Software Development Kit

Application Development Diversity Handling Guidelines

Status:	Final
Version:	4.0.0
Date:	18 June 2015
Author:	Smart TV Alliance inc.
Category:	Official

1. Change history

Version	Date	Changes
2.0	**2013-03-29**	**Final**
2.5	**2013-08-26**	**Final**
3.0	**2014-01-07**	**Final**
3.01	**2014-06-18**	**Final (4.2.4 is added)**
3.02	**2014-08-13**	**Final (3.1.2, 4.2.3 are updated, 4.2.5, 4.4.3 are added)**
4.0.0	**2015-06-18**	**Final (4.5 is added)**

2. Introduction

2.1. Overview

The Smart TV Alliance Specification describes the common set of API's which are supported on all STA platforms. However, platforms could still have slight differences, depending on the version of the Smart TV Alliance platform. These differences ('diversity') can be captured in three categories:

- means of identifying the platform to the Smart TV App
- specific user interface requirements due to platform design
- additional diversity handling, including optional features of the STA specification, that may be implemented only in specific platforms or platform-ranges

This document describes, for each STA-platform, means by which the specific diversity can be handled. It is meant to be used in combination with the Smart TV Alliance Guidelines documentation, that give generic guidelines for handling the diversity.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

Finally, while a lot of care has been taken to ensure the correctness of the information in this document, errors cannot be completely prevented. The latest version of this document, with possible corrections, is always available online. If you have questions and/or remarks regarding these guidelines, please post them through the designated support channels.

2.2. Conventions and styles

In this document, a number of code samples are provided. A code sample is displayed like this:

```
<!doctype html>
<html>
<head>
<title>Basic Example </title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
</head>
<body style="width:1280px;height:720px;margin:0px;overflow:hidden;">
Hello, world.<br/>
<br/>
This is a basic HTML 5 page.
</body>
</html>
```

Important hints for developing the app are formatted as follows:

This is an important hint for developing your app.

Where needed, references are made to the standards on which the Smart TV platform is based. These references are formatted as [n], where [n] is the referred document in 2.5.

2.3. Usage of code samples

All code samples can be used freely in your own code. However, the following terms apply on this code - explaining that you get no warranty on any of the code:

Acceptance of Terms of Use: By accessing and using this software you agree to be bound by the following Terms of Use and all terms and conditions contained and/or referenced herein or any additional terms and conditions set forth on this software and all such terms shall be deemed accepted by you. If you do NOT agree to all these Terms of Use, you should NOT use this software. If you do not agree to any additional specific terms which apply to particular Content (as defined below) then you should NOT use the part of the software which contains such Content.

These Terms of Use may be amended by the author at any time. Such amended Terms of Use shall be effective upon posting of this software. Other Software enclosed herein may have their own terms of use which apply to such Software. Also, specific terms and conditions may apply to specific content, products, materials, services or information contained in or available through this software (the Content). Such specific terms may be in addition to these Terms of Use or, where inconsistent with these Terms of Use, only to the extent the content or intent of such specific terms is inconsistent with these Terms of Use, such specific terms will supersede these Terms of Use.

The author reserves the right to make changes or updates with respect to or in the Content of the software or the format thereof at any time without notice. The author reserves the right to terminate or restrict access to the software for any reason whatsoever at its sole discretion.

Although care has been taken to ensure the accuracy of the information on this software, the author assumes no responsibility therefore. ALL CONTENT IS PROVIDED AS IS AND AS AVAILABLE. THE AUTHOR HEREBY EXPRESSLY DISCLAIMS ANY REPRESENTATIONS OR WARRANTIES OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION WARRANTIES OF MERCHANTABILITY, FITNESS FOR ANY PARTICULAR PURPOSE, NON-INFRINGEMENT, OR AS TO THE OPERATION OF THIS SOFTWARE OR THE CONTENT. THE AUTHOR DOES NOT WARRANT OR MAKE ANY REPRESENTATIONS AS TO THE SECURITY OF THIS SOFTWARE . YOU ACKNOWLEDGE ANY INFORMATION SENT MAY BE INTERCEPTED. THE AUTHOR DOES NOT WARRANT THAT THE SOFTWARE OR THE SERVERS WHICH MAKE THIS SOFTWARE AVAILABLE OR ELECTRONIC COMMUNICATIONS SENT BY THE AUTHOR ARE FREE FROM VIRUSES OR ANY OTHER HARMFUL ELEMENTS.

