International Telecommunication Union

ITU-T Technical Paper

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

(29 March 2019)

HSTP.CONF-H764
Conformance testing specification for H.764

ITU-T

International Telecommunication Union

Summary

This Technical Paper defines the conformance testing items for Recommendation ITU-T H.764 "*IPTV Service Enhanced Script Language*". It gives testing properties and the sample codes to be tested for web-based IPTV multimedia interaction.

This technical paper contains an electronic attachment found at https://www.itu.int/wftp3/Public/t/testsignal/SpeVideo/HSTP-CONF-H764/v2019_03/HSTP-CONF-H764-2019-Test_signals.zip.

Keywords

Conformance testing, IPTV multimedia interaction, script language

Change Log

This document contains Version 1 of the ITU-T Technical Paper on "*IPTV Service Enhanced Script Language*" approved at the ITU-T Study Group 16 meeting held in Geneva, 19-29 March 2019.

Editors:

Guqiao Zhu
China Telecom
P. R. China
Tel: + 86 21 58756453
Fax: + 86 21 58754490
E-mail: zhugq@sttri.com.cn

Chuanyang Miao
ZTE Corporation
P. R. China
Tel: +86-25-88014611
Fax: +86-25-88014843
Email: miao.chuanyang@zte.com.cn

CONTENTS

Page

List of Tables

Page

List of Figures

Page

Technical Paper ITU-T HSTP-CONF-H764

Conformance testing specification for H.764

1 Scope

This document defines the conformance testing items for Recommendation ITU-T H.764 "IPTV Service Enhanced Script Language (IPTV SESL)".

This technical paper has an electronic attachment that includes a reference video and test codes for test sample 1. The electronic attachment can be found in the ITU-T Test Signal Database at https://www.itu.int/wftp3/Public/t/testsignal/SpeVideo/HSTP-CONF-H764/v2019_03/HSTP-CONF-H764-2019-Test_signals.zip.

2 References

[ITU-T H.764] Recommendation ITU-T H.764 (2012), *IPTV Service Enhanced Script Language*.

[ITU-T H.762] Recommendation ITU-T H.762 (2011), *Lightweight interactive multimedia environment (LIME) for IPTV services*.

[ITU-T HSTP-CONF-H762] Technical Paper ITU-T HSTP-CONF-H762 (2013), *Conformance testing specification for H.762*.

3 Terms and definitions

3.1 Terms defined elsewhere

This Technical Paper uses the following terms defined elsewhere:

3.1.1 ECMAScript [b-ISO/IEC 16262]: The programming language defined by [b-ISO/IEC 16262].

3.1.2 Internet protocol television (IPTV) [ITU-T Y.1901]: Multimedia services such as television/video/audio/text/graphics/data delivered over IP-based networks that are managed to support the required level of QoS/QoE, security, interactivity and reliability.

3.1.3 linear TV [ITU-T Y.1901]: A television service in which a continuous stream flows in real time from the service provider to the terminal device and where the user cannot control the temporal order in which contents are viewed.

3.1.4 video-on-demand (VoD) [ITU-T Y.1901]: A service in which the end user can, on demand, select and view video content and where the end user can control the temporal order in which the video content is viewed (e.g., the ability to start the viewing, pause, fast forward, rewind, etc.).

3.1.5 trick mode functionality [ITU-T Y.1901]: The ability to pause, rewind or forward stored content.

3.1.6 TV with trick mode [ITU-T Y.1901]: TV service with trick mode functionality.

3.1.7 network personal video recorder (nPVR) [ITU-T Y.1901]: This is the same as PVR except that the recording device is located at the service provider's premises.

3.1.8 IPTV terminal device [b-ITU-T Y.1901]: A terminal device which has IPTV terminal function (ITF) functionality, e.g., a set-top box (STB).

3.1.9 user agent [b-W3C WebArch]: One type of Web agent; a piece of software acting on behalf of a person.

3.1.10 metadata [ITU-T Y.1901]: Structured, encoded data that describe characteristics of information-bearing entities to aid in the identification, discovery, assessment and management of the described entities.

3.2 Terms defined here

None.

4 Abbreviations

DOM	Document Object Model
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
LIME	Lightweight Interactive Multimedia Environment
PVR	Personal Video Recorder
TD	Terminal Device
URL	Universal Resource Location
XML	Extensible Markup Language

5 Conventions

None.

6 Introduction

Recommendation ITU-T H.764 provides an object-oriented programming language called "IPTV service enhanced script language", which is based on LIME-Script of [ITU-T H.762] with enhanced functionalities for IPTV services.

This document describes those points of [ITU-T H.764] that should be tested for conformance and interoperability. The “Core Script Profile” testing can refer to [ITU-T HSTP-CONF-H.762]. The details of “Extended Script Profile” testing are to be done using the test codes provided in the Appendix of this document.

7 Test assertions – Core ECMAScript methods and properties

ECMAScript is applied in web-based IPTV service to provide performing computations and interoperability with the end users within a host environment. The subset of ECMAScript, which is described in “Core Script Profile” section of [ITU-T H.764], has already been defined as LIME-Script in [ITU-T H.762].

For the correspondence of [ITU-T H.764] and [ITU-T H.762] Core Script part, test requirements and methods should be implemented and referred to clause 10 of [ITU-T HSTP-CONF-H.762]. The following built-in objects of Core Script Profile, as described in [ITU-T H.764] are required to be supported.

– Global
– Object
– Function
– Array
– String
– Boolean

– Number

– Date

8 Test assertions – Extended ECMAScript methods and properties

8.1 MediaController Object

The MediaController object encapsulates the capabilities of the terminal device to play live channel, nPVR, VoD, music and other media types. The MediaController object can provide necessary methods and properties to control a media. The following mandatory properties and methods are required to be supported.

8.1.1 MediaController

8.1.1.1 Property

The following mandatory properties of MediaController as described in [ITU-T H.764] are required to be supported.

Table 1 – Mandatory properties of MediaController object

Property	Description
allowTrickmode	The trick mode (fast forward/fast rewind/pause) operation during the life cycle of the player
currentPlayTime	Current position playing of the media
channelNum	The number of current channel
cycleFlag	Media's loop play
height	Height of the video output (in pixels)
instanceId	Unique ID of the local media player instance created
left	Relative excursion from the left-top corner of the browser window (in pixel)
playMode	Play mode of the MediaController
width	Width of the video output (in pixels)
mute	Mute/sound status of the MediaController
mediaCode	Unique identifier of the current media played by the MediaController instance
mediaDuration	Duration of the current media (in seconds)
playbackMode	Playback status of the MediaController
top	Relative excursion from the left-top corner of the browser window (in pixels)
videoDisplayMode	Video display mode of the MediaController

8.1.1.2 Method

The following mandatory methods of MediaController are required to be supported.

8.1.1.2.1 Constructor

- `MediaController ( ): to construct a MediaController object`

  `Syntax:`

  ```
  MediaController( )
  ```

8.1.1.2.2 bindNativePlayerInstance

- bindNativePlayerInstance ( ): to bind an existing native player instance

Syntax:

```
number bindNativePlayerInstance(input number instanceID)
```

8.1.1.2.3 fastForward

- fastForward ( ): to fast forward with the rate indicated by "speed"

Syntax:

```
fastForward(input number speed)
```

8.1.1.2.4 fastRewind

- fastRewind ( ): to fast rewind with the rate indicated by "speed"

Syntax:

```
fastRewind(input number speed)
```

8.1.1.2.5 getVolume

- getVolume ( ): to get the current volume of the IPTV TD

Syntax:

```
number getVolume()
```

8.1.1.2.6 gotoEnd

- gotoEnd ( ): to jump to the end of the media

Syntax:

```
gotoEnd()
```

8.1.1.2.7 gotoStart

- gotoStart ( ): to jump to the beginning of the media.

Syntax:

```
gotoStart()
```

8.1.1.2.8 initMediaController

- initMediaController ( ): to initialize some properties of a MediaController object

Syntax:

```
number initMediaController(input number instanceID,
                           input number playlistFlag,
                           input number videoDisplayMode,
                           input number height,
                           input number width,
                           input number left,
                           input number top,
                           input number muteFlag,
                           input number useNativeUIFlag,
                           input number subtitleFlag,
                           input number videoAlpha,
input number cycleFlag)
```

8.1.1.2.9 joinChannel

- joinChannel ( ): to join in a channel

Syntax:

```
number joinChannel(input number userchannelid)
```

8.1.1.2.10 launchIPTVContent

- launchIPTVContent ( ): to launch an IPTV content

Syntax:

```
number launchIPTVContent(input String content_uri,
                         input String ret_uri,
                         input number start_npt[,
input number license_id])
```

8.1.1.2.11 leaveChannel

- leaveChannel ( ): to quit the current channel

Syntax:

```
number leaveChannel()
```

8.1.1.2.12 pause

- pause ( ): to pause the media which is playing

Syntax:

```
pause()
```

8.1.1.2.13 playFromStart

- playFromStart ( ): to play from the beginning of the media

Syntax:

```
number playFromStart()
```

8.1.1.2.14 playByTime

- playByTime ( ): to play from a certain time of the current media

Syntax:

```
number playByTime(input number type,input String timestamp[,input
number speed])
```

8.1.1.2.15 refreshVideoDisplay

- refreshVideoDisplay ( ): to adjust the display of the video

Syntax:

```
refreshVideoDisplay()
```

8.1.1.2.16 releaseMediaController

- releaseMediaController ( ): to release the resources occupied by the player

Syntax:

```
number releaseMediaController(input number instanceID)
```

8.1.1.2.17 resume

- resume ( ): to resume from the pause/forward/rewind mode of the current media

Syntax:

```
resume()
```

8.1.1.2.18 setSingleMedia

- setSingleMedia ( ): to locate the resource of a single media

Syntax:

```
setSingleMedia(input anySimpleType mediaStr)
```

8.1.1.2.19 setVolume

- setVolume ( ): to set the volume of the IPTV TD

Syntax:

```
setVolume(input number volume)
```

8.1.1.2.20 stop

- stop ( ): to stop the current media

Syntax:

```
stop()
```

8.1.2 Graphic user interface

Properties and methods are optional. For more details, see clause 8.1.2 of [ITU-T H.764].

8.1.3 Audio and subtitles

Properties and methods are optional. For more details, see clause 8.1.3 of [ITU-T H.764].

8.1.4 Playlist

Properties and methods are optional. For more details, see clause 8.1.4 of [ITU-T H.764].

8.2 Event object

The Event object is designed for event notification from the Service Component Layer of the IPTV TD to the web-based user agent. The types of events include channel change, play mode change, system error and so on.

8.2.1 Event type

8.2.1.1 EVENT_GO_CHANNEL

The event is triggered when a new channel is joined by the IPTV TD. The following mandatory properties as described in [ITU-T H.764] are required to be supported.

Table 2 – Mandatory properties of EVENT_GO_CHANNEL

Property	Description
type	The value is EVENT_GO_CHANNEL.
instance_id	the native MediaController instance ID of IPTV TD
channel_code	the unified code of the channel

8.2.1.2 EVENT_MEDIA_END

The event is triggered when the media in the MediaController is played to the end. The following mandatory properties are required to be supported.

Table 3 – Mandatory properties of EVENT_MEDIA_END

Property	Description
type	The value is EVENT_MEDIA_END.
instance_id	the native MediaController instance ID of IPTV TD

8.2.1.3 EVENT_MEDIA_BEGINNING

The event is triggered when the media in the MediaController is played to the start. The following mandatory properties are required to be supported.

Table 4 – Mandatory properties of EVENT_MEDIA_BEGINNING

Property	Description
type	The value is EVENT_MEDIA_BEGINNING.
instance_id	the native MediaController instance ID of IPTV TD

8.2.1.4 EVENT_MEDIA_ERROR

The event is triggered when some errors happen in the MediaController. The following mandatory properties are required to be supported.

Table 5 – Mandatory properties of EVENT_MEDIA_ERROR

Property	Description
type	The value is EVENT_MEDIA_ERROR.
instance_id	the native MediaController instance ID of IPTV TD
error_code	the code of the error

8.2.1.5 EVENT_PLAYMODE_CHANGE

The event is triggered when the playback mode of the MediaController changes. The following properties are required to be supported.

Table 6 – Properties of EVENT_PLAYMODE_CHANGE

Property	Description
type	The value is EVENT_PLAYMODE_CHANGE.
instance_id	the native MediaController instance ID of IPTV TD
new_play_mode	It is the play mode after the play mode changes.
old_play_mode	It is the play mode before the play mode changes.
new_play_rate	It is the play rate after the play mode changes. This property is conditionally provided when the new play mode is trick mode.
old_play_rate	It is the play rate before the play mode changes. This property is conditionally provided when the old play mode is trick mode.

8.2.1.6 EVENT_REMINDER

The event is triggered when some messages need to be reminded. The following mandatory properties are required to be supported.

Table 7 – Mandatory properties of EVENT_REMINDER

Property	Description
type	The value is EVENT_REMINDER.
message	the content of the reminder

8.3 Service object

The Service object is used by IPTV service providers to specify necessary service information for service procedure of end users.

8.3.1 Parental control

8.3.1.1 Properties

The following mandatory properties on parental control are required to be supported.

Table 8 – Mandatory properties of service object on parental control

Property	Description
parentalCtrlEnabled	be used to enable/disable the function of parental control
parentalCtrlPassword	the parental control password

8.3.2 Licence & DRM

8.3.2.1 Properties

The following mandatory property on license and DRM is required to be supported.

Table 9 – Mandatory properties of service object on license & DRM

Property	Description
drmSystem	the location identifier of DRM system

8.4 XMLHttpRequest object

The XMLHttpRequest object implements an interface exposed by a scripting engine that allows scripts to perform HTTP client functionality, such as submitting form data or loading data from a server.

