtitleOn');
            setTestProgress( "The subtitle has been turned off.<br>Check that subtitle
is no longer displayed.<br>Press RED-Key to turn subtitle on.", "SUBTITLE ON");
            break;
         case 8 :
            //Turn subtitle on
            if(isLGEBrowser){video.subtitleOn=true;}
            jsLog.lgproperty('subtitleOn');
            setTestProgress("The subtitle has been turned on.<br>Check whether
subtitle is displayed in proper sync.<br>Compare subtitle with time shown at the
bottom.<br>Press RED-Key to change subtitle.", "SUBTITLE CHANGE TO SMI");
                break;
          case 9 :
             //change subtitle to different type of files :smi
             if(isLGEBrowser)
             {
                video.subtitle="/ApiTutorial/mediafile/smi-timing.1.smi";
             }
             jsLog.lgproperty('subtitle');
             setTestProgress("The subtitle has been changed.<br>Check that subtitle of
'SMI file' is displayed in proper sync.<br>Compare subtitle with time shown at the
bottom.", "SUBTITLE CHANGE TO XML");
             break;
          case 10 :
             //change subtitle to different type of files :xml
             if(isLGEBrowser)
             {
                video.subtitle="/ApiTutorial/mediafile/danish.xml";
             }
             jsLog.lgproperty('subtitle');
             setTestProgress("The subtitle has been changed.<br>Check that subtitle of
'XML file' is displayed in proper sync.<br>Compare subtitle with time shown at the
bottom.<br>No more test is left on this page.", "");
             document.getElementById("btn_red").style.display="none";
             break;
       }
       if(testStep == 10) testStep = 1;
       testStep ++;
    }

</script>
</head>


<body ondragstart='return false' onselectstart='return false'
onload="javascript:initPage();">

<div>
    <object
       id="video"
       type="application/x-netcast-av"
       width=1280
       height=720
       data="/ApiTutorial/mediafile/timer.mp4"
       style="float: left; z-index: 1; visibility: hidden;"
       subtitle="/ApiTutorial/mediafile/cinecanvas-timing.1.dcs"
       subtitleOn=true
       autoStart=false
       PlayCount="0">
    </object>
</div>

<div id='bodycontent'>
    <!-- title -->
    <div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>

    <!-- navigation -->
    <div class='SuiteNavigation'>
       <div style="float:left;">File : /mediaplayer/subtitle/subtitle.html</div>
    </div>

    <div class='SuiteTitleLine'> </div>

    <!-- test contents -->
    <div class='ContentArea'>
       <div class='ApiListTitleArea'>API List</div>
```

```
       <div class='ApiListArea'>
          <div class='MethodTitleArea'>
             Methods
             <div class='MethodListArea'>
             play(0)<br>
             play(1)<br>
             stop()<br>
             seek(time)<br>
             </div>
          </div>
          <div class='PropertyTitleArea'>
             Properties
             <div class='PropertyListArea'>
             subtitle<br>
             subtitleOn<br>
             autoStart
             </div>
          </div>
          <div class='EventTitleArea'>
             Events
             <div class='EventListArea'>

             </div>
          </div>
       </div>

       <div class='ViewTitleArea'>
          <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
          <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
       </div>

       <div id='view'>
          <div class='ViewArea'>
             <div class='centerTestGuide' style="position: relative;left: 200px;
top:100px;">Press RED-Key to start test.</div>
             <div class='ViewArea whiteColor' id='subtitle_test_description'
style="position: relative;left: 200px; top:-70px; font-size: 30px; visibility:
hidden;">Press RED-Key to start test.</div>
          </div>
       </div>

       <div style="visibility: hidden" id='codeview'>
          <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
      </div>
   </div>

</div>

<!-- button and copyright -->
<div class='SuiteButtonArea '>
   <!-- back key description -->
   <div id='btn_back' class='buttonDescription '>BACK</div>

   <!-- exit key description -->
   <div id='btn_exit' class='buttonDescription'>EXIT</div>

   <!-- red key description -->
   <div id='btn_red' class='buttonDescription redColor'>PLAY</div>
   <!-- copyright -->
   <div class='copyright '>Copyright LG Electronics</div>
</div>

</body>
</html>
```

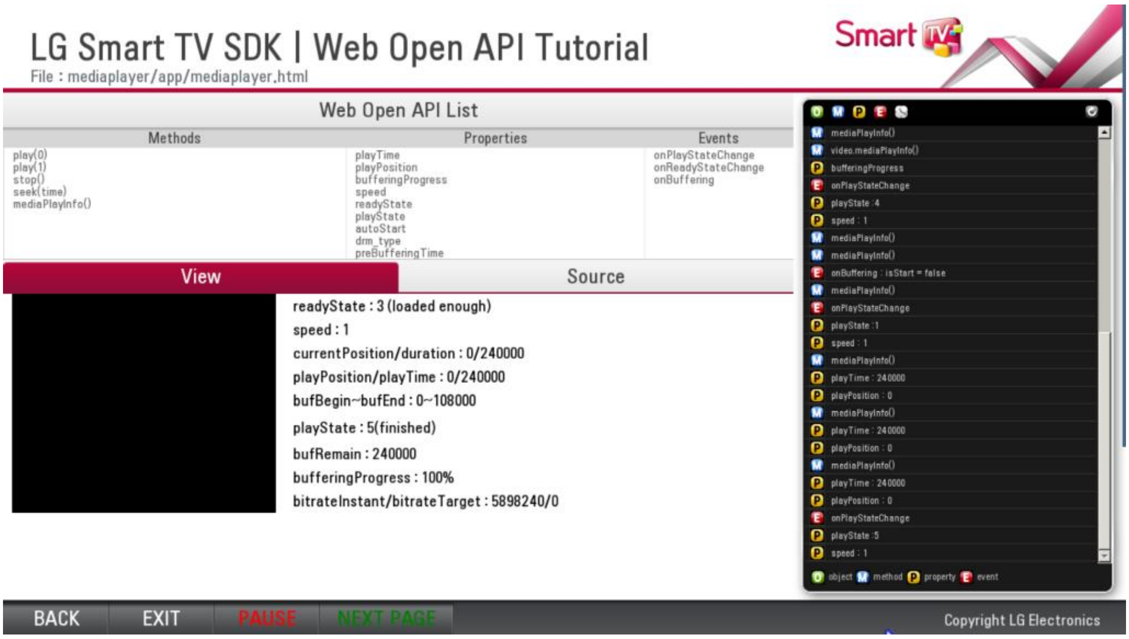
Executing Windows Media Player 1

This section describes how to execute the windows media player in web application.

- Initializing the Page
- Handling Media Events
- Controlling Play Mode
- Displaying Text
- Inputting Keys
- Setting Media Object
- Source Code of mediaplayer.html

This application is designed to shows which method, properties, and events of LG Web Open API are used to get basic information when executing media player in LG Smart TV.

This application shows how to control media and get property values. Also, occurred and used events when media status is changed are shown.



LG Smart TV SDK | Web Open API Tutorial

File : mediaplayer/app/mediaplayer.html

Web Open API List

Methods	Properties	Events

play(0)
play(1)
stop()
seek(time)
mediaPlayInfo()
	playTime
	playPosition
	bufferingProgress
	speed
	readyState
	playState
	autoStart
	drm_type
	preBufferingTime
		onPlayStateChange
		onReadyStateChange
		onBuffering

```
readyState : 3 (loaded enough)
speed : 1
currentPosition/duration : 0/240000
playPosition/playTime : 0/240000
bufBegin~bufEnd : 0~108000
playState : 5(finished)
bufRemain : 240000
bufferingProgress : 100%
bitrateInstant/bitrateTarget : 5898240/0
```

```
M mediaPlayInfo()
M video.mediaPlayInfo()
P bufferingProgress
E onPlayStateChange
P playState :4
P speed : 1
M mediaPlayInfo()
M mediaPlayInfo()
E onBuffering : isStart = false
M mediaPlayInfo()
E onPlayStateChange
P playState :1
P speed : 1
M mediaPlayInfo()
P playTime : 240000
P playPosition : 0
M mediaPlayInfo()
P playTime : 240000
P playPosition : 0
M mediaPlayInfo()
P playTime : 240000
P playPosition : 0
E onPlayStateChange
P playState :5
P speed : 1
```

Needed APIs

This application uses following Web Open API:

API Class	Function Name	Description
Method	play(0)	Pauses media.
	play(1)	Plays media.
	stop()	Stops media.
	seek(time)	Seeks media to specific time.
	mediaPlayInfo	Media playbackinformation
Property	playTime	Total play time of media
	playPosition	Current play position
	bufferingProgress	Buffering status
	Speed	Play speed
	readyState	Returns current media ready status.
	playState	Returns media play status by number.
	autoStart	Set whether media file playout to be started automatically.
	drm_type	Set drm type.

API Class	Function Name	Description
	preBufferingTime	Adjust buffering time before playback.
Event	onPlayStateChange	The event occurs when play status of media item changes.

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

04: Record the last visited page when running the application.
07: Initialize the page.
08: Get the source code of the page using the XMLHttpRequest object.
09: Set the page ID.
10: Initialize the Log function.
13-16: Add an event handler which will executed when the corresponding button is pressed.
18: Set text of btn_red to ”PAUSE”.
21: Declare video.
24-28: Add event handlers related to media.

```
01 : function initPage()
02 : {
03 :    //save page as last visited page
04 :    setLastVisitPage();
05 :
06 :   //common initialize function
07 :   commonInitialize();
08 :   requestSourceCode();
09 :   setPageID("Media Player");
10 :   jsLog.initLG();
11 :
12 :   //add onclick event handler
13 :   addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
14 :   addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
15 :   addEventHandler(document.getElementById("btn_green"),"click",onClickHandler);
16 :   addEventHandler(document.getElementById("btn_exit"),"click",onClickHandler);
17 :
18 :   setInnerTextById("btn_red", "PAUSE");
19 :
20 :   //add event handler for test
21 :   var video = document.getElementById("video");
22 :
23 :   //add onReadyStateChange event handler
24 :   video.onReadyStateChange = setReadyState;
25 :
26 :   //add playState event handler
27 :   video.onBuffering = processBufferingStateChangeFunction;
28 :   video.onPlayStateChange = processPlayStateChangeFunction;
29 :
30 :   jsLog.lgobject('application/x-netcast-av');
31 :   jsLog.lgproperty('autoStart');
32 :   jsLog.lgproperty('drm_type');
33 :   jsLog.lgproperty('preBufferingTime');
34 : }
```

Handling Media Events

The following functions are media event handlers which were added in initPage() function.

setReadyState

This function is handler of onReadStateChange event, displayed whenever readystate is changed.
03-05: Declare video, readyStateValue, and readyStateDesc.
07-11: Describe the value of readystate according to the return value of onReadStateChange event.
13: Display the value of readystate and description.

processBufferingStateChangeFunction

This function is handler of onBuffering event, displayed whenever bufferingstate is changed.
21: Call function that will display buffering information.
23-28: If buffering is started, update the buffering information on screen periodically by calling setBufferingInfo() function while buffering is proceeding.
29-32: If buffering ends, the calling of setBufferingInfo() function ends.

processPlayStateChangeFunction

This function is handler of onPlayStateChange event, displayed whenever playstate is changed.
39-41: Declare playStateExplain, video, and playState.
46: Display property speed.
48-54: Describe the value of playstate according to the return value ofonPlayStateChange event.
56: Display the value of playstate and description.
58-64: While the video is playing, update the playtime information on screen periodically by calling setPlayTimeInfo () function. If playing ends, the calling of setPlayTimeInfo() function ends.

```
01 : function setReadyState(readyState)
02 : {
03 :    var video = document.getElementById("video");
04 :    var readyStateValue = video.readyState;
05 :    var readyStateDesc = "";
06 :
07 :    if(readyStateValue == 0){readyStateDesc = "0 (not set)";}
08 :    else if(readyStateValue == 1){readyStateDesc = "1 (loading)";}
09 :    else if(readyStateValue == 3){readyStateDesc = "3 (loaded enough)";}
10 :    else if(readyStateValue == 4){readyStateDesc = "4 (loaded all)";}
11 :    else{readyStateDesc = readyStateValue + " (unknown state)";}
12 :
13 :    setInnerTextById("ready_state_value", "readyState : " + readyStateDesc);
14 :    jsLog.lgevent('onReadyStateChange : readyState = ' + readyState);
15 : }
16 :
17 : function processBufferingStateChangeFunction(isStarted)
18 : {
19 :    jsLog.lgevent('onBuffering : isStart = ' + isStarted);
20 :
21 :    setBufferingInfo();
22 :
23 :    if(isStarted)
24 :    {
25 :       buffertimer = setInterval(setBufferingInfo, 400);
26 :      jsLog.lgmethod('video.mediaPlayInfo()');
27 :      jsLog.lgproperty('bufferingProgress');
28 :   }
29 :   else
30 :   {
31 :      clearInterval(buffertimer);
32 :   }
33 : }
34 :
35 : function processPlayStateChangeFunction()
36 : {
37 :    jsLog.lgevent('onPlayStateChange');
38 :
```

```
39 :    var playStateExplain = "";
40 :    var video = document.getElementById("video");
41 :    var playState = video.playState;
42 :
43 :    jsLog.lgproperty('playState :' + playState);
44 :    jsLog.lgproperty('speed : ' + video.speed);
45 :
46 :    setInnerTextById("speed_value", "speed : " + video.speed);
47 :
48 :    if(playState == 0){playStateExplain = "stopped";}
49 :    else if(playState == 1){playStateExplain = "playing";}
50 :    else if(playState == 2){playStateExplain = "paused";}
51 :    else if(playState == 3){playStateExplain = "connecting";}
52 :    else if(playState == 4){playStateExplain = "buffering";}
53 :    else if(playState == 5){playStateExplain = "finished";}
54 :    else if(playState == 6){playStateExplain = "error";}
55 :
56 :    setInnerTextById("play_state_value", "playState : " + playState + "(" +
playStateExplain + ")");
57 :
58 :    if(playState == 1 )
59 :    { playtimer = setInterval(setPlayTimeInfo,2000);
60 :    }
61 :    else
62 :    {
63 :       clearInterval(playtimer);
64 :    }
65 : }
```

Controlling Play Mode

The following code shows how to control media play mode.

processRedKey

This function changes play status according to the value of the testProcess.
01: Initialize testProcess.
05: Declare video.
07-41: Using switch-case, control the play mode according to the value of testProcess. Set text of Red button for each status.
42: Whenever Red button is pressed, the value of testProcess is increased by 1.
43: Reinitialize testProcess if the value gets bigger than the setup value.

```
01 : var testProcess = 2;
02 :
03 : function processRedKey()
04 : {
05 :    var video = document.getElementById("video");
06 :
07 :   switch(testProcess)
08 :   {
09 :      case 0 :
10 :         setInnerTextById("btn_red", "Press RED-key to play video.");
11 :
12 :      case 1 :
13 :         video.play(1);
14 :         setInnerTextById("btn_red", "PAUSE");
15 :         jsLog.lgmethod('video.play(1)');
16 :         break;
17 :
18 :      case 2 :
19 :         video.play(0);
20 :          setInnerTextById("btn_red", "PLAY");
21 :          jsLog.lgmethod('video.play(0)');
22 :          break;
23 :
24 :      case 3 :
25 :          video.play(1);
26 :          setInnerTextById("btn_red", "SEEK");
27 :          jsLog.lgmethod('video.play(1)');
28 :          break;
29 :
30 :      case 4 :
31 :          video.seek(60 * 1000);
32 :          setInnerTextById("btn_red", "STOP");
33 :          jsLog.lgmethod('video.seek(time) ');
34 :          break;
35 :
36 :      case 5 :
37 :          video.stop();
38 :          setInnerTextById("btn_red", "Press RED-key to play video.");
39 :          jsLog.lgmethod('video.stop()');
40 :          break;
41 :   }
42 :   testProcess++;
43 :   if(testProcess==6) testProcess=1;
44 : }
```

Displaying Text

The following code displays the media object information by using properties and methods.

setPlayTimeInfo

This function displays playtime information on screen.
05: Declare playInfo and call mediaPlayInfo() function to save media play information into playInfo.
06-07: Display currentPosition, duration, and bufRemain saved in playInfo.
11: Display the values of playtime and playPosition properties.

setBufferingInfo

This function displays buffering information on screen.
16: Declare video.
19: Declare playInfo and call mediaPlayInfo() function to save media play information into playInfo.
21-24: Display the values of buffering properties saved in playInfo.

```
01 : function setPlayTimeInfo()
02 : {
03 :    jsLog.lgmethod('mediaPlayInfo()');
04 :
05 :    var playInfo = document.getElementById("video").mediaPlayInfo();
06 :    setInnerTextById("currentPosition_duration_value", "currentPosition/duration :
" + playInfo.currentPosition + "/" + playInfo.duration);
07 :    setInnerTextById("buf_remain_value", "bufRemain : " + playInfo.bufRemain);
08 :
09 :    jsLog.lgproperty('playTime : ' + video.playTime);
10 :    jsLog.lgproperty('playPosition : '+ video.playPosition);
11 :    setInnerTextById("playPosition_playTime_value", "playPosition/playTime : " +
video.playPosition + "/" + video.playTime);
12 : }
13 :
14 : function setBufferingInfo()
15 : {
16 :    var video = document.getElementById("video");
17 :
18 :    jsLog.lgmethod('mediaPlayInfo()');
19 :    var playInfo = video.mediaPlayInfo();
20 :
21 :    setInnerTextById("buf_Begin_end_value", "bufBegin~bufEnd : " +
playInfo.bufBegin + "~" + playInfo.bufEnd);
22 :    setInnerTextById("buf_remain_value", "bufRemain : " + playInfo.bufRemain);
23 :    setInnerTextById("buf_progress_value", "bufferingProgress : " +
video.bufferingProgress + "%");
24 :    setInnerTextById("bitrate_value", "bitrateInstant/bitrateTarget : " +
playInfo.bitrateInstant + "/" + playInfo.bitrateTarget);
25 : }
```

Inputting Keys

Use **onUserInput** function is called by onClickHandler function; it receives a key value as the userInput parameter from onClickHandler and creates the corresponding function for each key value to operate the key.

06: When the Back key is pressed, this code is executed.
07: When the Red key is pressed, this code is executed.
08: When the Green key is pressed, this code is executed.

```
01 : <//onUserInput function should be implemented
02 : function onUserInput(userInput)
03 : {
04 :    switch(userInput)
05 :    {
06 :       case VK_BACK : window.location.replace("../menu_mediaPlayer.html"); break;
07 :       case VK_RED : case 82 : processRedKey(); break;
08 :       case VK_GREEN : case 71 : window.location.replace("./mediaplayer2.html");
break;
09 :    }
10 : }
```

Setting Media Object

The following code shows how to set Media object.

03: Set data type. Refer to App Development Guide.
04: Set the media file playout to be started automatically.
05: Set drm type.
06: Set preBufferingTime.
07-08: Set width and height.

```
01 : <object
02 :    id="video"
03 :    type="application/x-netcast-av"
04 :    autoStart="true"
05 :    drm_type="wm-drm"
06 :    preBufferingTime = 5
07 :    width=300
08 :    height=250
09 :    data="../../mediafile/timer.mp4"
10 :    style="float: left">
11 : </object>
```

Source Code of mediaplayer.html

Source code of mediaplayer.html is as follows:

```
<!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=UTF-8">
<title>Windows Media Player API Test Page(1/3)</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/common.js"></script>
<script language="javascript" src="../../js/menu.js"></script>
<script language="javascript" src="../../js/media.js"></script>

<script type="text/javascript" src="../../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../../blackbirdjs/blackbird.css" />

<script>

   //initialize page
   var buffertimer;
   var playtimer;

   function initPage()
   {
       //save page as last visited page
       setLastVisitPage();

       //common initialize function
       commonInitialize();
       requestSourceCode();
       setPageID("Media Player");
       jsLog.initLG();

       //add onclick event handler
       addEventHandler(document.getElementById("btn_back"), "click", onClickHandler);
       addEventHandler(document.getElementById("btn_red"), "click", onClickHandler);
       addEventHandler(document.getElementById("btn_green"), "click", onClickHandler);
       addEventHandler(document.getElementById("btn_exit"), "click", onClickHandler);

       setInnerTextById("btn_red", "PAUSE");

      //add event handler for test
      var video = document.getElementById("video");
      //add onReadyStateChange event handler
       video.onReadyStateChange = setReadyState;

      //add playState event handler
      video.onBuffering = processBufferingStateChangeFunction;
      video.onPlayStateChange = processPlayStateChangeFunction;

      jsLog.lgobject('application/x-netcast-av');
      jsLog.lgproperty('autoStart');
      jsLog.lgproperty('drm_type');
      jsLog.lgproperty('preBufferingTime');
   }

   function setReadyState(readyState)
   {
       var video = document.getElementById("video");
       var readyStateValue = video.readyState;
       var readyStateDesc = "";

       if(readyStateValue == 0){readyStateDesc = "0 (not set)";}
       else if(readyStateValue == 1){readyStateDesc = "1 (loading)";}
       else if(readyStateValue == 3){readyStateDesc = "3 (loaded enough)";}
       else if(readyStateValue == 4){readyStateDesc = "4 (loaded all)";}
       else{readyStateDesc = readyStateValue + " (unknown state)";}

       setInnerTextById("ready_state_value", "readyState : " + readyStateDesc);
       jsLog.lgevent('onReadyStateChange : readyState = ' + readyState);
   }

   function processBufferingStateChangeFunction(isStarted)
   {
       jsLog.lgevent('onBuffering : isStart = ' + isStarted);
       setBufferingInfo();

       if(isStarted)
       {
          buffertimer = setInterval(setBufferingInfo, 400);
          jsLog.lgmethod('video.mediaPlayInfo()');
          jsLog.lgproperty('bufferingProgress');
       }
       else
       {
          clearInterval(buffertimer);
       }
   }

   function processPlayStateChangeFunction()
   {
       jsLog.lgevent('onPlayStateChange');

       var playStateExplain = "";
       var video = document.getElementById("video");
       var playState = video.playState;

       jsLog.lgproperty('playState :' + playState);
       jsLog.lgproperty('speed : ' + video.speed);
       setInnerTextById("speed_value", "speed : " + video.speed);

       if(playState == 0){playStateExplain = "stopped";}
       else if(playState == 1){playStateExplain = "playing";}
       else if(playState == 2){playStateExplain = "paused";}
       else if(playState == 3){playStateExplain = "connecting";}
       else if(playState == 4){playStateExplain = "buffering";}
       else if(playState == 5){playStateExplain = "finished";}
       else if(playState == 6){playStateExplain = "error";}

       setInnerTextById("play_state_value", "playState : " + playState + "(" +
playStateExplain + ")");

       if(playState == 1 ){
          playtimer = setInterval(setPlayTimeInfo,2000);
       }
       else {
          clearInterval(playtimer);
       }
   }

   var testProcess = 2;
   function processRedKey()
   {
       var video = document.getElementById("video");
       switch(testProcess)
       {
          case 0 :
             setInnerTextById("btn_red", "Press RED-key to play video.");

          case 1 :
             video.play(1);
             setInnerTextById("btn_red", "PAUSE");
             jsLog.lgmethod('video.play(1)');
             break;

          case 2 :
             video.play(0);
             setInnerTextById("btn_red", "PLAY");
             jsLog.lgmethod('video.play(0)');
             break;

          case 3 :
             video.play(1);
             setInnerTextById("btn_red", "SEEK");
             jsLog.lgmethod('video.play(1)');
             break;

          case 4 :
             video.seek(60 * 1000);
             setInnerTextById("btn_red", "STOP");
             jsLog.lgmethod('video.seek(time) ');
             break;

          case 5 :
             video.stop();
             setInnerTextById("btn_red", "Press RED-key to play video.");
             jsLog.lgmethod('video.stop()');
             break;
          }
          testProcess++;
          if(testProcess==6) testProcess=1;
   }

   function setPlayTimeInfo()
   {
       jsLog.lgmethod('mediaPlayInfo()');
       var playInfo = document.getElementById("video").mediaPlayInfo();
       setInnerTextById("currentPosition_duration_value", "currentPosition/duration :
" + playInfo.currentPosition + "/" + playInfo.duration);
       setInnerTextById("buf_remain_value", "bufRemain : " + playInfo.bufRemain);

       jsLog.lgproperty('playTime : ' + video.playTime);
       jsLog.lgproperty('playPosition : '+ video.playPosition);
       setInnerTextById("playPosition_playTime_value", "playPosition/playTime : " +
video.playPosition + "/" + video.playTime);
   }

   function setBufferingInfo()
   {
       var video = document.getElementById("video");

       jsLog.lgmethod('mediaPlayInfo()');
       var playInfo = video.mediaPlayInfo();
```