IN NO EVENT SHALL THE AUTHOR OR ANY OF ITS AFFILIATES BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS, CONTRACT, REVENUE, DATA, INFORMATION OR BUSINESS INTERRUPTION) RESULTING FROM, ARISING OUT OF OR IN CONNECTION WITH THE USE OF, OR INABILITY TO USE THIS SOFTWARE OR THE CONTENT, EVEN IF THE AUTHOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. ANY ACTION BROUGHT AGAINST THE AUTHOR PERTAINING TO OR IN CONNECTION WITH THIS SOFTWARE MUST BE COMMENCED AND NOTIFIED TO THE AUTHOR IN WRITING WITHIN ONE (1) YEAR AFTER THE DATE THE CAUSE FOR ACTION AROSE.

This software may provide other Software that is not under the control of the author. The author shall not be responsible in any way for the content of such other Software. The author provides such software only as a convenience to the user of this software, and the inclusion of any such Software does not imply endorsement by the author of the content of such Software.

The software may contain references to specific products and services that may not be (readily) available in a particular country. Any such reference does not imply or warrant that any such products or services shall be available at any time in any particular country. Please contact your local business contact for further information.

WARRANTIES, IF ANY, WITH RESPECT TO SUCH SOFTWARE SHALL ONLY APPLY AS EXPRESSLY SET FORTH IN THE APPLICABLE LICENSE AGREEMENT. THE AUTHOR HEREBY EXPRESSLY DISCLAIMS ALL FURTHER REPRESENTATIONS AND WARRANTIES OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WARRANTIES OF MERCHANTABILITY, FITNESS FOR ANY PARTICULAR PURPOSE OR NON-INFRINGEMENT WITH RESPECT TO THE SOFTWARE.

2.4. Definitions

UA	User Agent String

2.5. References

[1] Software Development Kit - Application Development and UI Guidelines, latest version

2.6. Trademarks and copyrights

3. Platform identification

3.1.1. Introduction

Platform identification occurs using the user-agent string. This happens in two ways:

- For Smart TV Alliance 2.5 Specification (and later) devices platforms expose a specific Smart TV Alliance user agent string extension
- For Smart TV Alliance 2.0 Specification (and older) devices, different manufacturers that support the Smart TV Alliance specification employ slightly varying ways of exposing the platform identity via this string.

Below is an overview of the various methods.

3.1.2. Identifying the platform

In general, for all Smart TV Alliance compliant devices the returned UA string can be used to further identify the platform manufacturer for additional diversity handling as described in this document. Although some examples are included in this document, the exact method of identifying the platform manufacturer differs per manufacturer.

For Smart TV Alliance Specification 2.5 (and later) devices, a new platform identification mechanism has been added using an extension to the user agent string. This extension to the user agent string is "**SmartTvA/2.5.0**" for Specification 2.5, and e.g. "**SmartTvA/3.0.0**" for Specification 3.0. An example of a (part of the) user agent string in 2.5 compliant devices would be:
" **Opera/9.80 (Linux mips** (……) **Presto/2.6.33 Version/10.70 SmartTvA/2.5.0**"

For Smart TV Alliance Specification 2.0 and older devices, the following method applies:

If you need to identify the platform you are running your app on, you can use a part of the user agent (UA) string to do so. An example of such a user-agent string would be:

```
Opera/9.80 (Linux mips ; U; HbbTV/1.1.1 (; Philips; ; ; ; ) CE-HTML/1.0 NETTV/4.2.0; en)
Presto/2.6.33 Version/10.70
```

Or

```
Mozilla/5.0 (DirectFB; U; Linux armv71) AppleWebKit/534.26+ (KHTML, like Gecko,
Safari/534.26+); LG Browser/5.00.00.9(+mouse+3D+SCREEN+TUNER; LGE; 42LE7500-ZA;
03.05.04; 0x00000001;); LG NetCast.TV-2013
```