8.4.1 Properties

The following mandatory properties of XMLHttpRequest object are required to be supported.

Table 10 – Mandatory properties of the XMLHttpRequest object

Property	Description
onreadystatechange	event handler for an event that is triggered at every state change
readyState	be used inside the event handler function that processes request object state change events
responseText	string version of data returned from server process
responseXML	DOM-compatible document object of data returned from server process
status	numeric status code returned by server
statusText	string message accompanying the status code

8.4.2 Methods

The following mandatory methods of XMLHttpRequest object are required to be supported.

8.4.2.1 Constructor

- `XMLHttpRequest( ): to construct a XMLHttpRequest object with default value`

Syntax:

```
XMLHttpRequest( )
```

8.4.2.2 abort

- `abort( ): to stop the current request`

Syntax:

```
abort( )
```

8.4.2.3 getAllResponseHeaders

- getAllResponseHeaders ( ): to get a complete set of headers (labels and values)

Syntax:

```
String getAllResponseHeaders( )
```

8.4.2.4 getResponseHeader

- getResponseHeader ( ): to return the string value of a single header label

Syntax:

```
String getResponseHeader(input String headerLabel)
```

8.4.2.5 open

- open ( ): to assign destination URL, method, and other optional attributes of a pending request

Syntax:

```
open(input String method,
    input String URL
    [, input Boolean asyncFlag
    [, input String userName
[,input String password]]])
```

8.4.2.6 send

- send ( ): to transmit the request to the server

Syntax:

```
send(input String content)
```

8.4.2.7 setRequestHeader

- setRequestHeader ( ): to assign a label/value pair to the header to be sent with a request

Syntax:

```
setRequestHeader(input String headerlabel,
                 input String value)
```

Appendix I

H.764 sample test codes

I.1 Test Sample 0

This test sample is designed to verify the mandatory properties and methods of MediaController object in this TP.

Deploying requirements:

– Each of the following code snippets is deployed as one test page.

– The naming of each test page is required to be consistent with the property value of HTML TITLE (e.g. the test page of code snippet 2 should be named as VOD_Test_1.html).

– All the test pages are required to be deployed in the same folder on the web server (e.g. /MediaController/VOD_Test_1.html).

– Each background image is required to have the same name as the corresponding test page.

– All the test images are required to be deployed in the subfolder of test pages on the web server (e.g. /MediaController/img/VOD_Test_1.jpg).

– Multicast-based and unicast-based IPTV content is required to be deployed on the video content server and can be delivered to the terminal device.

– The metadata of deployed content is required to be re-configured in the corresponding code snippets of test pages.

NOTE – Reference video for this appendix is contained in the electronic attachment.

The composition of five test pages of this sample and the configuration requirements are presented in the following Table I.1.

Table I.1 – Test page composition and configuration requirements (sample 0)

Test Identifier	Test Page	Relevant Small Pages	Background image	Content & Metadata Configuration
S0T1	VOD_Test_1.html (code snippet 2)	VOD_Test_1_Smallvod.html (code snippet 1)	VOD_Test_1.jpg (Figure I.1-1-a)	A VOD content with configured metadata in code snippet 1
S0T2	VOD_Test_2.html (code snippet 3)	VOD_Test_1_Smallvod.html (code snippet 1)	VOD_Test_2.jpg (Figure I.1-2-a)	Same as S0T1
S0T3	VOD_Test_3.html (code snippet 5)	VOD_Test_3_Smallvod (code snippet 4)	VOD_Test_3.jpg (Figure I.1-3-a)	A VOD content with configured metadata in code snippet 4
S0T4	VOD_Test_4.html (code snippet 7)	VOD_Test_1_Smallvod.html (code snippet 1) VOD_Test_4_Smallvod.html (code snippet 6)	VOD_Test_4.jpg (Figure I.1-4-a)	Same as S0T1
S0T5	VOD_Test_5.html (code snippet 10)	VOD_Test_5_Smallvod1.html (code snippet 8) VOD_Test_5_Smallvod2.html (code snippet 9)	VOD_Test_5.jpg (Figure I.1-5-a)	Two linear TV streams with configured channel ID a VOD content with configured metadata in code snippet 9
NOTE – The switch between test pages can be re-configured as required in the corresponding code snippets (2/3/5/7/10).				

Test guide:

– Read the brief introduction of codes on the test page before testing.

– Follow the guideline on test page and choose test options one by one.

– Validate the property values of MediaController object and the playback effect of video and audio by comparing to the corresponding reference images and described expected behaviour.

Test sample 0 code snippet 1

```
<!-- create a new MediaController object in the small window with specific location and
size -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
 <HEAD>
  <TITLE> VOD_Test_1_Smallvod </TITLE>
  <META NAME="Generator" CONTENT="EditPlus">
  <META NAME="Author" CONTENT="">
  <META NAME="Keywords" CONTENT="">
  <META NAME="Description" CONTENT="">
 </HEAD>
<script language="javascript">
var mp;
var InstanceID = -1;

function pageload()
{
      //new a MediaController object
      mp = new MediaController();
      InstanceID = mp.instanceId;
      //initialization of video play mode, location and size of the object
      mp.playMode = 0;
      mp.videoDisplayMode = 0;
      mp.allowTrickmode = 1;

      mp.left = 389;
      mp.top = 88;
      mp.width = 240;
      mp.height = 180;
      mp.refreshVideoDisplay();

      //a sample of formatted playURL string containing content metadata
      //to be adjusted in actual testing
      var mediaStr =
'[{mediaUrL:"'+"rtsp://124.75.34.37/88888888/16/20140925/269703862/269703862.ts"+'",med
iaCode: "code1",'+'mediaType:2,'+'audioType:1,'+'videoType:1,'+
'streamType:2,'+'drmType:1,'+'fingerPrint:0,'+'copyProtection:1,'+'allowTrickmode:1,'+'
startTime:0,'+'endTime:5000,'+'entryID:"entry1"}]';

      //locate a media and play from the start
      mp.setSingleMedia( mediaStr );
      mp.playFromStart();
}

//functions for small VOD of VOD_test_1
//return playMode property value of MediaController object in the small window
function getSingleOrPlaylistMode()
{
      return mp.playMode;
}

//return videoDisplayMode property value of MediaController object in the small
//window
function getVideoDisplayMode()
{
      return mp.videoDisplayMode;
}

//return left property value of MediaController object in the small window
function getVideoDisplayLeft()
{
```

```
	return mp.left;
}

//return top property value of MediaController object in the small window
function getVideoDisplayTop()
{
	return mp.top;
}

//return width property value of MediaController object in the small window
function getVideoDisplayWidth()
{
	return mp.width;
}

//return height property value of MediaController object in the small window
function getVideoDisplayHeight()
{
	return mp.height;
}

//call playByTime method of MediaController object
function playByTime( mode, time )
{
	mp.playByTime( mode, time );
}

//return currentPlayTime property value of MediaController object in the small
//window
function getCurrentPlayTime()
{
	return mp.currentPlayTime;
}

//functions for small VOD of VOD_test_2
//return allowTrickmode property value of MediaController object in the small
//window
function getAllowTrickmodeFlag()
{
	return mp.allowTrickmode;
}

//return mediaDuration property value of MediaController object in the small
//window
function getMediaDuration()
{
	return mp.mediaDuration;
}

//return mediaCode property value of MediaController object in the small
//window
function getMediaCode()
{
	return mp.mediaCode;
}

//call pause method of MediaController object
function pause()
{
	mp.pause();
}

//call resume method of MediaController object
function resume()
{
	mp.resume();
}

//return playbackMode property value of MediaController object in the small
//window
function getPlayBackMode()
{
```

```
      return mp.playbackMode;
}

//call fastForward method of MediaController object
function fastForward( speed )
{
      mp.fastForward( speed );
}

//call fastRewind method of MediaController object
function fastRewind( speed )
{
      mp.fastRewind( speed );
}

//call gotoEnd method of MediaController object
function gotoEnd()
{
      mp.gotoEnd();
}

//call gotoStart method of MediaController object
function gotoStart()
{
      mp.gotoStart();
}

//return getVolume property value of MediaController object in the small window
function getVolume()
{
      return mp.getVolume();
}

//call setVolume method of MediaController object
function setVolume( volume )
{
      mp.setVolume( volume );
}

//call stop & releaseMediaController method of MediaController object
function stop()
{
          mp.stop();mp.releaseMediaController( InstanceID );
}

//functions for small VOD of VOD_test_4
//set mute property value of MediaController object in the small window
function setMuteFlag( MuteFlag )
{
      mp.mute = MuteFlag;
}

//return mute property value of MediaController object in the small window
function getMuteFlag()
{
      return mp.mute;
}

//set VideoDisplayMode property value of MediaController object in the small
//window
function setVideoDisplayMode( mode )
{
      mp.VideoDisplayMode = mode;
}

//call refreshVideoDisplay method of MediaController object
function refreshVideoDisplay()
{
      mp.refreshVideoDisplay();
}

</script>
```

```
 <!-- load a page and execute the script -->
 <BODY width="240px" height="180px" bgcolor="transparent" onload="pageload()">
 </BODY>
</HTML>
```

Test sample 0 code snippet 2

```
<!-- base on VOD_test_1 image to create div elements for test options and define
functions to respond to user interactivities -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
 <HEAD>
  <TITLE> VOD_Test_1 </TITLE>
  <META NAME="Generator" CONTENT="EditPlus">
  <META NAME="Author" CONTENT="">
  <META NAME="Keywords" CONTENT="">
  <META NAME="Description" CONTENT="">
 </HEAD>
<script language="javascript">
var InstanceID = -1;

//get instanceId property value in the small window and return this value to the
//box of divInstanceID
function getNativePlayerInstanceID()
{
      InstanceID = window.frames["if_smallscreen"].instanceId;
      document.getElementById("divInstanceID").innerHTML = InstanceID;
}

//get playMode property value in the small window and return this value to the
//box of divMode
function getSingleOrPlaylistMode()
{
      var Mode = window.frames["if_smallscreen"].getSingleOrPlaylistMode();
      document.getElementById("divMode").innerHTML = Mode;
}

//get videoDisplayMode property value in the small window and return this value
//to the box of divDisplayMode
function getVideoDisplayMode()
{
      var DisplayMode = window.frames["if_smallscreen"].getVideoDisplayMode();
      document.getElementById("divDisplayMode").innerHTML = DisplayMode;
}

//get left property value in the small window and return this value to the
//box of divDisplayLeft
function getVideoDisplayLeft()
{
      var DisplayLeft = window.frames["if_smallscreen"].getVideoDisplayLeft();
      document.getElementById("divDisplayLeft").innerHTML = DisplayLeft;
}

//get top property value in the small window and return this value to the
//box of divDiaplayTop
function getVideoDisplayTop()
{
      var DisplayTop = window.frames["if_smallscreen"].getVideoDisplayTop();
      document.getElementById("divDisplayTop").innerHTML = DisplayTop;
}

//get top width value in the small window and return this value to the
//box of divDiaplayWidth
function getVideoDisplayWidth()
{
      var DisplayWidth = window.frames["if_smallscreen"].getVideoDisplayWidth();
      document.getElementById("divDisplayWidth").innerHTML = DisplayWidth;
}

//get height property value in the small window and return this value to the
//box of divDiaplayHeight
function getVideoDisplayHeight()
{
```

```
	var DisplayHeight = window.frames["if_smallscreen"].getVideoDisplayHeight();
	document.getElementById("divDisplayHeight").innerHTML = DisplayHeight;
}

//call playByTime method in the small window
function playByTime( mode, time )
{
	window.frames["if_smallscreen"].playByTime( mode, time );
}

//get currentPlayTime property value in the small window and return this value
//to the box of divCurrentPlayTime
function getCurrentPlayTime()
{
	var CurrentPlayTime = window.frames["if_smallscreen"].getCurrentPlayTime();
	document.getElementById("divCurrentPlayTime").innerHTML = CurrentPlayTime;
}

//call stop method in the small window and link to next page
function GoNextPage()
{
	window.frames["if_smallscreen"].stop();
	document.location.href = "VOD_Test_2.html";
}

</script>
<!-- create div elements based on the background image -->
<BODY width="640" height="530" style="background-Repeat:no-repeat"
background="img/VOD_Test_1.jpg" >
<!-- div elements with red border are used to display property value -->
<!-- InstanceID -->
 <div id="divInstanceID" style="color: #F00; width: 49px; height: 22px; position:
absolute; left: 138px; top: 377px; border: 1px solid red;"></div>

 <!-- iframe element to load VOD_Test_1_Smallvod.html -->
<div id="smallvod" style="left: 389px; top: 88px; width: 240px; height: 180px;
position: absolute; border: 1px solid red;">
<iframe id="if_smallscreen" width="240px" height="180px" src="VOD_Test_1_Smallvod.html"
frameborder="no" scrolling="no">
</iframe>
</div>

 <!-- getSingleOrPlaylistMode -->
 <div id="divMode" style="color: #F00; width: 50px; height: 22px; position: absolute;
left: 314px; top: 377px; border: 1px solid red;"></div>

 <!-- getVideoDisplayMode -->
 <div id="divDisplayMode" style="color: #F00; width: 57px; height: 22px; position:
absolute; left: 560px; top: 377px; border: 1px solid red;"></div>

 <!-- getVideoDisplayLeft -->
 <div id="divDisplayLeft" style="color: #F00; width: 45px; height: 22px; position:
absolute; left: 450px; top: 291px; border: 1px solid red;"></div>

 <!-- getVideoDisplayTop -->
 <div id="divDisplayTop" style="color: #F00; width: 50px; height: 22px; position:
absolute; left: 567px; top: 291px; border: 1px solid red;"></div>

 <!-- getVideoDisplayWidth -->
 <div id="divDisplayWidth" style="color: #F00; width: 41px; height: 22px; position:
absolute; left: 454px; top: 333px; border: 1px solid red;"></div>

 <!-- getVideoDisplayHeight -->
 <div id="divDisplayHeight" style="color: #F00; width: 40px; height: 22px; position:
absolute; left: 577px; top: 333px; border: 1px solid red;"></div>

 <!-- CurrentPlayTime -->
 <div id="divCurrentPlayTime" style="color: #F00; width: 53px; height: 22px; position:
absolute; left: 363px; top: 414px; border: 1px solid red;"></div>

<!-- div elements with blue border are used for user operation -->
 <!-- click to execute getNativePlayerInstanceID function -->
```

```
  <div style="width: 85px; height: 22px; position: absolute; left: 40px; top: 377px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="getNativePlayerInstanceID()">
<img src="img/space.gif" style="width:85px;height:20px;"/>
</a>
</div>
 <!-- click to execute getSingleOrPlaylistMode function -->
  <div style="width: 83px; height: 22px; position: absolute; left: 218px; top: 377px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="getSingleOrPlaylistMode()">
<img src="img/space.gif" style="width:83px;height:20px;"/>
</a>
</div>
 <!-- click to execute getVideoDisplayMode function -->
  <div style="width: 154px; height: 22px; position: absolute; left: 396px; top: 377px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="getVideoDisplayMode()">
<img src="img/space.gif" style="width:149px;height:20px;"/>
</a>
 </div>
 <!-- click to execute getVideoDisplayLeft function -->
  <div style="width: 45px; height: 22px; position: absolute; left: 396px; top: 291px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="getVideoDisplayLeft()">
<img src="img/space.gif" style="width:45px;height:20px;"/>
</a>
 </div>
 <!-- click to execute getVideoDisplayTop function -->
  <div style="width: 45px; height: 22px; position: absolute; left: 512px; top: 291px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="getVideoDisplayTop()">
<img src="img/space.gif" style="width:45px;height:21px;"/>
</a>
 </div>
 <!-- click to execute getVideoDisplayWidth function -->
  <div style="width: 50px; height: 22px; position: absolute; left: 396px; top: 333px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="getVideoDisplayWidth()">
<img src="img/space.gif" style="width:50px;height:20px;"/>
</a>
 </div>
 <!-- click to execute getVideoDisplayHeight function -->
  <div style="width: 58px; height: 22px; position: absolute; left: 512px; top: 333px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="getVideoDisplayHeight()">
<img src="img/space.gif" style="width:55px;height:20px;"/>
</a>
</div>
<!-- click to execute playByTime(1, 100) function -->
 <div style="width: 145px; height: 22px; position: absolute; left: 40px; top: 414px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="playByTime(1, 100)">
<img src="img/space.gif" style="width:145px;height:20px;"/>
</a>
</div>
<!-- click to execute getCurrentPlayTime function -->
 <div style="width: 131px; height: 22px; position: absolute; left: 218px; top: 414px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="getCurrentPlayTime()">
<img src="img/space.gif" style="width:131px;height:20px;"/>
</a>
</div>
 <!-- click to execute GoNextPage function -->
  <div style="width: 55px; height: 22px; position: absolute; left: 560px; top: 414px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="GoNextPage()">
<img src="img/space.gif" style="width:55px;height:22px;"/>
</a>
</div>
</BODY>
</HTML>
```