```
        setInnerTextById("buf_Begin_end_value", "bufBegin~bufEnd : " +
playInfo.bufBegin + "~" + playInfo.bufEnd);
        setInnerTextById("buf_remain_value", "bufRemain : " + playInfo.bufRemain);
        setInnerTextById("buf_progress_value", "bufferingProgress : " +
video.bufferingProgress + "%");
        setInnerTextById("bitrate_value", "bitrateInstant/bitrateTarget : " +
playInfo.bitrateInstant + "/" + playInfo.bitrateTarget);
    }

    //onUserInput function should be implemented
    function onUserInput(userInput)
    {
        switch(userInput)
        {
            case VK_BACK : window.location.replace("../menu_mediaPlayer.html"); break;
            case VK_RED : case 82 : processRedKey(); break;
            case VK_GREEN : case 71 : window.location.replace("./mediaplayer2.html");
break;
        }
    }

</script>
</head>

<body ondragstart='return false' onselectstart='return false'
onload="javascript:initPage();">

<!-- title -->
<div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>

<!-- navigation -->
<div class='SuiteNavigation'>
    <div style="float:left;">File : mediaplayer/app/mediaplayer.html</div>
</div>

<div class='SuiteTitleLine'> </div>

<!-- test contents -->
<div id='content_body' class='ContentArea'>

    <div class='ApiListTitleArea'>Web Open API List</div>
    <div class='ApiListArea'>
        <div class='MethodTitleArea'>
            Methods
            <div class='MethodListArea'>
                play(0)<br>
                play(1)<br>
                stop()<br>
                seek(time)<br>
                mediaPlayInfo()
            </div>
        </div>
        <div class='PropertyTitleArea'>
            Properties
            <div class='PropertyListArea'>
                playTime<br>
                playPosition<br>
                bufferingProgress<br>
                speed<br>readyState<br>
```

```
             playState<br>
             autoStart<br>
             drm_type<br>
             preBufferingTime
          </div>
       </div>
       <div class='EventTitleArea'>
          Events
          <div class='EventListArea'>
             onPlayStateChange<br>
             onReadyStateChange<br>
             onBuffering<br>

          </div>
       </div>
   </div>

   <div class='ViewTitleArea'>
       <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
       <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
   </div>

   <div id='view'>
       <div class='ViewArea'>
       <object
          id="video"
          type="application/x-netcast-av"
          autoStart="true"
          drm_type="wm-drm"
          preBufferingTime = 5
          width=300
          height=250
          data="../../mediafile/timer.mp4"
          style="float: left">
       </object>

       <table border="0" cellpadding="0" cellspacing="0" style="position: relative;
left: 10px;    width:450px; height:250px;">
          <tr height="11%" >
             <td ><div class="eachTestGuide " id="ready_state_value"
>readyState :</div></td>
          </tr>
          <tr height=11%>
             <td ><div class="eachTestGuide " id="speed_value">speed :</div></td>
          </tr>
          <tr height="11%">
            <td ><div class="eachTestGuide "
id="currentPosition_duration_value">currentPosition/duration :</div></td>
          </tr>
          <tr height="11%">
            <td><div class="eachTestGuide "
id="playPosition_playTime_value">playPosition/playTime :</div></td>
          </tr>
          <tr height="11%">
             <td ><div class="eachTestGuide " id="buf_Begin_end_value">bufBegin -
bufEnd :</div></td>
          </tr>
          <tr>
```

```
              <td><div class="eachTestGuide "
id="play_state_value">playState :</div></td>
          </tr>
          <tr height="11%">
              <td><div class="eachTestGuide "
id="buf_remain_value">bufRemain :</div></td>
          </tr>
          <tr height="11%">
              <td><div class="eachTestGuide "
id="buf_progress_value">bufferingProgress :</div></td>
          </tr>
          <tr height="11%">
              <td><div class="eachTestGuide "
id="bitrate_value">bitrateInstant/bitrateTarget :</div></td>
          </tr>
       </table>
       </div>
   </div>

   <div style="visibility: hidden" id='codeview'>
       <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
   </div>

</div>


<!-- button and copyright -->
<div class='SuiteButtonArea'>

   <!-- button -->
   <div id='btn_back' class='buttonDescription'>BACK</div>

   <!-- exit key description -->
   <div id='btn_exit' class='buttonDescription'>EXIT</div>

   <!-- red key description -->
   <div id='btn_red' class='buttonDescription redColor'>PLAY</div>

   <!-- green key description -->
   <div id='btn_green' class='buttonDescription greenColor'>NEXT PAGE</div>

   <!-- copyright -->
   <div class='copyright'>Copyright LG Electronics</div>
</div>

</body>
</html>
```

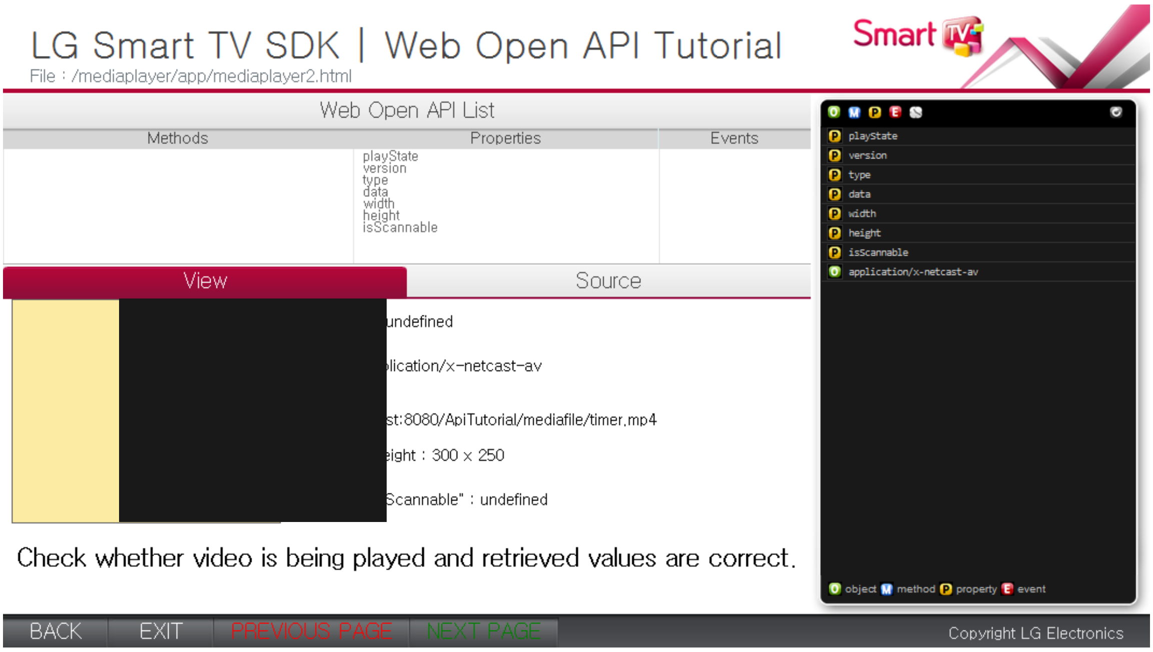
Executing Windows Media Player 2

This section describes how to execute the windows media player in web application.

- Initializing the Page
- Inputting Keys
- Displaying Text
- Setting Media Object
- Source Code of mediaplayer2.html

Following the “LG Web_Executing Window Media Play #1”, this application shows which properties of LG Web Open API are used to get media information when executing media player in LG Smart TV.

This application shows how to get and set property values.



File : /mediaplayer/app/mediaplayer2.html

```
undefined
lication/x-netcast-av
st:8080/ApiTutorial/mediafile/timer.mp4
eight : 300 x 250
Scannable" : undefined
```

Check whether video is being played and retrieved values are correct.

Needed APIs

This application uses following Web Open API:

API Class	Name	Description
Property	playState	Returns current media play status by number.
	version	Returns media version by String type.
	type	Returns media type by String type.
	data	Returns media URL by String type.
	width / height	Returns media object size by String type.
	isScannable	If returned value is true, fast forward or rewind is available.

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

04: Record the last visited page when running the application.
07: Initialize the page.
08: Get the source code of the page using the XMLHttpRequest object.
09: Set the page ID.
10: Initialize the Log function.

13-16: Add an event handler which will executed when the corresponding button is pressed.
18: Call function that displays property values on screen.

```
01 : function initPage()
02 : {
03 :   //save page as last visited page
04 :   setLastVisitPage();
05 :
06 :   //common initialize function
07 :   commonInitialize();
08 :   requestSourceCode();
09 :   setPageID("Media Player");
10 :   jsLog.initLG();
11 :
12 :   //add onclick event handler
13 :   addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
14 :   addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
15 :   addEventHandler(document.getElementById("btn_green"),"click",onClickHandler);
16 :   addEventHandler(document.getElementById("btn_exit"),"click",onClickHandler);
17 :
18 :   processTest();
19 :
20 :   jsLog.lgobject('application/x-netcast-av');
21 : }
```

Inputting Keys

Use **onUserInput** function is called by onClickHandler function; it receives a key value as the userInput parameter from onClickHandler and creates the corresponding function for each key value to operate the key.

06: When the Back key is pressed, this code is executed.
07: When the Red key is pressed, this code is executed.
08: When the Green key is pressed, this code is executed.

```
01 : function onUserInput(userInput)
02 : {
03 :  switch(userInput)
04 :  {
05 :    case VK_BACK : window.location.replace("../menu_mediaPlayer.html"); break;
06 :   case VK_RED : case 82 : window.location.replace("./mediaplayer.html"); break;
07 :    case VK_GREEN : case 71 : window.location.replace("./mediaplayer2.html");
break;
08 :  }
09 : }
```

Displaying Text

The following code displays the media object information by using properties.

processTest

This function displays media property values on screen.
04: Declare video.
08-13: If media is not playing, call processTest function at intervals of 4000 milliseconds.
16-34: Display media property values on screen.

```
01 : var oldPlayState;
02 : function processTest()
03 : {
04 :   var video = document.getElementById("video");
05 :   jsLog.lgproperty('playState');
06 :
07 :   //check if video is now being played
```

```
    if((video.playState != 1) && (oldPlayState != video.playState))
    {
        oldPlayState = video.playState;
        setInterval(processTest, 4000);
        return;
    }

    //version
    jsLog.lgproperty('version');
    setInnerTextById("version_value", "1. version : " + video.version);

    //type
    jsLog.lgproperty('type');
    setInnerTextById("type_value", "2. type : " + video.type);

    //data
    jsLog.lgproperty('data');
    setInnerTextById("data_value", "3. data : " + video.data);

    //width x height
    jsLog.lgproperty('width');
    jsLog.lgproperty('height');
    setInnerTextById("width_height_value", "4. width x height : " + video.width + " x " + video.height);

    //isScannable
    jsLog.lgproperty('isScannable');
    setInnerTextById("isScannable_value", "6. Check \"isScannable\" : " + video.isScannable);
}
```

Setting Media Object

The following code shows how to set Media object.

03: Set data type. Refer to App Development Guide.
04-05: Set width and height.
06: Set the path of media file which will be played.

```
01 : <object
02 :   id="video"
03 :   type="application/x-netcast-av"
04 :   width=300
05 :   height=250
06 :   data="/ApiTutorial/mediafile/timer.mp4"
07 :   style="float: left">
08 : </object>
```

Source Code of mediaplayer2.html

Source code of mediaplayer2.html is as follows:

```
!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=UTF-8">
<title>Windows Media Player API Test Page(2/3)</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/common.js"></script>
<script language="javascript" src="../../js/menu.js"></script>
```

```
<script language="javascript" src="../../js/media.js"></script>

<script type="text/javascript" src="../../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../../blackbirdjs/blackbird.css" />

<script>

    //initialize page
    function initPage()
    {
        //save page as last visited page
        setLastVisitPage();

        //common initialize function
        commonInitialize();
        requestSourceCode();
        setPageID("Media Player");
        jsLog.initLG();

        //add onclick event handler
        addEventHandler(document.getElementById("btn_back"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_red"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_green"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_exit"), "click", onClickHandler);

        processTest();

        jsLog.lgobject('application/x-netcast-av');
    }

    //onUserInput function should be implemented
    function onUserInput(userInput)
    {
        switch(userInput)
        {
            case VK_BACK : window.location.replace("../menu_mediaPlayer.html"); break;
            case VK_RED : case 82 : window.location.replace("./mediaplayer.html");
break;
            case VK_GREEN : case 71 : window.location.replace("./mediaplayer3.html");
break;
        }
    }

    var oldPlayState;
    function processTest()
    {
        var video = document.getElementById("video");
        jsLog.lgproperty('playState');

        //check if video is now being played
        if((video.playState != 1) && (oldPlayState != video.playState))
        {
            oldPlayState = video.playState;
            setInterval(processTest, 4000);
            return;
        }

        //version
        jsLog.lgproperty('version');
        setInnerTextById("version_value", "1. version : " + video.version);
        //type
        jsLog.lgproperty('type');
        setInnerTextById("type_value", "2. type : " + video.type);

        //data
        jsLog.lgproperty('data');
        setInnerTextById("data_value", "3. data : " + video.data);

        //width x height
        jsLog.lgproperty('width');
        jsLog.lgproperty('height');
        setInnerTextById("width_height_value", "4. width x height : " + video.width + "
x " + video.height);

        //isScannable
        jsLog.lgproperty('isScannable');
        setInnerTextById("isScannable_value", "6. Check \"isScannable\" : " +
video.isScannable);
    }

</script>
</head>


<body ondragstart='return false' onselectstart='return false'
onload="javascript:initPage();">

<!-- title -->
<div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>

<!-- navigation -->
<div class='SuiteNavigation'>
    <div style="float:left;">File : /mediaplayer/app/mediaplayer2.html</div>
</div>

<div class='SuiteTitleLine'> </div>

<!-- test contents -->
<div class='ContentArea'>
    <div class='ApiListTitleArea'>Web Open API List</div>
    <div class='ApiListArea'>
        <div class='MethodTitleArea'>
            Methods
            <div class='MethodListArea'>

            </div>
        </div>
        <div class='PropertyTitleArea'>
            Properties
            <div class='PropertyListArea'>
                playState<br>
                version<br>
                type<br>
                data<br>
                width<br>
                height<br>
                isScannable<br>
            </div>
        </div>
```

```
      <div class='EventTitleArea'>
         Events
         <div class='EventListArea'>

         </div>
      </div>
   </div>

   <div class='ViewTitleArea'>
      <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
      <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
      </div>

   <div id='view'>
      <div class='ViewArea'>
      <!-- video object -->
         <object
            id="video"
            type="application/x-netcast-av"
            width=300
            height=250
            data="/ApiTutorial/mediafile/timer.mp4"
            style="float: left";>
         </object>

         <table border="0" cellpadding="0" cellspacing="0" style="position: relative;
left: 10px; width:450px; height:250px;">
            <tr height="40px">
               <td width="100%" colspan="2"><div class="eachTestGuide "
id="version_value">1. version :</div></td>
            </tr>
            <tr height="40px">
               <td width="100%"><div class="eachTestGuide " id="type_value">2.
type :</div></td>
            </tr>
            <tr height="40px">
               <td width="100%"><div class="eachTestGuide " id="data_value">3.
data :</div></td>
            </tr>
            <tr height="40px">
               <td width="100%"><div class="eachTestGuide "
id="width_height_value">4. width x height :</div></td>
            </tr>
            <tr height="40px">
               <td width="100%"><div class="eachTestGuide " id="isScannable_value">5.
Check &quot;isScannable&quot; :</div></td>
             </tr>
         </table>
      </div>
   </div>

   <div style="visibility: hidden" id='codeview'>
      <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
   </div>

</div>
<!-- description -->
<div id='testdescription' class='SuiteDescription' style="position: absolute; left:-
```

```
50px; top:600px">Check whether video is being played and retrieved values are
correct.</div>

<!-- button and copyright -->
<div class='SuiteButtonArea'>
   <!-- back key description -->
   <div id='btn_back' class='buttonDescription '>BACK</div>

   <!-- exit key description -->
   <div id='btn_exit' class='buttonDescription'>EXIT</div>

   <!-- red key description -->
   <div id='btn_red' class='buttonDescription redColor'>PREVIOUS PAGE</div>

   <!-- green key description -->
   <div id='btn_green' class='buttonDescription greenColor'>NEXT PAGE</div>

   <!-- copyright -->
   <div class='copyright'>Copyright LG Electronics</div>
</div>

</body>
</html>
```

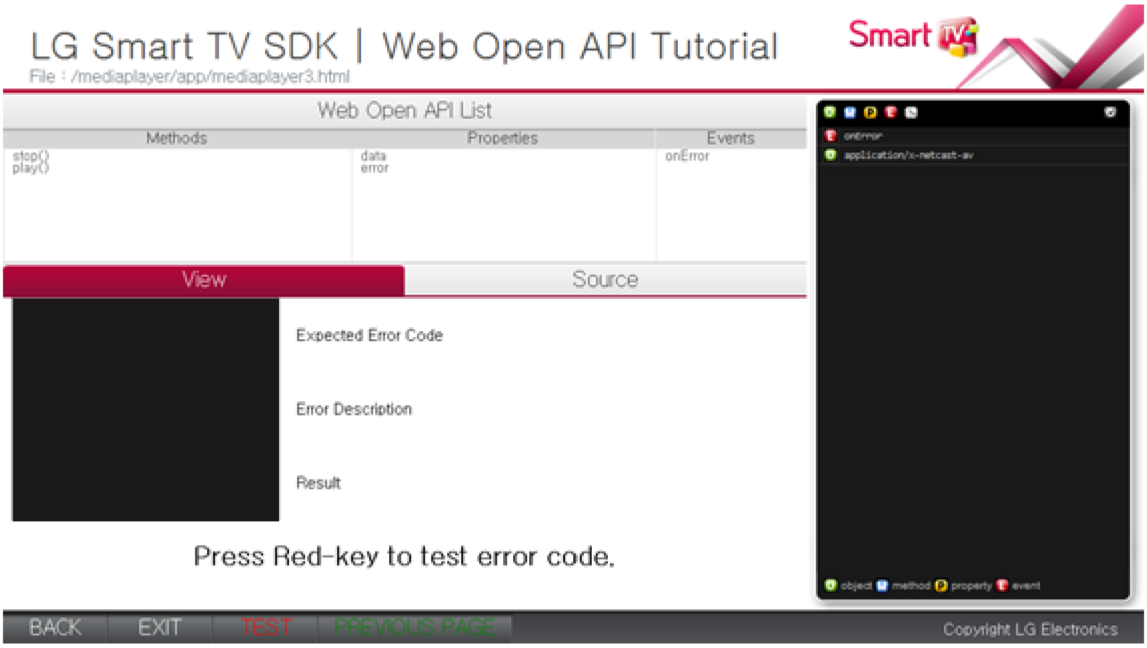
Executing Windows Media Player 3

This section describes how to activate and deactivate browser page loading icon.

- Initializing the Page
- Inputting Keys
- Displaying Text
- Setting Media Object
- Source Code of mediaplayer3.html

Following the “LG Web_Executing Window Media Player #2”, this application is designed to show which method, properties, and events of LG Web Open API are used to get basic information when executing media player in LG Smart TV.

This application shows how to get property values.



LG Smart TV SDK | Web Open API Tutorial

File : /mediaplayer/app/mediaplayer3.html

Web Open API List

Methods	Properties	Events

stop()
play()
	data
	error
		onError

Expected Error Code

Error Description

Result

Press Red-key to test error code.

onError

application/x-netcast-av

Needed APIs
This application uses following Web Open API:

API Class	Name	Description
Method	play()	Plays media.
	stop()	Stops media.
Property	data	Returns media URL with String type.
	error	Returns error code if error occurs during media play.
Event	onError	The event occurs when an error occurs during media play.
	play()	Plays media.

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

04: Record the last visited page when running the application.
07: Initialize the page.
08: Get the source code of the page using the XMLHttpRequest object.
09: Set the page ID.

10: Initialize the Log function.
13-16: Add an event handler which will executed when the corresponding button is pressed.
18: Call function that displays property values on screen.
21: Add onError event handler of media.

```
01 : function initPage()
02 : {
03 :   //save page as last visited page
04 :   setLastVisitPage();
05 :
06 :   //common initialize function
07 :   commonInitialize();
08 :   requestSourceCode();
09 :   setPageID("Media Player");
10 :   jsLog.initLG();
11 :
12 :   //add onclick event handler
13 :   addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
14 :   addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
15 :   addEventHandler(document.getElementById("btn_green"),"click",onClickHandler);
16 :   addEventHandler(document.getElementById("btn_exit"),"click",onClickHandler);
17 :
18 :   var video = document.getElementById("video");
19 :
20 :   //add onError handler
21 :   video.onError = processError;
22 :
23 :   jsLog.lgobject('application/x-netcast-av');
24 : }
```

Inputting Keys

Use **onUserInput** function is called by onClickHandler function; it receives a key value as the userInput parameter from onClickHandler and creates the corresponding function for each key value to operate the key.

06: When the Back key is pressed, this code is executed.
07: When the Red key is pressed, this code is executed.
08: When the Green key is pressed, this code is executed.

```
01 : function onUserInput(userInput)
02 : {
03 :  switch(userInput)
04 :  {
05 :    case VK_BACK : window.location.replace("../menu_mediaPlayer.html"); break;
06 :   case VK_RED : case 82 : window.location.replace("./mediaplayer.html"); break;
07 :    case VK_GREEN : case 71 : window.location.replace("./mediaplayer3.html");
break;
08 :  }
09 : }
```

Displaying Text

The following code displays the media object information by using events and properties.

Declaring and initializing variables
01: Declare array variable errorsToTest.
03-12: Set description for each error code.
14: Declare and initialize currentStep.

setNextStep

This function displays error code and message saved in each step.
15: Change currentStep value when this function is called.
16-20: When this function is called, the test is started and the button is set to next.

22: If the value of currentStep is bigger than the size of array the function ends.
24: Declare media.
25: Stop media.
27-29: Displays error code, description, and result error code of errorsToTest on screen.
30-31: Set text of test_result class.
33-37: If the value of currentStep equals the size of errorsToTest array, hide the Next button and displays description text on screen.
39-46: Set data (media source) which is got with currentStep to media. If there is data, media is played.

processError

This function is onError event handler; it displays corresponding error code on screen if event occurs.

53-54: Returns if video is not played.
56-57: Set error property of value to errorCode and displays returned number.

