LG Electronics

[Tutorial] LG Web_Generating ‘Outofmemory’ Event

Version 1.1 – February 2012

LGDEV-053

Home Entertainment Company
LG Electronics, Inc.

Copyright

Copyright © 2011 LG Electronics, Inc. All Rights Reserved.

Though every care has been taken to ensure the accuracy of this document, LG Electronics, Inc. cannot accept responsibility for any errors or omissions or for any loss occurred to any person, whether legal or natural, from acting, or refraining from action, as a result of the information contained herein. Information in this document is subject to change at any time without obligation to notify any person of such changes.

LG Electronics, Inc. may have patents or patent pending applications, trademarks copyrights or other intellectual property rights covering subject matter in this document. The provision of this document does not give the recipient or reader any license to these patents, trademarks copyrights or other intellectual property rights.

No part of this document may be communicated, distributed, reproduced or transmitted in any form or by any means, electronic or mechanical or otherwise, for any purpose, without the prior written permission of LG Electronics, Inc.

This document is subject to revision without further notice.

All brand names and product names mentioned in this document are trademarks or registered trademarks of their respective owners.

About This Document

Revision History

Document Version	Date	Comment
1.1	February 13, 2012	Needed APIs and Source codes are updated. Section 2.6 is added.
1.0	October 13, 2011	Initial Version

Purpose

This document describes how to generate ‘outofmemory’ event.

Reference Documents

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

Conventions

Codes

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

Note, Caution

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

Note

Contains information about something that is helpful to you.

Caution

Contains important information about something that you should know.

Abbreviation

The following table defines the abbreviations used in this document.

Abbreviation	Description
API	Application Programming Interface

Contents

Tables

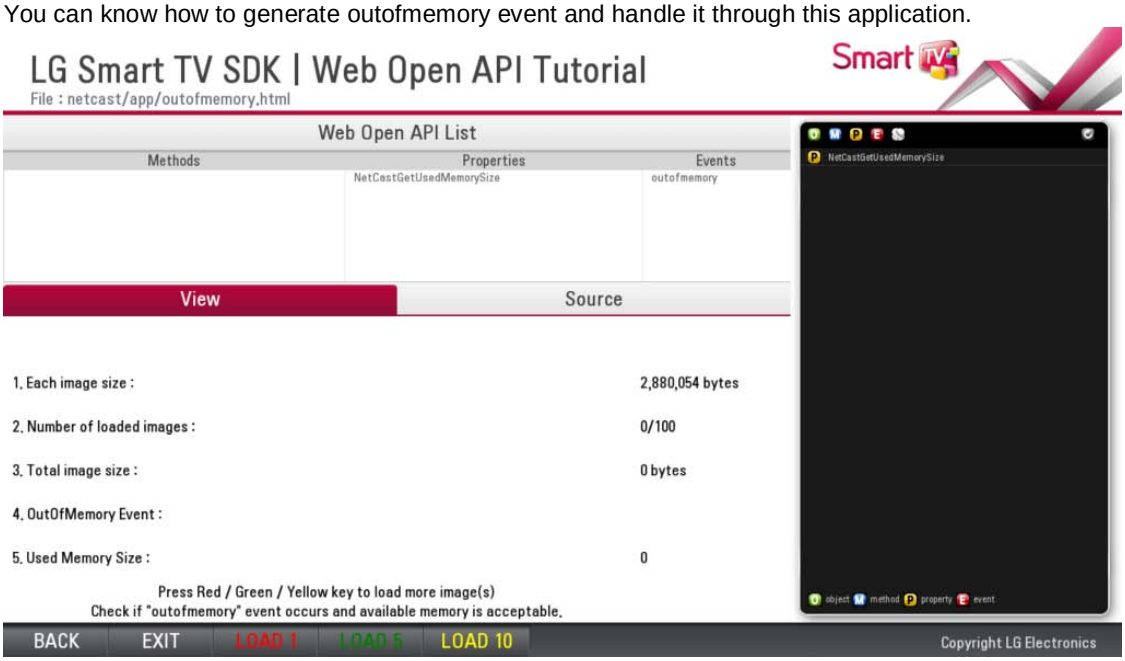
Figures

1 Introduction

This chapter provides an overview of this application and needed APIs.