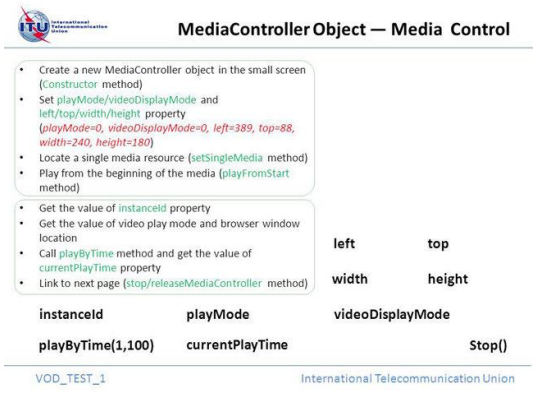

Figure I.1-1-a – Original image

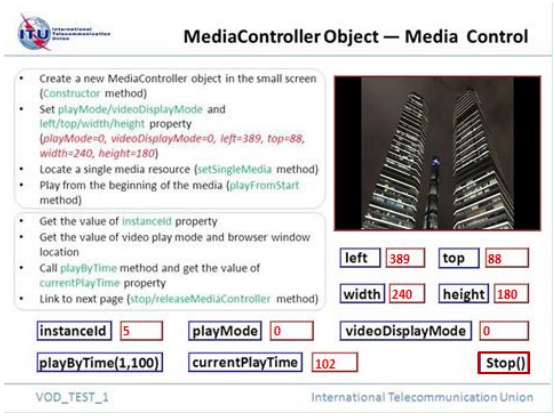

Figure I.1-1-b – Test result

Figure I.1-1 – Reference image, test sample 0 code snippet 2

Test sample 0 code snippet 3

```
<!-- base on VOD_test_2 image to create div elements for test options and define
functions to respond to user interactivities -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
 <HEAD>
  <TITLE> VOD_Test_2 </TITLE>
  <META NAME="Generator" CONTENT="EditPlus">
  <META NAME="Author" CONTENT="">
  <META NAME="Keywords" CONTENT="">
  <META NAME="Description" CONTENT="">
 </HEAD>
<script language="javascript">

//get allowTrickmode property value in the small window and return this value to
//the box of divTrickMode
function getAllowTrickmodeFlag()
{
      var TrickMode = window.frames["if_smallscreen"].getAllowTrickmodeFlag();
      document.getElementById("divTrickMode").innerHTML = TrickMode;
}

//get mediaDuration property value in the small window and return this value to
//the box of divDuration
function getMediaDuration()
{
      var Duration = window.frames["if_smallscreen"].getMediaDuration();
      document.getElementById("divDuration").innerHTML = Duration;
}

//get mediaCode property value in the small window and return this value to
//the box of divMediaCode
function getMediaCode()
{
      var MediaCode = window.frames["if_smallscreen"].getMediaCode();
      document.getElementById("divMediaCode").innerHTML = MediaCode;
}

//call pause method in the small window
function pause()
{
      window.frames["if_smallscreen"].pause();
}

//call resume method in the small window
function resume()
{
```

```
      window.frames["if_smallscreen"].resume();
}

//call fastForward method in the small window and call fastforwardupdate
//function after 1000 milliseconds
function fastForward( speed )
{
      window.frames["if_smallscreen"].fastForward( speed );
      setTimeout("fastForwardUpdate("+speed+")", 1000);
}

//compare the playback mode of user operation and the actual fastforward
//execution of the small window, if they are the same then return OK result to
//the box of divPlayBackMode1 or divPlayBackMode3, else return NOK result
function fastForwardUpdate(speed)
{
      var PlayBackMode;
      eval( 'PlayBackMode = ' + window.frames["if_smallscreen"].getPlayBackMode() );

 //alert($PlayBackMode);
  if( speed == 2 )
      {
            if ( PlayBackMode. CurrentMode == "Trickmode" && PlayBackMode.Speed == "2x"
)
            {
                  document.getElementById("divPlayBackMode1").innerHTML = "OK";
            }
            else
            {
                  document.getElementById("divPlayBackMode1").innerHTML = "NOK";
            }
      }
      else if( speed == 8 )
      {
            if ( PlayBackMode. CurrentMode == "Trickmode" && PlayBackMode.Speed == "8x"
)
            {
                  document.getElementById("divPlayBackMode3").innerHTML = "OK";
            }
            else
            {
                  document.getElementById("divPlayBackMode3").innerHTML = "NOK";
            }
      }
}

//call fastRewind method in the small window and call fastRewindupdate
//function after 1000 milliseconds
function fastRewind( speed )
{
      window.frames["if_smallscreen"].fastRewind( speed );
      setTimeout("fastRewindUpdate("+speed+")", 1000);
}

//compare the playback mode of user operation and the actual fastRewind
//execution of the small window, if they are the same then return OK result to
//the box of divPlayBackMode1 or divPlayBackMode3, else return NOK result
function fastRewindUpdate(speed)
{
      var PlayBackMode;
      eval( 'PlayBackMode = ' + window.frames["if_smallscreen"].getPlayBackMode() );

      if( speed == -4 )
      {
            if ( PlayBackMode.CurrentMode == "Trickmode" && PlayBackMode.Speed == "-4x"
)
            {
                  document.getElementById("divPlayBackMode1").innerHTML = "OK";
            }
            else
            {
                  document.getElementById("divPlayBackMode1").innerHTML = "NOK";
```

```
            }
      }
      else if( speed == -32 )
      {
            if ( PlayBackMode.CurrentMode == "Trickmode" && PlayBackMode.Speed == "-32x"
)
            {
                  document.getElementById("divPlayBackMode3").innerHTML = "OK";
            }
            else
            {
                  document.getElementById("divPlayBackMode3").innerHTML = "NOK";
            }
      }
}

//call gotoEnd method in the small window
function gotoEnd()
{
      window.frames["if_smallscreen"].gotoEnd();
}

//call gotoStart method in the small window
function gotoStart()
{
      window.frames["if_smallscreen"].gotoStart();
}

//call playByTime method in the small window
function playByTime( mode, time )
{
      window.frames["if_smallscreen"].playByTime( mode, time );
}

//get currentPlayTime property value in the small window and return this value
//to the box of divCurrentPlayTime
function getCurrentPlayTime()
{
      var CurrentPlayTime = window.frames["if_smallscreen"].getCurrentPlayTime();
      document.getElementById("divCurrentPlayTime").innerHTML = CurrentPlayTime;
}

//call setVolume function in the small window and call getVolume function to
//output the volume value in the box of divVolume
function setVolume( volume )
{
      window.frames["if_smallscreen"].setVolume( volume );
      var Volume = window.frames["if_smallscreen"].getVolume();
      document.getElementById("divVolume").innerHTML = Volume;
}

//call stop method in the small window and link to next page
function GoNextPage()
{
      window.frames["if_smallscreen"].stop();
      document.location.href = "VOD_Test_3.html";
}

</script>
<!-- create div elements based on the background image -->
 <BODY width="640" height="530" style="background-Repeat:no-repeat"
background="img/VOD_Test_2.jpg" >
<!-- div elements with red border are used to display property value -->
 <!-- TrickMode -->
 <div id="divTrickMode" style="color: #F00; width: 40px; height: 22px; position:
absolute; left: 175px; top: 282px; border: 1px solid red;"></div>

 <!-- Duration -->
 <div id="divDuration" style="color: #F00; width: 42px; height: 22px; position:
absolute; left: 557px; top: 282px; border: 1px solid red;"></div>

 <!-- MediaCode -->
```

```
 <div id="divMediaCode" style="color: #F00; width: 53px; height: 22px; position:
absolute; left: 338px; top: 282px; border: 1px solid red;"></div>

 <!-- PlayBackMode1 -->
 <div id="divPlayBackMode1" style="color: #F00; width: 67px; height: 22px; position:
absolute; left: 324px; top: 317px; border: 1px solid red;"></div>

 <!-- PlayBackMode3 -->
 <div id="divPlayBackMode3" style="color: #F00; width: 67px; height: 22px; position:
absolute; left: 324px; top: 352px; border: 1px solid red;"></div>

 <!-- Volume -->
 <div id="divVolume" style="color: #F00; width: 97px; height: 22px; position: absolute;
left: 294px; top: 386px; border: 1px solid red;"></div>

 <!-- iframe element to load VOD_Test_1_Smallvod.html -->
<div id="smallvod" style="left: 389px; top: 88px; width: 240px; height: 180px;
position: absolute; border: 1px solid red;">
<iframe id="if_smallscreen" width="240px" height="180px" src="VOD_Test_1_Smallvod.html"
frameborder="no" scrolling="no">
</iframe>
</div>

<!-- div elements with blue border are used for user operation -->
 <!-- click to execute getAllowTrickFlag function -->
  <div style="width: 130px; height: 20px; position: absolute; left: 34px; top: 282px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="getAllowTrickmodeFlag()">
<img src="img/space.gif" style="width:130px;height:20px;"/>
</a>
</div>
 <!-- click to execute getMediaDuration function -->
  <div style="width: 125px; height: 20px; position: absolute; left: 421px; top: 282px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="getMediaDuration()">
<img src="img/space.gif" style="width:125px;height:20px;"/>
</a>
</div>
 <!-- click to execute getMediaCode function -->
  <div style="width: 90px; height: 20px; position: absolute; left: 241px; top: 282px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="getMediaCode()">
<img src="img/space.gif" style="width:90px;height:20px;"/>
</a>
</div>
 <!-- click to execute pause function -->
  <div style="width: 68px; height: 20px; position: absolute; left: 34px; top: 418px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="pause()">
<img src="img/space.gif" style="width:68px;height:20px;"/>
</a>
</div>
 <!-- click to execute resume function -->
  <div style="width: 75px; height: 20px; position: absolute; left: 148px; top: 418px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="resume()">
<img src="img/space.gif" style="width:75px;height:20px;"/>
</a>
</div>
 <!-- click to execute fastForward(2) function -->
  <div style="width: 130px; height: 20px; position: absolute; left: 34px; top: 317px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="fastForward(2)">
<img src="img/space.gif" style="width:130px;height:20px;"/>
</a>
</div>
 <!-- click to execute fastForward(8) function -->
  <div style="width: 130px; height: 20px; position: absolute; left: 34px; top: 352px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="fastForward(8)">
<img src="img/space.gif" style="width:130px;height:20px;"/>
</a>
```

```
</div>
 <!-- click to execute fastRewind(-4) function -->
  <div style="width: 125px; height: 20px; position: absolute; left: 421px; top: 317px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="fastRewind(-4)">
<img src="img/space.gif" style="width:125px;height:20px;"/>
</a>
</div>
 <!-- click to execute fastRewind(-32) function -->
  <div style="width: 125px; height: 20px; position: absolute; left: 421px; top: 352px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="fastRewind(-32)">
<img src="img/space.gif" style="width:125px;height:20px;"/>
</a>
</div>
 <!-- click to execute gotoEnd function -->
  <div style="width: 85px; height: 20px; position: absolute; left: 414px; top: 418px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="gotoEnd()">
<img src="img/space.gif" style="width:85px;height:20px;"/>
</a>
</div>
 <!-- click to execute gotoStart function -->
  <div style="width: 90px; height: 20px; position: absolute; left: 272px; top: 418px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="gotoStart()">
<img src="img/space.gif" style="width:90px;height:20px;"/>
</a>
</div>

 <!-- click to execute setVolume(0) function -->
  <div style="width: 130px; height: 20px; position: absolute; left: 34px; top: 386px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="setVolume(0)">
<img src="img/space.gif" style="width:130px;height:20px;"/>
</a>
</div>
 <!-- click to execute setVolume(50) function -->
  <div style="width: 125px; height: 20px; position: absolute; left: 421px; top: 386px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="setVolume(50)">
<img src="img/space.gif" style="width:125px;height:20px;"/>
</a>
</div>

<!-- click to execute GoNextPage function -->
  <div style="width: 58px; height: 20px; position: absolute; left: 542px; top: 418px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onClick="GoNextPage()">
<img src="img/space.gif" style="width:58px;height:20px;"/>
</a>
</div>

</BODY>
</HTML>
```