Or

```
Mozilla/5.0 (Linux mipsel; U; en) AppleWebKit/534.1 (KHTML, like Gecko) TOSHIBA-DTV
(L7300; 7.2.11.0.0.1; 2013A; NA)
```

It exists of a few different parts:

```
Opera/9.80 (Linux mips ; U; ...
```

and

```
Mozilla/5.0 (DirectFB; U; Linux armv71) AppleWebKit/534.26+ (KHTML, like Gecko,
Safari/534.26+); ...
```

This is the specific browser used for the platform. It can be different for each platform, however, or could not be there at all. Your app should not use this for identification purposes.

The other part of the UA string differs also between manufacturers:

```
HbbTV/1.1.1 (; Philips; ; ; ; ) CE-HTML/1.0 NETTV/4.2.0;
```

Or e.g.:

```
LG Browser/5.00.00.9(+mouse+3D+SCREEN+TUNER; LGE; 42LE7500-ZA; 03.05.04; 0x00000001;);
LG NetCast.TV-2013
```

Or e.g.:

```
TOSHIBA-DTV
```

Here NETTV/4.2.0," LG NetCast.TV-2013" and “TOSHIBA-DTV” are specifically referring to the platform type you are working with. Capabilities can be different for the various platforms.

Platform identification Javascript sample code

To determine the platform version, you can use this Javascript function (indicative, refer to the code samples for the latest version):

```
// parameters: none
// returns: associative array
// smarttv_platform["manufacturer"] = manufacturer name - for STA 2.5+ platforms, this
variable is set to Smart TV Alliance
// smarttv_platform["type"] = type name (e.g. NetCast.TV-2013 - for STA 2.5+ platforms,
this variable is filled with the full UA string)
// smarttv_platform["ua"] = full user agent string
// smarttv_platform["version"] = smart tv specification version (e.g. 2.0) - for STA
2.5+ platforms, this variable is equal to the version number part of the
"SmartTvA/X.Y.Z" string (e.g. 2.5.0)
function smarttv_getPlatform ()
{
  /*jshint sub: true */
  var userAgent = navigator.userAgent;
  var smarttv_platform = [];
  smarttv_platform["ua"] = userAgent;
  smarttv_platform["manufacturer"] = "unknown";
  smarttv_platform["type"] = "unknown";
  smarttv_platform["version"] = "unknown";
  var substr_pos;

  // check if this is a Smart TV Alliance 2.5+ compliant platform
  substr_pos=userAgent.search(/SmartTvA/i);
  if (substr_pos > -1)
  {
    // this is a Smart TV Alliance 2.5+ compliant platform with a new UA identification
    smarttv_platform["type"]=userAgent;
    smarttv_platform["manufacturer"]="Smart TV Alliance";
    smarttv_platform["version"]=userAgent.substr(substr_pos+9,5);
    return smarttv_platform;
  }

  // for STA Specification 2.0 (or older) devices, the older identification method is
used as shown below (this example is not exhaustive for all STA manufacturers):

  // check if this is the Smart TV Alliance SDK
  substr_pos=userAgent.search(/Chromium\/18/i);
  if (substr_pos > -1)
  {
    substr_pos=userAgent.search(/Linux i686/i);
    if (substr_pos > -1)
    {
      smarttv_platform["type"]=userAgent;
      smarttv_platform["manufacturer"]="Smart TV Alliance SDK";
      smarttv_platform["version"]="2.0";
    }
  }

  // check for specific manufacturers (overriding STA SDK option)

  substr_pos=userAgent.search(/NetCast.TV-2012/i);
  if (substr_pos > -1)
  {
    smarttv_platform["type"]=userAgent.substr(substr_pos, 15);
    smarttv_platform["manufacturer"]="LG";
```

```
    smarttv_platform["version"]="1.0";
  }
  else
  {
    substr_pos=userAgent.search(/NetCast.Media/i);
    if (substr_pos > -1)
    {
      smarttv_platform["type"]=userAgent.substr(substr_pos, 18);
      smarttv_platform["manufacturer"]="LG";
    }
    else
    {
      substr_pos=userAgent.search(/NetCast/i);
      if (substr_pos > -1)
      {
       smarttv_platform["type"]=userAgent.substr(substr_pos, 12);
       smarttv_platform["manufacturer"]="LG";
      }
    }
  }
  substr_pos=userAgent.search(/NetCast.TV-2013/i);
  if (substr_pos > -1)
  {
    smarttv_platform["type"]=userAgent.substr(substr_pos, 15);
    smarttv_platform["manufacturer"]="LG";
    smarttv_platform["version"]="2.5";
  }
  substr_pos=userAgent.search(/NETTV\/4/i);
  if (substr_pos > -1)
  {
    smarttv_platform["type"]=userAgent.substr(substr_pos, 11);
    smarttv_platform["manufacturer"]="PHILIPS";
    smarttv_platform["version"]="1.0";
  }
  substr_pos=userAgent.search(/NETTV\/4.2/i);
  if (substr_pos > -1)
  {
    smarttv_platform["type"]=userAgent.substr(substr_pos, 11);
    smarttv_platform["manufacturer"]="PHILIPS";
    smarttv_platform["version"]="2.0";
  }
  else
  {
    substr_pos=userAgent.search(/NETTV/i);
    if (substr_pos > -1)
    {
      smarttv_platform["type"]=userAgent.substr(substr_pos, 11);
      smarttv_platform["manufacturer"]="PHILIPS";
    }
  }
  substr_pos=userAgent.search(/TOSHIBA-DTV/i);
  if (substr_pos > -1)
  {
    smarttv_platform["type"]=userAgent.substr(substr_pos, 11);
    smarttv_platform["manufacturer"]="TOSHIBA";
    smarttv_platform["version"]="2.0";
  }
  return smarttv_platform;
}
```