```
01 : var errorsToTest = new Array();
02 :
03 : errorsToTest[errorsToTest.length] = [0, "A/V format not supported",
getMediaFileUrl("samplevideo.flv")];
04 : errorsToTest[errorsToTest.length] = [1, "Cannot connect to server or connection
lost", "http://not.exist.xxx/ncts/mediafiles/NetCastGeneratorClient_H.264_MP3.mov"];
05 : errorsToTest[errorsToTest.length] = [1000, "File is not found",
getMediaFileUrl("not_exist.mp4")];
06 : errorsToTest[errorsToTest.length] = [1001, "Invalid protocol", "xxxx" +
getMediaFileUrl("timer.mp4").substring(4)];
07 : errorsToTest[errorsToTest.length] = [1003, "Play list is empty",
getMediaFileUrl("empty_list.asx")];
08 : errorsToTest[errorsToTest.length] = [1004, "Unrecognized play list",
getMediaFileUrl("pls_test.pls")];
09 : errorsToTest[errorsToTest.length] = [1005, "Invalid ASX format",
getMediaFileUrl("invalid_format.asx")];
10 :
11 : var currentStep = -1;
12 :
13 : function setNextStep()
14 : {
15 :   currentStep++;
16 :   if(currentStep == 0)
17 :   {
18 :       setInnerTextById("error_test_description", "Check whether expected error is
occurred.<br>Press Red-key to test next error code.");
19 :       setInnerTextById("btn_red", "NEXT");
20 :   }
21 :
22 :   if(currentStep >= errorsToTest.length){return;}
23 :
24 :   var media = document.getElementById("video");
25 :   media.stop();
26 :   jsLog.lgmethod('stop()');
27 :   setInnerTextById("error_code", errorsToTest[currentStep][0]);
28 :   setInnerTextById("error_desc", errorsToTest[currentStep][1]);
29 :   setInnerTextById("result_error_code", "");
30 :   document.getElementById("test_result").className = "";
31 :   setInnerTextById("test_result", "");
32 :
33 :   if(currentStep == (errorsToTest.length - 1))
34 :   {
35 :       setInnerTextById("error_test_description", "Check whether expected error is
occurred.<br>No more test is left on this page.");
36 :       document.getElementById("btn_red").style.visibility = "hidden";
37 :   }
38 :
39 :   media.data = errorsToTest[currentStep][2];
40 :   jsLog.lgproperty('data');
41 :
42 :   if(errorsToTest[currentStep][2] != "")
43 :   {
44 :      media.play();
45 :      jsLog.lgmethod('play()');
46 :   }
47 : }
48 :
49 : function processError()
50 : {
51 :   jsLog.lgevent('onError');
52 :
53 :   if(currentStep < 0 || currentStep >= errorsToTest.length)
54 :      return;
55 :
56 :   var errorCode = document.getElementById("video").error;
57 :   setInnerTextById("result_error_code", errorCode);
58 : }
```

Setting Media Object

The following code shows how to set Media object.

03: Set data type. Refer to App Development Guide.
04-05: Set width and height.
06: Set the path of media file which will be played.

```
01 : <object
02 :   id="video"
03 :   type="application/x-netcast-av"
04 :   width=300
05 :   height=250
06 :   data="/ApiTutorial/mediafile/timer.mp4"
07 :   style="float: left">
08 : </object>
```

Source Code of mediaplayer3.html

Source code of mediaplayer3.html is as follows:

```
<!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=UTF-8">
<title>Windows Media Player API Test Page(3/3)</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/common.js"></script>
<script language="javascript" src="../../js/menu.js"></script>
<script language="javascript" src="../../js/media.js"></script>

<script type="text/javascript" src="../../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../../blackbirdjs/blackbird.css" />

<script>

   //initialize page
   function initPage()
   {
      //save page as last visited page
      setLastVisitPage();

      //common initialize function
      commonInitialize();
      requestSourceCode();
      setPageID("Media Player");
      jsLog.initLG();

      //add onclick event handler
      addEventHandler(document.getElementById("btn_back"), "click", onClickHandler);
      addEventHandler(document.getElementById("btn_red"), "click", onClickHandler);
      addEventHandler(document.getElementById("btn_green"), "click", onClickHandler);
      addEventHandler(document.getElementById("btn_exit"), "click", onClickHandler);

      var video = document.getElementById("video");

      //add onError handler
      video.onError = processError;

      jsLog.lgobject('application/x-netcast-av');
   }

   //onUserInput function should be implemented
   function onUserInput(userInput)
   {
      switch(userInput)
      {
         case VK_BACK : window.location.replace("../menu_mediaPlayer.html"); break;
         case VK_RED : case 82 : setNextStep(); break;
         case VK_GREEN : case 71 : window.location.replace("./mediaplayer2.html");
break;
      }
   }

   var errorsToTest = new Array();

   errorsToTest[errorsToTest.length] = [0, "A/V format not supported",
getMediaFileUrl("samplevideo.flv")];
   errorsToTest[errorsToTest.length] = [1, "Cannot connect to server or connection
lost", "http://not.exist.xxx/ncts/mediafiles/NetCastGeneratorClient_H.264_MP3.mov"];
   errorsToTest[errorsToTest.length] = [1000, "File is not found",
getMediaFileUrl("not_exist.mp4")];
   errorsToTest[errorsToTest.length] = [1001, "Invalid protocol", "xxxx" +
getMediaFileUrl("timer.mp4").substring(4)];
   errorsToTest[errorsToTest.length] = [1003, "Play list is empty",
getMediaFileUrl("empty_list.asx")];
   errorsToTest[errorsToTest.length] = [1004, "Unrecognized play list",
getMediaFileUrl("pls_test.pls")];
   errorsToTest[errorsToTest.length] = [1005, "Invalid ASX format",
getMediaFileUrl("invalid_format.asx")];

   var currentStep = -1;

   function setNextStep()
   {
      currentStep++;
      if(currentStep == 0)
      {
         setInnerTextById("error_test_description", "Check whether expected error is
occurred.<br>Press Red-key to test next error code.");
         setInnerTextById("btn_red", "NEXT");
      }

      if(currentStep >= errorsToTest.length){return;}

      var media = document.getElementById("video");
      media.stop();
      jsLog.lgmethod('stop()');
      setInnerTextById("error_code", errorsToTest[currentStep][0]);
      setInnerTextById("error_desc", errorsToTest[currentStep][1]);
      setInnerTextById("result_error_code", "");
      document.getElementById("test_result").className = "";
      setInnerTextById("test_result", "");

      if(currentStep == (errorsToTest.length - 1))
      {
         setInnerTextById("error_test_description", "Check whether expected error is
occurred.<br>No more test is left on this page.");
         document.getElementById("btn_red").style.visibility = "hidden";
      }

      media.data = errorsToTest[currentStep][2];
      jsLog.lgproperty('data');

      if(errorsToTest[currentStep][2] != "")
      {
         media.play();
         jsLog.lgmethod('play()');
      }
   }

   function processError()
   {
      jsLog.lgevent('onError');

      if(currentStep < 0 || currentStep >= errorsToTest.length)
         return;

      var errorCode = document.getElementById("video").error;
      setInnerTextById("result_error_code", errorCode);
      jsLog.lgproperty('video.error');
   }

</script>
</head>


<body ondragstart='return false' onselectstart='return false'
onload="javascript:initPage();">

<!-- title -->
<div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>

<!-- navigation -->
<div class='SuiteNavigation'>
   <div style="float:left;">File : /mediaplayer/app/mediaplayer3.html</div>
</div>
```

```
<div class='SuiteTitleLine'> </div>

<!-- test contents -->
<div class='ContentArea ' id="content_body">
   <div class='ApiListTitleArea'>API List</div>
   <div class='ApiListArea'>
      <div class='MethodTitleArea'>
         Methods
         <div class='MethodListArea'>
            stop()<br>
            play()<br>
         </div>
      </div>
      <div class='PropertyTitleArea'>
         Properties
         <div class='PropertyListArea'>
            data<br>
            error
         </div>
      </div>
      <div class='EventTitleArea'>
         Events
         <div class='EventListArea'>
            onError<br>

         </div>
      </div>
   </div>

   <div class='ViewTitleArea'>
      <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
      <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
   </div>

   <div id='view'>
      <div class='ViewArea'>
         <object
            id="video"
            type="application/x-netcast-av"
            width=300
            height=250
            data="/ApiTutorial/mediafile/timer.mp4"
            style="float: left";>
         </object>

         <table id="temp_result" style="position: relative; left: 10px;  width:570px;
height:250px;" border="0" cellpadding="0" cellspacing="0">
            <tr height="80px">
               <td width="40%"><div class="eachTestGuide ">Expected Error
Code</div></td>
               <td width="60%" colspan="2"><div class="eachTestGuide "
id="error_code"></div></td>
            </tr>
            <tr height="80px">
               <td width="30%"><div class="eachTestGuide ">Error
Description</div></td>
               <td width="60%" colspan="2"><div class="eachTestGuide "
id="error_desc"></div></td>
```

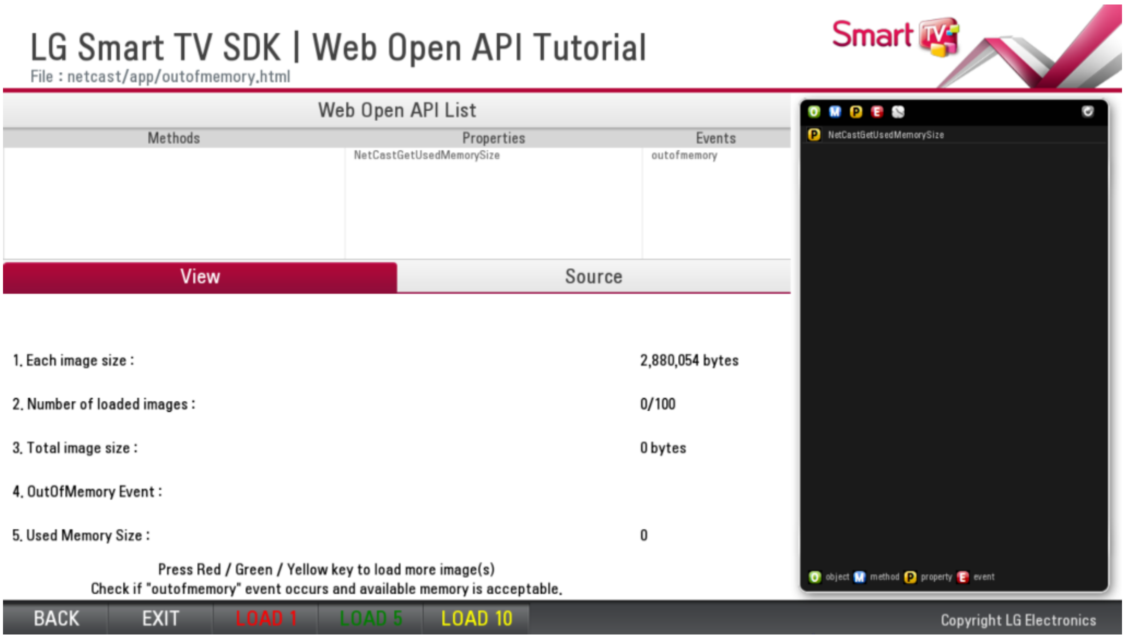
```
                </tr>
                <tr height="80px">
                    <td width="30%"><div class="eachTestGuide ">Result</div></td>
                    <td width="30%"><div class="eachTestGuide "
id="result_error_code"></div></td>
                    <td width="40%"><div id="test_result" style="float:left;"></div></td>
                </tr>
            </table>
        </div>
    </div>

    <div style="visibility: hidden" id='codeview'>
        <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
    </div>
</div>

<!-- description -->
<div class='SuiteDescriptionTwoLine ' id="error_test_description" style="position:
absolute; left:-50px; top:600px">Press Red-key to test error code.</div>

<!-- button and copyright -->
<div class='SuiteButtonArea'>
    <!-- back key description -->
    <div id='btn_back' class='buttonDescription '>BACK</div>

    <!-- exit key description -->
    <div id='btn_exit' class='buttonDescription'>EXIT</div>

    <!-- red key description -->
    <div id='btn_red' class='buttonDescription redColor'>TEST</div>

    <!-- green key description -->
    <div id='btn_green' class='buttonDescription greenColor'>PREVIOUS PAGE</div>

    <!-- copyright -->
    <div class='copyright '>Copyright LG Electronics</div>
</div>

</body>
</html>
```

Generating Outofmemory Event

This section describes how to generate outofmenory event in web application.

- Initializing the Page
- Inputting Keys
- Loadind Images
- Displaying Information of Loaded Image and Event
- Handling Events
- Displaying usedMemorySize
- Source Code of outofmemory.html

This application generates outofmemory event by loading image and increasing memory size.
You can know how to generate outofmemory event and handle it through this application.



LG Smart TV SDK | Web Open API Tutorial

File : netcast/app/outofmemory.html

Web Open API List

Methods	Properties	Events
	NetCastGetUsedMemorySize	outofmemory

1. Each image size : 2,880,054 bytes
2. Number of loaded images : 0/100
3. Total image size : 0 bytes
4. OutOfMemory Event :
5. Used Memory Size : 0

Press Red / Green / Yellow key to load more image(s)
Check if "outofmemory" event occurs and available memory is acceptable.

Needed APIs
This application uses following Web Open API:

API Class	Function Name	Description
Method	N/A	N/A
Property	NetCastGetUsedMemorySize	Total memory size used by the browser process
Event	outofmemory	This event gives the remaining memory size to application author.

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

04: Records the last visited page when running the application.
07: Initializes the page.
08: Gets the source code of the page using the XMLHttpRequest object.
09: Sets the page ID.
10: Initializes the Log function.
13-17: Registers an event handler which will executed when the corresponding button is pressed.
19: Calls displayStatus function.
22: Registers an event handler which will be executed when the outofmemory event occurs.
24: Displays the memory size used by the application on screen.

```
01 : function initPage()
02 : {
03 :   //save page as last visited page
04 :   setLastVisitPage();
05 :
06 :   //common initialize function
07 :   commonInitialize();
08 :   requestSourceCode();
09 :   setPageID("Out Of Memory");
10 :   jsLog.initLG();
11 :
12 :   //add onclick event handler
13 :   addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
14 :   addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
15 :   addEventHandler(document.getElementById("btn_green"),"click",onClickHandler);
16 :   addEventHandler(document.getElementById("btn_yellow"),"click",onClickHandler);
17 :   addEventHandler(document.getElementById("btn_exit"),"click",onClickHandler);
18 :
19 :   displayStatus();
20 :
21 :   //add outofmemory event handler
22 :   addEventHandler(window, "outofmemory", outOfMemoryHandler);
23 :
24 :   getUsedMemorySize();
25 : }
```

Inputting Keys

The **onUserInput** function is called by the onClickHandler function; it receives a key value as the userInput parameter from onClickHandler and creates the corresponding function for each key value to operate the key.

06-07: When the Back key is pressed, this code is executed.
08: When the Red key is pressed, one image is loaded.
09: When the Green key is pressed, five images are loaded.
10: When the Yellow key is pressed, ten images are loaded.

```
01 : //onUserInput function should be implemented
02 : function onUserInput(userInput)
03 : {
04 :   switch(userInput)
05 :  {
06 :     case VK_BACK:
07 :        window.location.replace("../menu_netcast.html"); break;
08 :     case VK_RED : case 82 : loadMoreImages(1); break;
09 :     case VK_GREEN : case 71 : loadMoreImages(5); break;
10 :     case VK_YELLOW : case 89 : loadMoreImages(10); break;
11 :   }
12 : }
```

Loading Images

The following functions load the specified number of images.

loadMoreImages(numberToLoad)

This function is called when multiple images are loaded.
04: Calls loadOneImage() function as the number of images.

loadOneImage:

Loads one image.
20-21: Returns if there is no image to load
24-25: Specifies size of the image to load.

26-27: Specifies the position that the image is loaded on the screen.
30-35: Specifies the image to load
36: Creates element of div tag.

Increases number of images by 1.
48: Increases number of images by 1.
49: Calls displayStatus() function.

```
01 : function loadMoreImages(numberToLoad)
02 : {
03 :    for(var i = 0 ; i < numberToLoad ; i++)
04 :       loadOneImage();
05 : }
06 :
07 : var numberOfLoadedImages = 0;
08 : var EACH_IMG_SIZE = 2880054;
09 : var DEFAULT_LEFT = 0;
10 : var DEFAULT_TOP = 0;
11 : var DEFAULT_IMG_LEFT = 100;
12 : var DEFAULT_IMG_TOP = 500;
13 : var TOTAL_WIDTH = 900;
14 : var TOTAL_HEIGHT = 50;
15 : var MAX_COLUMN_INDEX = 50;
16 : var MAX_ROW_INDEX = 2;
17 :
18 : function loadOneImage()
19 : {
20 :    if(MAX_COLUMN_INDEX * MAX_ROW_INDEX == numberOfLoadedImages)
21 :       return;
22 :
23 :    //create div and set an image as background
24 :    var widthToSet = Math.floor(TOTAL_WIDTH / MAX_COLUMN_INDEX);
25 :    var heightToSet = Math.floor(TOTAL_HEIGHT / MAX_ROW_INDEX);
26 :    var leftToSet = Math.floor((TOTAL_WIDTH / MAX_COLUMN_INDEX) *
(numberOfLoadedImages % MAX_COLUMN_INDEX));
27 :    var topToSet = Math.floor((TOTAL_HEIGHT / MAX_ROW_INDEX) *
Math.floor(numberOfLoadedImages / MAX_COLUMN_INDEX));
28 :    var bgImgIndex = "";
29 :
30 :    if(String(numberOfLoadedImages).length == 1){bgImgIndex = "00";}
31 :    else if(String(numberOfLoadedImages).length == 2){bgImgIndex = "0";}
32 :    else if(String(numberOfLoadedImages).length == 3){bgImgIndex = "";}
33 :
34 :    bgImgIndex += String(numberOfLoadedImages);
35 :    var bgUrlToSet = getMediaFileUrl("memoryTestImages") + "lake-nature_" +
bgImgIndex + ".bmp";
36 :    var divToAppend = document.createElement("div");
37 :
38 :    divToAppend.style.position = "absolute";
39 :    divToAppend.style.width = (widthToSet - 2) + "px";
40 :    divToAppend.style.height = (heightToSet - 2) + "px";
41 :    divToAppend.style.border = "1px solid #000000";
42 :    divToAppend.style.left = leftToSet + "px";
43 :    divToAppend.style.top = topToSet + "px";
44 :    divToAppend.style.background = "url(" + bgUrlToSet + ")";
45 :    divToAppend.style.backgroundPosition = "-" + (DEFAULT_IMG_LEFT + leftToSet) +
"px -" + (DEFAULT_IMG_TOP + topToSet) + "px";
46 :    divToAppend.style.backgroundRepeat = "no-repeat";
47 :    document.getElementById("ViewArea").appendChild(divToAppend);
48 :    numberOfLoadedImages++;
49 :    displayStatus();
50 : }
```

Displaying Information of Loaded Image and Event

The following functions are for event handlers added by the initPage function.

displayStatus:

Displays information regarding the loaded image.
03 : Displays the number of loaded images on the screen.
05: Displays size of loaded images on the screen.

toCommaNotation(origString)

Inserts commas in numbers.
15-22: If the number of digits is greater than three, comma is inserted in number for each three digits, which is saved in resultString.

```
01 : function displayStatus()
02 : {
03 :   setInnerTextById("number_of_loaded_image_result", numberOfLoadedImages + "/" +
(MAX_COLUMN_INDEX * MAX_ROW_INDEX));
04 :   var totalSize = EACH_IMG_SIZE * numberOfLoadedImages;
05 :   setInnerTextById("total_image_size_result", toCommaNotation(String(totalSize))
+ " bytes");
06 : }
07 :
08 :
09 : function toCommaNotation(origString)
10 : {
11 :   var unitToSep = 3;
12 :   var resultString = "";
13 :   var processString = origString;
14 :
15 :   while(processString.length > unitToSep)
16 :   {
17 :       var cutIndex = processString.length - unitToSep;
18 :       var resultString = "," + processString.substring(cutIndex) + resultString;
19 :       processString = processString.substring(0, cutIndex);
20 :   }
21 :   resultString = processString + resultString;
22 :   return resultString;
23 : }
```

```
51 : function loadMoreImages(numberToLoad)
52 : {
53 :   for(var i = 0 ; i < numberToLoad ; i++)
54 :       loadOneImage();
55 : }
56 :
57 : var numberOfLoadedImages = 0;
58 : var EACH_IMG_SIZE = 2880054;
59 : var DEFAULT_LEFT = 0;
60 : var DEFAULT_TOP = 0;
61 : var DEFAULT_IMG_LEFT = 100;
62 : var DEFAULT_IMG_TOP = 500;
63 : var TOTAL_WIDTH = 900;
64 : var TOTAL_HEIGHT = 50;
65 : var MAX_COLUMN_INDEX = 50;
66 : var MAX_ROW_INDEX = 2;
67 :
68 : function loadOneImage()
```

```
{
   if(MAX_COLUMN_INDEX * MAX_ROW_INDEX == numberOfLoadedImages)
      return;

   //create div and set an image as background
   var widthToSet = Math.floor(TOTAL_WIDTH / MAX_COLUMN_INDEX);
   var heightToSet = Math.floor(TOTAL_HEIGHT / MAX_ROW_INDEX);
   var leftToSet = Math.floor((TOTAL_WIDTH / MAX_COLUMN_INDEX) * (numberOfLoadedImages % MAX_COLUMN_INDEX));
   var topToSet = Math.floor((TOTAL_HEIGHT / MAX_ROW_INDEX) * Math.floor(numberOfLoadedImages / MAX_COLUMN_INDEX));
   var bgImgIndex = "";

   if(String(numberOfLoadedImages).length == 1){bgImgIndex = "00";}
   else if(String(numberOfLoadedImages).length == 2){bgImgIndex = "0";}
   else if(String(numberOfLoadedImages).length == 3){bgImgIndex = "";}

   bgImgIndex += String(numberOfLoadedImages);
   var bgUrlToSet = getMediaFileUrl("memoryTestImages") + "lake-nature_" + bgImgIndex + ".bmp";
   var divToAppend = document.createElement("div");

   divToAppend.style.position = "absolute";
   divToAppend.style.width = (widthToSet - 2) + "px";
   divToAppend.style.height = (heightToSet - 2) + "px";
   divToAppend.style.border = "1px solid #000000";
   divToAppend.style.left = leftToSet + "px";
   divToAppend.style.top = topToSet + "px";
   divToAppend.style.background = "url(" + bgUrlToSet + ")";
   divToAppend.style.backgroundPosition = "-" + (DEFAULT_IMG_LEFT + leftToSet) + "px -" + (DEFAULT_IMG_TOP + topToSet) + "px";
   divToAppend.style.backgroundRepeat = "no-repeat";
   document.getElementById("ViewArea").appendChild(divToAppend);
   numberOfLoadedImages++;
   displayStatus();
}
```

Handling Events

The following function is for event handler, which handles outofmemory event.
outOfMemoryHandler is called when the outofmemory event occurs.

07: Increases eventCount by 1.
09-14: Saves counted value and event.available (size of usable memory) value to resultString and displays the values on the screen.

```
01 : var resultString = " ";
02 : var eventCount = 0;
03 :
04 : function outOfMemoryHandler()
05 : {
06 :   jsLog.lgevent('outofmemory');
07 :   eventCount++;
08 :
09 :   if(eventCount == 1)
10 :      resultString += eventCount + " - " + event.available + " MB";
11 :   else
12 :      resultString += " / " + eventCount + " - " + event.available + " MB";
13 :
14 :   setInnerTextById("out_of_memory_event_occurred", resultString);
15 : }
```

Displaying usedMemorySize

The following function is for displaying the: memory size

01: Set usedMemorySize.
04-09: If the return value of NetCastGetUsedMemorySize exists, display the value.