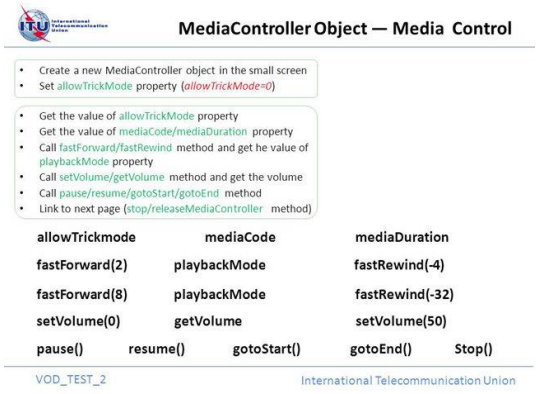

Figure I.1-2-a – Original image

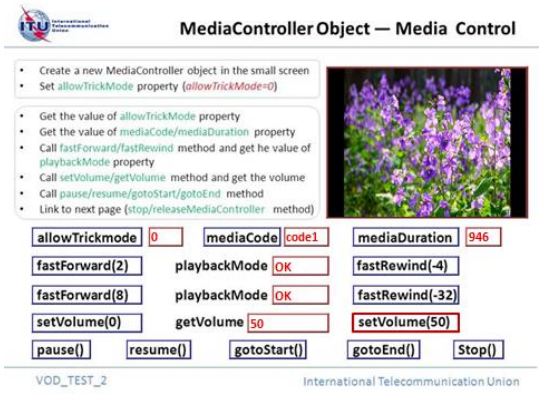

Figure I.1-2-b – Test result

Figure I.1-2 – Reference image, test sample 0 code snippet 3

Test sample 0 code snippet 4

```
<!-- create a new MediaController object in the small window by  initMediaController
method -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
 <HEAD>
  <TITLE> VOD_Test_3_Smallvod </TITLE>
  <META NAME="Generator" CONTENT="EditPlus">
  <META NAME="Author" CONTENT="">
  <META NAME="Keywords" CONTENT="">
  <META NAME="Description" CONTENT="">
 </HEAD>
<script language="javascript">
var mp;
var InstanceID = -1;

function pageload()
{

}

function initMediaPlayer()
{
      mp = new MediaController();
      InstanceID = mp.instanceId;
      //call initMediaController method of MediaController object
      //capture initialization error in case of unsupported properties
      try
      {
      mp.initMediaController( InstanceID, 0, 0, 147, 196, 399, 100, 0, 0, 0, 100, 1 );
      }
     catch(err)
      {
      alert("Error description: property or method is unsupported!");
      return(-1);
      }

      //a sample of formatted playURL string containing content metadata
      //to be adjusted in actual testing
      var mediaStr =
'[{mediaUrL:"'+"rtsp://124.75.34.37/88888888/16/20140925/269703862/269703863.ts"+'",med
iaCode: "code2",'+'mediaType:2,'+'audioType:1,'+'videoType:1,'+
'streamType:2,'+'drmType:1,'+'fingerPrint:0,'+'copyProtection:1,'+'allowTrickmode:1,'+'
startTime:0,'+'endTime:5000,'+'entryID:"entry1"}]';
      mp.setSingleMedia( mediaStr );

      mp.refreshVideoDisplay();
      mp.playFromStart();
}
```

```
//return playMode property value of MediaController object in the small
//window
function getSingleOrPlaylistMode()
{
     return mp.playMode;
}

//return videoDisplayMode property value of MediaController object in the small
//window
function getVideoDisplayMode()
{
     return mp.videoDisplayMode;
}

//return left property value of MediaController object in the small window
function getVideoDisplayLeft()
{
     return mp.left;
}

//return width property value of MediaController object in the small window
function getVideoDisplayWidth()
{
     return mp.width;
}

//return top property value of MediaController object in the small window
function getVideoDisplayTop()
{
     return mp.top;
}

//return height property value of MediaController object in the small window
function getVideoDisplayHeight()
{
     return mp.height;
}

//return left mute value of MediaController object in the small window
function getMuteFlag()
{
     return mp.mute;
}

//return cycleFlag property value of MediaController object in the small
//window
function getCycleFlag()
{
     return mp.cycleFlag;
}

//call stop & releaseMediaController method of MediaController object
function stop()
{
     mp.stop();
     mp.releaseMediaPlayer( InstanceID );
}

</script>
 <BODY width="196px" height="147px" bgcolor="transparent" onload="pageload()">
 </BODY>
</HTML>
```

Test sample 0 code snippet 5

```
<!-- base on VOD_test_3 image to create div elements for test options and define
functions to respond to user interactivities -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
 <HEAD>
```

```
  <TITLE> VOD_Test_3 </TITLE>
  <META NAME="Generator" CONTENT="EditPlus">
  <META NAME="Author" CONTENT="">
  <META NAME="Keywords" CONTENT="">
  <META NAME="Description" CONTENT="">
 </HEAD>
<script language="javascript">

//call initMediaPlayer method in the small window and call a series of functions
//to show property values of the MediaController object
function initMediaPlayer()
{
      //stop execution and link to the next page in case of initialization
      //failure
      var y = window.frames["if_smallscreen"].initMediaPlayer();
      if ( y == -1 )
      {
            document.location.href = "VOD_Test_4.html"
      }
      getSingleOrPlaylistMode();
      getVideoDisplayMode();
      getVideoDisplayLeft();
      getVideoDisplayTop();
      getVideoDisplayWidth();
      getVideoDisplayHeight();
      getMuteFlag();
      getCycleFlag();
}

//get playMode property value in the small window and return this value
//to the box of divSingleOrPlaylistMode
function getSingleOrPlaylistMode()
{
      var SingleOrPlaylistMode =
window.frames["if_smallscreen"].getSingleOrPlaylistMode();
      document.getElementById("divSingleOrPlaylistMode").innerHTML =
SingleOrPlaylistMode;
}

//get videoDisplayMode property value in the small window and return this value
//to the box of divVideoDisplayMode
function getVideoDisplayMode()
{
      var VideoDisplayMode = window.frames["if_smallscreen"].getVideoDisplayMode();
      document.getElementById("divVideoDisplayMode").innerHTML = VideoDisplayMode;
}

//get left property value in the small window and return this value to the box
//of divLeft
function getVideoDisplayLeft()
{
      var Left = window.frames["if_smallscreen"].getVideoDisplayLeft();
      document.getElementById("divLeft").innerHTML = Left;
}

//get width property value in the small window and return this value to the box
//of divWidth
function getVideoDisplayWidth()
{
      var Width = window.frames["if_smallscreen"].getVideoDisplayWidth();
      document.getElementById("divWidth").innerHTML = Width;
}

//get top property value in the small window and return this value to the box
//of divTop
function getVideoDisplayTop()
{
      var Top = window.frames["if_smallscreen"].getVideoDisplayTop();
      document.getElementById("divTop").innerHTML = Top;
}

//get height property value in the small window and return this value to the box
```

```
//of divHeight
function getVideoDisplayHeight()
{
     var Height = window.frames["if_smallscreen"].getVideoDisplayHeight();
     document.getElementById("divHeight").innerHTML = Height;
}

//get mute property value in the small window and return this value to the box
//of divMuteFlag
function getMuteFlag()
{
     var MuteFlag = window.frames["if_smallscreen"].getMuteFlag();
     document.getElementById("divMuteFlag").innerHTML = MuteFlag;
}

//get cycleFlag property value in the small window and return this value to
//the box of divCycleFlag
function getCycleFlag()
{
     var CycleFlag = window.frames["if_smallscreen"].getCycleFlag();
     document.getElementById("divCycleFlag").innerHTML = CycleFlag;
}

//call stop method in the small window and link to next page
function GoNextPage()
{
     window.frames["if_smallscreen"].stop();
     document.location.href = "VOD_Test_4.html";
}

</script>
<!-- create div elements based on the background image -->
 <BODY width="640" height="530" style="background-Repeat:no-repeat"
background="img/VOD_Test_3.jpg" >
<!-- iframe element to load VOD_Test_3_Smallvod.html -->
<div id="smallvod" style="left: 399px; top: 100px; width: 196px; height: 147px;
position: absolute; border: 1px solid red;">
<iframe id="if_smallscreen" width="196px" height="147px" src="VOD_Test_3_Smallvod.html"
frameborder="no" scrolling="no">
</iframe>
</div>

<!-- div elements with red border are used to display property value -->
 <!-- divSingleOrPlaylistMode -->
 <div id="divSingleOrPlaylistMode" style="color: #F00; width: 55px; height: 22px;
position: absolute; left: 355px; top: 298px; border: 1px solid red;"></div>

 <!-- divVideoDisplayMode -->
 <div id="divVideoDisplayMode" style="color: #F00; width: 49px; height: 22px; position:
absolute; left: 193px; top: 298px; border: 1px solid red;"></div>

 <!-- divHeight -->
 <div id="divHeight" style="color: #F00; width: 71px; height: 22px; position: absolute;
left: 105px; top: 335px; border: 1px solid red;"></div>

 <!-- divTop -->
 <div id="divTop" style="color: #F00; width: 71px; height: 22px; position: absolute;
left: 299px; top: 374px; border: 1px solid red;"></div>

 <!-- divLeft -->
 <div id="divLeft" style="color: #F00; width: 71px; height: 22px; position: absolute;
left: 299px; top: 335px; border: 1px solid red;"></div>

 <!-- divWidth -->
 <div id="divWidth" style="color: #F00; width: 71px; height: 22px; position: absolute;
left: 105px; top: 374px; border: 1px solid red;"></div>

 <!-- divMuteFlag -->
 <div id="divMuteFlag" style="color: #F00; width: 55px; height: 22px; position:
absolute; left: 537px; top: 298px; border: 1px solid red;"></div>

<!-- divCycleFlag -->
```

```
 <div id="divCycleFlag" style="color: #F00; width: 55px; height: 22px; position:
absolute; left: 537px; top: 335px; border: 1px solid red;"></div>

<!-- div elements with blue border are used for user operation -->
 <!-- click to execute initMediaPlayer function -->
  <div style="width: 228px; height: 40px; position: absolute; left: 39px; top: 239px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="initMediaPlayer()">
<img src="img/space.gif" style="width:228px;height:40px;"/>
</a>
</div>
 <!-- click to execute GoNextPage function -->
  <div style="width: 196px; height: 22px; position: absolute; left: 401px; top: 406px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="GoNextPage()">
<img src="img/space.gif" style="width:196px;height:22px;"/>
</a>
</div>

</BODY>
</HTML>
```