1.1 Overview

This application generates outofmemory event by loading image and increasing memory size. You can know how to generate outofmemory event and handle it through this application.



[Figure 1] Application to Generate 'outofmemory' Event

Note

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

1.2 Needed APIs

This application uses following Web Open API:

[Table 1] Description of the Needed APIs

API Class	Name	Description
Method	N/A	N/A
Property	NetCastGetUsedMemorySize	Total memory size used by the browser process
Event	outofmemory	This event gives the remaining memory size to application author.

For more information on these functions, refer to “LG Web Open API Reference Guide”.

2 Creating Application

This chapter describes how to generate ‘outofmemory’ event.

2.1 Initializing the Page

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

04: Records the last visited page when running the application.
07: Initializes the page.
08: Gets the source code of the page using the XMLHttpRequest object.
09: Sets the page ID.
10: Initializes the Log function.
13-17: Registers an event handler which will executed when the corresponding button is pressed.
19: Calls displayStatus function.
22: Registers an event handler which will be executed when the outofmemory event occurs.
24: Displays the memory size used by the application on screen.

Sample Code

```
01 : function initPage()
02 : {
03 :  //save page as last visited page
04 :  setLastVisitPage();
05 :
06 :  //common initialize function
07 :  commonInitialize();
08 :  requestSourceCode();
09 :  setPageID("Out Of Memory");
10 :  jsLog.initLG();
11 :
12 :  //add onclick event handler
13 :  addEventHandler(document.getElementById("btn_back"),"click",onClickHandler);
14 :  addEventHandler(document.getElementById("btn_red"),"click", onClickHandler);
15 :  addEventHandler(document.getElementById("btn_green"),"click",onClickHandler);
16 :  addEventHandler(document.getElementById("btn_yellow"),"click",onClickHandler);
17 :  addEventHandler(document.getElementById("btn_exit"),"click",onClickHandler);
18 :
19 :   displayStatus();
20 :
21 :  //add outofmemory event handler
22 :  addEventHandler(window, "outofmemory", outOfMemoryHandler);
23 :
24 :  getUsedMemorySize();
25 : }
```

2.2 Inputting Keys

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

06-07: When the Back key is pressed, this code is executed.
08: When the Red key is pressed, one image is loaded.
09: When the Green key is pressed, five images are loaded.
10: When the Yellow key is pressed, ten images are loaded.

Sample Code

```
01 : //onUserInput function should be implemented
02 : function onUserInput(userInput)
03 : {
04 :   switch(userInput)
05 :   {
06 :          case VK_BACK:
07 :             window.location.replace("../menu_netcast.html"); break;
08 :          case VK_RED : case 82 : loadMoreImages(1); break;
09 :          case VK_GREEN : case 71 : loadMoreImages(5); break;
10 :          case VK_YELLOW : case 89 : loadMoreImages(10); break;
11 :     }
12 : }
```

2.3 Loading Images

The following functions load the specified number of images.

loadMoreImages(numberToLoad)

This function is called when multiple images are loaded.
04: Calls loadOneImage() function as the number of images.

loadOneImage:

Loads one image.
20-21: Returns if there is no image to load
24-25: Specifies size of the image to load.
26-27: Specifies the position that the image is loaded on the screen.
30-35: Specifies the image to load
36: Creates element of div tag.

Increases number of images by 1.
48: Increases number of images by 1.
49: Calls displayStatus() function.