```
01 : var usedMemorySize;
02 : function getUsedMemorySize()
03 : {
04 :    if(window.NetCastGetUsedMemorySize)
05 :    {
06 :        usedMemorySize = window.NetCastGetUsedMemorySize();
07 :        setInnerTextById("used_memory_result", usedMemorySize);
08 :        jsLog.lgproperty('NetCastGetUsedMemorySize');
09 :    }
10 : }
```

Source Code of outofmemory.html

Source code of outofmemory.html is as follows:

```
<!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=UTF-8">
<title>OutOfMemory Event Test Page</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/media.js"></script>
<script language="javascript" src="../../js/common.js"></script>
<script language="javascript" src="../../js/menu.js"></script>

<script type="text/javascript" src="../../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../../blackbirdjs/blackbird.css" />

<script>

    //initialize page
    function initPage()
    {
        //save as last visited page
        setLastVisitPage();

        //common initialize function
        commonInitialize();
        requestSourceCode();
        setPageID("Out Of Memory");
        jsLog.initLG();

        //add onclick event handler
        addEventHandler(document.getElementById("btn_back"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_red"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_green"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_yellow"), "click",
onClickHandler);
        addEventHandler(document.getElementById("btn_exit"), "click", onClickHandler);

        displayStatus();
        //add outofmemory event handler
        addEventHandler(window, "outofmemory", outOfMemoryHandler);

        getUsedMemorySize();
    }

    function onUserInput(userInput)
    {
        switch(userInput)
        {
            case VK_BACK : window.location.replace("../menu_netcast.html"); break;
            case VK_RED : case 82 : loadMoreImages(1); break;
            case VK_GREEN : case 71 : loadMoreImages(5); break;
            case VK_YELLOW : case 89 : loadMoreImages(10); break;
        }
    }

    function loadMoreImages(numberToLoad)
    {
        for(var i = 0 ; i < numberToLoad ; i++)
            loadOneImage();
    }

    var numberOfLoadedImages = 0;
    var EACH_IMG_SIZE = 2880054;
    var DEFAULT_LEFT = 0;
    var DEFAULT_TOP = 0;
    var DEFAULT_IMG_LEFT = 100;
    var DEFAULT_IMG_TOP = 500;
    var TOTAL_WIDTH = 900;
    var TOTAL_HEIGHT = 50;
    var MAX_COLUMN_INDEX = 50;
    var MAX_ROW_INDEX = 2;

    function loadOneImage()
    {
        if(MAX_COLUMN_INDEX * MAX_ROW_INDEX == numberOfLoadedImages)
            return;

        //create div and set an image as background
        var widthToSet = Math.floor(TOTAL_WIDTH / MAX_COLUMN_INDEX);
        var heightToSet = Math.floor(TOTAL_HEIGHT / MAX_ROW_INDEX);
        var leftToSet = Math.floor((TOTAL_WIDTH / MAX_COLUMN_INDEX) *
(numberOfLoadedImages % MAX_COLUMN_INDEX));
        var topToSet = Math.floor((TOTAL_HEIGHT / MAX_ROW_INDEX) *
Math.floor(numberOfLoadedImages / MAX_COLUMN_INDEX));

        var bgImgIndex = "";

        if(String(numberOfLoadedImages).length == 1){bgImgIndex = "00";}
        else if(String(numberOfLoadedImages).length == 2){bgImgIndex = "0";}
        else if(String(numberOfLoadedImages).length == 3){bgImgIndex = "";}

        bgImgIndex += String(numberOfLoadedImages);
        var bgUrlToSet = getMediaFileUrl("memoryTestImages") + "lake-nature_" +
bgImgIndex + ".bmp";

        var divToAppend = document.createElement("div");
```

```
      divToAppend.style.position = "absolute";
      divToAppend.style.width = (widthToSet - 2) + "px";
      divToAppend.style.height = (heightToSet - 2) + "px";
      divToAppend.style.border = "1px solid #000000";
      divToAppend.style.left = leftToSet + "px";
      divToAppend.style.top = topToSet + "px";
      divToAppend.style.background = "url(" + bgUrlToSet + ")";
      divToAppend.style.backgroundPosition = "-" + (DEFAULT_IMG_LEFT + leftToSet) +
"px -" + (DEFAULT_IMG_TOP + topToSet) + "px";
      divToAppend.style.backgroundRepeat = "no-repeat";
      document.getElementById("ViewArea").appendChild(divToAppend);
      numberOfLoadedImages++;
      displayStatus();
   }


   function displayStatus()
   {
      setInnerTextById("number_of_loaded_image_result", numberOfLoadedImages + "/" +
(MAX_COLUMN_INDEX * MAX_ROW_INDEX));
      var totalSize = EACH_IMG_SIZE * numberOfLoadedImages;
      setInnerTextById("total_image_size_result", toCommaNotation(String(totalSize))
+ " bytes");
   }


   function toCommaNotation(origString)
   {
      var unitToSep = 3;
      var resultString = "";
      var processString = origString;

      while(processString.length > unitToSep)
      {
         var cutIndex = processString.length - unitToSep;
         var resultString = "," + processString.substring(cutIndex) + resultString;
         processString = processString.substring(0, cutIndex);
      }
      resultString = processString + resultString;
      return resultString;
   }

   var resultString = " ";
   var eventCount = 0;

   function outOfMemoryHandler()
   {
      jsLog.lgevent('outofmemory');
      eventCount++;

      if(eventCount == 1)
         resultString += eventCount + " - " + event.available + " MB";
      else
         resultString += " / " + eventCount + " - " + event.available + " MB";

      setInnerTextById("out_of_memory_event_occurred", resultString);
   }

   var usedMemorySize;
   function getUsedMemorySize()
   {
      if(window.NetCastGetUsedMemorySize)
      {
         usedMemorySize = window.NetCastGetUsedMemorySize();
         setInnerTextById("used_memory_result", usedMemorySize);
         jsLog.lgproperty('NetCastGetUsedMemorySize');
      }
   }

</script>
</head>

<body ondragstart='return false' onselectstart='return false'
onload="javascript:initPage();">

<!-- title -->
<div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>

<!-- navigation -->
<div class='SuiteNavigation'>
   <div style="float:left;">File : netcast/app/outofmemory.html</div>
</div>

<div class='SuiteTitleLine'> </div>

<!-- test contents -->
<div class='ContentArea'>
   <div class='ApiListTitleArea'>Web Open API List</div>
   <div class='ApiListArea'>
      <div class='MethodTitleArea'>
         Methods
         <div class='MethodListArea'>

         </div>
      </div>
      <div class='PropertyTitleArea'>
         Properties
         <div class='PropertyListArea'>
            NetCastGetUsedMemorySize
         </div>
      </div>
      <div class='EventTitleArea'>
         Events
         <div class='EventListArea'>
            outofmemory
         </div>
      </div>
   </div>

   <div class='ViewTitleArea'>
      <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
      <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
   </div>

   <div id='view'>
      <div class='ViewArea' id= 'ViewArea' style="position: relative;">
         <table border="0" cellpadding="0" cellspacing="0">
            <tr height="50px">
```

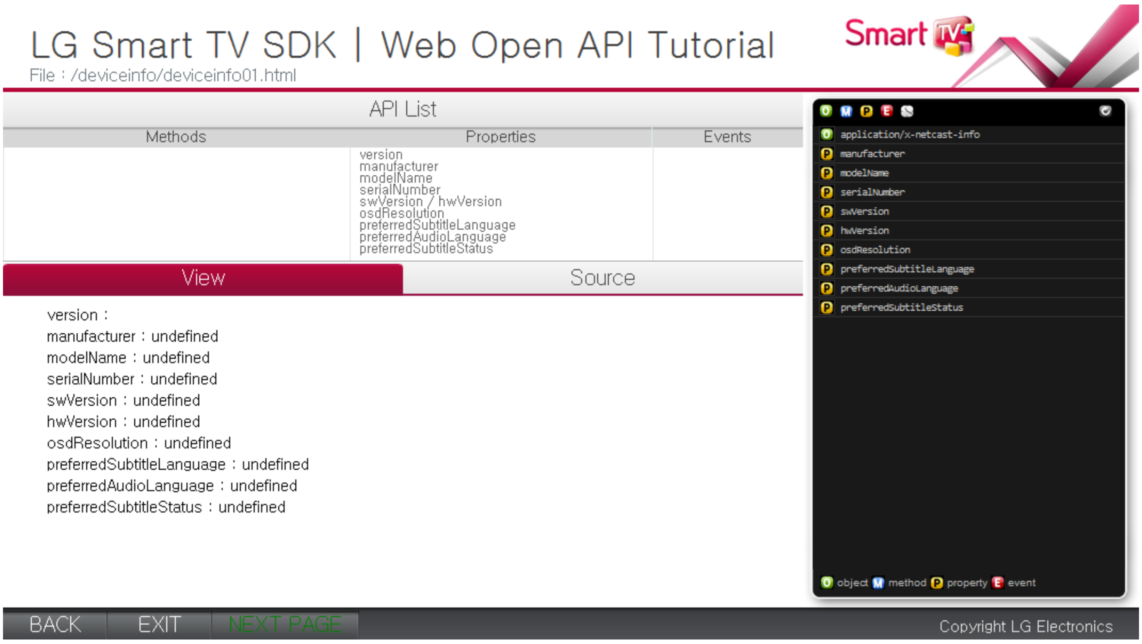
```
                 <td width=100px align="left"></td>
              </tr>
              <tr height="50px">
                 <td width=300px align=left>1. Each image size :</td>
                 <td width=500px align="left">2,880,054 bytes</td>
                 <td width=100px align="left"></td>
              </tr>
              <tr height="50px">
                 <td width=300px align=left>2. Number of loaded images :</td>
                 <td width=500px align="left"><div
id="number_of_loaded_image_result"></div></td>
                <td width=100px align="left"></td>
              </tr>
              <tr height="50px">
                <td width=300px align=left>3. Total image size :</td>
                <td width=500px align="left"><div
id="total_image_size_result"></div></td>
                <td width=100px align="left"></td>
              </tr>
              <tr height="50px">
                 <td width=300px align=left>4. OutOfMemory Event :</td>
                 <td width=500px align="left"><div
id="out_of_memory_event_occurred"></div></td>
                 <td width=100px align="left"></td>
              </tr>
              <tr height="50px">
                <td width="200px" align=left><div>5. Used Memory Size : </div></td>
                <td align="left"><div id="used_memory_result"></div></td>
              </tr>
              <tr height="50px">
                <td width=3000px align=center> Press Red / Green / Yellow key to load
more image(s)<br>
                Check if "outofmemory" event occurs and available memory is
acceptable.</td>
              </tr>
          </table>
       </div>
   </div>

   <div style="visibility: hidden" id='codeview'>
       <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
   </div>

</div>


<!-- button and copyright -->
<div class='SuiteButtonArea'>
   <!-- back key description -->
   <div id='btn_back' class='buttonDescription '>BACK</div>

   <!-- exit button -->
   <div id='btn_exit' class='buttonDescription'>EXIT</div>

   <!-- red key description -->
   <div id='btn_red' class='buttonDescription redColor'>LOAD 1</div>

   <!-- green key description -->
   <div id='btn_green' class='buttonDescription greenColor'>LOAD 5</div>
   <!-- yellow key description -->
   <div id='btn_yellow' class='buttonDescription yellowColor'>LOAD 10</div>

   <!-- copyright -->
   <div class='copyright'>Copyright LG Electronics</div>
</div>

</body>
</html>
```

Getting and Displaying Device Information

This section describes how to get and display the device information in web application.

- Initializing the Page
- Inputting Keys
- Getting & Displaying Device Information
- Setting Device Object
- Source Code of deviceinfor01.html

This application shows how to get LG Smart TV information using properties of LG Web Open API.



LG Smart TV SDK | Web Open API Tutorial
File : /deviceinfo/deviceinfo01.html

API List

Methods	Properties	Events
	version manufacturer modelName serialNumber swVersion / hwVersion osdResolution preferredSubtitleLanguage preferredAudioLanguage preferredSubtitleStatus	

```
version :
manufacturer : undefined
modelName : undefined
serialNumber : undefined
swVersion : undefined
hwVersion : undefined
osdResolution : undefined
preferredSubtitleLanguage : undefined
preferredAudioLanguage : undefined
preferredSubtitleStatus : undefined
```

```
O application/x-netcast-info
P manufacturer
P modelName
P serialNumber
P swVersion
P hwVersion
P osdResolution
P preferredSubtitleLanguage
P preferredAudioLanguage
P preferredSubtitleStatus
```

Needed APIs
This application uses following Web Open API:

API Class	Name	Description
Property	version	Version of Device Info Plugin.
	manufacturer	Manufacturer ID (LGE)
	modelName	Device model name
	serialNumber	Device serial number
	swVersion	Software version.
	hwVersion	Hardware version
	osdResolution	OSD resolution (Returns in number)
	preferredSubtitleLanguage	Subtitle language set in TV
	preferredSubtitleStatus	On/off status of subtitle

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

04: Record the last visited page when running the application.
07: Initialize the page.
08: Get the source code of the page using the XMLHttpRequest object.

09: Initialize the Log function.
12-14: Add an event handler which will executed when the corresponding button is pressed.
18: Display device information on screen by calling getDeviceInfo() method.

```
01 : function initPage()
02 : {
03 :   //save page as last visited page
04 :   setLastVisitPage();
05 :
06 :   //common initialize function
07 :   commonInitialize();
08 :   requestSourceCode();
09 :   jsLog.initLG();
10 :
11 :   //add onclick event handler
12 :   addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
13 :   addEventHandler(document.getElementById("btn_exit"),"click",onClickHandler);
14 :   addEventHandler(document.getElementById("btn_green"),"click",onClickHandler);
15 :
16 :   jsLog.lgobject('application/x-netcast-info');
17 :
18 :   getDeviceInfo();
19 : }
```

Inputting Keys

Use **onUserInput** function is called by onClickHandler function; it receives a key value as the userInput parameter from onClickHandler and creates the corresponding function for each key value to operate the key.

06: When the Back key is pressed, this code is executed.
08: When the Green key is pressed, this code is executed.

```
01 : function onUserInput(userInput)
02 : {
03 :   switch(userInput)
04 :   {
05 :      case VK_BACK :
06 :         window.location.replace("../main_menu.html");
07 :         break;
08 :      case VK_GREEN :
09 :         window.location.replace("./deviceinfo02.html");
10 :         break;
11 :   }
12 : }
```

Getting & Displaying Device Information

The following code shows how to get and display device information of LG Smart TV.

getDeviceInfo

This method displays the information got from the device when the application is started.
03: Declare device.
05: Call getDeviceVersionInfo() method to display version information.
07: Call setInnerTextById method to display to display manufacturer information.
10: Call setInnerTextById method to display modelName information.
13: Call setInnerTextById method to display serialNumber information.
16: Call setInnerTextById method to display swVersion information.
19: Call setInnerTextById method to display hwVersion information.
22: Call setInnerTextById method to display osdResolution information.
25: Call setInnerTextById method to display preferredSubtitleLanguage information.
28: Call setInnerTextById method to display preferredAudioLangage information.
31: Call setInnerTextById method to display preferredSubtitleStatus information.

getDeviceVersionInfo

This method is called in getDeviceInfo method and display the device version information.
38: Declare device.
40: Check the device version.
42: Call setInnerTextById method to display version information.

```
01 : function getDeviceInfo()
02 : {
03 :   var device = document.getElementById('device');
04 :
05 :   getDeviceVersionInfo();
06 :
07 :   setInnerTextById("manufacturer_value","manufacturer:"+ device.manufacturer);
08 :   jsLog.lgproperty('manufacturer');
09 :
10 :   setInnerTextById("modelName_value", "modelName : " + device.modelName);
11 :   jsLog.lgproperty('modelName');
12 :
13 :   setInnerTextById("serialNumber_value","serialNumber:"+ device.serialNumber);
14 :   jsLog.lgproperty('serialNumber');
15 :
16 :   setInnerTextById("swVersion_value", "swVersion : " + device.swVersion);
17 :   jsLog.lgproperty('swVersion');
18 :
19 :   setInnerTextById("hwVersion_value", "hwVersion : " + device.hwVersion);
20 :   jsLog.lgproperty('hwVersion');
21 :
22 :   setInnerTextById("osdResolution_value","osdResolution:"+device.osdResolution);
23 :   jsLog.lgproperty('osdResolution');
24 :
25 :   setInnerTextById("preferredSubtitleLanguage_value",
"preferredSubtitleLanguage : " + device.preferredSubtitleLanguage);
26 :   jsLog.lgproperty('preferredSubtitleLanguage');
27 :
28 :   setInnerTextById("preferredAudioLanguage_value", "preferredAudioLanguage : " +
device.preferredAudioLanguage);
29 :   jsLog.lgproperty('preferredAudioLanguage');
30 :
31 :   setInnerTextById("preferredSubtitleStatus_value", "preferredSubtitleStatus : "
+ device.preferredSubtitleStatus);
32 :   jsLog.lgproperty('preferredSubtitleStatus');
33 :
34 : }
35 :
36 : function getDeviceVersionInfo()
37 : {
38 :   var device = document.getElementById('device');
39 :
40 :   if(device.version)
41 :   {
42 :      setInnerTextById("version_value", "version : " + device.version);
43 :      jsLog.lgproperty('version');
44 :   }
45 : }
```

Setting Device Object

The following code shows how to set Media object.

03: Set deviceinfo type. Refer to App Development Guide.
04-05: Set width and height.

```
<object
  type="application/x-netcast-info"
  id="device"
  width="0"
  height="0"
  style="float: left">
</object>
```

Source Code of deviceinfo01.html

Source code of deviceinfo01.html is as follows:

```
<!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=EUC-KR">
<title>Insert title here</title>
<link rel="stylesheet" href="../css/style.css">
<script language="javascript" src="../js/common.js"></script>
<script language="javascript" src="../js/keycode.js"></script>
<script language="javascript" src="../js/menu.js"></script>

<script type="text/javascript" src="../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../blackbirdjs/blackbird.css" />

<script>

   //initialize page
   function initPage()
   {
      //save page as last visited page
      setLastVisitPage();

      //common initialize function
      commonInitialize();
      requestSourceCode();
      jsLog.initLG();

      //add onclick event handler
      addEventHandler(document.getElementById("btn_back"), "click", onClickHandler);
      addEventHandler(document.getElementById("btn_exit"), "click", onClickHandler);
      addEventHandler(document.getElementById("btn_green"), "click", onClickHandler);

      jsLog.lgobject('application/x-netcast-info');

      getDeviceInfo();
   }

   function onUserInput(userInput)
   {
      switch(userInput)
      {
         case VK_BACK :
            window.location.replace("../main_menu.html");
            break;
         case VK_GREEN :
            window.location.replace("./deviceinfo02.html");
            break;
      }
```

```
    }

    function getDeviceInfo()
    {
        var device = document.getElementById('device');

        getDeviceVersionInfo();

        setInnerTextById("manufacturer_value", "manufacturer : " +
device.manufacturer);
        jsLog.lgproperty('manufacturer');

        setInnerTextById("modelName_value", "modelName : " + device.modelName);
        jsLog.lgproperty('modelName');

        setInnerTextById("serialNumber_value", "serialNumber : " +
device.serialNumber);
        jsLog.lgproperty('serialNumber');

        setInnerTextById("swVersion_value", "swVersion : " + device.swVersion);
        jsLog.lgproperty('swVersion');

        setInnerTextById("hwVersion_value", "hwVersion : " + device.hwVersion);
        jsLog.lgproperty('hwVersion');

        setInnerTextById("osdResolution_value", "osdResolution : " +
device.osdResolution);
        jsLog.lgproperty('osdResolution');

        setInnerTextById("preferredSubtitleLanguage_value",
"preferredSubtitleLanguage : " + device.preferredSubtitleLanguage);
        jsLog.lgproperty('preferredSubtitleLanguage');

        setInnerTextById("preferredAudioLanguage_value", "preferredAudioLanguage : " +
device.preferredAudioLanguage);
        jsLog.lgproperty('preferredAudioLanguage');

        setInnerTextById("preferredSubtitleStatus_value", "preferredSubtitleStatus : "
+ device.preferredSubtitleStatus);
        jsLog.lgproperty('preferredSubtitleStatus');

    }

    function getDeviceVersionInfo()
    {
        var device = document.getElementById('device');

        if(device.version)
        {
            setInnerTextById("version_value", "version : " + device.version);
            jsLog.lgproperty('version');
        }

    }

</script>

</head>
<body ondragstart='return false' onselectstart='return false'
onload="javascript:initPage();" >
```

```
<div id='bodycontent'>
   <!-- title -->
   <div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>

   <!-- navigation -->
   <div class='SuiteNavigation'>
      <div style="float:left;">File : /deviceinfo/deviceinfo01.html</div>
   </div>

   <div class='SuiteTitleLine'> </div>

   <!-- test contents -->
   <div class='ContentArea'>
      <div class='ApiListTitleArea'>API List</div>
      <div class='ApiListArea'>
         <div class='MethodTitleArea'>
            Methods
            <div class='MethodListArea'>
            </div>
         </div>
         <div class='PropertyTitleArea'>
            Properties
            <div class='PropertyListArea'>
            version<br>
            manufacturer<br>
            modelName<br>
            serialNumber<br>
            swVersion / hwVersion<br>
            osdResolution<br>
            preferredSubtitleLanguage<br>
            preferredAudioLanguage<br>
            preferredSubtitleStatus<br>
            </div>
         </div>
         <div class='EventTitleArea'>
            Events
            <div class='EventListArea'>

            </div>
         </div>
      </div>

      <div class='ViewTitleArea'>
         <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
        <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
      </div>

      <div id='view'>
         <div class='ViewArea'>
            <object
               type="application/x-netcast-info"
               id="device"
               width="0"
               height="0"
               style="float: left">
            </object>
            <table border="0" cellpadding="0" cellspacing="0" style="position:
```

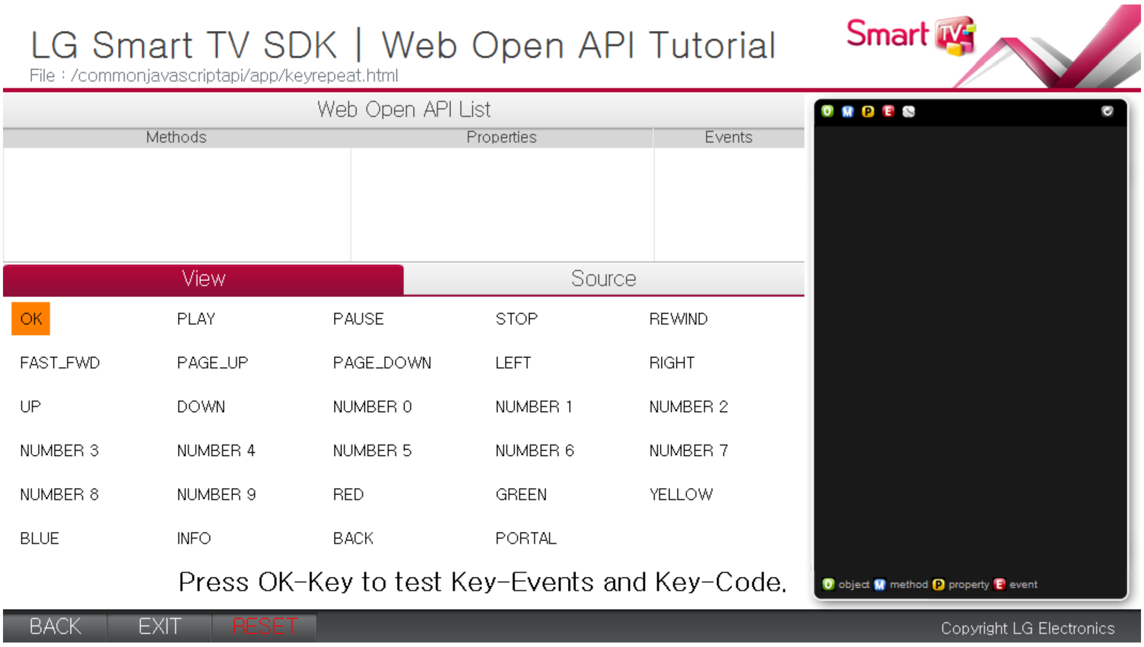
```
relative; left: 30px; top:10px; width:450px; height:250px;">
                 <tr height="10%"  >
                    <td ><div class="eachTestGuide " id="version_value"
>version :</div></td>
                 </tr>
                 <tr height=10%>
                    <td ><div class="eachTestGuide "
id="manufacturer_value">manufacturer :</div></td>
                 </tr>
                 <tr height="10%">
                    <td ><div class="eachTestGuide "
id="modelName_value">modelName :</div></td>
                 </tr>
                 <tr height="10%">
                    <td><div class="eachTestGuide "
id="serialNumber_value">serialNumber :</div></td>
                 </tr>
                 <tr height="10%">
                    <td><div class="eachTestGuide "
id="swVersion_value">swVersion :</div></td>
                 </tr>
                 <tr height="10%">
                    <td><div class="eachTestGuide "
id="hwVersion_value">hwVersion :</div></td>
                 </tr>
                 <tr height="10%">
                    <td><div class="eachTestGuide "
id="osdResolution_value">osdResolution :</div></td>
                 </tr>
                 <tr height="10%">
                    <td><div class="eachTestGuide "
id="preferredSubtitleLanguage_value">preferredSubtitleLanguage :</div></td>
                 </tr>
                 <tr height="10%">
                    <td><div class="eachTestGuide "
id="preferredAudioLanguage_value">preferredAudioLanguage :</div></td>
                 </tr>
                 <tr height="10%">
                    <td><div class="eachTestGuide "
id="preferredSubtitleStatus_value">preferredSubtitleStatus :</div></td>
                 </tr>
             </table>
          </div>
       </div>

       <div style="visibility: hidden" id='codeview'>
          <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
       </div>
   </div>

</div>


<!-- button and copyright -->
<div class='SuiteButtonArea'>

   <!-- button -->
   <div id='btn_back' class='buttonDescription'>BACK</div>

   <!-- exit key description -->
   <div id='btn_exit' class='buttonDescription'>EXIT</div>

   <!-- green key description -->
   <div id='btn_green' class='buttonDescription greenColor'>NEXT PAGE</div>

   <!-- copyright -->
   <div class='copyright'>Copyright LG Electronics</div>
</div>
</body>
</html>
```

Handling Key Inputs Using JavaScript Events

This section describes how to handle the key inputs using JavaScript evnets.

- Initializing the Page
- Handling Events
- Displaying Text
- Executing Test
- Source Code of keydownuppress.html

This application shows how to use JavaScript events for general key inputs in applications.
When receiving a key from IR Remote, you can check and control the received key using JavaScript event.



LG Smart TV SDK | Web Open API Tutorial

File : /commonjavascriptapi/app/keyrepeat.html

Press OK-Key to test Key-Events and Key-Code.

Copyright LG Electronics

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

Declare the variable.
01-15: Declare keysToTestArray and set key names and values.

initPage

This function initializes the page.
21: Record the last visited page when running the application.
24: Initialize the page.
25: Get the source code of the page using the XMLHttpRequest object.
26: Set the page ID.
27: Initialize the Log function.
30-31: Add an event handler which will executed when the corresponding button is pressed.
34: Add keydown event handler which will executed when menu keys (Back, Red, Green, and Yellow buttons) are pressed.
37: Call setKeysOnScreen() function.
40: Call setTimeout function, then, call initTest() function after 700 milliseconds.
42: Declare device.
43-46: Check device and determine whether to display portal key or not.