This should provide your app with a means to properly identify the platform for your own purposes.
It is also possible to do part of this identification server-side: by retrieving the user agent string on your server and analyzing it using server-side scripting - in a similar way as above - you can provide a smaller set of HTML and Javascript code to the platform. In this way, the network usage of your Smart TV App is lower and the loading speed of your app will increase.

4. Specific platform requirements

4.1. Introduction

Certain platforms may require specific elements to be included in your Smart TV App to allow the user to control the App. An example could be the inclusion of a specific screen element that replaces functionality that is not available on the remote control of a specific manufacturer. This chapter describes these elements in more detail.

4.2. LG

4.2.1. Key constants

The VK_-key constants need to be defined in your application for the LG platform. Below is an example how you can do this, the key codes can differ. Please refer to the separate "lg_keyconstants.js" file for the detailed and up to date key constants.

```
var myplatform = smarttv_getPlatform();

if (myplatform["manufacturer"] == "LG")
{
      var VK_ENTER    =  13;
      var VK_PAUSE    =  19;
      var VK_LEFT     =  37;
      var VK_UP       =  38;
      var VK_RIGHT    =  39;
      var VK_DOWN     =  40;
      var VK_0        =  48;
      var VK_1        =  49;
      var VK_2        =  50;
      var VK_3        =  51;
      var VK_4        =  52;
      var VK_5        =  53;
      var VK_6        =  54;
      var VK_7        =  55;
      var VK_8        =  56;
      var VK_9        =  57;
      var VK_RED      = 403;
      var VK_GREEN    = 404;
      var VK_YELLOW   = 405;
      var VK_BLUE     = 406;
      var VK_REWIND   = 412;
      var VK_STOP     = 413;
      var VK_PLAY     = 415;
      var VK_FAST_FWD = 417;
      var VK_BACK     = 461;
}
```

4.2.2. Implement a Q.Menu (Quick Menu) button on video full-screen for LG TV

This function is only needed for LG TV, not for non-TV products such as blu-ray players.
For these products, implement diversity handling that avoids showing the Quick Menu button.