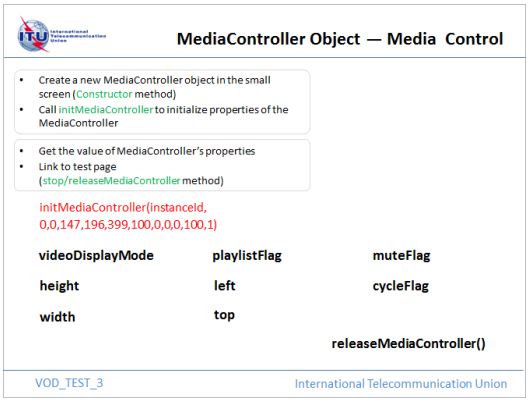

Figure I.1-3-a – Background image

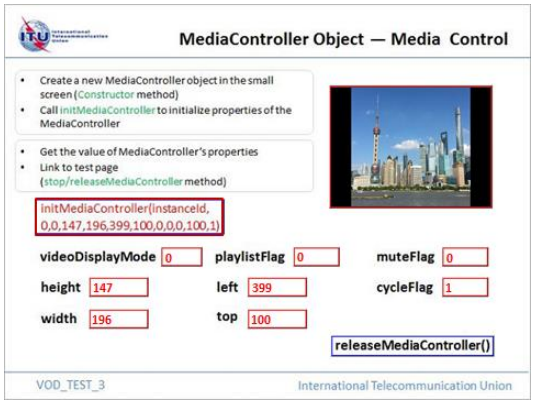

Figure I.1-3-b – Test result

Figure I.1-3 – Reference image, test sample 0 code snippet 5

Test sample 0 code snippet 6

```
<!-- create a new MediaController object in the small window with specific location and
size -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
 <HEAD>
  <TITLE> VOD_Test_4_Smallvod </TITLE>
  <META NAME="Generator" CONTENT="EditPlus">
  <META NAME="Author" CONTENT="">
  <META NAME="Keywords" CONTENT="">
  <META NAME="Description" CONTENT="">
 </HEAD>
<script language="javascript">
var mp;
var InstanceID = -1;

function bindNativePlayerInstanceID ( mpID )
{
      //new a MediaController object and bind with the given instanceId
      mp = new MediaController();
      mp.bindNativePlayerInstance( mpID );
      InstanceID = mp.instanceId;
      //initialization of video play mode, location and size of the object
      mp.playMode = 0;
      mp.videoDisplayMode = 0;
      mp.allowTrickmode = 1;
```

```
      mp.left = 38;
      mp.top = 285;
      mp.width = 200;
      mp.height = 150;
      mp.refreshVideoDisplay();
}

//set mute property value of MediaController object in the small window
function setMuteFlag( MuteFlag )
{
      mp.mute = MuteFlag;
}

// return mute property value of MediaController object in the small window
function getMuteFlag()
{
      return mp.mute;
}

//set VideoDisplayMode property value of MediaController object in the small window
function setVideoDisplayMode( mode )
{
      mp.VideoDisplayMode = mode;
}

//return VideoDisplayMode property value of MediaController object in the small window
function getVideoDisplayMode()
{
      return mp.VideoDisplayMode;
}

//call refreshVideoDisplay method of MediaController object
function refreshVideoDisplay()
{
      mp.refreshVideoDisplay();
}

//call stop & releaseMediaPlayer method of MediaController object
function stop()
{
      mp.stop();
      mp.releaseMediaPlayer( InstanceID );
}
</script>

<BODY width="200px" height="150px" bgcolor="transparent">
 </BODY>
</HTML>
```

Test sample 0 code snippet 7

```
<!-- base on VOD_test_4 image to create div elements for test options and define
functions to respond to user interactivities -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
 <HEAD>
  <TITLE> VOD_Test_4 </TITLE>
  <META NAME="Generator" CONTENT="EditPlus">
  <META NAME="Author" CONTENT="">
  <META NAME="Keywords" CONTENT="">
  <META NAME="Description" CONTENT="">
 </HEAD>
<script language="javascript">
var InstanceID = -1;
var isOne = -1;

//call bindNativePlayerInstance function in small window 2 by binding
//instanceId property of MediaController object in small window 1
function bindNativePlayerInstanceID()
{
```

```
      InstanceID = window.frames["if_smallscreen1"].instanceId;
      window.frames["if_smallscreen2"].bindNativePlayerInstance ( InstanceID );
      isOne = 0;
}

//call setMuteFlag and getMuteFlag functions in the corresponding small
//window and return mute property value
function setMuteFlag( MuteFlag )
{
      var muteflag = -1;
      if ( isOne == -1 )
      {
            window.frames["if_smallscreen1"].setMuteFlag( MuteFlag );
            muteflag = window.frames["if_smallscreen1"].getMuteFlag();
      }
      else
      {
            window.frames["if_smallscreen2"].setMuteFlag( MuteFlag );
            muteflag = window.frames["if_smallscreen2"].getMuteFlag();
      }
       document.getElementById("divMuteFlag").innerHTML = muteflag;
}

//call setvideoDisplayMode and getVideoDisplayMode functions in the
//corresponding small window and return videoDisplayMode property value
function setVideoDisplayMode( mode )
{
      var DisplayMode = -1;
      if ( isOne == -1 )
      {
            window.frames["if_smallscreen1"].setVideoDisplayMode( mode );
            window.frames["if_smallscreen1"].refreshVideoDisplay();
            DisplayMode = window.frames["if_smallscreen1"].getVideoDisplayMode();
      }
      else
      {
            window.frames["if_smallscreen2"].setVideoDisplayMode( mode );
            window.frames["if_smallscreen2"].refreshVideoDisplay();
            DisplayMode = window.frames["if_smallscreen2"].getVideoDisplayMode();
      }
      document.getElementById("divDisplayMode").innerHTML = DisplayMode;
}

//call stop function in the small window and link to next page
function GoNextPage()
{
      if ( isOne == -1 )
      {
            window.frames["if_smallscreen1"].stop();
      }
      else
      {
            window.frames["if_smallscreen2"].stop();
      }
      document.location.href = "VOD_Test_5.html"
}

</script>
<!-- create div elements based on the background image -->
<BODY width="640" height="530" style="background-Repeat:no-repeat"
background="img/VOD_Test_4.jpg" >

<!-- iframe element 1 to load VOD_Test_1_Smallvod.html -->
<div id="smallvod1" style="left: 389px; top: 88px; width: 240px; height: 180px;
position: absolute; border: 1px solid red;">
<iframe id="if_smallscreen1" width="240px" height="180px"
src="VOD_Test_1_Smallvod.html" frameborder="no" scrolling="no">
</iframe>
</div>

<!-- iframe element 2 to load VOD_Test_4_Smallvod.html -->
```

```
<div id="smallvod2" style="left: 38px; top: 285px; width: 200px; height: 150px;
position: absolute; border: 1px solid red;">
<iframe id="if_smallscreen2" width="200px" height="150px"
src="VOD_Test_4_Smallvod.html" frameborder="no" scrolling="no">
</iframe>
</div>

<!-- div elements with red border are used to display property value -->
<div id="divMuteFlag" style="width: 60px; height: 20px; position: absolute; left:
436px; top: 323px; border: 1px solid red;"></div>
<div id="divDisplayMode" style="width: 95px; height: 20px; position: absolute; left:
402px; top: 416px; border: 1px solid red;"></div>

<!-- div elements with blue border are used for user operation -->
<!-- click to execute bindNativePlayerInstance function -->
<div style="width: 180px; height: 20px; position: absolute; left: 268px; top: 292px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="bindNativePlayerInstanceID()">
<img src="img/space.gif" style="width:180px;height:20px;"/>
</a>
</div>

<!-- click to execute setMuteFlag(1) function -->
<div style="width: 110px; height: 20px; position: absolute; left: 268px; top: 323px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="setMuteFlag(1)">
<img src="img/space.gif" style="width:110px;height:20px;"/>
</a>
</div>

<!-- click to execute setMuteFlag(0) function -->
<div style="width: 103px; height: 20px; position: absolute; left: 516px; top: 323px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="setMuteFlag(0)">
<img src="img/space.gif" style="width:103px;height:20px;"/>
</a>
</div>

<!-- click to execute setVideoDisplayMode(1) function -->
<div style="width: 166px; height: 20px; position: absolute; left: 268px; top: 354px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="setVideoDisplayMode(1)">
<img src="img/space.gif" style="width:166px;height:20px;"/>
</a>
</div>

<!-- click to execute setVideoDisplayMode(2) function -->
<div style="width: 169px; height: 20px; position: absolute; left: 450px; top: 354px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="setVideoDisplayMode(2)">
<img src="img/space.gif" style="width:169px;height:20px;"/>
</a>
</div>

<!-- click to execute setVideoDisplayMode(3) function -->
<div style="width: 166px; height: 20px; position: absolute; left: 268px; top: 385px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="setVideoDisplayMode(3)">
<img src="img/space.gif" style="width:166px;height:20px;"/>
</a>
</div>

<!-- click to execute setVideoDisplayMode(255) function -->
<div style="width: 169px; height: 20px; position: absolute; left: 450px; top: 385px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="setVideoDisplayMode(255)">
<img src="img/space.gif" style="width:169px;height:20px;"/>
</a>
</div>

<!-- click to execute GoNextPage function -->
```

```
<div style="width: 66px; height: 20px; position: absolute; left: 553px; top: 416px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="GoNextPage()">
<img src="img/space.gif" style="width:66px;height:20px;"/>
</a>
</div>
</BODY>
</HTML>
```

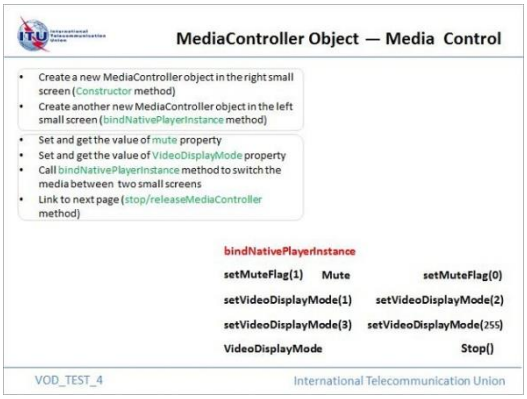

Figure I.1-4-a Background image

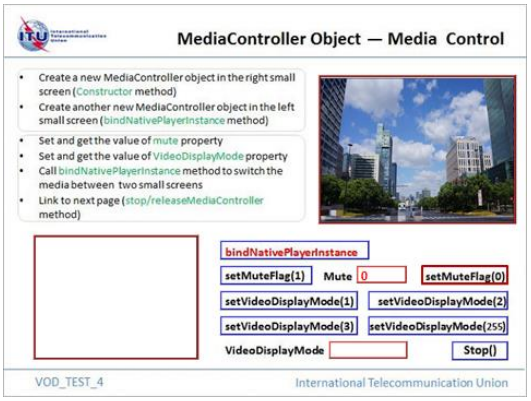

Figure I.1-4-b After setting ‘mute=0’ for the right small window

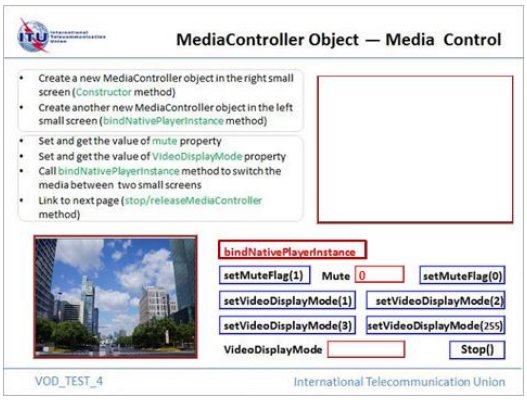

Figure I.1-4-c – After clicking the ‘bindNativePlayerIntance’ button

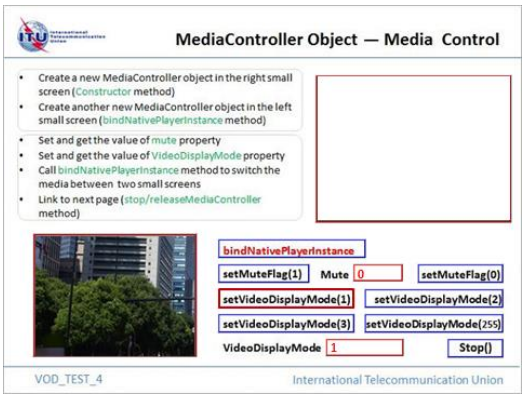

Figure I.1-4-d – After setting ‘videoDisplay Mode=1’ for the left small window

Figure I.1-4 – Reference image, test sample 0 code snippet 7

Test sample 0 code snippet 8

```
<!-- create a new MediaController object in the small window with specific location and
size -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
 <HEAD>
  <TITLE> VOD_Test_5_Smallvod1 </TITLE>
  <META NAME="Generator" CONTENT="EditPlus">
  <META NAME="Author" CONTENT="">
  <META NAME="Keywords" CONTENT="">
  <META NAME="Description" CONTENT="">
 </HEAD>
<script language="javascript">
var mp1;
var InstanceID1 = -1;

function pageload()
{
     //new a MediaController object
     mp1 = new MediaController();
```

```
      InstanceID1 = mp.instanceId;
      //initialization of video play mode, location and size of the object
      mp1.playMode = 0;
      mp1.videoDisplayMode = 0;
      mp1.allowTrickmode = 1;

      mp1.left = 61;
      mp1.top = 214;
      mp1.width = 200;
      mp1.height = 150;
      mp1.refreshVideoDisplay();
}

//call joinChannel method of MediaController object
function joinChannel( userChannelID )
{
      mp1.joinChannel( userChannelID );
}


//call leaveChannel method of MediaController object
function leaveChannel()
{
      mp1.leaveChannel();
}

//call stop method of MediaController object
//this method doesn't work on linear TV service
function stop()
{
    mp1.stop();
}

//call releaseMediaController method of MediaController object
function releaseMediaController()
{
    mp1.releaseMediaController( InstanceID1 );
}

</script>

<!-- load a page and execute the script -->
 <BODY width="200px" height="150px" bgcolor="transparent" onload="pageload()">
 </BODY>
</HTML>
```

Test sample 0 code snippet 9

```
<!-- create a new MediaController object in the small window with specific location and
size -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
 <HEAD>
  <TITLE> VOD_Test_5_Smallvod2 </TITLE>
  <META NAME="Generator" CONTENT="EditPlus">
  <META NAME="Author" CONTENT="">
  <META NAME="Keywords" CONTENT="">
  <META NAME="Description" CONTENT="">
 </HEAD>
<script language="javascript">
var mp;
var InstanceID = -1;

function pageload()
{

}

function launchIPTVContent()
{
      //new a MediaController object
```

```
      mp = new MediaController();
      InstanceID = mp.instanceId;
      //initialization of mute,video play mode location and size of the object
      mp.playMode = 0;
      mp.videoDisplayMode = 0;
      mp.mute = 1;
      mp.left = 382;
      mp.top = 214;
      mp.width = 200;
      mp.height = 150;
      mp.refreshVideoDisplay();

      //a sample of playURL, returnURL and starttime string
      //to be adjusted in actual testing
      var content_uri =
"rtsp://124.75.34.37/88888888/16/20140925/269703862/269703862.ts"
     var ret_uri = "Event_Test_1.html"
     var start_npt = 100.0

      //call launchIPTVContent method of MediaController object
      mp.launchIPTVContent(content_uri, ret_uri, start_npt );
}

//return currentPlayTime property value of MediaController object in the small
//window
function getCurrentPlayTime ()
{
     return mp.currentPlayTime;
}

//call stop method of MediaController object
function stop()
{
     mp.stop();
     mp.releaseMediaController( InstanceID );
}

</script>
<!-- load a page and execute the script -->
 <BODY width="200px" height="150px" bgcolor="transparent" onload="pageload()">
 </BODY>
</HTML>
```