```
01 : var keysToTestArray = [["div_01", "OK", 13, "d_p_u"]
02 :    ,["div_02", "PLAY", 415, "d_u"],["div_03", "PAUSE", 19, "d_u"]
03 :    ,["div_04", "STOP", 413, "d_u"],["div_05", "REWIND", 412, "d_u"]
04 :    ,["div_06", "FAST_FWD", 417, "d_u"],["div_07", "PAGE_UP", 33, "d_u"]
05 :    ,["div_08", "PAGE_DOWN", 34, "d_u"],["div_09", "LEFT", 37, "d_u"]
06 :    ,["div_10", "RIGHT", 39, "d_u"],["div_11", "UP", 38, "d_u"]
07 :    ,["div_12", "DOWN", 40, "d_u"],["div_13", "NUMBER 0", 48, "d_p_u"]
```

```
   ,["div_14", "NUMBER 1", 49, "d_p_u"],["div_15", "NUMBER 2", 50, "d_p_u"]
   ,["div_16", "NUMBER 3", 51, "d_p_u"],["div_17", "NUMBER 4", 52, "d_p_u"]
   ,["div_18", "NUMBER 5", 53, "d_p_u"],["div_19", "NUMBER 6", 54, "d_p_u"]
   ,["div_20", "NUMBER 7", 55, "d_p_u"],["div_21", "NUMBER 8", 56, "d_p_u"]
   ,["div_22", "NUMBER 9", 57, "d_p_u"],["div_23", "RED", 403, "d_u"]
   ,["div_24", "GREEN", 404, "d_u"],["div_25", "YELLOW", 405, "d_u"]
   ,["div_26", "BLUE", 406, "d_u"],["div_27", "INFO", 457, "d_u"]
   ,["div_28", "BACK", 461, "d_u"],["div_29", "PORTAL", 1000, "d_u"]];

//initialize page
function initPage()
{
   //save page as last visited page
   setLastVisitPage();

   //common initialize function
   commonInitialize();
   requestSourceCode();
   setPageID("Key Down/Up/Press");
   jsLog.initLG();

   //add onclick event handler
   addEventHandler(document.getElementById("btn_back"), "click", backHandler);
   addEventHandler(document.getElementById("btn_reset"), "click", resetHandler);

   //add event handler for menu
   addEventHandler(document.body, "keydown", menuKeyHandler);

   //set keys to test
   setKeysOnScreen();

   //add event handler to test
   setTimeout(initTest,700);

   var device = document.getElementById("device");
   if(! device.supportPortalKey)
   {
      document.getElementById("div_29").style.visibility = "hidden";
   }
}
```

Handling Events

The following code shows event handler functions which are added from initPage() function.

initTest

This function adds handlers of key events.
03: Add keydown event handler.
04: Add keyup event handler.
05: Add keypress event handler.
06: Call initTestProgress() function.

backHandler

When the Back button is pressed, the page moves to the assigned location.

resetHandler

When the Reset button is pressed, Key test is initialized by calling initTestProgress() function.

menuKeyHandler

This function is keydown event handler and executes corresponding function for each menu button.
19-21: Declare and initialize backToYellow, resetToGreen, and hasMenuExecuted. These variables are used to distinguish the test key and bottom menu.
25: Declare userInput and set key code of the event.
26-32: Using switch-case, execute the corresponding function for received userInput and set hasMenuExecuted to true.
28: When Back key is pressed and the Back button is not yellow, call backHandler() function.
29: When Red key is pressed and the Reset button is not green, call resetHandler() function.
30: When Green key is pressed and the Reset button is green, call resetHandler() function.
31: When Yellow key is pressed and the Back button is yellow, call backHandler() function.

testKeyDown

This function is a handler that is executed when keydown event occurs.
35: Declare and initialize keyEventGathering. keyEventGathering is used to check if the pressed key is for testing purpose.
39: Set "_down_" and inputted value of event key to keyEventGathering.

testKeyPress

This function is keypress event handler.
46: Check menuKeyHandler() function and if menu button is executed, just return.
47: If menu button is not executed and only key inputs are received, set "_press_" and inputted value of event key to keyEventGathering.

testKeyUp

This function is a handler that is executed when keyup event occurs.
52: Set "_up_" and inputted value of event key to keyEventGathering.
54-57: If hasMenuExecuted is true, initialize to false.
57-59: If hasMenuExecuted is false, call checkEachResult() function.

```
01 : function initTest()
02 : {
03 :    addEventHandler(document.body, "keydown", testKeyDown);
04 :    addEventHandler(document.body, "keyup", testKeyUp);
05 :    addEventHandler(document.body, "keypress", testKeyPress);
06 :    initTestProgress();
07 : }
08 :
09 : function backHandler()
10 : {
11 :    window.location.replace("../menu_commonjavascriptapi.html");
12 : }
13 :
14 : function resetHandler()
15 : {
16 :    initTestProgress();
17 : }
18 :
19 : var backToYellow = false;
20 : var resetToGreen = false;
21 : var hasMenuExecuted = false;
22 :
23 : function menuKeyHandler(event)
24 : {
25 :    var userInput = getKeyCode(event);
26 :    switch(userInput)
27 :    {
28 :        case VK_BACK : if(!backToYellow){backHandler(); hasMenuExecuted = true;}
break;
29 :        case VK_RED : case 82 : if(!resetToGreen){resetHandler(); hasMenuExecuted =
true;} break;
30 :        case VK_GREEN : case 71 : if(resetToGreen){resetHandler(); hasMenuExecuted
= true;} break;
31 :        case VK_YELLOW : case 89 : if(backToYellow){backHandler(); hasMenuExecuted
= true;} break;
32 :     }
33 : }
34 :
35 : var keyEventGathering = "";
36 :
37 : function testKeyDown(event)
38 : {
39 :    keyEventGathering += "_down_" + getKeyCode(event);
40 :    jsLog.lgevent('javascript: Key Event Handling : KeyDown, Key : ' +
getKeyName(event));
41 : }
42 :
43 : function testKeyPress(event)
44 : {
45 :    jsLog.lgevent('javascript: Key Event Handling : KeyPress, Key : ' +
getKeyName(event));
46 :    if(menuKeyHandler(event, true) == "menuExecuted"){return;}
47 :    keyEventGathering += "_press_" + getKeyCode(event);
48 : }
49 :
50 : function testKeyUp(event)
51 : {
52 :    keyEventGathering += "_up_" + getKeyCode(event);
53 :    jsLog.lgevent('javascript: Key Event Handling : KeyUp, Key : ' +
getKeyName(event));
54 :    if(hasMenuExecuted){
55 :        hasMenuExecuted = false;
56 :        keyEventGathering = "";
57 :    }else{
58 :        checkEachResult();
59 :    }
60 : }
```

Displaying Text

The following code displays test keys on screen with text and change the color of text and background according to the proceeding status of the test.

setSelectedKey

Receives selectedKeyIndex and changes the background color of key which will be tested next.
03-10: The function is executed by number of the size of keysToTestArray.
05-07: Change the background color of key text of received selectedKeyIndex which is the key index of the next test.
07-09: Background colors of other key texts are set with default value.

initTestProgress

Set key text and class for testing.
13: Declare and initialize currentKeyIndex.
17: Initialize currentKeyIndex to 0.
18-21: Using For loop, set class to each key by number keysToTestArray size.
22: Call setSelectedKey() function and initialize to test the first key.
23: Display the description on screen.

setKeysOnScreen

Displays key text on screen which is assigned to keysToTestArray.

```
01 : function setSelectedKey(selectedKeyIndex)
02 : {
03 :    for(var i = 0 ; i < keysToTestArray.length ; i++)
04 :    {
05 :        if(selectedKeyIndex == i){
06 :            document.getElementById(keysToTestArray[i][0]).style.backgroundColor =
"#FF8000";
07 :        }else{
08 :           document.getElementById(keysToTestArray[i][0]).style.backgroundColor =
"";
09 :        }
10 :    }
11 : }
12 :
13 : var currentKeyIndex = 0;
14 :
15 : function initTestProgress()
16 : {
17 :    currentKeyIndex = 0;
18 :    for(var i = 0 ; i < keysToTestArray.length ; i++)
19 :    {
20 :        document.getElementById(keysToTestArray[i][0]).className="eachTestGuide";
21 :    }
22 :    setSelectedKey(0);
23 :    setInnerTextById("keydownuppress_test_description", "Press OK-Key to test Key-
Events and Key-Code.");
24 : }
25 :
26 : function setKeysOnScreen()
27 : {
28 :    for(var i = 0 ; i < keysToTestArray.length ; i++)
29 :    {
30 :        setInnerTextById(keysToTestArray[i][0], keysToTestArray[i][1]);
31 :    }
32 : }
```

Executing Test

Use **checkEachResult** function to test each key event and display in order.

03: Return if currentKeyIndex is bigger than the size of keysToTestArray
05: Declare and initialize keyGatheringToCompare.
07-16: Set the string to keyGatheringToCompare to compare if the correct key is pressed when key event is occurred according to the keys of keysToTestArray[currentKeyIndex].
18-20: If the value of keyEventGathering is same with the value of keyGatheringToCompare, the key text is displayed with green.
21-22: If it is not same, display with red.
24: Initialize keyEventGathering.
26-28: If currentKeyIndex is Red button, set resetToGreen to false, make button executable, and display button text with red.
31-34: If currentKeyIndex is Back button, set backToYellow to false, make button executable, and display button text with white.
36: Increase the value of currentKeyIndex by 1.
37: If the value of currentKeyIndex is smaller than the size of keysToTestArray execute the following codes.
38: Call setSelectedKey function with currentKeyIndex parameter.
40-42: If the value of keysToTestArray is “PORTAL” and the device does not supprt “PORTAL” key, display the corresponding text.
43-45: Otherwise, display the corresponding text.
47-50: If the value of keysToTestArray is Red and it is executed from test menu, set the key text in menu to change into green.
52-55: If the value of keysToTestArray is Back and it is executed in test key, set the key text in menu to change into yellow.
56-59: If the value of currentKeyIndex is bigger than the size of keysToTestArray, display the orresponding text.

```
01 : function checkEachResult()
02 : {
    if(currentKeyIndex >= keysToTestArray.length){return;}

    var keyGatheringToCompare = "";

    if(keysToTestArray[currentKeyIndex][3] == "d_p_u"){
       keyGatheringToCompare = "_down_" + keysToTestArray[currentKeyIndex][2]
           + "_press_" + keysToTestArray[currentKeyIndex][2]
           + "_up_" + keysToTestArray[currentKeyIndex][2];
    }else if(keysToTestArray[currentKeyIndex][3] == "d_u"){
       keyGatheringToCompare = "_down_" + keysToTestArray[currentKeyIndex][2]
           + "_up_" + keysToTestArray[currentKeyIndex][2];
    }else{
       keyGatheringToCompare = "";
    }

    if(keyEventGathering == keyGatheringToCompare){
       document.getElementById(keysToTestArray[currentKeyIndex][0]).className += " greenColor";
    }else{
       document.getElementById(keysToTestArray[currentKeyIndex][0]).className += " redColor";
    }

    keyEventGathering = "";

    if(keysToTestArray[currentKeyIndex][1] == "RED"){
       resetToGreen = false;
       document.getElementById("btn_reset").className = "buttonDescription redColor";
    }

    if(keysToTestArray[currentKeyIndex][1] == "BACK"){
       backToYellow = false;
       document.getElementById("btn_back").className = "buttonDescription whiteColor";
    }

    currentKeyIndex++;
    if(currentKeyIndex < keysToTestArray.length){
       setSelectedKey(currentKeyIndex);

       var device = document.getElementById("device");
       if((keysToTestArray[currentKeyIndex][1]=="PORTAL") && !device.supportPortalKey){
          setInnerTextById("keydownuppress_test_description", "Press " + "RESET" + "-Key to test Key-Events and Key-Code.");
       }else{
          setInnerTextById("keydownuppress_test_description", "Press " + keysToTestArray[currentKeyIndex][1] + "-Key to test Key-Events and Key-Code.");
       }

       if(keysToTestArray[currentKeyIndex][1] == "RED"){
          resetToGreen = true;
          document.getElementById("btn_reset").className = "buttonDescription yellowColor";
       }

       if(keysToTestArray[currentKeyIndex][1] == "BACK"){
          backToYellow = true;
          document.getElementById("btn_back").className = "buttonDescription
```

```
yellowColor";
        }
    }else{
        setSelectedKey(-1);
        setInnerTextById("keydownuppress_test_description", "No more test is left
on this page.");
    }
}
```

Source Code of keydownuppress.html

Source code of keydownuppress.html is as follows:

```
<!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=UTF-8">
<title>Key Down/Up/Press Test Page</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/common.js"></script>
<script language="javascript" src="../../js/menu.js"></script>

<script type="text/javascript" src="../../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../../blackbirdjs/blackbird.css" />

<script>

   var keysToTestArray = [["div_01", "OK", 13, "d_p_u"]
   ,["div_02", "PLAY", 415, "d_u"]
   ,["div_03", "PAUSE", 19, "d_u"]
   ,["div_04", "STOP", 413, "d_u"]
   ,["div_05", "REWIND", 412, "d_u"]
   ,["div_06", "FAST_FWD", 417, "d_u"]
   ,["div_07", "PAGE_UP", 33, "d_u"]
   ,["div_08", "PAGE_DOWN", 34, "d_u"]
   ,["div_09", "LEFT", 37, "d_u"]
   ,["div_10", "RIGHT", 39, "d_u"]
   ,["div_11", "UP", 38, "d_u"]
   ,["div_12", "DOWN", 40, "d_u"]
   ,["div_13", "NUMBER 0", 48, "d_p_u"]
   ,["div_14", "NUMBER 1", 49, "d_p_u"]
   ,["div_15", "NUMBER 2", 50, "d_p_u"]
   ,["div_16", "NUMBER 3", 51, "d_p_u"]
   ,["div_17", "NUMBER 4", 52, "d_p_u"]
   ,["div_18", "NUMBER 5", 53, "d_p_u"]
   ,["div_19", "NUMBER 6", 54, "d_p_u"]
   ,["div_20", "NUMBER 7", 55, "d_p_u"]
   ,["div_21", "NUMBER 8", 56, "d_p_u"]
   ,["div_22", "NUMBER 9", 57, "d_p_u"]
   ,["div_23", "RED", 403, "d_u"]
   ,["div_24", "GREEN", 404, "d_u"]
   ,["div_25", "YELLOW", 405, "d_u"]
   ,["div_26", "BLUE", 406, "d_u"]
   ,["div_27", "INFO", 457, "d_u"]
   ,["div_28", "BACK", 461, "d_u"]
   ,["div_29", "PORTAL", 1000, "d_u"]];

   //initialize page
   function initPage()
   {
      //save page as last visited page
      setLastVisitPage();

      //common initialize function
      commonInitialize();
      requestSourceCode();
      setPageID("Key Down/Up/Press");
      jsLog.initLG();

      //add onclick event handler
      addEventHandler(document.getElementById("btn_back"), "click", backHandler);
      addEventHandler(document.getElementById("btn_reset"), "click", resetHandler);

      //add event handler for menu
      addEventHandler(document.body, "keydown", menuKeyHandler);

      //set keys to test
      setKeysOnScreen();

      //add event handler to test
      setTimeout(initTest,700);

      var device = document.getElementById("device");
      if(! device.supportPortalKey)
      {
         document.getElementById("div_29").style.visibility = "hidden";
      }
   }

   function initTest()
   {
      addEventHandler(document.body, "keydown", testKeyDown);
      addEventHandler(document.body, "keyup", testKeyUp);
      addEventHandler(document.body, "keypress", testKeyPress);
      initTestProgress();
   }

   function backHandler()
   {
      window.location.replace("../menu_commonjavascriptapi.html");
   }

   function resetHandler()
   {
      initTestProgress();
   }

   var backToYellow = false;
   var resetToGreen = false;
   var hasMenuExecuted = false;

   function menuKeyHandler(event)
   {
      var userInput = getKeyCode(event);
      switch(userInput)
      {
         case VK_BACK : if(!backToYellow){backHandler(); hasMenuExecuted = true;}
break;
```

```
         case VK_RED : case 82 : if(!resetToGreen){resetHandler(); hasMenuExecuted =
true;} break;
         case VK_GREEN : case 71 : if(resetToGreen){resetHandler(); hasMenuExecuted =
true;} break;
         case VK_YELLOW : case 89 : if(backToYellow){backHandler(); hasMenuExecuted =
true;} break;
      }
   }

   var keyEventGathering = "";

   function testKeyDown(event)
   {
      keyEventGathering += "_down_" + getKeyCode(event);
      jsLog.lgevent('javascript: Key Event Handling : KeyDown, Key : ' +
getKeyName(event));
   }

   function testKeyPress(event)
   {
      jsLog.lgevent('javascript: Key Event Handling : KeyPress, Key : ' +
getKeyName(event));
      if(menuKeyHandler(event, true) == "menuExecuted"){return;}
         keyEventGathering += "_press_" + getKeyCode(event);
   }

   function testKeyUp(event)
   {
      keyEventGathering += "_up_" + getKeyCode(event);
      jsLog.lgevent('javascript: Key Event Handling : KeyUp, Key : ' +
getKeyName(event));
      if(hasMenuExecuted){
         hasMenuExecuted = false;
         keyEventGathering = "";
      }else{
         checkEachResult();
      }
   }

   function setSelectedKey(selectedKeyIndex)
   {
      for(var i = 0 ; i < keysToTestArray.length ; i++)
      {
         if(selectedKeyIndex == i){
            document.getElementById(keysToTestArray[i][0]).style.backgroundColor =
"#FF8000";
         }else{
            document.getElementById(keysToTestArray[i][0]).style.backgroundColor =
"";
         }
      }
   }

   var currentKeyIndex = 0;

   function initTestProgress()
   {
      currentKeyIndex = 0;
      for(var i = 0 ; i < keysToTestArray.length ; i++)
      {
         document.getElementById(keysToTestArray[i][0]).className = "eachTestGuide";
      }
         setSelectedKey(0);
         setInnerTextById("keydownuppress_test_description", "Press OK-Key to test
Key-Events and Key-Code.");
   }

   //set keys to test
   function setKeysOnScreen()
   {
      for(var i = 0 ; i < keysToTestArray.length ; i++)
      {
         setInnerTextById(keysToTestArray[i][0], keysToTestArray[i][1]);
      }
   }

   function checkEachResult()
   {
      if(currentKeyIndex >= keysToTestArray.length){return;}
      var keyGatheringToCompare = "";

      if(keysToTestArray[currentKeyIndex][3] == "d_p_u"){
         keyGatheringToCompare = "_down_" + keysToTestArray[currentKeyIndex][2]
            + "_press_" + keysToTestArray[currentKeyIndex][2]
            + "_up_" + keysToTestArray[currentKeyIndex][2];
      }else if(keysToTestArray[currentKeyIndex][3] == "d_u"){
         keyGatheringToCompare = "_down_" + keysToTestArray[currentKeyIndex][2]
            + "_up_" + keysToTestArray[currentKeyIndex][2];
      }else{
         keyGatheringToCompare = "";
      }

      if(keyEventGathering == keyGatheringToCompare){
         document.getElementById(keysToTestArray[currentKeyIndex][0]).className += "
greenColor";
      }else{
         document.getElementById(keysToTestArray[currentKeyIndex][0]).className += "
redColor";
      }

      keyEventGathering = "";
      //in case just tested red key
      if(keysToTestArray[currentKeyIndex][1] == "RED"){
         resetToGreen = false;
         document.getElementById("btn_reset").className = "buttonDescription
redColor";
      }
      //in case just tested back key
      if(keysToTestArray[currentKeyIndex][1] == "BACK"){
         backToYellow = false;
         document.getElementById("btn_back").className = "buttonDescription
whiteColor";
      }

      currentKeyIndex++;
      if(currentKeyIndex < keysToTestArray.length){
         setSelectedKey(currentKeyIndex);
         var device = document.getElementById("device");
         if((keysToTestArray[currentKeyIndex][1]=="PORTAL")
&& !device.supportPortalKey){
```

```
            if(! device.supportPortalKey){
                setInnerTextById("keydownuppress_test_description", "Press " + "RESET" +
"-Key to test Key-Events and Key-Code.");
            }
            }else{
                setInnerTextById("keydownuppress_test_description", "Press " +
keysToTestArray[currentKeyIndex][1] + "-Key to test Key-Events and Key-Code.");
            }

            //in case next test is red key
            if(keysToTestArray[currentKeyIndex][1] == "RED"){
                resetToGreen = true;
                document.getElementById("btn_reset").className = "buttonDescription
yellowColor";
            }
            //in case next testis back key
            if(keysToTestArray[currentKeyIndex][1] == "BACK"){
                backToYellow = true;
                document.getElementById("btn_back").className = "buttonDescription
yellowColor";
            }
        }else{
            setSelectedKey(-1);
            setInnerTextById("keydownuppress_test_description", "No more test is left on
this page.");
        }
    }

    function getKeyName(event)
    {
        var key = getKeyCode(event);
        for(var i = 0 ; i < keysToTestArray.length ; i++)
        {
            if(keysToTestArray[i][2] == key)
            {
                return keysToTestArray[i][1];
            }
        }

        return "";
    }

</script>
</head>

<body ondragstart='return false' onselectstart='return false'
onload="javascript:initPage();">

<!-- title -->
<div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>
<object type="application/x-netcast-info"
    id="device">
</object>
<!-- navigation -->

<div class='SuiteNavigation'>
    <div style="float:left;">File : /commonjavascriptapi/app/keyrepeat.html</div>
</div>
<div class='SuiteTitleLine'> </div>
<!-- test contents -->
```

```
<div class='ContentArea'>
   <div class='ApiListTitleArea'>Web Open API List</div>
   <div class='ApiListArea'>
      <div class='MethodTitleArea'>
         Methods
         <div class='MethodListArea'>

         </div>
      </div>
      <div class='PropertyTitleArea'>
         Properties
         <div class='PropertyListArea'>

         </div>
      </div>
      <div class='EventTitleArea'>
         Events
         <div class='EventListArea'>