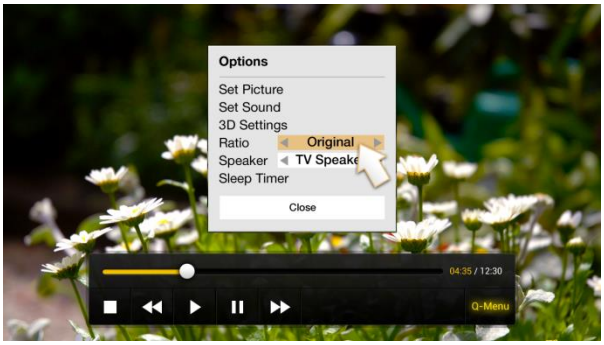
Quick Menu (QMENU) is a special natively built-in function for LG TVs. It essentially allows users to set Video/Audio options, picture size and speaker modes and applies to media applications (for details please refer to LG Web Open API Reference Guide). Since it is a built-in function, manual implementation is not necessary. LG requires this function be implemented for TVs and provides the QMENU with a simple function called NetCastLaunchQMENU.

The QMENU shall be applied for only LG TV platforms. One thing to note is that LG currently supports another media product called BDP, which is a Media Product other than TV. Developers should take caution

to clearly distinguish between the two since QMENU should not be activated or displayed for BDPs. QMENU should be hidden or disabled in the latter case. Refer to the diversity handling guidelines in **Error! Reference source not found.** and the separate video example code snippet.

Also, QMENU can only be called when the video is in full screen mode. Please make sure to have the playing video in this mode prior to calling QMENU (see LGE Web Open API Reference Guide).
QMENU is launched by using the following method. Please note that the video should be in full screen mode.

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



4.2.3. On-screen keyboard

An on-screen keyboard is necessary if there is any text/numeric input for the LG platform: a library for an on-screen keyboard needs to be included within the code (see below for details).
On-screen Keyboard is designed to appear as below when a text input element is focused. Developers can easily use virtual keyboard by including on-screen keyboard library.
Please download the latest JavaScript library at:
http://developer.lge.com/resource/tv/RetrieveSampleCodeContent.dev?resourceId=RS00000586
Regarding how to use this library, please find “LG Virtual Keyboard User Guide.pdf” located in “jsLgVKeyboard” folder of the download file.

4.2.4. Multiscreen DIAL Service Discovery

In UPnP Device Architecture, SSDP messages use part of the header field format of HTTP1.1 as defined in RFC 2616. LG TV device is case-sensitive and then the search target header field shall be capitalized "ST". The client application shall use the M-SEARCH request as

```
ST: urn:dial-multiscreen-org:service:dial:1
```

SDK3.0 sample code supports it and it covers the other v2.5 and later conformant TV devices.

4.2.5. Exit (Back Navigation)

The remote control's Back button is supported in LG TV web apps through the use of the DOM's history object. For details please refer to: http://developer.lge.com/webOSTV/develop/web-app/app-developer-guide/remote-back-button-support/.

By default, when a user presses on the remote's Back button, the running app is replaced with the Home screen. To respond to the Back, use the history object to push state onto the history stack.

```
history.pushState({ "data": "some data" });
```

As long as there is something on the stack, LG TV web apps will receive a popstate event. When the stack is empty, control will pass to the Home screen. You can subscribe to popstate events using a history library or by subscribing to the window event:

```
window.addEventListener("popstate", function(inEvent) {
     // received back, check inEvent.state if you want the data from the history push
 });
```

Use the history events and pushstate to keep track of application state, and allow users to navigate back within your app. You should only use the remote's Back button where it makes sense within your app, and not prevent users from returning to the Home screen using the remote's Back button.

4.3. Toshiba

4.3.1. Exit (Back navigation)

If the device is older than Smart TV Alliance Specification 3.0 Toshiba platform, optionally contact Toshiba to get the information of Toshiba JavaScript library to handle the back navigation. Otherwise your app shall *NOT* exit or return control back to the source application, also not in case the user navigates back 'beyond' the first (entry) page of your Smart TV app. Please note that Smart TV Alliance Specification 3.0 and later compliant Toshiba platform supports the exit method as written in [1] 3.3.2.

4.4. TP Vision

4.4.1. Spatial navigation

The Opera browser inside the TP Vision platform supports built-in *proprietary* spatial navigation. However, it is recommended to use CSS3 spatial navigation instead, as that is available on all Smart TV Alliance platforms. For this reason, make sure that your application defines CSS3 navigation paths for all elements on the page.