Test sample 0 code snippet 10

```
<!-- base on VOD_test_5 image to create div elements for test options and define
functions to respond to user interactivities -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
 <HEAD>
  <TITLE> VOD_Test_5 </TITLE>
  <META NAME="Generator" CONTENT="EditPlus">
  <META NAME="Author" CONTENT="">
  <META NAME="Keywords" CONTENT="">
  <META NAME="Description" CONTENT="">
 </HEAD>
<script language="javascript">
var joinChannelFlg = -1;
var launchContentFlg = -1;

//call joinChannel function in the small window 1
function joinChannel( userChannelID )
{
      if ( joinChannelFlg == -1 )
      {
            window.frames["if_smallscreen1"].joinChannel( userChannelID );
            joinChannelFlg = 0;
      }
      else
      {
            window.frames["if_smallscreen1"].leaveChannel();
```

```
            window.frames["if_smallscreen1"].joinChannel( userChannelID );
      }
}

//call leaveChannel function in the small window 1
function leaveChannel()
{
      window.frames["if_smallscreen1"].leaveChannel();
      joinChannelFlg = -1;
}

//call stop function in the small window 1
function stop()
{
      window.frames["if_smallscreen1"].stop();
}

//call launchIPTVContent function and return currentPlayTime property value of
//the MediaController object in the small window 2
function launchIPTVContent()
{
      window.frames["if_smallscreen2"].launchIPTVContent();
      document.getElementById("divstarttime").innerHTML =
window.frames["if_smallscreen2"].getCurrentPlayTime();
      launchContentFlg = 0;
}

//call leaveChannel function in small window 1, call stop function in small window 2
//and then link to next page
function GoNextPage()
{
      if ( joinChannelFlg != -1 )
       {
            window.frames["if_smallscreen1"].leaveChannel();
               window.frames["if_smallscreen1"].releaseMediaController();
       }
      if ( launchContentFlg != -1 )
       {
            window.frames["if_smallscreen2"].stop();
       }
      document.location.href = "Event_Test_1.html";
}

</script>
<!-- create div elements based on the background image -->
<BODY width="640" height="530" style="background-Repeat:no-repeat"
background="img/VOD_Test_5.jpg" >

<!-- iframe element 1 to load VOD_Test_5_Smallvod1.html -->
<div id="smallvod1" style="left: 61px; top: 214px; width: 200px; height: 150px;
position: absolute; border: 1px solid red;">
<iframe id="if_smallscreen1" width="200px" height="150px"
src="VOD_Test_5_Smallvod1.html" frameborder="no" scrolling="no">
</iframe>
</div>

<!-- iframe element 2 to load VOD_Test_5_Smallvod2.html -->
<div id="smallvod2" style="left:382px; top:214px; width:200px; height:150px;
position:absolute;border:1px solid red;">
<iframe id="if_smallscreen2" width="200px" height="150px"
src="VOD_Test_5_Smallvod2.html" frameborder="no" scrolling="no">
</iframe>
</div>

<!-- div elements with red border are used to display property value -->
<div id="divstarttime"
style="width:66px;height:20px;position:absolute;left:443px;top:411px;border:1px solid
red;"></div>

<!-- div elements with blue border are used for user operation -->
<!-- click to execute joinChannel(1) function -->
<div style="width:114px;height:20px;position:absolute;left:34px;
```

```
top:380px;cursor:hand;border:1px solid blue;"  >
  <a href="#" onclick="joinChannel(1)">
<img src="img/space.gif" style="width:114px;height:20px;"/>
</a>
</div>

<!-- click to execute joinChannel(2) function -->
<div style="width:114px;height:20px;position:absolute;left:34px;
top:411px;cursor:hand;border:1px solid blue;"  >
  <a href="#" onclick="joinChannel(2)">
<img src="img/space.gif" style="width:114px;height:20px;"/>
</a>
</div>

<!-- click to execute leaveChannel() function -->
<div style="width:114px;height:20px;position:absolute;left:182px;
top:380px;cursor:hand;border:1px solid blue;"  >
  <a href="#" onclick="leaveChannel()">
<img src="img/space.gif" style="width:114px;height:20px;"/>
</a>
</div>

<!-- click to execute stop() function -->
<div style="width:58px;height:20px;position:absolute;left:238px;
top:411px;cursor:hand;border:1px solid blue;"  >
  <a href="#" onclick="stop()">
<img src="img/space.gif" style="width:58px;height:20px;"/>
</a>
</div>

<!-- click to execute launchIPTVContent() function -->
<div style="width:146px;height:20px;position:absolute;left:361px;
top:380px;cursor:hand;border:1px solid blue;"  >
  <a href="#" onclick="launchIPTVContent()">
<img src="img/space.gif" style="width:146px;height:20px;"/>
</a>
</div>

 <!-- click to execute GoNextPage function -->
  <div style="width: 65px; height: 22px; position: absolute; left: 547px; top: 419px;
cursor: hand; border: 1px solid blue;"  >
  <a href="#" onclick="GoNextPage()">
<img src="img/space.gif" style="width:65px;height:22px;"/>
</a>
</div>

</BODY>
</HTML>
```

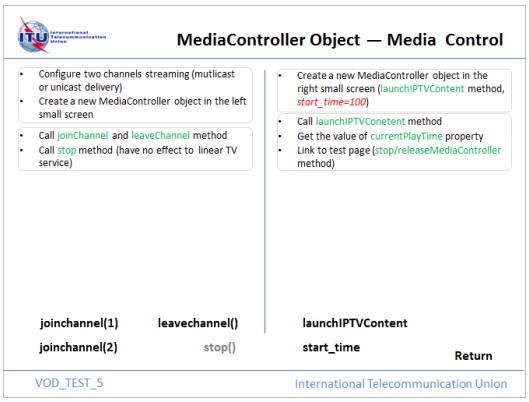

Figure I.1-5-a – Background image

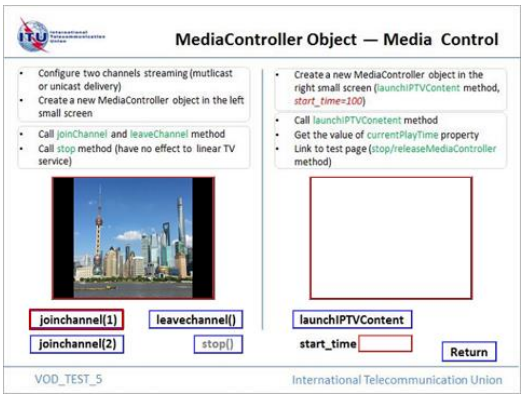

Figure I.1-5-b – After clicking to join channel(1)

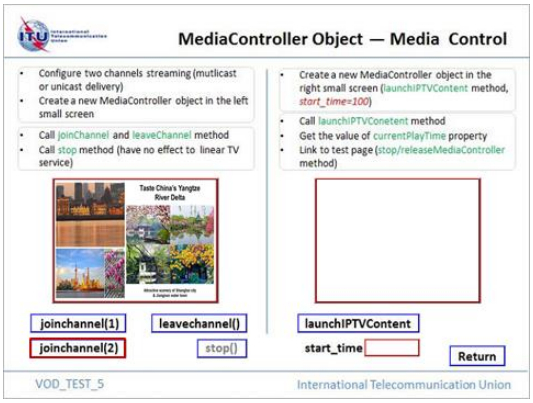

Figure I.1-5-c – After switching to join channel(2)

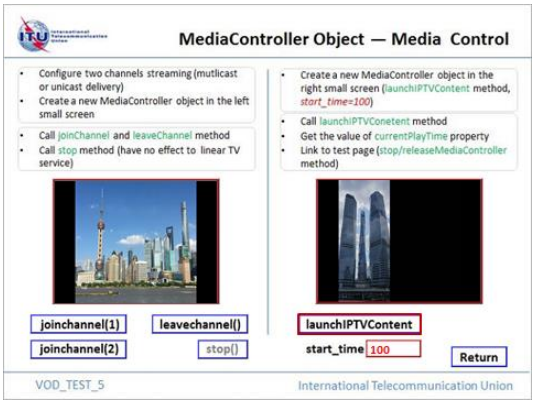

Figure I.1-5-d – After clicking the 'launchIPTVContent' button

Figure I.1-5 – Reference image, test sample 0 code snippet 10

I.2 Test Sample 1

This test sample is designed to verify Event types and mandatory properties of Event object in this TP.

Deploying requirements:

– Each of the following code snippets is deployed as one test page.

– The naming of each test page is required to be consistent with the property value of HTML TITLE (e.g. the test page of code snippet 1 should be named as Event_Test_1.html).

– All the test pages are required to be deployed in the same folder on the web server (e.g. /Event/ Event_Test_1.html).

– Each background image is required to have the same name as the corresponding test page.

– All the test images are required to be deployed in the subfolder of test pages on the web server (e.g. /Event/img/Event_Test_1.jpg).

– Multicast-based and unicast-based IPTV content is required to be deployed on the video content server and can be delivered to the terminal device.

– The metadata of deployed content is required to be re-configured in the corresponding code snippets of test pages.

The composition of two test pages of this sample and the configuration requirements are presented in the following table I.2.

Table I.2 – Test page composition and configuration requirements (sample 1)

Test Identifier	Test Page	Background image	Content & Metadata Configuration
S1T1	Event_Test_1.html (code snippet 1)	Event_Test_1.jpg (Figure I.2-1-a)	A VOD content with configured metadata
S1T2	Event_Test_2.html (code snippet 2)	Event_Test_2.jpg (Figure I.2-2-a)	At least one linear TV stream by multicast delivery, with configured metadata of channel_num and channel_code, trick mode supported
NOTE 1 – The method of Event capture and acquisition is to be adjusted into the code snippets with regards to practical implementation. NOTE 2 – The switch between test pages can be re-configured as required in the corresponding code snippets.			

Test guide:

– Read the brief introduction of codes on the test page before testing.

– Follow the guideline on test page and choose test options one by one.

– Validate the playback effect of video/audio and the conformance of actual property values and expected ones of Event object by comparing to the described testing procedure and corresponding reference images.

Test sample 1 code snippet 1

This code snippet is contained in the electronic attachment (with the file name of Event_Test_1.html).

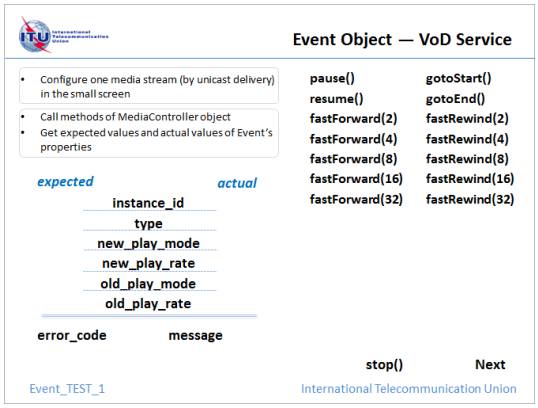

Figure I.2-1-a – Background image

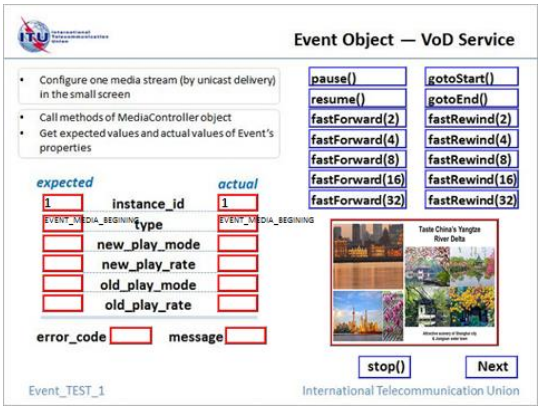

Figure I.2-1-b – After the page load (E1S1 in Table I.3)

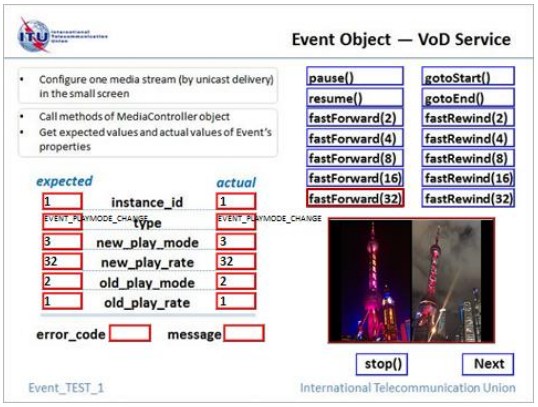

Figure I.2-1-c – From normal play to 'fastForward(32)' operation (E1S2 in Table I.3)

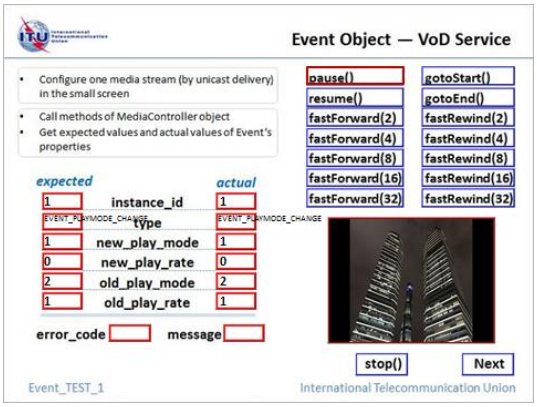

Figure I.2-1-d – From 'resume' to 'pause' operation (E1S13 in Table I.3)

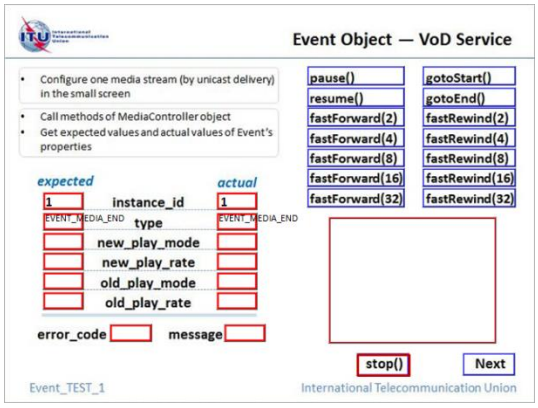

Figure I.2-1-e – From 'fastRewind(32)' to 'stop' operation (E1S19 in Table I.3)

Figure I.2-1 – Reference image, test sample 1 code snippet 1

Table I.3 indicates an instance of testing procedure and result set corresponding to this sample code of Event object in VoD service. Property values of types of Event object shown from column 3 to column 7 in this table are used to verify the correctness of expected results and actual results of Event object in this test case.