         </div>
      </div>
   </div>
   <div class='ViewTitleArea'>
      <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
      <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
   </div>
   <div id='view'>
      <div class = 'ViewArea'>
         <table width="880px" border="0" cellpadding="0" cellspacing="0">
            <tr height="50px">
               <td width=200px align=left><div class='eachTestGuide'
id="div_01"></div></td>
               <td width=200px align=left><div class='eachTestGuide'
id="div_02"></div></td>
               <td width=200px align=left><div class='eachTestGuide'
id="div_03"></div></td>
               <td width=200px align=left><div class='eachTestGuide'
id="div_04"></div></td>
               <td width=200px align=left><div class='eachTestGuide'
id="div_05"></div></td>
            </tr>
            <tr height="50px">
               <td width=200px align=left><div class='eachTestGuide'
id="div_06"></div></td>
               <td width=200px align=left><div class='eachTestGuide'
id="div_07"></div></td>
               <td width=200px align=left><div class='eachTestGuide'
id="div_08"></div></td>
               <td width=200px align=left><div class='eachTestGuide'
id="div_09"></div></td>
               <td width=200px align=left><div class='eachTestGuide'
id="div_10"></div></td>
            </tr>
            <tr height="50px">
               <td width=200px align=left><div class='eachTestGuide'
id="div_11"></div></td>
               <td width=200px align=left><div class='eachTestGuide'
id="div_12"></div></td>
```

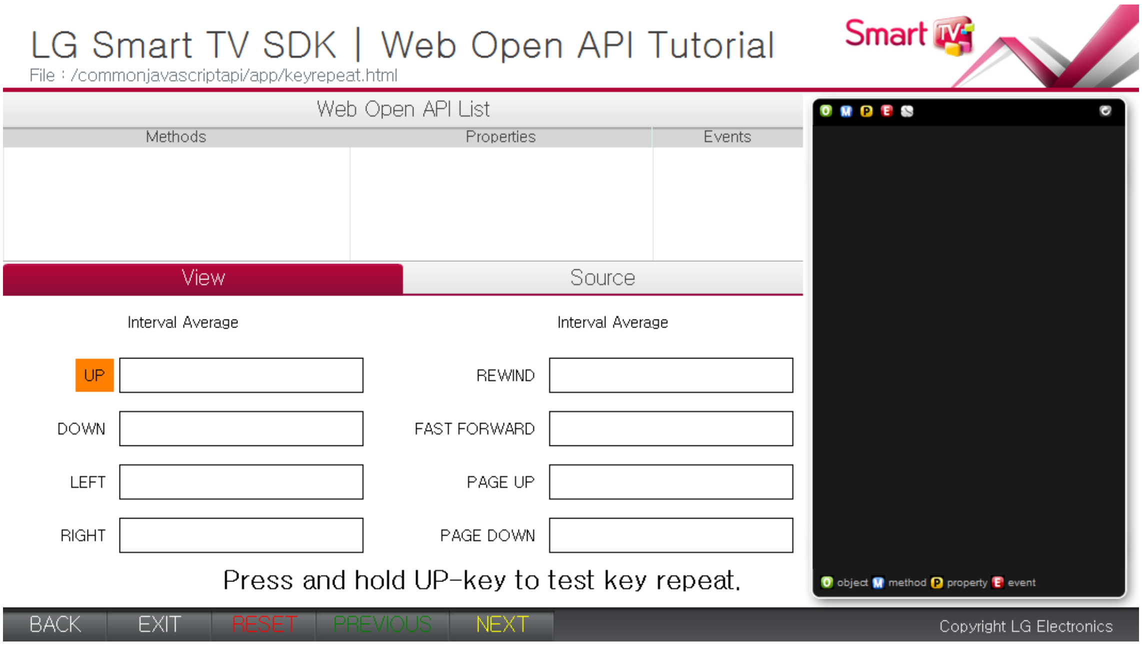
```
                 <td width=200px align=left><div class='eachTestGuide'
id="div_13"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_14"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_15"></div></td>
             </tr>
             <tr height="50px">
                 <td width=200px align=left><div class='eachTestGuide'
id="div_16"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_17"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_18"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_19"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_20"></div></td>
             </tr>
             <tr height="50px">
                 <td width=200px align=left><div class='eachTestGuide'
id="div_21"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_22"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_23"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_24"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_25"></div></td>
             </tr>
             <tr height="50px">
                 <td width=200px align=left><div class='eachTestGuide'
id="div_26"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_27"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_28"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_29"></div></td>
                 <td width=200px align=left><div class='eachTestGuide'
id="div_30"></div></td>
             </tr>
          </table>
       </div>
   </div>
   <div scrolling="AUTO" style="visibility: hidden" id='codeview'>
       <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
   </div>
</div>

<!-- description -->
<div id='keydownuppress_test_description' class='SuiteDescription' style="position:
absolute; left: 40px; top:630px"></div>

<!-- button and copyright -->
<div class='SuiteButtonArea'>
   <!-- back key description -->
    div id='btn_back' class='buttonDescription'>BACK</div>
   <!-- exit key description -->
   <div id='btn_exit' class='buttonDescription'>EXIT</div>

   <!-- red key description -->
   <div id='btn_reset' class='buttonDescription redColor'>RESET</div>

   <!-- copyright -->
   <div class='copyright'>Copyright LG Electronics</div>
</div>

</body>
</html>
```

Handling Key Repeat Using JavaScript Events

This section describes how to handle the key repeat using JavaScript events.

- Initializing the Page
- Handling Events
- Source Code of keyrepeat.html

This application shows how JavaScript events are occurred when a key is pressed from IR Remote and to use these events.



LG Smart TV SDK | Web Open API Tutorial

File : /commonjavascriptapi/app/keyrepeat.html

Press and hold UP-key to test key repeat.

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

04: Record the last visited page when running the application.
07: Initialize the page.
08: Get the source code of the page using the XMLHttpRequest object.
09: Set the page ID.
10: Initialize the Log function.
13-16: Add an event handler which will executed when the corresponding button is pressed.
19-20: Add event handlers for key inputs (keydown, keyup) from IR Remote.
23: Set the background color of key to test.
24: Display description for test on screen.

```
01 : function initPage()
02 : {
03 :    //save page as last visited page
04 :    setLastVisitPage();
05 :
06 :    //common initialize function
07 :    commonInitialize();
08 :    requestSourceCode();
09 :    setPageID("Key Repeat");
10 :    jsLog.initLG();
11 :
12 :    //add onclick event handler
13 :   addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
14 :   addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
15 :   addEventHandler(document.getElementById("btn_green"),"click",onClickHandler);
16 :   addEventHandler(document.getElementById("btn_yellow"),"click",onClickHandler);
17 :
18 :   //add key repeat test event handler
19 :   addEventHandler(document.body, "keydown", onKeyDownHandler);
20 :   addEventHandler(document.body, "keyup", onKeyUpHandler);
21 :
22 :   //select current position
23 :   document.getElementById(keysToTest[currentTestPosition][0]).className =
"eachTestGuide orangeBgColor";
24 :   setInnerTextById("keyrepeat_test_description", "Press and hold " +
keysToTest[currentTestPosition][3] + "-key to test key repeat.");
25 : }
```

Handling Events

The following code shows event handler functions which are added from initPage() function.

02-16: Declare and initialize keyToTest, currentTestPosition, maxRepeatCount, maxGaugeWidth, repeatCount, startTime, and propressStatus.

onKeyDownHandler

This event handler called when key down is occurred.
20: Declare userInput and save key code of the event.
23: If userInput is same with setup key and the repeat number is not bigger than the maximum value, execute the following codes.
25-30: If repeatCount is 0, declare currentDate and initialize startTime to current time.
31: Increase repeatCount by 1.
32: Increase gauge bar by number of repeatCount.
34-40: If number of key repeat is bigger than maxRepeatCount, declare elapsedTime, count the key repeat time, and display it on screen.

onKeyUpHandler

This event handler is called when key up is occurred.
46: Declare userInput and save key code of the event.
49-52: If inputted key value is same with setup key value and the repeat number does not exceed the maximum number, change the color of corresponding gauge bar.

resetCurrentTest

Execute reset function.
57-59: Initialize repeatCount, startTime, and progressStatus.
60: Delete the child nodes of currentTestPosition.
61-63: Initialize the bar length of currentTestPosition and display the corresponding text on screen.

changeTestPosition

Change key to test.
68-70: Initialize repeatCount, startTime, and progressStatus.
71: Set class to unselect the selected key.
73-78: To go to previous key, decrease the value of currentTestPosition by 1 and move the position of the key. If the value of currentTestPosition gets smaller than 0, move the position to the last key position according to the size of keysToTest.
80-85: To go to the next key, increase the value of currentTestPosition by 1 and move the key position. If the value of currentTestPosition gets bigger than keysToTest, set to 0 and move to the first key position.
87-88: Changing the background color of class to orange to show that the testing key is changed. Also, display description text on screen.

```
01 : //[0] : title id, [1] : gauge id, [2] : key code, [3] : key name
02 : var keysToTest = [ ["up_title", "up_progress", 38, "UP"]
03 :     , ["down_title", "down_progress", 40, "DOWN"]
04 :     , ["left_title", "left_progress", 37, "LEFT"]
```

```
    , ["right_title", "right_progress", 39, "RIGHT"]
    , ["rewind_title", "rewind_progress", 412, "REWIND"]
    ,["fast_forward_title" , "fast_forward_progress" , 417 , "FAST_FORWARD"]
    , ["page_up_title", "page_up_progress", 33, "PAGE_UP"]
    , ["page_down_title", "page_down_progress", 34, "PAGE_DOWN"]];

var currentTestPosition = 0;
var maxRepeatCount = 40;
var maxGaugeWidth = 273;
var repeatCount = 0;
var startTime = 0;
var progressStatus = "";

function onKeyDownHandler(event)
{
    var userInput = getKeyCode(event);
    jsLog.lgevent('javascript: Key Event Handling : KeyDown, Key : ' + getKeyName(event));

    if(userInput==keysToTest[currentTestPosition][2]&&repeatCount<maxRepeatCount)
    {
        if(repeatCount == 0)
        {
            var currentDate = new Date();
            startTime = currentDate.getTime();
            setInnerTextById(keysToTest[currentTestPosition][1], "");
        }
        repeatCount++;
        document.getElementById(keysToTest[currentTestPosition][1]).style.width = Math.floor((maxGaugeWidth * (repeatCount / maxRepeatCount))) + "px";

        if(repeatCount >= maxRepeatCount)
        {
            var currentDate = new Date();
            var elapsedTime = currentDate.getTime() - startTime;
            setInnerTextById(keysToTest[currentTestPosition][1], (elapsedTime / repeatCount) + "ms");
            setInnerTextById("keyrepeat_test_description", "Press YELLOW-Key to test next key.");
        }
    }
}

function onKeyUpHandler(event)
{
    var userInput = getKeyCode(event);
    jsLog.lgevent('javascript: Key Event Handling : KeyUp, Key : ' + getKeyName(event));

    if(userInput==keysToTest[currentTestPosition][2]&&repeatCount<maxRepeatCount)
    {
document.getElementById(keysToTest[currentTestPosition][1]).style.backgroundColor="#FF0000";
    }
}

function resetCurrentTest()
{
    repeatCount = 0;
    startTime = 0;
    progressStatus = "";
    clearChildNodes(document.getElementById(keysToTest[currentTestPosition][1]));
    document.getElementById(keysToTest[currentTestPosition][1]).style.width="0px";
    document.getElementById(keysToTest[currentTestPosition][1]).style.backgroundColor = "#008000";
    setInnerTextById("keyrepeat_test_description", "Press and hold " + keysToTest[currentTestPosition][3] + "-key to test key repeat.");
}

function changeTestPosition(direction)
{
    repeatCount = 0;
    startTime = 0;
    progressStatus = "";
    document.getElementById(keysToTest[currentTestPosition][0]).className="eachTestGuide";

    if(direction == "PREVIOUS"){
        currentTestPosition--;
        if(currentTestPosition < 0){
            currentTestPosition = keysToTest.length - 1;
        }
    }

    if(direction == "NEXT"){
        currentTestPosition++;
        if(currentTestPosition >= keysToTest.length){
            currentTestPosition = 0;
        }
    }

    document.getElementById(keysToTest[currentTestPosition][0]).className = "eachTestGuide orangeBgColor";
    setInnerTextById("keyrepeat_test_description", "Press and hold " + keysToTest[currentTestPosition][3] + "-key to test key repeat.");
}
```

Source Code of keyrepeat.html

Source code of keyrepeat.html is as follows:

```
<!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=UTF-8">
<title>Key Repeat Test Page</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/common.js"></script>
<script language="javascript" src="../../js/menu.js"></script>

<script type="text/javascript" src="../../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../../blackbirdjs/blackbird.css" />

<script>

    //initialize page
    function initPage()
    {
        //save page as last visited page
        setLastVisitPage();

        //common initialize function
        commonInitialize();
        requestSourceCode();
        setPageID("Key Repeat");
        jsLog.initLG();

        //add onclick event handler
        addEventHandler(document.getElementById("btn_back"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_red"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_green"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_yellow"), "click",
onClickHandler);

        //add key repeat test event handler
        addEventHandler(document.body, "keydown", onKeyDownHandler);
        addEventHandler(document.body, "keyup", onKeyUpHandler);

        //select current position
        document.getElementById(keysToTest[currentTestPosition][0]).className =
"eachTestGuide orangeBgColor";
        setInnerTextById("keyrepeat_test_description", "Press and hold " +
keysToTest[currentTestPosition][3] + "-key to test key repeat.");
    }

    function onUserInput(userInput)
    {
        switch(userInput)
        {
            case VK_BACK : window.location.replace("../menu_commonjavascriptapi.html");
break;
            case VK_RED : case 82 : resetCurrentTest(); break;
            case VK_GREEN : case 71 : changeTestPosition("PREVIOUS"); break;
            case VK_YELLOW : case 89 : changeTestPosition("NEXT"); break;
        }
    }

    //array of keys to test
    //[0] : title id, [1] : gauge id, [2] : key code, [3] : key name
    var keysToTest = [ ["up_title", "up_progress", 38, "UP"]
        , ["down_title", "down_progress", 40, "DOWN"]
        , ["left_title", "left_progress", 37, "LEFT"]
        , ["right_title", "right_progress", 39, "RIGHT"]
        , ["rewind_title", "rewind_progress", 412, "REWIND"]
        , ["fast_forward_title", "fast_forward_progress", 417, "FAST_FORWARD"]
        , ["page_up_title", "page_up_progress", 33, "PAGE_UP"]
        , ["page_down_title", "page_down_progress", 34, "PAGE_DOWN"]];

    var currentTestPosition = 0;
    var maxRepeatCount = 40;
    var maxGaugeWidth = 273;
    var repeatCount = 0;
    var startTime = 0;
    var progressStatus = "";

    function onKeyDownHandler(event)
    {
```

```
    var userInput = getKeyCode(event);
    jsLog.lgevent('javascript: Key Event Handling : KeyDown, Key : ' + 
getKeyName(event));

    if(userInput == keysToTest[currentTestPosition][2] && repeatCount < 
maxRepeatCount){
      if(repeatCount == 0){
        var currentDate = new Date();
        startTime = currentDate.getTime();
        setInnerTextById(keysToTest[currentTestPosition][1], "");
      }
      repeatCount++;
      document.getElementById(keysToTest[currentTestPosition][1]).style.width = 
Math.floor((maxGaugeWidth * (repeatCount / maxRepeatCount))) + "px";
      if(repeatCount >= maxRepeatCount){
        var currentDate = new Date();
        var elapsedTime = currentDate.getTime() - startTime;
        setInnerTextById(keysToTest[currentTestPosition][1], (elapsedTime / 
repeatCount) + "ms");
        setInnerTextById("keyrepeat_test_description", "Press YELLOW-Key to test 
next key.");
      }
    }
  }

  function onKeyUpHandler(event)
  {
    var userInput = getKeyCode(event);
    jsLog.lgevent('javascript: Key Event Handling : KeyUp, Key : ' + 
getKeyName(event));

    if(userInput == keysToTest[currentTestPosition][2] && repeatCount < 
maxRepeatCount){

document.getElementById(keysToTest[currentTestPosition][1]).style.backgroundColor = 
"#FF0000";
    }
  }
  //reset current key test status
  function resetCurrentTest()
  {
    repeatCount = 0;
    startTime = 0;
    progressStatus = "";
    clearChildNodes(document.getElementById(keysToTest[currentTestPosition][1]));
    document.getElementById(keysToTest[currentTestPosition][1]).style.width = 
"0px";

document.getElementById(keysToTest[currentTestPosition][1]).style.backgroundColor = 
"#008000";
    setInnerTextById("keyrepeat_test_description", "Press and hold " + 
keysToTest[currentTestPosition][3] + "-key to test key repeat.");
  }
  //change test position
  function changeTestPosition(direction)
  {
    repeatCount = 0;
    startTime = 0;
    progressStatus = "";
    document.getElementById(keysToTest[currentTestPosition][0]).className = 
"eachTestGuide";

    if(direction == "PREVIOUS"){
      currentTestPosition--;
      if(currentTestPosition < 0){
        currentTestPosition = keysToTest.length - 1;
      }
    }
    if(direction == "NEXT"){
      currentTestPosition++;
      if(currentTestPosition >= keysToTest.length){
        currentTestPosition = 0;
      }
    }
    document.getElementById(keysToTest[currentTestPosition][0]).className =
"eachTestGuide orangeBgColor";
    setInnerTextById("keyrepeat_test_description", "Press and hold " +
keysToTest[currentTestPosition][3] + "-key to test key repeat.");
  }

  function getKeyName(event)
  {
    var key = getKeyCode(event);
    for(var i = 0 ; i < keysToTest.length ; i++)
    {
      if(keysToTest[i][2] == key)
      {
        return keysToTest[i][3];
      }
    }

    return "-";
  }

</script>
</head>
<body ondragstart='return false' onselectstart='return false'
onload="javascript:initPage();">

<!-- title -->
<div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>

<!-- navigation -->
<div class='SuiteNavigation'>
  <div style="float:left;">File : /commonjavascriptapi/app/keyrepeat.html</div>
</div>
<div class='SuiteTitleLine'> </div>

<!-- test contents -->
<div id='ContentArea' class='ContentArea' >
  <div class='ApiListTitleArea'>Web Open API List</div>
  <div class='ApiListArea'>
    <div class='MethodTitleArea'>
      Methods
      <div class='MethodListArea'>

      </div>
    </div>
    <div class='PropertyTitleArea'>
      Properties
```

```
            <div class='PropertyListArea'>

            </div>
        </div>
        <div class='EventTitleArea'>
            Events
            <div class='EventListArea'>

            </div>
        </div>
    </div>
    <div class='ViewTitleArea'>
        <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
        <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
    </div>
    <div id='view'>
    <div class = 'ViewArea'>
        <table width="880px" border="0" cellpadding="0" cellspacing="0">
        <tr height="60px">
            <td width="150px"></td>
            <td width="275px"><div class='eachTestGuide'>Interval Average</div></td>
            <td width="275px"></td>
            <td width="275px"><div class='eachTestGuide'>Interval Average</div></td>
        </tr>
        <tr height="60px">
            <td width="150px" align="right"><div class='eachTestGuide'
id="up_title">UP</div>&nbsp;</td>
            <td width="275px" align="left"><div class='gaugeBorder'><div
class='gaugeBar' id="up_progress"></div></div></td>
            <td width="275px" align="right"><div class='eachTestGuide'
id="rewind_title">REWIND</div>&nbsp;</td>
            <td width="275px" align="left"><div class='gaugeBorder'><div
class='gaugeBar' id="rewind_progress"></div></div></td>
        </tr>
        <tr height="60px">
            <td width="150px" align="right"><div class='eachTestGuide'
id="down_title">DOWN</div>&nbsp;</td>
            <td width="275px" align="left"><div class='gaugeBorder'><div
class='gaugeBar' id="down_progress"></div></div></td>
            <td width="275px" align="right"><div class='eachTestGuide'
id="fast_forward_title">FAST FORWARD</div>&nbsp;</td>
            <td width="275px" align="left"><div class='gaugeBorder'><div
class='gaugeBar' id="fast_forward_progress"></div></div></td>
        </tr>
        <tr height="60px">
            <td width="150px" align="right"><div class='eachTestGuide'
id="left_title">LEFT</div>&nbsp;</td>
            <td width="275px" align="left"><div class='gaugeBorder'><div
class='gaugeBar' id="left_progress"></div></div></td>
            <td width="275px" align="right"><div class='eachTestGuide'
id="page_up_title">PAGE UP</div>&nbsp;</td>
            <td width="275px" align="left"><div class='gaugeBorder'><div
class='gaugeBar' id="page_up_progress"></div></div></td>
        </tr>
        <tr height="60px">
            <td width="150px" align="right"><div class='eachTestGuide'
id="right_title">RIGHT</div>&nbsp;</td>
            <td width="275px" align="left"><div class='gaugeBorder'><div
```

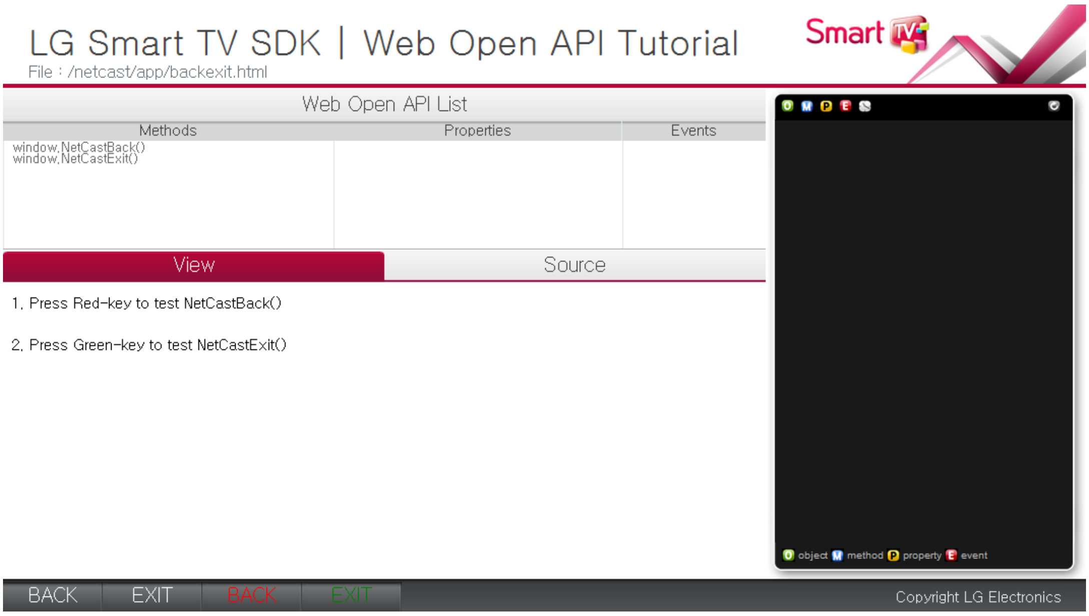
```
class='gaugeBar' id="right_progress"></div></div></td>
          <td width="275px" align="right"><div class='eachTestGuide'
id="page_down_title">PAGE DOWN</div>&nbsp;</td>
          <td width="275px" align="left"><div class='gaugeBorder'><div
class='gaugeBar' id="page_down_progress"></div></div></td>
       </tr>
       </table>
   </div>
   </div>
   <divscrolling="AUTO" style="visibility: hidden" id='codeview'>
       <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
   </div>

</div>

<!-- description -->
<div id='keyrepeat_test_description' class='SuiteDescription' style="position:
absolute; left: 40px; top:630px"></div>

<!-- button and copyright -->
<div class='SuiteButtonArea'>
   <!-- back key description -->
   <div id='btn_back' class='buttonDescription '>BACK</div>

   <!-- exit key description -->
   <div id='btn_exit' class='buttonDescription'>EXIT</div>

   <!-- red key description -->
   <div id='btn_red' class='buttonDescription redColor'>RESET</div>

   <!-- green key description -->
   <div id='btn_green' class='buttonDescription greenColor'>PREVIOUS</div>

   <!-- yellow key description -->
   <div id='btn_yellow' class='buttonDescription yellowColor'>NEXT</div>

   <!-- copyright -->
   <div class='copyright'>Copyright LG Electronics</div>
</div>

</body>
</html>
```

Implementing Functionality of Back and Exit Keys

This section describes how to implement functionality of Back and Exit keys in web application.

- Initializing the Page
- Inputting Keys
- Handling Keys
- Source Code of backexit.html

This application is designed to show how to implement the functionality of Back, and Exit using Web open API.
This application shows which method is used to exit the application and move back to the MyApps menu or TV video in each page.


LG Smart TV SDK | Web Open API Tutorial

File : /netcast/app/backexit.html

Web Open API List

Methods	Properties	Events

window.NetCastBack()
window.NetCastExit()		

1. Press Red-key to test NetCastBack()

2. Press Green-key to test NetCastExit()

Copyright LG Electronics

Needed APIs
This application uses following Web Open API:

API Class	Name	Description
Method	window.NetCastExit()	This API is used to exit or quit the application.
	window.NetCastBack()	This API is used to move back to the MyApps menu.
Property	N/A	N/A
Event	N/A	N/A

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

04: Records the last visited page when running the application.
07: Initializes the page.
08: Gets the source code of the page using the XMLHttpRequest object.
09: Sets the page ID.
10: Initializes the Log function.
13-16: Registers an event handler which will executed when the corresponding button is pressed.
18-21: Deletes the cookie related the application if it has been saved.

