LG Electronics

[Tutorial] LG Web_Handling Key Inputs Using JavaScript Events

Version 1.0 – October 2011

LGDEV-054

Home Entertainment Company
LG Electronics, Inc.

Copyright

About This Document

Revision History

Document Version	Date	Comment
1.0	October 12, 2011	Initial Version

Purpose

This document describes how to use JavaScript events for general key inputs in application.

Reference Documents

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

Conventions

Codes

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

Note, Caution

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

Note

Contains information about something that is helpful to you.

Caution

Contains important information about something that you should know.

Abbreviation

The following table defines the abbreviations used in this document.

Abbreviation	Description
API	Application Programming Interface

Contents

Figures

1 Introduction

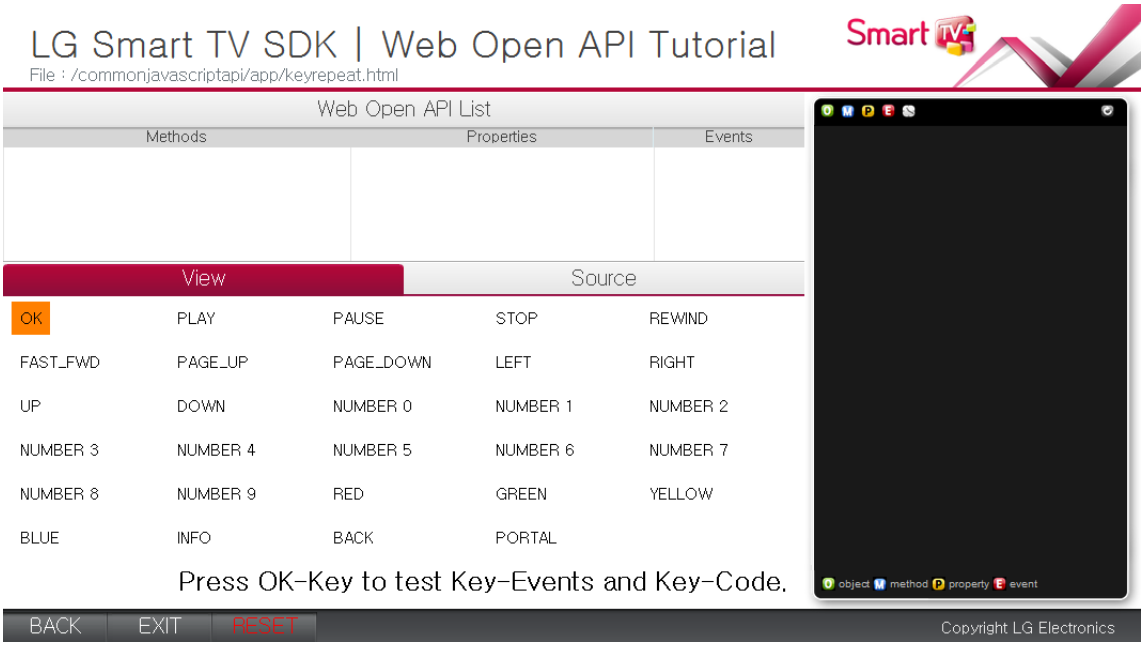
This chapter provides an overview of this application.

1.1 Overview

1.1 Overview

This application shows how to use JavaScript events for general key inputs in applications.

When receiving a key from remote control unit, you can check and control the received key using JavaScript event.



[Figure 1] Key Up/Down Application

Note

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

2 Creating Application

This chapter describes how to use JavaScript events for general key inputs.

2.1 Initializing the Page

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

Declare the variable.
01-15: Declare keysToTestArray and set key names and values.

initPage

This function initializes the page.
21: Record the last visited page when running the application.
24: Initialize the page.
25: Get the source code of the page using the XMLHttpRequest object.
26: Set the page ID.
27: Initialize the Log function.
30-31: Add an event handler which will executed when the corresponding button is pressed.
34: Add keydown event handler which will executed when menu keys (Back, Red, Green, and Yellow buttons) are pressed.
37: Call setKeysOnScreen() function.
40: Call setTimeout function, then, call initTest() function after 700 milliseconds.
42: Declare device.
43-46: Check device and determine whether to display portal key or not.