Sample Code

```
01 : function loadMoreImages(numberToLoad)
02 : {
03 :  for(var i = 0 ; i < numberToLoad ; i++)
04 :         loadOneImage();
05 : }
06 :
07 : var numberOfLoadedImages = 0;
08 : var EACH_IMG_SIZE = 2880054;
09 : var DEFAULT_LEFT = 0;
10 : var DEFAULT_TOP = 0;
11 : var DEFAULT_IMG_LEFT = 100;
12 : var DEFAULT_IMG_TOP = 500;
13 : var TOTAL_WIDTH = 900;
14 : var TOTAL_HEIGHT = 50;
15 : var MAX_COLUMN_INDEX = 50;
16 : var MAX_ROW_INDEX = 2;
17 :
18 : function loadOneImage()
19 : {
20 :  if(MAX_COLUMN_INDEX * MAX_ROW_INDEX == numberOfLoadedImages)
21 :         return;
22 :
23 :  //create div and set an image as background
24 :  var widthToSet = Math.floor(TOTAL_WIDTH / MAX_COLUMN_INDEX);
25 :  var heightToSet = Math.floor(TOTAL_HEIGHT / MAX_ROW_INDEX);
26 :  var  leftToSet  =  Math.floor((TOTAL_WIDTH  /  MAX_COLUMN_INDEX)  *
   (numberOfLoadedImages % MAX_COLUMN_INDEX));
27 :  var  topToSet  =  Math.floor((TOTAL_HEIGHT  /  MAX_ROW_INDEX)  *
   Math.floor(numberOfLoadedImages / MAX_COLUMN_INDEX));
28 :  var bgImgIndex = "";
29 :
30 :  if(String(numberOfLoadedImages).length == 1){bgImgIndex = "00";}
31 :  else if(String(numberOfLoadedImages).length == 2){bgImgIndex = "0";}
32 :  else if(String(numberOfLoadedImages).length == 3){bgImgIndex = "";}
33 :
34 :  bgImgIndex += String(numberOfLoadedImages);
35 :  var  bgUrlToSet  =  getMediaFileUrl("memoryTestImages")  +  "lake-nature_"  +
   bgImgIndex + ".bmp";
36 :  var divToAppend = document.createElement("div");
37 :
38 :  divToAppend.style.position = "absolute";
39 :  divToAppend.style.width = (widthToSet - 2) + "px";
40 :  divToAppend.style.height = (heightToSet - 2) + "px";
41 :  divToAppend.style.border = "1px solid #000000";
42 :  divToAppend.style.left = leftToSet + "px";
```

```
43 :   divToAppend.style.top = topToSet + "px";
44 :   divToAppend.style.background = "url(" + bgUrlToSet + ")";
45 :   divToAppend.style.backgroundPosition = "-" + (DEFAULT_IMG_LEFT + leftToSet)
     + "px -" + (DEFAULT_IMG_TOP + topToSet) + "px";
46 :   divToAppend.style.backgroundRepeat = "no-repeat";
47 :   document.getElementById("ViewArea").appendChild(divToAppend);
48 :   numberOfLoadedImages++;
49 :   displayStatus();
50 : }
```

2.4 Displaying Information of Loaded Image and Event

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

displayStatus:

Displays information regarding the loaded image.
03: Displays the number of loaded images on the screen.
05: Displays size of loaded images on the screen.

toCommaNotation(origString)

Inserts commas in numbers.
15-22: If the number of digits is greater than three, comma is inserted in number for each three digits, which is saved in resultString.

Sample Code

```
01 : function displayStatus()
02 : {
03 :   setInnerTextById("number_of_loaded_image_result", numberOfLoadedImages + "/"
     + (MAX_COLUMN_INDEX * MAX_ROW_INDEX));
04 :   var totalSize = EACH_IMG_SIZE * numberOfLoadedImages;
05 :   setInnerTextById("total_image_size_result",
     toCommaNotation(String(totalSize)) + " bytes");
06 : }
07 :
08 :
09 : function toCommaNotation(origString)
10 : {
11 :   var unitToSep = 3;
12 :   var resultString = "";
13 :   var processString = origString;
14 :
15 :   while(processString.length > unitToSep)
16 :   {
17 :         var cutIndex = processString.length - unitToSep;
18 :         var   resultString   =   ","   +   processString.substring(cutIndex)   +
     resultString;
19 :         processString = processString.substring(0, cutIndex);
20 :   }
21 :   resultString = processString + resultString;
22 :   return resultString;
23 : }
```