```
01 : function initPage()
02 : {
03 :    //save page as last visited page
```

```
   setLastVisitPage();

   //common initialize function
   commonInitialize();
   requestSourceCode();
   setPageID("Back, Exit");
   jsLog.initLG();

   //add onclick event handler
   addEventHandler(document.getElementById("btn_back"),"click", onClickHandler);
   addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
   addEventHandler(document.getElementById("btn_green"),"click",onClickHandler);
   addEventHandler(document.getElementById("btn_exit"),"click", onClickHandler);

   if(readCookie("back_exit_test_status") != "")
   {
      //clear back_exit_test_status cookie if it have been set
      deleteCookie("back_exit_test_status", "/");
   }
}
```

Inputting Keys

Use **onUserInput** function is called by onClickHandler function; it receives a key value as the userInput parameter from onClickHandler and creates the corresponding function for each key value to operate the key.

06: When the Back key is pressed, this code is executed.
07: When the Red key is pressed, this code is executed.
08: When the Green key is pressed, this code is executed.

```
01 : //onUserInput function should be implemented
02 : function onUserInput(userInput)
03 : {
04 :    switch(userInput)
05 :    {
06 :       case VK_BACK : window.location.replace("../menu_netcast.html"); break;
07 :       case VK_RED : case 82 : processRedKey(); break;
08 :       case VK_GREEN : case 71 : processGreenKey(); break;
09 :    }
10 : }
```

Handling Keys

The following functions are executed when each key of onUserInput() is pressed.

processRedKey

This function is called when the Red key is pressed.
05-07: Gets the type and version of the browser through the navigator object.
11: Saves “back_exit_test_status” cookie.
13: Moves back to the MyApps menu.

processGreenKey

This function is called when the Green key is pressed.
27: Exits the application.
30: Shows the text on the screen if the browser of app is not a LG Browser.

Note

If you open the main menu when “back_exit_test_status” cookie has been saved, you can move to the current application through a new popup window.

```
function processRedKey()
{
   if(isThisLGEBrowser())
   {
      var userAgentString = new String(navigator.userAgent);
      var BrowserVer = userAgentString.substring(userAgentString.search(/LG Browser/) + 11);
      BrowserVer = BrowserVer.split("(", 1);
      if(BrowserVer >= "3.0.23")
      {
         //set cookie which will be checked at main_menu.html
         writeCookie("return_back_exit_test_status", "under_testing", "/", 1 * 24 * 60 * 60 * 1000);
         jsLog.lgmethod('window.NetCastBack()');
         window.NetCastBack();
      }
      else
         setInnerTextById("result_3", "Need Version over 3.0.23");
   }
   else
      setInnerTextById("result_3", "This is not a LG Browser");
}

function processGreenKey()
{
   if(isThisLGEBrowser())
   {
      jsLog.lgmethod('window.NetCastExit()');
      window.NetCastExit();
   }
   else
      setInnerTextById("result_4", "This is not a LG Browser");
}
```

Source Code of backexit.html

Source code of backexit.html is as follows:

```
<!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=UTF-8">
<title>App Template of API Unit Sample App</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/common.js"></script>
<script language="javascript" src="../../js/menu.js"></script>

<script type="text/javascript" src="../../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../../blackbirdjs/blackbird.css" />

<script>


   //initialize page
   function initPage()
   {
      //save as last visited page
      setLastVisitPage();
```

```
        //common initialize function
        commonInitialize();
        requestSourceCode();
        setPageID("Back, Exit");
        jsLog.initLG();

        //add onclick event handler
        addEventHandler(document.getElementById("btn_back"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_red"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_green"), "click", onClickHandler);
        addEventHandler(document.getElementById("btn_exit"), "click", onClickHandler);

        if(readCookie("back_exit_test_status") != "")
        {   //clear back_exit_test_status cookie if it have been set
            deleteCookie("back_exit_test_status", "/");
        }
    }


    function onUserInput(userInput)
    {
        switch(userInput)
        {
            case VK_BACK : window.location.replace("../menu_netcast.html"); break;
            case VK_RED : case 82 : processRedKey(); break;
            case VK_GREEN : case 71 : processGreenKey(); break;
        }
    }

    function processRedKey()
    {
        if(isThisLGEBrowser())
        {
            var userAgentString = new String(navigator.userAgent);
            var BrowserVer = userAgentString.substring(userAgentString.search(/LG
Browser/) + 11);
            BrowserVer = BrowserVer.split("(", 1);
            if(BrowserVer >= "3.0.23")
            {
                //set cookie which will be checked at main_menu.html
                writeCookie("back_exit_test_status", "under_testing", "/", 1 * 24 * 60 *
60 * 1000);
                jsLog.lgmethod('window.NetCastBack()');
                window.NetCastBack();
            }
            else
                setInnerTextById("result_3", "Need Version over 3.0.23");
        }
        else
            setInnerTextById("result_3", "This is not a LG Browser");
    }

    function processGreenKey()
    {
        if(isThisLGEBrowser())
        {
            jsLog.lgmethod('window.NetCastExit()');
            window.NetCastExit();
        }
        else
            setInnerTextById("result_4", "This is not a LG Browser");
    }

</script>
</head>

<body ondragstart='return false' onselectstart='return false'
onload="javascript:initPage();">

<!-- title -->
<div class='SuiteTitle'>LG Smart TV SDK | Web Open API Tutorial</div>

<!-- navigation -->
<div class='SuiteNavigation'>
    <div style="float:left;">File : /netcast/app/backexit.html</div>
</div>

<div class='SuiteTitleLine'> </div>

<!-- test contents -->
<div id='content_body' class='ContentArea'>
    <div class='ApiListTitleArea'>Web Open API List</div>
    <div class='ApiListArea'>
        <div class='MethodTitleArea'>
            Methods
            <div class='MethodListArea'>
                window.NetCastBack()<br>
                window.NetCastExit()
            </div>
        </div>
        <div class='PropertyTitleArea'
            Properties
            <div class='PropertyListArea'>

            </div>
        </div>
        <div class='EventTitleArea'>
            Events
            <div class='EventListArea'>

            </div>
        </div>
    </div>

    <div class='ViewTitleArea'>
        <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
        <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
    </div>

    <div id='view' class='ViewArea'>
        <table border="0" cellpadding="0" cellspacing="0">
            <tr height="50px">
                <td width=500px align=left><div>1. Press Red-key to test
NetCastBack()</div></td>
                <td width=200px align="left"><div id="result_1"></div></td>
                <td width=100px align="left"></td>
            </tr>
```

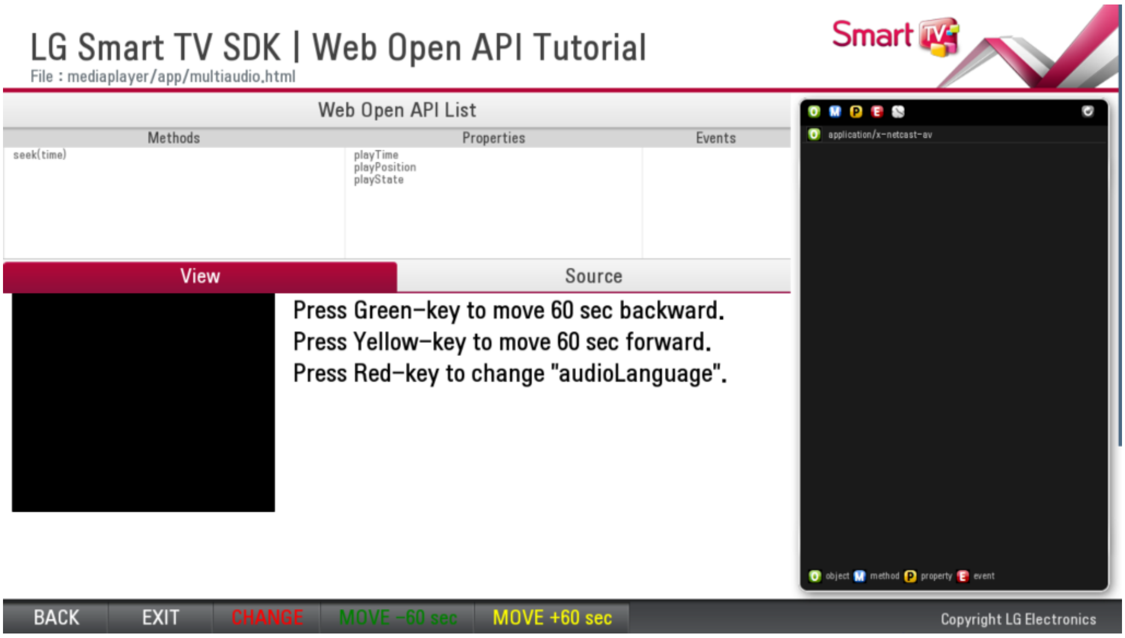
```
          <tr height="50px">
             <td width=500px align=left><div>2. Press Green-key to test 
NetCastExit()</div></td>
             <td width=200px align="left"><div id="result_2"></div></td>
             <td width=100px align="left"></td>
          </tr>
       </table>
       <table border="0" cellpadding="0" cellspacing="0">
          <tr height="50px">
             <td align=left><div id="description" class="blueColor"></div></td>
          </tr>
       </table>
   </div>

   <div style="visibility: hidden" id='codeview'>
       <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
   </div>
</div>


<!-- description -->
<div class='SuiteDescription' id='back_exit_test_description'></div>

<!-- button and copyright -->
<div class='SuiteButtonArea'>
   <!-- back key description -->
   <div id='btn_back' class='buttonDescription'>BACK</div>

   <!-- exit button -->
   <div id='btn_exit' class='buttonDescription'>EXIT</div>

   <!-- red key description -->
   <div id='btn_red' class='buttonDescription redColor'>BACK</div>

   <!-- green key description -->
   <div id='btn_green' class='buttonDescription greenColor'>EXIT</div>

   <!-- copyright -->
   <div class='copyright'>Copyright LG Electronics</div>
</div>

</body>
</html>
```

Implementing Functionality of Multi Audio

This section describes how to implement functionality of multi audio in web application.

- [Initializing the Page]
- [Inputting Keys]
- [Changing Language]
- [Moving the Video Playback Position]
- [Setting Object]
- [Source Code of multiaudio.html]

This application shows how to use the Multi Audio function. The Multi Audio function plays a video supporting multiple languages and allows **the user to listen to voice data in desired language.**



Press Green-key to move 60 sec backward.
Press Yellow-key to move 60 sec forward.
Press Red-key to change "audioLanguage".

Needed APIs
This application uses following Web Open API:

API Class	Name	Description
Method	Seek(time)	Sets the time position of playback.
Property	playTime	Returns the duration of the currently playing media item
	playPosition	Returns the play position of the currently playing media item
	playState	Returns the play state of the currently playing media item as an enumerated number.
Event	N/A	N/A

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

04: Records the last visited page when running the application.
07: Initializes the page.
08: Gets the source code of the page using the XMLHttpRequest object.
09: Sets the page ID.
10: Initializes the Log function.
13-17: Registers an event handler which will executed when the corresponding button is pressed.
19: Calls checkInitStatus() function.

```
function initPage()
{
  //save page as last visited page
  setLastVisitPage();

  //common initialize function
  commonInitialize();
  requestSourceCode();
  setPageID("Multi Audio");
  jsLog.initLG();

  //add onclick event handler
  addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
  addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
  addEventHandler(document.getElementById("btn_green"),"click",onClickHandler);
  addEventHandler(document.getElementById("btn_yellow"),"click",onClickHandler);
  addEventHandler(document.getElementById("btn_exit"),"click",onClickHandler);

 checkInitStatus();

  jsLog.lgobject('application/x-netcast-av');
}
```

Inputting Keys

Use **onUserInput** function is called by onClickHandler function; it receives a key value as the userInput parameter from onClickHandler and creates the corresponding function for each key value to operate the key.

06: When the Back key is pressed, this code is executed.
07: When the Red key is pressed, toggleAudioLanguage() is called.
08: When the Green key is pressed, movePosition(-60) is called.
09: When the Yellow key is pressed, movePosition(60) is called.

```
01 : //onUserInput function should be implemented
02 : function onUserInput(userInput)
03 : {
04 :    switch(userInput)
05 :    {
06 :       case VK_BACK: window.location.replace("../menu_commonjavascriptapi.html");
break;
07 :       case VK_RED : toggleAudioLanguage(); break;
08 :       case VK_GREEN : movePosition(-60);  break;
09 :       case VK_YELLOW : movePosition(60); break;
10 :   }
11 : }
```

Changing Language

The following code shows that the language changes when the video plays back.

checkInitStatus

Checks the playback status.
06-11: If playState is 1 (playing), toggleAudioLanguage() is called.
12: If playState is not 1, the setTimeOut() method is called to call checkInitStatus() every 1 ms.

toggleAudioLanguage

Changes the language.
16-17: Declares the curLangIdx variable and stores the supported languages in supportLanguage.
21-24: Since two languages are supported, they are stored in curLangIdx and nextlangIdx alternately.
25: Changes the language using the audioLanuage property.
28-31: Displays the currently selected language on the screen.

```
01 : //check until playstate become 1(playing) to set initial "audioLanguage"
02 : function checkInitStatus()
03 : {
04 :    var videoObj = document.getElementById("video");
05 :
06 :    if(videoObj.playState == 1)
07 :    {
08 :       jsLog.lgproperty('playState :' + videoObj.playState);
09 :
10 :       toggleAudioLanguage();
11 :    }
12 :    else{setTimeout(checkInitStatus, 1);}
13 : }
14 :
15 : //toggle "audioLanguage" among supportLanguage
16 : var curLangIdx = -1;
17 : var supportLanguage = ["en", "fr"];
18 :
19 : function toggleAudioLanguage()
20 : {
21 :    curLangIdx++;
22 :    if(curLangIdx >= supportLanguage.length){curLangIdx = 0;}
23 :    var nextlangIdx = curLangIdx + 1;
24 :    if(nextlangIdx >= supportLanguage.length){nextlangIdx = 0;}
25 :    document.getElementById("video").audioLanguage= supportLanguage[curLangIdx];
26 :    var guideStrToSet = "";
27 :
28 :    guideStrToSet +='Current "audioLanguage" is "' + supportLanguage[curLangIdx] +
'".<br>';
29 :    guideStrToSet += 'Check whether audio track is in "' +
supportLanguage[curLangIdx] + '".';
30 :
31 :    setInnerText(document.getElementById("multiAudioTestGuide"), guideStrToSet);
32 : }
```

Moving the Video Playback Position

The following code moves the current playback position of the video to the desired position using the seek API.

movePosition

04: Stores the video object in the videoObj variable.
05: Gets the total playback time of the video using the playTime property.
08: Adds the setToMove parameter value passed to this function when it was called to the current playback position (playPosition) and stores the result in the positionToSet variable.
11: If positionToSet is less than 0, positionToSet is set to 0 to move the playback position of the video to the start position.
12: If positionToSet is greater than the total playback time of the video, positionToSet is set to the totalPlayTime value to move the playback position of the video to the end position.
14: Moves the current playback position of the video to the position stored in positionToSet.

```
01 : //move position
02 : function movePosition(secToMove)
03 : {
04 :    var videoObj = document.getElementById("video");
05 :    var totalPlayTime = videoObj.playTime;
06 :    jsLog.lgproperty('playTime :' + totalPlayTime);
07 :
08 :    var positionToSet = videoObj.playPosition + (secToMove * 1000);
09 :    jsLog.lgproperty('playPosition :' + videoObj.playPosition);
10 :
11 :    if(positionToSet < 0 ){positionToSet = 0;}
```

```
    if(positionToSet > totalPlayTime){positionToSet = totalPlayTime;}

    video.seek(positionToSet);
    jsLog.lgmethod('seek(time)');
}
```

Setting Object

The following code shows how to set object.

03: Set data type. Refer to App Development Guide.
04-05: Set width and height.

```
01 : <object
02 :    id="video"
03 :    type="application/x-netcast-av"
04 :    width=300
05 :    height=250
06 :    data="../../mediafile/ multi_audio2.wmv"
07 :    style="float: left">
08 : </object>
```

Source Code of multiaudio.html

Source code of multiaudio.html is as follows:

```
<!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=UTF-8">
<title>Multi Audio Test Page</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/common.js"></script>
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/media.js"></script>
<script language="javascript" src="../../js/menu.js"></script>

<script type="text/javascript" src="../../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../../blackbirdjs/blackbird.css" />

<script>

//initialize page
function initPage()
{
   //save page as last visited page
   setLastVisitPage();

   //common initialize function
   commonInitialize();
   requestSourceCode();
   setPageID("Multi Audio");
   jsLog.initLG();

   //add onclick event handler
   addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
   addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
   addEventHandler(document.getElementById("btn_green"),"click",onClickHandler);
   addEventHandler(document.getElementById("btn_yellow"),"click",onClickHandler);
   addEventHandler(document.getElementById("btn_exit"),"click",onClickHandler);
```

```
    checkInitStatus();

    jsLog.lgobject('application/x-netcast-av');

}

function onUserInput(userInput)
{
    switch(userInput)
    {
        case VK_BACK : window.location.replace("../menu_commonjavascriptapi.html");
break;
        case VK_RED : toggleAudioLanguage(); break;
        case VK_GREEN : movePosition(-60); break;
        case VK_YELLOW : movePosition(60); break;
    }
}

//check until playstate become 1(playing) to set initial "audioLanguage"
function checkInitStatus()
{
    var videoObj = document.getElementById("video");

    if(videoObj.playState == 1)
    {
        jsLog.lgproperty('playState :' + videoObj.playState);

        toggleAudioLanguage();
    }
    else{setTimeout(checkInitStatus, 1);}
    }

    //toggle "audioLanguage" among supportLanguage
    var curLangIdx = -1;
    var supportLanguage = ["en", "fr"];

    function toggleAudioLanguage()
    {
        curLangIdx++;
        if(curLangIdx >= supportLanguage.length){curLangIdx = 0;}
        var nextlangIdx = curLangIdx + 1;
        if(nextlangIdx >= supportLanguage.length){nextlangIdx = 0;}
        document.getElementById("video").audioLanguage = supportLanguage[curLangIdx];
        var guideStrToSet = "";

        guideStrToSet += 'Current "audioLanguage" is "' + supportLanguage[curLangIdx] +
'".<br>';
        guideStrToSet += 'Check whether audio track is in "' +
supportLanguage[curLangIdx] + '".';

        setInnerText(document.getElementById("multiAudioTestGuide"), guideStrToSet);
}

//move position
function movePosition(secToMove)
{
    var videoObj = document.getElementById("video");
    var totalPlayTime = videoObj.playTime;
    jsLog.lgproperty('playTime :' + totalPlayTime);
```

```
   var positionToSet = videoObj.playPosition + (secToMove * 1000);
   jsLog.lgproperty('playPosition :' + videoObj.playPosition);

   if(positionToSet < 0 ){positionToSet = 0;}
   if(positionToSet > totalPlayTime){positionToSet = totalPlayTime;}

   video.seek(positionToSet);
   jsLog.lgmethod('seek(time)');
}

</script>
</head>


<body ondragstart='return false' onselectstart='return false'
onload="javascript:initPage();">

<!-- title -->
<div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>

<!-- navigation -->
<div class='SuiteNavigation'>
   <div style="float:left;">File : /commonjavascriptapi/app/multiaudio.html</div>
</div>

<div class='SuiteTitleLine'> </div>

<!-- test contents -->
<div id='content_body' class='ContentArea'>
   <div class='ApiListTitleArea'>Web Open API List</div>
      <div class='ApiListArea'>
         <div class='MethodTitleArea'>
            Methods
            <div class='MethodListArea'>
            seek(time)<br>
            </div>
         </div>
         <div class='PropertyTitleArea'>
            Properties
         <div class='PropertyListArea'>
            playTime<br>
            playPosition<br>
            playState<br>
audioLanguage<br>
         </div>
      </div>
      <div class='EventTitleArea'>
         Events
         <div class='EventListArea'>
         </div>
      </div>
   </div>

   <div class='ViewTitleArea'>
      <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
      <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
   </div>
```

```
    <div id = 'view'>
        <div class='ViewArea'>
            <object
                id="video"
                type="application/x-netcast-av"
                width=300
                height=250
                data="/ApiTutorial/mediafile/multi_audio2.wmv"
                style="float: left">
            </object>

            <table border="0" cellpadding="0" cellspacing="0" style="position: relative;
left: 10px; width:600px; height:150px;">
                <tr height="10%" >
                    <td ><div class="centerTestGuide ">Press Green-key to move 60 sec
backward.</div></td>
                </tr>
                <tr height=10%>
                    <td ><div class="centerTestGuide ">Press Yellow-key to move 60 sec
forward.</div></td>
                </tr>
                <tr height=10%>
                    <td ><div class="centerTestGuide ">Press Red-key to change
"audioLanguage".</div></td>
                </tr>
                <tr height=10%>
                    <td ><div class="centerTestGuide "></div></td>
                </tr>
                <tr height="10%">
                    <td align="center" valign="middle"> <div class="centerTestGuide "
id="multiAudioTestGuide"></div></td>
                </tr>
            </table>
        </div>
    </div>
    <div style="visibility: hidden" id='codeview'>
        <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
    </div>
</div>

<!-- button and copyright -->
<div class='SuiteButtonArea'>
    <!-- back key description -->
    <div id='btn_back' class='buttonDescription '>BACK</div>

    <!-- exit key description -->
    <div id='btn_exit' class='buttonDescription'>EXIT</div>

    <!-- red key description -->
    <div id='btn_red' class='buttonDescription redColor'>CHANGE</div>

    <!-- green key description -->
    <div id='btn_green' class='buttonDescription greenColor'>MOVE -60 sec</div>

    <!-- yellow key description -->
    <div id='btn_yellow' class='buttonDescription yellowColor'>MOVE +60 sec</div>

    <!-- copyright -->
    <div class='copyright '>Copyright LG Electronics</div>
```

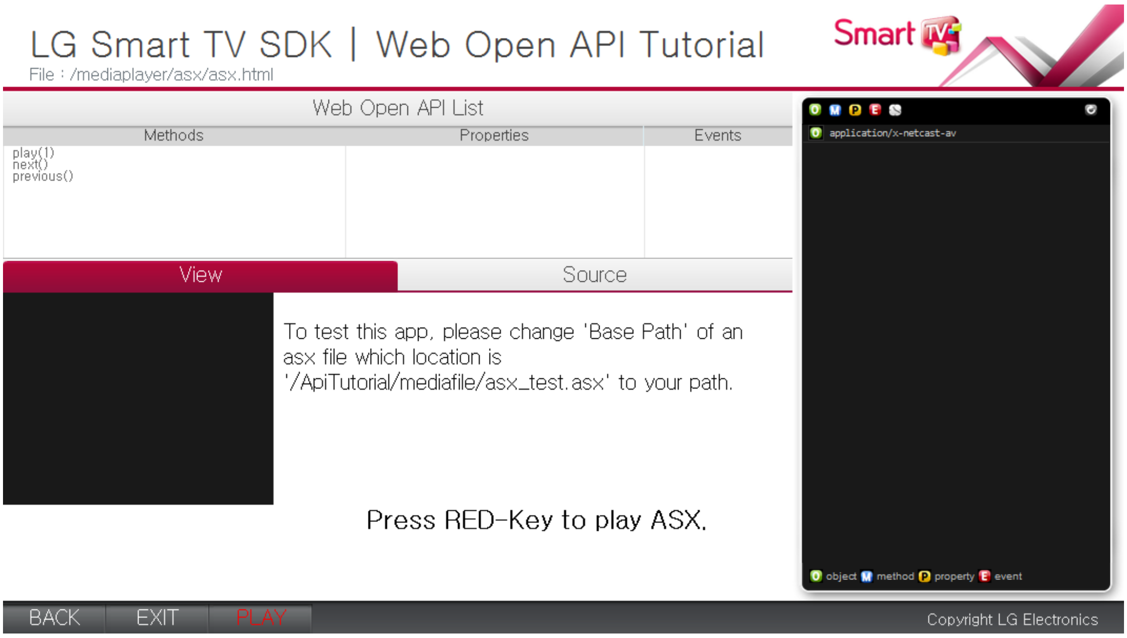
```
</div>

</body>
</html>
```

Playing Media with ASX File

This section describes how to play media with asx file.

- Initializing the Page
- Inputting Keys
- Displaying Text
- Executing Test
- Setting Object
- Source Code of asx.html

This application show how to play media with ASX file, which is play list file format, using the Web Open API of Smart TV.
Refer to **App Development Guide > Specifications >** Play list (ASX) for supported elements.



To test this app, please change 'Base Path' of an asx file which location is '/ApiTutorial/mediafile/asx_test.asx' to your path.

Press RED-Key to play ASX.

Needed APIs
This application uses following Web Open API:

API Class	Name	Description
Method	play(1)	Plays media.
	next()	Plays the next media.
	previous()	Plays the previous media.

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

04: Record the last visited page when running the application.
07: Initialize the page.
08: Get the source code of the page using the XMLHttpRequest object.
09: Set the page ID.
10: Initialize the Log function.
13-15: Add an event handler which will executed when the corresponding button is pressed.
18: Declare userAgent and set the value of navigator.userAgent.
20~23: Check if the device uses LG Browser and set true or false to isLGEBrowser.

Refer to **App Development Guide > Specifications >** userAgent string.