Table I.3 – Test sequence and result reference (VoD service)

Sequence Identifier	Operation	Event Type	New Play Mode	New Play Rate	Old Play Mode	Old Play Rate
E1S1	page load	EVENT_MEDIA_BEGINING				
E1S2	fastForward(32)	EVENT_PLAYMODE_CHANGE	3	32	2	1
E1S3	fastRewind(2)	EVENT_PLAYMODE_CHANGE	3	-2	3	32
E1S4	resume	EVENT_PLAYMODE_CHANGE	2	1	3	-2
E1S5	fastForward(16)	EVENT_PLAYMODE_CHANGE	3	16	2	1
E1S6	fastForward(8)	EVENT_PLAYMODE_CHANGE	3	8	3	16
E1S7	pause	EVENT_PLAYMODE_CHANGE	1	0	3	8
E1S8	fastRewind(4)	EVENT_PLAYMODE_CHANGE	3	-4	1	0
E1S9	fastRewind(8)	EVENT_PLAYMODE_CHANGE	3	-8	3	-4
E1S10	gotoStart	1)EVENT_MEDIA_BEGINING				
		2)EVENT_PLAYMODE_CHANGE	2	1	3	-8
E1S11	fastForward(4)	EVENT_PLAYMODE_CHANGE	3	4	2	1
E1S12	resume	EVENT_PLAYMODE_CHANGE	2	1	3	4
E1S13	pause	EVENT_PLAYMODE_CHANGE	1	0	2	1
E1S14	gotoEnd	EVENT_MEDIA_END				
E1S15	fastRewind(16)	EVENT_PLAYMODE_CHANGE	3	-16	2	1
E1S16	fastForward(2)	EVENT_PLAYMODE_CHANGE	3	2	3	-16
E1S17	resume	EVENT_PLAYMODE_CHANGE	2	1	3	2
E1S18	fastRewind(32)	EVENT_PLAYMODE_CHANGE	3	-32	2	1
E1S19	stop	EVENT_MEDIA_END				

NOTE 1 – As for E1S10, either of the two rows of result listed behind is correct.

NOTE 2 – When the property value of play rate equals to 0 or 1 (as shown in column 5 and column 7 of this table), the practical test result displayed as null is also correct.

Test sample 1 code snippet 2

This code snippet is contained in the electronic attachment (with the file name of Event_Test_2.html)..

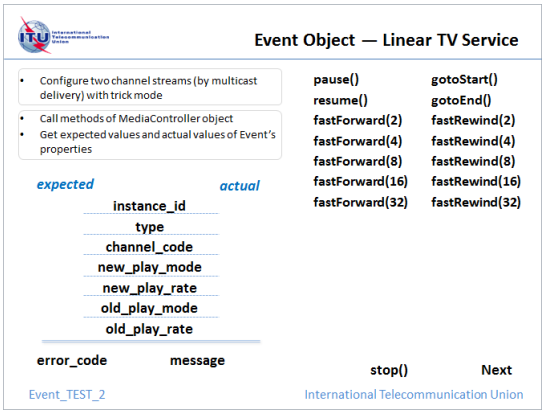

Figure I.2-2-a – Background image

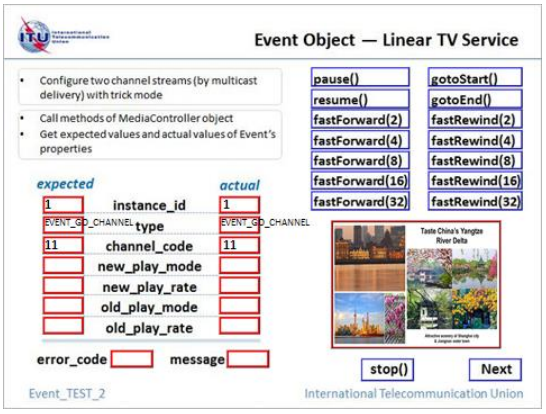

Figure I.2-2-b – After the page load (E2S1 in Table I.4)

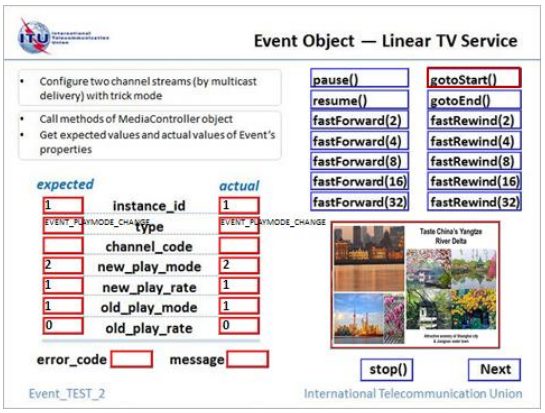

Figure I.2-2-c – From 'pause' to 'gotoStart' operation (E2S6 in Table I.4)

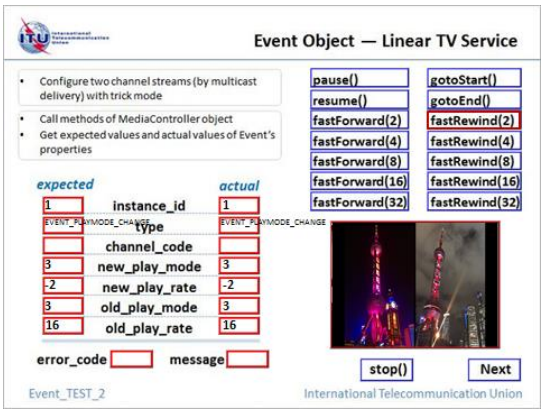

Figure I.2-2-d – From 'fastForward(16)' to 'fastRewind(2)' operation (E2S9 in Table I.4)

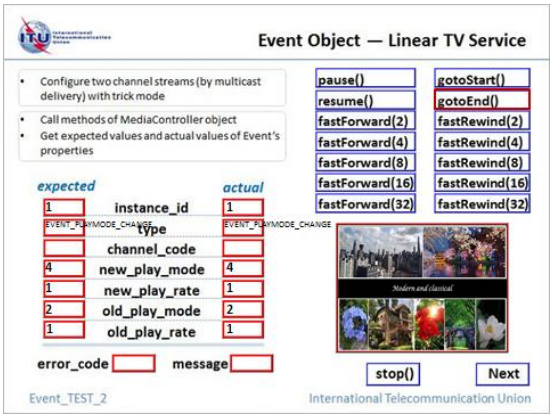

Figure I.2-2-e – From 'resume' to 'gotoEnd' operation (E2S17 in Table I.4)

Figure I.2-2 – Reference image, test sample 1 code snippet 2

Table I.4 indicates an instance of testing procedure and result set corresponding to this sample code of Event object in linear TV service.

Table I.4 – Test sequence and result reference (Linear TV service)

Sequence Identifier	Operation	Event Type	New Play Mode	New Play Rate	Old Play Mode	Old Play Rate
E2S1	page load	EVENT_GO_CHANNEL				
E2S2	pause	EVENT_PLAYMODE_CHANGE	1	0	4	1
E2S3	fastForward(8)	(1)EVENT_PLAYMODE_CHANGE	3	8	1	0
		(2)EVENT_PLAYMODE_CHANGE	4	1	3	8
E2S4	fastRewind(8)	EVENT_PLAYMODE_CHANGE	3	-8	4	1
E2S5	pause	EVENT_PLAYMODE_CHANGE	1	0	3	-8
E2S6	gotoStart	EVENT_PLAYMODE_CHANGE	2	1	1	0
E2S7	fastForward(4)	EVENT_PLAYMODE_CHANGE	3	4	2	1
E2S8	fastForward(16)	EVENT_PLAYMODE_CHANGE	3	16	3	4
E2S9	fastRewind(2)	EVENT_PLAYMODE_CHANGE	3	-2	3	16
E2S10	fastRewind(4)	EVENT_PLAYMODE_CHANGE	3	-4	3	-2
E2S11	resume	EVENT_PLAYMODE_CHANGE	2	1	3	-4
E2S12	fastForward(32)	EVENT_PLAYMODE_CHANGE	3	32	2	1
E2S13	fastRewind(32)	EVENT_PLAYMODE_CHANGE	3	-32	3	32
E2S14	fastRewind(16)	EVENT_PLAYMODE_CHANGE	3	-16	3	-32
E2S15	fastForward(2)	EVENT_PLAYMODE_CHANGE	3	2	3	-16
E2S16	resume	EVENT_PLAYMODE_CHANGE	2	1	3	2
E2S17	gotoEnd	EVENT_PLAYMODE_CHANGE	4	1	2	1
E2S18	switch channel	EVENT_GO_CHANNEL				
E2S19	stop	EVENT_MEDIA_END				

NOTE 1 – As for E2S3, fast forward operation is required to last until multicast channel is re-joined. The two rows of result listed behind will be shown in order.

NOTE 2 – When the property value of play rate equals to 0 or 1 (as shown in column 5 and column 7 of this table), the practical test result displayed as null is also correct.

I.3 Test Sample 2

This test sample is designed to verify the mandatory properties of Service object in this TP.

It is suggested to follow the guideline on test page and validate the property values of Service object by comparing to reference images.

Test sample 2 code snippet 1

```
<!-- base on Service_test image to create div elements for test options and define
functions to respond to user interactivities -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
 <HEAD>
  <TITLE> Service_Test </TITLE>
  <META NAME="Generator" CONTENT="EditPlus">
  <META NAME="Author" CONTENT="">
  <META NAME="Keywords" CONTENT="">
  <META NAME="Description" CONTENT="">
 </HEAD>
<script language="javascript">

function pageload()
{
      //new a Service object
      si = new Service();
```

```
      //initialization of drmSystem property on Licence & DRM of Service object
      si.drmSystem = 'https://11.22.21.9:700/DRM/servlet';
      //initialization of parentalCtrlEnabled, parentalCtrlPassword properties
      //on Parental control of Service object
      si.parentalCtrlEnabled = 'true';
      si.parentalCtrlPassword = 'test@ITU';
}

//return drmSystem property value of Service object
function getdrmSystem()
{
      document.getElementById("divdrmSystem").innerHTML = si.drmSystem;
}

//return parentalCtrlEnabled property value of Service object
function getparentalCtrlEnabled()
{
      document.getElementById("divParentalCtrl").innerHTML = si.parentalCtrlEnabled;
}

//return parentalCtrlPassword property value of Service object
function getparentalCtrlPassword()
{
      document.getElementById("divParentalCtrlPwd").innerHTML =
si.parentalCtrlPassword;
}

function GoNextPage()
{
      document.location.href = "XMLHttpRequest_Test.html";
}

</script>

<!-- create div elements based on the background image -->
 <!-- background image should be deployed in the subfolder of this test page -->
 <BODY width="640" height="530" style="background-Repeat:no-repeat"
background="img/Service_Test.jpg" onload="PageLoad()">

<!-- div elements with red border are used to display property value -->
 <!-- divdrmSystem -->
 <div id="divdrmSystem" style="color: #F00; width: 307px; height: 30px; position:
absolute; left: 38px; top: 390px; border: 1px solid red;"></div>

 <!-- divParentalCtrl -->
 <div id="divParentalCtrl" style="color: #F00; width: 86px; height: 30px; position:
absolute; left: 631px; top: 346px; border: 1px solid red;"></div>

 <!-- divParentalCtrlPwd -->
 <div id="divParentalCtrlPwd" style="color: #F00; width: 86px; height: 30px; position:
absolute; left: 631px; top: 390px; border: 1px solid red;"></div>

<!-- div elements with blue border are used for user operation -->
 <!-- click to execute getdrmSystem function -->
  <div style="width: 110px; height: 30px; position: absolute; left: 38px; top: 345px;
cursor: hand; border: 1px solid blue;" >
  <a href="#" onclick="getdrmSystem()">
<img src="img/space.gif" style="width:110px;height:30px;"/>
</a>
</div>

<!-- click to execute getparentalCtrlEnabled function -->
  <div style="width: 196px; height: 30px; position: absolute; left: 420px; top: 345px;
cursor: hand; border: 1px solid blue;" >
  <a href="#" onclick="getparentalCtrlEnabled()">
<img src="img/space.gif" style="width:196px;height:30px;"/>
</a>
</div>

 <!-- click to execute getparentalCtrlPassword function -->
  <div style="width: 196px; height: 30px; position: absolute; left: 420px; top: 389px;
cursor: hand; border: 1px solid blue;" >
```

```
  <a href="#" onclick="getparentalCtrlPassword()">
<img src="img/space.gif" style="width:196px;height:30px;"/>
</a>
</div>

<!-- click to execute GoNextPage function -->
  <div style="width: 75px; height: 26px; position: absolute; left: 664px; top: 513px;
cursor: hand; border: 1px solid blue;" >
  <a href="#" onclick="GoNextPage()">
<img src="img/space.gif" style="width:75px;height:26px;"/>
</a>
</div>

</BODY>
</HTML>
```

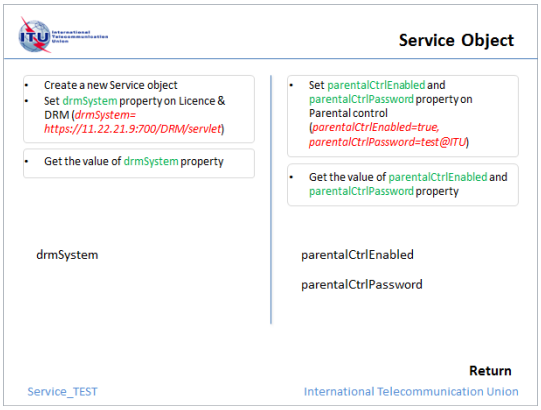

Figure I.3-a – Original image

Figure I.3-b – Test result

Figure I.3 – Reference image, test sample 2 code snippet 1

I.4 Test Sample 3

This test sample is designed to verify the mandatory properties and methods of XMLHttpRequest object in this TP.