Sample Code

```
51 : function loadMoreImages(numberToLoad)
52 : {
53 :  for(var i = 0 ; i < numberToLoad ; i++)
54 :         loadOneImage();
55 : }
56 :
57 : var numberOfLoadedImages = 0;
58 : var EACH_IMG_SIZE = 2880054;
59 : var DEFAULT_LEFT = 0;
60 : var DEFAULT_TOP = 0;
61 : var DEFAULT_IMG_LEFT = 100;
62 : var DEFAULT_IMG_TOP = 500;
63 : var TOTAL_WIDTH = 900;
64 : var TOTAL_HEIGHT = 50;
65 : var MAX_COLUMN_INDEX = 50;
66 : var MAX_ROW_INDEX = 2;
67 :
68 : function loadOneImage()
69 : {
70 :  if(MAX_COLUMN_INDEX * MAX_ROW_INDEX == numberOfLoadedImages)
71 :         return;
72 :
73 :  //create div and set an image as background
74 :  var widthToSet = Math.floor(TOTAL_WIDTH / MAX_COLUMN_INDEX);
75 :  var heightToSet = Math.floor(TOTAL_HEIGHT / MAX_ROW_INDEX);
76 :  var  leftToSet  =  Math.floor((TOTAL_WIDTH  /  MAX_COLUMN_INDEX)  *
   (numberOfLoadedImages % MAX_COLUMN_INDEX));
77 :  var  topToSet  =  Math.floor((TOTAL_HEIGHT  /  MAX_ROW_INDEX)  *
   Math.floor(numberOfLoadedImages / MAX_COLUMN_INDEX));
78 :  var bgImgIndex = "";
79 :
80 :  if(String(numberOfLoadedImages).length == 1){bgImgIndex = "00";}
81 :  else if(String(numberOfLoadedImages).length == 2){bgImgIndex = "0";}
82 :  else if(String(numberOfLoadedImages).length == 3){bgImgIndex = "";}
83 :
84 :  bgImgIndex += String(numberOfLoadedImages);
85 :  var  bgUrlToSet  =  getMediaFileUrl("memoryTestImages")  +  "lake-nature_"  +
   bgImgIndex + ".bmp";
86 :  var divToAppend = document.createElement("div");
87 :
88 :  divToAppend.style.position = "absolute";
89 :  divToAppend.style.width = (widthToSet - 2) + "px";
90 :  divToAppend.style.height = (heightToSet - 2) + "px";
91 :  divToAppend.style.border = "1px solid #000000";
92 :  divToAppend.style.left = leftToSet + "px";
93 :  divToAppend.style.top = topToSet + "px";
94 :  divToAppend.style.background = "url(" + bgUrlToSet + ")";
95 :  divToAppend.style.backgroundPosition = "-" + (DEFAULT_IMG_LEFT + leftToSet)
   + "px -" + (DEFAULT_IMG_TOP + topToSet) + "px";
96 :  divToAppend.style.backgroundRepeat = "no-repeat";
97 :  document.getElementById("ViewArea").appendChild(divToAppend);
98 :  numberOfLoadedImages++;
99 :  displayStatus();
100 :  }
```

2.5 Handling Events

The following function is for event handler, which handles outofmemory event.
outOfMemoryHandler is called when the outofmemory event occurs.

07: Increases eventCount by 1.
09-14: Saves counted value and event.available (size of usable memory) value to resultString and displays the values on the screen.

Sample Code

```
01 : var resultString = " ";
02 : var eventCount = 0;
03 :
04 : function outOfMemoryHandler()
05 : {
06 :  jsLog.lgevent('outofmemory');
07 :  eventCount++;
08 :
09 :  if(eventCount == 1)
10 :         resultString += eventCount + " - " + event.available + " MB";
11 :  else
12 :         resultString += " / " + eventCount + " - " + event.available + "
     MB";
13 :
14 :   setInnerTextById("out_of_memory_event_occurred", resultString);
15 : }
```

2.6 Displaying usedMemorySize

The following function is for displaying the: memory size
01: Set usedMemorySize.
04-09: If the return value of NetCastGetUsedMemorySize exists, display the value.