```
function initPage()
{
  //save page as last visited page
  setLastVisitPage();

  //common initialize function
  commonInitialize();
  requestSourceCode();
  setPageID("Asx");
  jsLog.initLG();

  //add onclick event handler
  addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
  addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
  addEventHandler(document.getElementById("btn_exit"),"click",onClickHandler);

  //check if this is LG Browser
  var userAgent = new String(navigator.userAgent);

  if (userAgent != null && userAgent.search(/LG Browser/) > -1)
     isLGEBrowser = true;
  else
     isLGEBrowser = false;

  jsLog.lgobject('application/x-netcast-av');
}
```

Inputting Keys

Use **onUserInput** function is called by onClickHandler function; it receives a key value as the userInput parameter from onClickHandler and creates the corresponding function for each key value to operate the key.

05: When the Back key is pressed, this code is executed.
07: When the Red key is pressed, this code is executed.

```
01 : function onUserInput(userInput)
02 : {
03 :  switch(userInput)
04 :  {
05 :    case VK_BACK : window.location.replace ("../menu_mediaPlayer.html");
06 :        break;
07 :    case VK_RED : case 82 changeTestProgress(); break;
08 :        break;
09 :  }
10 : }
```

Displaying Text

Use **setVideoDescription** function receives text as the descriptionToSet parameter and display the text on corresponding ID.

```
function setVideoDescription(descriptionToSet)
{
  document.getElementById("video_description").style.zIndex = "2";
  document.getElementById("test_result").style.zIndex = "1";
  setInnerTextById("video_description", descriptionToSet);
}
```

Executing Test

The **changeTestProgress** function is called when user pressed Red button.

Using switch-case, execute testing according to the value of testStep.

01: Declare initialize testStep.
05-08: Declare video, asxPropertiesDiv, descriptionDiv, and redButtonDiv and save each element.
10-53: Using switch-case, execute the functions according to the value of testStep.
12-25: (case1) If the browser is LG Browser, declare video and play media.
27-34: (case2) If the browser is LG Browser, move to the next media.
36-43: (case3) If the browser is LG Browser, move to the next media.
45-52: (case4) If the browser is LG Browser, move to the previous media.
54: Increase testStep by 1.

```
01 : var testStep = 1;
02 :
03 : function changeTestProgress()
04 : {
05 :   var video = document.getElementById("video");
06 :   var asxPropertiesDiv = document.getElementById("asxProperties");
07 :   var descriptionDiv = document.getElementById("buttonDescription");
08 :   var redButtonDiv = document.getElementById("btn_red");
09 :
10 :   switch(testStep)
11 :   {
12 :      case 1:
13 :         if(isLGEBrowser)
14 :         {
15 :            var video = document.getElementById("video");
16 :            video.play(1);
17 :            jsLog.lgmethod('video.play()');
18 :         }
19 :         else
20 :            setVideoDescription("This is not a LG Browser");
21 :
22 :         setVideoDescription("Check whether snow boarding video is displayed.");
23 :         setInnerTextById("buttonDescription", "Press RED-Key to play next
video.");
24 :         setInnerTextById("btn_red", "PLAY NEXT");
25 :         break;
26 :
27 :      case 2:
28 :         if(isLGEBrowser){video.next();}
29 :         jsLog.lgmethod('video.next()');
30 :
31 :         setVideoDescription("Check whether NetCast advertising video is
displayed.");
32 :         setInnerTextById("buttonDescription", "Press RED-Key to play next
video.");
33 :         setInnerTextById("btn_red", "PLAY NEXT");
34 :         break;
35 :
36 :      case 3:
37 :         if(isLGEBrowser){ video.next(); }
38 :         jsLog.lgmethod('video.next()');
39 :
40 :         setVideoDescription("Check whether video with timer at the bottom is
displayed.");
41 :         setInnerTextById("buttonDescription", "Press RED-Key to play previous
video.");
42 :         setInnerTextById("btn_red", "PLAY PREVIOUS");
43 :         break;
44 :
45 :     case 4:
        if(isLGEBrowser){video.previous();}
        jsLog.lgmethod('video.previous()');

        setVideoDescription("Check whether NetCast advertising video is displayed.");
        setInnerTextById("buttonDescription", "No more test is left on this page.");
        document.getElementById("btn_red").style.visibility = "hidden";
        break;
  }
  testStep++;
}
```

Setting Object

The following code shows how to set object.

03: Set data type. Refer to App Development Guide.
04-05: Set width and height.

```
01 : <object
02 :   id="video"
03 :   type="application/x-netcast-av"
04 :   width=300
05 :   height=250
06 :   data="../../mediafile/asx_test.asx"
07 :   style="float: left">
08 :   autoStart=false>
09 : </object>
```

Source Code of asx.html

Source code of asx.html is as follows:

```
<!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=UTF-8">
<title>ASX Test Page</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/common.js"></script>
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/menu.js"></script>
<script language="javascript" src="../../js/media.js"></script>

<script type="text/javascript" src="../../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../../blackbirdjs/blackbird.css" />

<script>

    //initialize page
    function initPage()
    {
        //save page as last visited page
        setLastVisitPage();

        //common initialize function
        commonInitialize();
        requestSourceCode();
        setPageID("Asx");
```

```
   jsLog.initLG();

   //add onclick event handler
   addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
   addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
   addEventHandler(document.getElementById("btn_exit"),"click",onClickHandler);

   //check if this is LG Browser
   var userAgent = new String(navigator.userAgent);

   if (userAgent != null && userAgent.search(/LG Browser/) > -1)
      isLGEBrowser = true;
   else
      isLGEBrowser = false;

   jsLog.lgobject('application/x-netcast-av');
}

//onUserInput function should be implemented
function onUserInput(userInput)
{
   switch(userInput)
   {
      case VK_BACK :
         window.location.replace("../menu_mediaPlayer.html"); break;
      case VK_RED : case 82 : changeTestProgress(); break;
   }
}

//set video description
function setVideoDescription(descriptionToSet)
{
   document.getElementById("video_description").style.zIndex = "2";
   document.getElementById("test_result").style.zIndex = "1";
   setInnerTextById("video_description", descriptionToSet);
}

//change test progress
var testStep = 1;
function changeTestProgress()
{
   var video = document.getElementById("video");
   var asxPropertiesDiv = document.getElementById("asxProperties");
   var descriptionDiv = document.getElementById("buttonDescription");
   var redButtonDiv = document.getElementById("btn_red");

   switch(testStep)
   {
      case 1:
         if(isLGEBrowser)
         {
            var video = document.getElementById("video");
            video.play(1);
            jsLog.lgmethod('video.play()');
         }
         else
            setVideoDescription("This is not a LG Browser");

         setVideoDescription("Check whether snow boarding video is displayed.");
         setInnerTextById("buttonDescription", "Press RED-Key to play next
video.");
             setInnerTextById("btn_red", "PLAY NEXT");
             break;

          case 2:
             if(isLGEBrowser){video.next();}
             jsLog.lgmethod('video.next()');

             setVideoDescription("Check whether NetCast advertising video is 
displayed.");
             setInnerTextById("buttonDescription", "Press RED-Key to play next 
video.");
             setInnerTextById("btn_red", "PLAY NEXT");
             break;

          case 3:
             if(isLGEBrowser){       video.next(); }
             jsLog.lgmethod('video.next()');

             setVideoDescription("Check whether video with timer at the bottom is 
displayed.");
             setInnerTextById("buttonDescription", "Press RED-Key to play previous 
video.");
             setInnerTextById("btn_red", "PLAY PREVIOUS");
             break;

          case 4:
             if(isLGEBrowser){video.previous();}
             jsLog.lgmethod('video.previous()');

             setVideoDescription("Check whether NetCast advertising video is 
displayed.");
             setInnerTextById("buttonDescription", "No more test is left on this 
page.");
             document.getElementById("btn_red").style.visibility = "hidden";
             break;
          }
      testStep++;
   }


</script>

</head>
<body ondragstart='return false' onselectstart='return false' 
onload="javascript:initPage();">
   <!-- title -->
   <div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>

   <!-- navigation -->
   <div class='SuiteNavigation'>
      <div style="float:left;">File : /mediaplayer/asx/asx.html</div>
   </div>

   <div class='SuiteTitleLine'> </div>

   <!-- test contents -->
   <div class='ContentArea'>
      <div class='ApiListTitleArea'>API List</div>
      <div class='ApiListArea'>
```

```
          <div class='MethodTitleArea'>
             Methods
             <div class='MethodListArea'>
                 play(1)<br>
                 next()<br>
                 previous()
             </div>
          </div>
          <div class='PropertyTitleArea'>
             Properties
             <div class='PropertyListArea'>

             </div>
          </div>
          <div class='EventTitleArea'>
             Events
             <div class='EventListArea'>

             </div>
          </div>
       </div>

       <div class='ViewTitleArea'>
          <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
          <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
       </div>

       <div id='view'>
          <div class='ViewArea'>
             <object
                 id="video"
                 type="application/x-netcast-av"
                 width=300
                 height=250
                 data="/ApiTutorial/mediafile/asx_test.asx"
                 style="float: left"
                 autoStart=false
             >
          </object>

          <!-- video description -->
          <div id='test_result' >
             <table width="550px" height="200px" border="0" cellpadding="0"
cellspacing="0" style="position :relative; top: 10px; left:10px; font-size: 25px">
                <tr><td>To test this app, please change 'Base Path' of an asx file
which location is '/ApiTutorial/mediafile/asx_test.asx' to your path.</td></tr>
                <tr ><td><div id="video_description" class="eachTestGuide"
style="font-size: 25px"></div></td></tr>
            </table>
          </div>

          <br><br>
          <div id='buttonDescription' style="font-size: 30px; text-align: center; ">
Press RED-Key to play ASX.</div>
       </div>
   </div>

   <div style="visibility: hidden" id='codeview'>
```

```
      <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
   </div>

</div>

<!-- button and copyright -->
<div class='SuiteButtonArea'>
   <!-- back key description -->
   <div id='btn_back' class='buttonDescription '>BACK</div>

   <!-- exit key description -->
   <div id='btn_exit' class='buttonDescription'>EXIT</div>

   <!-- red key description -->
   <div id='btn_red' class='buttonDescription redColor'>PLAY</div>

   <!-- copyright -->
   <div class='copyright'>Copyright LG Electronics</div>
</div>

</body>
</html>
```

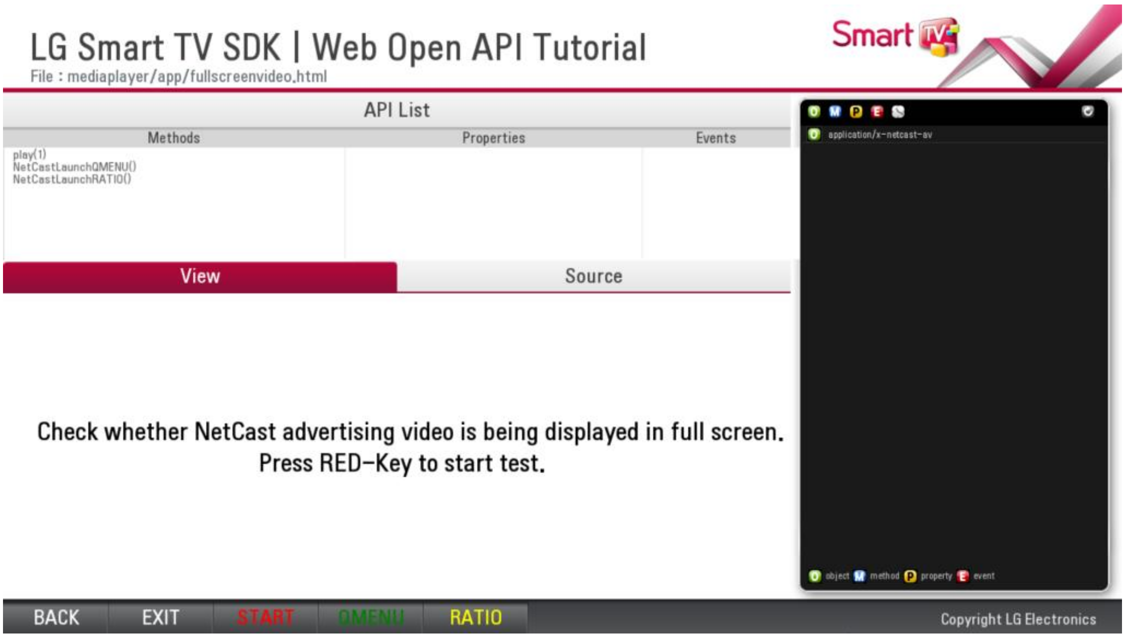
Playing Video in Full Screen Mode

This section describes how to play video in full screen mode.

- Initializing the Page
- Inputting Keys
- Displaying and Playing Media Object in Full Screen Mode
- Setting Object
- Source Code of fullscreenvideo.html

This application shows how to play a video in full screen mode.
The LG Smart TV SDK provides no separate Web Open API that plays media content in full screen mode.

Basically, when the start position of the media object is set to (0, 0) and its size is set to 1280 X 720, the content is played in full screen mode. In this application, the width and height are set within the <body> and </body> tags and the elements used to compose the screen are hidden using the code 'style.visibility = “hidden”' so that only the video is played in full screen mode.



LG Smart TV SDK | Web Open API Tutorial
File : mediaplayer/app/fullscreenvideo.html

API List

Methods	Properties	Events

play(1)
NetCastLaunchQMENU()
NetCastLaunchRATIO()		

Check whether NetCast advertising video is being displayed in full screen.
Press RED-Key to start test.

application/x-netcast-av

Copyright LG Electronics

Needed APIs
This application uses following Web Open API:

API Class	Name	Description
Method	Play(1)	Plays media.
	NetCastLaunchQMENU	Launches QMENU over a full screen video.
	NetCastLaunchRATIO	Sets aspect ratio of full screen video.
Property	N/A	N/A
Event	N/A	N/A

Note

Log is used for checking the sequence of Web Open API; this will not be covered in the sample code description.

Initializing the Page

Use the **initPage** function to set the basic functions of the application.

04: Records the last visited page when running the application.
07: Initializes the page.
08: Gets the source code of the page using the XMLHttpRequest object.
09: Sets the page ID.

10: Initializes the Log function.
13-15: Registers an event handler which will executed when the corresponding button is pressed.

```
01 : function initPage()
02 : {
03 :   //save page as last visited page
04 :   setLastVisitPage();
05 :
06 :   //common initialize function
07 :   commonInitialize();
08 :   requestSourceCode();
09 :   setPageID(“Full Screen");
10 :   jsLog.initLG();
11 :
12 :   //add onclick event handler
13 :   addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
14 :   addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
15 :   addEventHandler(document.getElementById("btn_exit"),"click",onClickHandler);
16 :
17 :   jsLog.lgobject('application/x-netcast-av');
18 : }
```

Inputting Keys

Use **onUserInput** function is called by onClickHandler function; it receives a key value as the userInput parameter from onClickHandler and creates the corresponding function for each key value to operate the key.

06: When the Back key is pressed, this code is executed.
07: When the Red key is pressed, changeTestProgress() is called.
08:   When the Green key is pressed, NetCastLaunchQMENU() is called.
10: When the Yellow key is pressed, NetCastLaunchRATIO() is called.

```
01 : //onUserInput function should be implemented
02 : function onUserInput(userInput)
03 : {
04 :   switch(userInput)
05 :   {
06 :      case VK_BACK : window.location.replace("../menu_mediaPlayer.html"); break;
07 :      case VK_RED : case 82 : changeTestProgress(); break;
08 :      case VK_GREEN : window.NetCastLaunchQMENU();
09 :         jsLog.lgmethod('NetCastLaunchQMENU()'); break;
10 :      case VK_YELLOW : window.NetCastLaunchRATIO();
11 :         jsLog.lgmethod('NetCastLaunchRATIO()'); break;
12 :   }
13 : }
```

Displaying and Playing Media Object in Full Screen Mode

The following function displays and plays a media object in full screen mode.

changeTestProgress

Changes the test progress.
03-05: Declares the video, bodycontent, and sourcecode variables.
07: Plays the video.
09: Sets the visibility of the video to "visible" to display it on the screen.
11-12: Sets the visibility of both of the bodycontent and sourcecode to "hidden" to hide it from the screen.
13: Sets the visibility of the Red key to "hidden" to hide it from the menu bar.

```
01 : function changeTestProgress()
02 : {
03 :   var video = document.getElementById("video");
04 :   var bodycontent = document.getElementById("bodycontent");
```

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

  video.play(1);
  jsLog.lgmethod('video.play(1)');
  video.style.visibility="visible";

  bodycontent.style.visibility="hidden";
  sourcecode.style.visibility = "hidden";
  document.getElementById("btn_red").style.visibility="hidden";
}
```

Setting Object

The following code shows how to set object.

03: Set data type. Refer to App Development Guide.
04-05: Set width and height.

```
01 : <object
02 :   id="video"
03 :   type="application/x-netcast-av"
04 :   width=1280
05 :   height=720
06 :   style="float: left; z-index: 1; visibility: hidden;"
07 :   autoStart=false
08 :   data="../../mediafile/NetCastGeneratorClient.avi">
09 : </object>
```

Source Code of fullscreenvideo.html

Source code of fullscreenvideo.html is as follows:

```
<!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=UTF-8">
<title>Full Screen Video Test Page</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/common.js"></script>
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/menu.js"></script>
<script language="javascript" src="../../js/media.js"></script>

<script type="text/javascript" src="../../blackbirdjs/blackbird.js"></script>
<link type="text/css" rel="Stylesheet" href="../../blackbirdjs/blackbird.css" />
<style type="text/css">
   .OverLay { position: relative; z-index:10; opacity: 1; filter: alpha(opacity
=100); }
   body { height: 100%; }
   html { height: 100%; }
</style>
<script>

   //initialize page
   function initPage()
   {
      //save page as last visited page
      setLastVisitPage();

      //common initialize function
      commonInitialize();
      requestSourceCode();
      setPageID("Full Screen");
      jsLog.initLG();

      //add onclick event handler
      addEventHandler(document.getElementById("btn_back"), "click", onClickHandler);
      addEventHandler(document.getElementById("btn_exit"), "click", onClickHandler);
      addEventHandler(document.getElementById("btn_red"), "click", onClickHandler);
      addEventHandler(document.getElementById("btn_green"), "click", onClickHandler);
      addEventHandler(document.getElementById("btn_yellow"), "click",
onClickHandler);

      jsLog.lgobject('application/x-netcast-av');
   }

   function onUserInput(userInput)
   {
      switch(userInput)
      {
         case VK_BACK : window.location.replace("../menu_mediaPlayer.html"); break;
         case VK_RED : case 82 : changeTestProgress(); break;
         case VK_GREEN : window.NetCastLaunchQMENU();
            jsLog.lgmethod('NetCastLaunchQMENU()'); break;
         case VK_YELLOW : window.NetCastLaunchRATIO();
            jsLog.lgmethod('NetCastLaunchRATIO()'); break;
       }
   }
   function changeTestProgress()
   {
      var video = document.getElementById("video");
      var bodycontent = document.getElementById("bodycontent");
      var sourcecode = document.getElementById("sourcecode");

      video.play(1);
      jsLog.lgmethod('video.play(1)');
      video.style.visibility="visible";

      bodycontent.style.visibility="hidden";
      sourcecode.style.visibility = "hidden";
      document.getElementById("btn_red").style.visibility="hidden";
   }

</script>
</head>

<body ondragstart='return false' onselectstart='return false'
onload="javascript:initPage();">
<div id='bodycontent'>
   <div>
      <object
         id="video"
         type="application/x-netcast-av"
         width=1280
         height=720
         style="float: left; z-index: 1; visibility: hidden;"
         autoStart=false
         data="../../mediafile/NetCastGeneratorClient.avi">
      </object>
   </div>
   <!-- title -->
   <div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>

   <!-- navigation -->
   <div class='SuiteNavigation'>
      <div style="float:left;">File : mediaplayer/app/fullscreenvideo.html</div>
   </div>

   <div class='SuiteTitleLine'> </div>

   <!-- test contents -->
   <div class='ContentArea'>
      <div class='ApiListTitleArea '>API List</div>
      <div class='ApiListArea OverLay'>
         <div class='MethodTitleArea'>
            Methods
            <div class='MethodListArea'>
            play(1)<br>
            NetCastLaunchQMENU()<br>
            NetCastLaunchRATIO()
            </div>
         </div>
         <div class='PropertyTitleArea'>
            Properties
            <div class='PropertyListArea'>
            </div>
         </div>
         <div class='EventTitleArea'>
            Events
            <div class='EventListArea'>

            </div>
         </div>
      </div>

      <div class='ViewTitleArea'>
         <div id='tabViewArea' class='SelectedViewArea' style='float:left;'
onclick="showView();">View</div>
         <div id='tabCodeArea' class='UnselectedViewArea' style='float:right;'
onclick='showCode();'>Source</div>
      </div>

      <div id='view'>
         <div class='ViewArea'>
            <table width="100%" height="100%" cellpadding="0" cellspacing="0"><tr><td
align="center" valign="middle">
            <div class='centerTestGuide '>Check whether NetCast advertising video is
being displayed in full screen.<br>Press RED-Key to start test.</div>
            </td></tr></table>
         </div>
      </div>

      <div style="visibility: hidden" id='codeview'>
         <textarea class="SourceCodeArea" value=""  id='sourcecode'></textarea>
      </div>
   </div>

</div>
```

```
<!-- button and copyright -->
<div class='SuiteButtonArea '>
   <!-- back key description -->
   <div id='btn_back' class='buttonDescription'>BACK</div>

   <!-- exit key description -->
   <div id='btn_exit' class='buttonDescription'>EXIT</div>

   <!-- red key description -->
   <div id='btn_red' class='buttonDescription redColor'>START</div>

   <!-- green key description -->
   <div id='btn_green' class='buttonDescription greenColor'>QMENU</div>

   <!-- yellow key description -->
   <div id='btn_yellow' class='buttonDescription yellowColor'>RATIO</div>

   <!-- copyright -->
   <div class='copyright '>Copyright LG Electronics</div>
</div>

</body>
</html>
```