4.4.2. Keyboard

The TP Vision platform supports a built-in keyboard that appears when the user presses OK (VK_ENTER) in a regular text-field. On some platforms, SMS-tap key entry is also supported for regular text-fields.

4.4.3. Exit

On TP Vision platforms older than Smart TV Alliance Specification version 3.0, the method by which to exit the application to the home screen is as follows:

```
// to exit the application and return to the home screen
window.history.go(-999);
```

4.5. Panasonic

4.5.1. Initialization file

Panasonic platform reads an initialization file placed at the location of the application.

Sample init.json file.

```
{
  "rendering_size"    :{
          "width"     : 1280,
```

```
            "height"    : 720,
    },
    "keyboard"          :{
            "use"       :true,
            "color"     :"BLACK"
    },
    "mode"              : "key",
    "spatial_navigation" : true,
}
```

This file has to be placed at the initial launch path. Otherwise, application will not be configured correctly.

For example, if the application path is http://smarttv-alliance.org/tvapp/v2/index.html then the init.json should be placed so that it is accessible as http://smarttv-alliance.org/tvapp/v2/init.json.

Features defined in init.json

Parent property	Child property	Available value	Definition
rendering_size	width	number	width of browser(px)
	height	number	height of browser(px)
keyboard	use	true	enable screen keyboard
		false	disable screen keyboard
	color	"BLACK"	make screen keyboard black
		"WHITE"	make screen keyboard white
mode	(none)	“free”	Free cursor mode
		“key”	arrow key mode
spatial_navigation	(none)	true	Turn spatial navigation on
		false	Turn spatial navigation off

Combination of rendering_size should be either width=1280, height=720 or width=1920, height=1080.

4.5.2. Browser default behavior

The Panasonic smart TV application platform will hook return key (VK_BACK) and automatically triggers history.back() as default behavior. Many applications can make use of this behavior to easily create the HTML application. If this default behavior is not desired, the developer can insert preventDefault() in key handler script.

```
function keyHandler (e) {
switch (e.keyCode) {
    // Ignore back key to prevent closing the application.
    case VK_BACK:
      e.preventDefault();
    break;
    }
}
```

5. Additional diversity handling

5.1. Introduction

This chapter describes additional features that can be detected for specific platforms. Some features are browser (not manufacturer) specific, other features are platform specific (such as 3D capability).

5.2. Browser specific

Based on the browser used in the platform, certain standards may require a specific browser-extension to function. Mostly these browser-extensions apply to CSS3 elements. Below is an overview of these elements and the browser-extensions that have to be used.

5.2.1. CSS3 Transform

Different browsers need specific browser-extensions in front of the CSS-style attribute that defines the transformation. Below is a list of browsers per manufacturer and their subsequent extension:

Manufacturer	Browser	Browser extension
Toshiba	Webkit	-webkit-transform-*
LG	Webkit	-webkit-transform-*
TP Vision	Opera	-o-transform-* (optional)

5.2.2. CSS3 Transitions

Different browsers need specific browser-extensions in front of the CSS-style attribute that defines the transition. Below is a list of browsers per manufacturer and their subsequent extension:

Manufacturer	Browser	Browser extension
Toshiba	Webkit	-webkit-transition-*
LG	Webkit	-webkit- transition-*
TP Vision	Opera	-o- transition-* (optional)

5.3. Optional platform specific features

For Smart TV Alliance Specification 2.5 and older compliant devices, below is a list of optional, platform specific features and the way support for these features can be identified from a Smart TV Application. Please refer to [1] 3.2.4 how to identify 3D support for the Smart TV Alliance Specification 3.0 and later compliant devices.

Feature	Manufacturer	Identification method
3D	Toshiba	No identification method available.
3D	LG	"+3D" in user agent string
3D	TP Vision	No identification method available. Advise is to include a message when content is provided in 3D so the user can make the determination based on platform support. For more information please contact TP Vision support.

Note: when a platform supports 3D video, sometimes this is indicated. This does not say anything on whether a user has switched their TV into 3D mode. For some platforms where 3D identification is not supported (but 3D support *is* provided), providing a specific message to the user that content is 3D allows them to make the proper determination if they want to play out the content.