Sample Code

```
01 : var usedMemorySize;
02 : function getUsedMemorySize()
03 : {
04 :  if(window.NetCastGetUsedMemorySize)
05 :  {
06 :         usedMemorySize = window.NetCastGetUsedMemorySize();
07 :         setInnerTextById("used_memory_result", usedMemorySize);
08 :         jsLog.lgproperty('NetCastGetUsedMemorySize');
09 :  }
10 : }
```

2.7 Source Code of outofmemory.html

Source code of outofmemory.html is as follows:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>OutOfMemory Event Test Page</title>
<link rel="stylesheet" href="../../css/style.css">
<script language="javascript" src="../../js/keycode.js"></script>
<script language="javascript" src="../../js/media.js"></script>
<script language="javascript" src="../../js/common.js"></script>
<script language="javascript" src="../../js/menu.js"></script>

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

<script>

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

		//common initialize function
		commonInitialize();
		requestSourceCode();
		setPageID("Out Of Memory");
		jsLog.initLG();

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

		displayStatus();

		//add outofmemory event handler
		addEventHandler(window, "outofmemory", outOfMemoryHandler);

		getUsedMemorySize();
	}

	function onUserInput(userInput)
	{
		switch(userInput)
		{
			case VK_BACK : window.location.replace("../menu_netcast.html");
break;
			case VK_RED : case 82 : loadMoreImages(1); break;
			case VK_GREEN : case 71 : loadMoreImages(5); break;
```

```
                case VK_YELLOW : case 89 : loadMoreImages(10); break;
            }
        }

        function loadMoreImages(numberToLoad)
        {
            for(var i = 0 ; i < numberToLoad ; i++)
                loadOneImage();
        }

        var numberOfLoadedImages = 0;
        var EACH_IMG_SIZE = 2880054;
        var DEFAULT_LEFT = 0;
        var DEFAULT_TOP = 0;
        var DEFAULT_IMG_LEFT = 100;
        var DEFAULT_IMG_TOP = 500;
        var TOTAL_WIDTH = 900;
        var TOTAL_HEIGHT = 50;
        var MAX_COLUMN_INDEX = 50;
        var MAX_ROW_INDEX = 2;

        function loadOneImage()
        {
            if(MAX_COLUMN_INDEX * MAX_ROW_INDEX == numberOfLoadedImages)
                return;

            //create div and set an image as background
            var widthToSet = Math.floor(TOTAL_WIDTH / MAX_COLUMN_INDEX);
            var heightToSet = Math.floor(TOTAL_HEIGHT / MAX_ROW_INDEX);
            var leftToSet = Math.floor((TOTAL_WIDTH / MAX_COLUMN_INDEX) *
(numberOfLoadedImages % MAX_COLUMN_INDEX));
            var topToSet = Math.floor((TOTAL_HEIGHT / MAX_ROW_INDEX) *
Math.floor(numberOfLoadedImages / MAX_COLUMN_INDEX));

            var bgImgIndex = "";

            if(String(numberOfLoadedImages).length == 1){bgImgIndex = "00";}
            else if(String(numberOfLoadedImages).length == 2){bgImgIndex = "0";}
            else if(String(numberOfLoadedImages).length == 3){bgImgIndex = "";}

            bgImgIndex += String(numberOfLoadedImages);
            var bgUrlToSet = getMediaFileUrl("memoryTestImages") + "lake-nature_" +
bgImgIndex + ".bmp";

            var divToAppend = document.createElement("div");

            divToAppend.style.position = "absolute";
            divToAppend.style.width = (widthToSet - 2) + "px";
            divToAppend.style.height = (heightToSet - 2) + "px";
            divToAppend.style.border = "1px solid #000000";
            divToAppend.style.left = leftToSet + "px";
            divToAppend.style.top = topToSet + "px";
            divToAppend.style.background = "url(" + bgUrlToSet + ")";
            divToAppend.style.backgroundPosition = "-" + (DEFAULT_IMG_LEFT +
leftToSet) + "px -" + (DEFAULT_IMG_TOP + topToSet) + "px";
            divToAppend.style.backgroundRepeat = "no-repeat";
            document.getElementById("ViewArea").appendChild(divToAppend);
            numberOfLoadedImages++;
            displayStatus();
        }
```