Deployment requirements of this sample is presented in the following table I.5.

Table I.5 – Deployment requirements (sample 3)

Test Page	Background image	File Configuration	
		File name	Textual content
XMLHttpRequest_Test.html (code snippet 1)	XMLHttpRequest_Test.jpg (Figure I.4-a)	test_xmlhttp.txt	This is a test using an XML HTTP request to show a text file. If this text shows, it worked!
		test_xmlhttp.xml	<?xml version="1.0" encoding="ISO-8859-1"?> <book> <title lang="en"> Harry Potter</title> <author>J K. Rowling</author> <year>2005</year> </book>

It is suggested to follow the guideline on test page and validate the property values of XMLHttpRequest object by comparing to reference images or actual configurations.

Test sample 3 code snippet 1

```
<!-- base on XMLHttpRequest_test image to create div elements for test options and
define functions to respond to user interactivities -->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
 <HEAD>
  <TITLE> XMLHttpRequest_Test </TITLE>
  <META NAME="Generator" CONTENT="EditPlus">
  <META NAME="Author" CONTENT="">
  <META NAME="Keywords" CONTENT="">
  <META NAME="Description" CONTENT="">
 </HEAD>
<script language="javascript">

function pageload()
{

}

//call open, send and getResponseHeader methods
//get responseText, Status, StatusText values of XMLHttpRequest object
function loadXMLDoc1()
{
      document.getElementById("divReadyState").innerHTML = '';
      document.getElementById("divResponseText").innerHTML = '';
      document.getElementById("divResponseXML").innerHTML = '';
      document.getElementById("divStatus").innerHTML = '';
      document.getElementById("divStatusText").innerHTML = '';
      document.getElementById("divgetRespHeader").innerHTML = '';
      document.getElementById("divgetAllRespHeaders").innerHTML = '';

      //new an XMLHttpRequest object
      var xhr;
      if (window.XMLHttpRequest)
      {
            //code for new browsers (e.g.IE7+, Firefox, Chrome, Opera, Safari)
            xhr = new XMLHttpRequest();
      }
      else
      {
            //code for old browsers (e.g.IE6, IE5)
             xhr = new ActiveXObject("Microsoft.XMLHTTP");
      }
      xhr.onreadystatechange = function()
      {
            if (xhr.readyState == 4)
            {
                  if (xhr.status == 200)
                  {
                        document.getElementById("divResponseText").innerHTML =
xhr.responseText;
                        document.getElementById("divResponseText").style.fontSize =
'small';
                        //a sample of response header label
                        //to be adjusted in actual testing
                        document.getElementById("divgetRespHeader").innerHTML =
xhr.getResponseHeader('Content-Type');
                        document.getElementById("divgetRespHeader").style.fontSize =
'small';
                        document.getElementById("divStatus").innerHTML = xhr.status;
                        document.getElementById("divStatusText").innerHTML =
xhr.statusText;
                  }
                  document.getElementById("divStatus").innerHTML = xhr.status;
                  document.getElementById("divStatusText").innerHTML = xhr.statusText;
            }
            document.getElementById("divReadyState").innerHTML = xhr.readyState;
      }
      //a sample of destination URL of 'test_xmlhttp.txt'
      //to be adjusted in actual testing
      xhr.open("GET","test_xmlhttp.txt",true);
```

```
      xhr.send();
}

//call open, send, setRequestHeader and getAllResponseHeaders methods
//get responseXML, Status, StatusText values of XMLHttpRequest object
function loadXMLDoc2()
{
      document.getElementById("divReadyState").innerHTML = '';
      document.getElementById("divResponseText").innerHTML = '';
      document.getElementById("divResponseXML").innerHTML = '';
      document.getElementById("divStatus").innerHTML = '';
      document.getElementById("divStatusText").innerHTML = '';
      document.getElementById("divgetRespHeader").innerHTML = '';
      document.getElementById("divgetAllRespHeaders").innerHTML = '';

      //new an XMLHttpRequest object
      var xhr;
      if (window.XMLHttpRequest)
      {
            //code for new browsers (e.g.IE7+, Firefox, Chrome, Opera, Safari)
            xhr = new XMLHttpRequest();
      }
      else
      {
            //code for old browsers (e.g.IE6, IE5)
             xhr = new ActiveXObject("Microsoft.XMLHTTP");
      }
      xhr.onreadystatechange = function()
      {
            if (xhr.readyState == 4)
            {
                  if (xhr.status == 200)
                  {
                        xmlDoc = xhr.responseXML;
                        //a sample of XML content, to be adjusted in actual testing
                        //if element node "title" is configured, the value "Harry
                        //Potter" of its text childnode will be shown in div
                        //element 'divResponseXML'
                        document.getElementById("divResponseXML").innerHTML =
xmlDoc.getElementsByTagName("title")[0].childNodes[0].nodeValue;
                        document.getElementById("divResponseXML").style.fontSize =
'small';
                        document.getElementById("divgetAllRespHeaders").innerHTML =
xhr.getAllResponseHeaders();
                        document.getElementById("divgetAllRespHeaders").style.fontSize =
'small';
                        document.getElementById("divStatus").innerHTML = xhr.status;
                        document.getElementById("divStatusText").innerHTML =
xhr.statusText;
                  }
                  document.getElementById("divStatus").innerHTML = xhr.status;
                  document.getElementById("divStatusText").innerHTML = xhr.statusText;
            }
            document.getElementById("divReadyState").innerHTML = xhr.readyState;
      }
      //a sample of destination URL of 'test_xmlhttp.xml'
      //to be adjusted in actual testing
      xhr.open("GET","test_xmlhttp.xml",true);
      //a sample of request header label, to be adjusted in actual testing
      //if the header label set by setRequestHeader method is configured on the
      //server to return in the response header, value of 'User-Agent' label
      //will be shown in div element 'divgetAllRespHeaders'
      xhr.setRequestHeader('User-Agent','webkit;Resolution(PAL,720P,1080P)');
      xhr.send();
}

//call open, send and abort methods of XMLHttpRequest object
//get responseText, readyState values of XMLHttpRequest object
function loadXMLDoc3()
{
      document.getElementById("divReadyState").innerHTML = '';
      document.getElementById("divResponseText").innerHTML = '';
```

```
      document.getElementById("divResponseXML").innerHTML = '';
      document.getElementById("divStatus").innerHTML = '';
      document.getElementById("divStatusText").innerHTML = '';
      document.getElementById("divgetRespHeader").innerHTML = '';
      document.getElementById("divgetAllRespHeaders").innerHTML = '';

      //new an XMLHttpRequest object
      var xhr;
      if (window.XMLHttpRequest)
      {
            //code for new browsers (e.g.IE7+, Firefox, Chrome, Opera, Safari)
            xhr = new XMLHttpRequest();
      }
      else
      {
            //code for old browsers (e.g.IE6, IE5)
             xhr = new ActiveXObject("Microsoft.XMLHTTP");
      }
      xhr.onreadystatechange = function()
      {
            if (xhr.readyState == 3)
            {
                  xhr.abort();
                  document.getElementById("divReadyState").innerHTML = xhr.readyState;
                  document.getElementById("divResponseText").innerHTML =
xhr.responseText;
            }
      }
      //a sample of destination URL, to be adjusted in actual testing
      xhr.open("GET","test_xmlhttp.txt",true);
      xhr.send();
}

function GoNextPage()
{
      //to be adjusted in actual testing
      document.location.href = "VOD_Test_1.html";
}

</script>

<!-- create div elements based on the background image -->
 <!-- background image should be deployed in the subfolder of this test page -->
 <BODY width="640" height="530" style="background-Repeat:no-repeat"
background="img/XMLHttpRequest_Test.jpg" onload="PageLoad()">

<!-- div elements with red border are used to display property value -->
 <!-- divReadyState -->
 <div id="divReadyState" style="color: #F00; width: 55px; height: 22px; position:
absolute; left: 537px; top: 240px; border: 1px solid red;"></div>

 <!-- divResponseText -->
 <div id="divResponseText" style="color: #F00; width: 123px; height: 150px; position:
absolute; left: 23px; top: 262px; border: 1px solid red;"></div>

 <!-- divResponseXML -->
 <div id="divResponseXML" style="color: #F00; width: 123px; height: 150px; position:
absolute; left: 158px; top: 262px; border: 1px solid red;"></div>

<!-- divStatus -->
 <div id="divStatus" style="color: #F00; width: 55px; height: 22px; position: absolute;
left: 537px; top: 270px; border: 1px solid red;"></div>

<!-- divStatusText -->
 <div id="divStatusText" style="color: #F00; width: 55px; height: 22px; position:
absolute; left: 537px; top: 300px; border: 1px solid red;"></div>

<!-- divgetRespHeader -->
 <div id="divgetRespHeader" style="color: #F00; width: 141px; height: 23px; position:
absolute; left: 452px; top: 374px; border: 1px solid red;"></div>
```

```
<!-- divgetAllRespHeaders -->
 <div id="divgetAllRespHeaders" style="color: #F00; width: 123px; height: 150px;
position: absolute; left: 293px; top: 262px; border: 1px solid red;"></div>

<!-- div elements with blue border are used for user operation -->
 <!-- click to execute the first combined function -->
  <div style="width: 148px; height: 40px; position: absolute; left: 29px; top: 190px;
cursor: hand; border: 1px solid blue;" >
  <a href="#" onclick="loadXMLDoc1()">
<img src="img/space.gif" style="width:148px;height:40px;"/>
</a>
</div>

 <!-- click to execute the second combined function -->
  <div style="width: 218px; height: 40px; position: absolute; left: 220px; top: 190px;
cursor: hand; border: 1px solid blue;" >
  <a href="#" onclick="loadXMLDoc2()">
<img src="img/space.gif" style="width:218px;height:40px;"/>
</a>
</div>
<!-- click to execute the third combined function -->
  <div style="width: 102px; height: 40px; position: absolute; left: 481px; top: 190px;
cursor: hand; border: 1px solid blue;" >
  <a href="#" onclick="loadXMLDoc3()">
<img src="img/space.gif" style="width:102px;height:40px;"/>
</a>
</div>

<!-- click to execute GoNextPage function -->
  <div style="width: 64px; height: 22px; position: absolute; left: 529px; top: 406px;
cursor: hand; border: 1px solid blue;" >
  <a href="#" onclick="GoNextPage()">
<img src="img/space.gif" style="width:64px;height:22px;"/>
</a>
</div>

</BODY>
</HTML>
```

Figure I.4-a – Original image

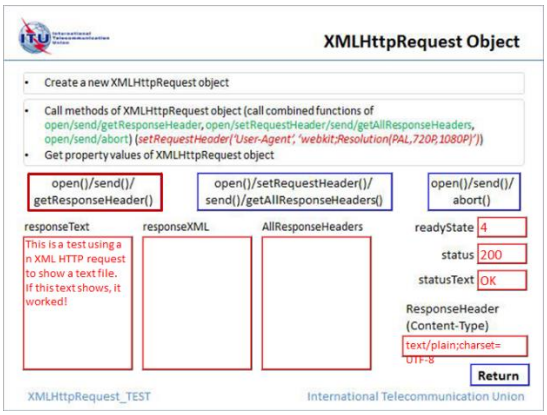

Figure I.4-b – After selecting the first combined methods on the left

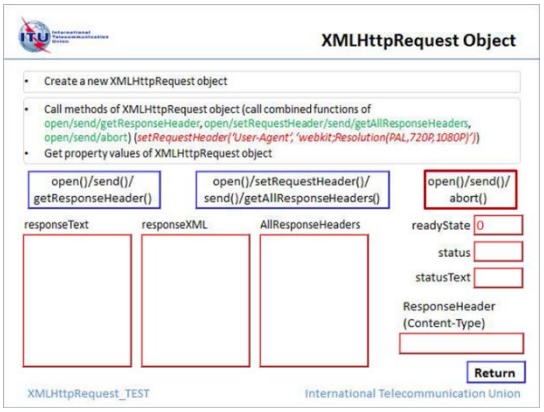

Figure I.4-c – After selecting the first combined methods on the right

Figure I.4-d – After selecting the second combined methods on the right

NOTE 1 – When selecting the second combined methods on the right, the returned value for getAllResponseHeaders() method will be displayed according to actual configurations.

Figure I.4 – Reference image, test sample 3 code snippet 1