```
	function displayStatus()
	{
			setInnerTextById("number_of_loaded_image_result",
numberOfLoadedImages + "/" + (MAX_COLUMN_INDEX * MAX_ROW_INDEX));
			var totalSize = EACH_IMG_SIZE * numberOfLoadedImages;
			setInnerTextById("total_image_size_result",
toCommaNotation(String(totalSize)) + " bytes");
	}


	function toCommaNotation(origString)
	{
		var unitToSep = 3;
		var resultString = "";
		var processString = origString;

		while(processString.length > unitToSep)
		{
			var cutIndex = processString.length - unitToSep;
			var resultString = "," + processString.substring(cutIndex) +
resultString;
			processString = processString.substring(0, cutIndex);
		}
		resultString = processString + resultString;
		return resultString;
	}

	var resultString = " ";
	var eventCount = 0;

	function outOfMemoryHandler()
	{
		jsLog.lgevent('outofmemory');
		eventCount++;

		if(eventCount == 1)
			resultString += eventCount + " - " + event.available + " MB";
		else
			resultString += " / " + eventCount + " - " + event.available + "
MB";

		setInnerTextById("out_of_memory_event_occurred", resultString);
	}

	var usedMemorySize;
	function getUsedMemorySize()
	{
		if(window.NetCastGetUsedMemorySize)
		{
			usedMemorySize = window.NetCastGetUsedMemorySize();
			setInnerTextById("used_memory_result", usedMemorySize);
			jsLog.lgproperty('NetCastGetUsedMemorySize');
		}
	}

</script>
</head>

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

<!-- title -->
```

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

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

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

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

                        </div>
                  </div>
                  <div class='PropertyTitleArea'>
                        Properties
                        <div class='PropertyListArea'>
                        NetCastGetUsedMemorySize
                        </div>
                  </div>
                  <div class='EventTitleArea'>
                        Events
                        <div class='EventListArea'>
                        outofmemory
                        </div>
                  </div>
            </div>

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

            <div id='view'>
                  <div class='ViewArea' id= 'ViewArea' style="position: relative;">
                        <table border="0" cellpadding="0" cellspacing="0">
                              <tr height="50px">
                                    <td width=100px align="left"></td>
                              </tr>
                              <tr height="50px">
                                    <td width=300px align=left>1. Each image
size :</td>
                                    <td width=500px align="left">2,880,054
bytes</td>
                                    <td width=100px align="left"></td>
                              </tr>
                              <tr height="50px">
                                    <td width=300px align=left>2. Number of
loaded images :</td>
                                    <td width=500px align="left"><div
id="number_of_loaded_image_result"></div></td>
                                    <td width=100px align="left"></td>
                              </tr>
                              <tr height="50px">
                                    <td width=300px align=left>3. Total image
```

```
size :</td>
                                        <td width=500px align="left"><div
id="total_image_size_result"></div></td>
                                        <td width=100px align="left"></td>
                                  </tr>
                                  <tr height="50px">
                                        <td width=300px align=left>4. OutOfMemory
Event :</td>
                                        <td width=500px align="left"><div
id="out_of_memory_event_occurred"></div></td>
                                        <td width=100px align="left"></td>
                                  </tr>
                                  <tr height="50px">


                                        <td width="200px" align=left><div>5. Used
Memory Size : </div></td>
                                        <td align="left"><div
id="used_memory_result"></div></td>
                                  </tr>
                                  <tr height="50px">
                                        <td width=3000px align=center> Press Red /
Green / Yellow key to load more image(s)<br>
                                        Check if "outofmemory" event occurs and
available memory is acceptable.</td>
                                  </tr>
                            </table>
                      </div>
               </div>

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

        </div>



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

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

        <!-- red key description -->
        <div id='btn_red' class='buttonDescription redColor'>LOAD 1</div>

        <!-- green key description -->
        <div id='btn_green' class='buttonDescription greenColor'>LOAD 5</div>

        <!-- yellow key description -->
        <div id='btn_yellow' class='buttonDescription yellowColor'>LOAD 10</div>

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

</body>
</html>
```