Sample Code

```
01 : var keysToTestArray = [["div_01", "OK", 13, "d_p_u"]
02 :   ,["div_02", "PLAY", 415, "d_u"],["div_03", "PAUSE", 19, "d_u"]
03 :   ,["div_04", "STOP", 413, "d_u"],["div_05", "REWIND", 412, "d_u"]
04 :   ,["div_06", "FAST_FWD", 417, "d_u"],["div_07", "PAGE_UP", 33, "d_u"]
05 :   ,["div_08", "PAGE_DOWN", 34, "d_u"],["div_09", "LEFT", 37, "d_u"]
06 :   ,["div_10", "RIGHT", 39, "d_u"],["div_11", "UP", 38, "d_u"]
07 :   ,["div_12", "DOWN", 40, "d_u"],["div_13", "NUMBER 0", 48, "d_p_u"]
08 :   ,["div_14", "NUMBER 1", 49, "d_p_u"],["div_15", "NUMBER 2", 50, "d_p_u"]
09 :   ,["div_16", "NUMBER 3", 51, "d_p_u"],["div_17", "NUMBER 4", 52, "d_p_u"]
10 :   ,["div_18", "NUMBER 5", 53, "d_p_u"],["div_19", "NUMBER 6", 54, "d_p_u"]
11 :   ,["div_20", "NUMBER 7", 55, "d_p_u"],["div_21", "NUMBER 8", 56, "d_p_u"]
12 :   ,["div_22", "NUMBER 9", 57, "d_p_u"],["div_23", "RED", 403, "d_u"]
13 :   ,["div_24", "GREEN", 404, "d_u"],["div_25", "YELLOW", 405, "d_u"]
14 :   ,["div_26", "BLUE", 406, "d_u"],["div_27", "INFO", 457, "d_u"]
15 :   ,["div_28", "BACK", 461, "d_u"],["div_29", "PORTAL", 1000, "d_u"]];
16 :
17 : //initialize page
18 : function initPage()
19 : {
20 :   //save page as last visited page
21 :   setLastVisitPage();
22 :
23 :   //common initialize function
24 :   commonInitialize();
25 :   requestSourceCode();
26 :   setPageID("Key Down/Up/Press");
27 :   jsLog.initLG();
28 :
29 :   //add onclick event handler
30 :   addEventHandler(document.getElementById("btn_back"), "click", backHandler);
31 :   addEventHandler(document.getElementById("btn_reset"),              "click",
     resetHandler);
32 :
33 :   //add event handler for menu
34 :   addEventHandler(document.body, "keydown", menuKeyHandler);
35 :
36 :   //set keys to test
37 :   setKeysOnScreen();
38 :
39 :   //add event handler to test
40 :   setTimeout(initTest,700);
41 :
42 :   var device = document.getElementById("device");
43 :   if(! device.supportPortalKey)
```

```
44 :   {
45 :           document.getElementById("div_29").style.visibility = "hidden";
46 :   }
47 : }
```

2.2 Handling Events

The following code shows event handler functions which are added from initPage() function.

initTest

This function adds handlers of key events.
03: Add keydown event handler.
04: Add keyup event handler.
05: Add keypress event handler.
06: Call initTestProgress() function.

backHandler

When the Back button is pressed, the page moves to the assigned location.

resetHandler

When the Reset button is pressed, Key test is initialized by calling initTestProgress() function.

menuKeyHandler

This function is keydown event handler and executes corresponding function for each menu button.
19-21: Declare and initialize backToYellow, resetToGreen, and hasMenuExecuted. These variables are used to distinguish the test key and bottom menu.
25: Declare userInput and set key code of the event.
26-32: Using switch-case, execute the corresponding function for received userInput and set hasMenuExecuted to true.
28: When Back key is pressed and the Back button is not yellow, call backHandler() function.
29: When Red key is pressed and the Reset button is not green, call resetHandler() function.
30: When Green key is pressed and the Reset button is green, call resetHandler() function.
31: When Yellow key is pressed and the Back button is yellow, call backHandler() function.

testKeyDown

This function is a handler that is executed when keydown event occurs.
35: Declare and initialize keyEventGathering. keyEventGathering is used to check if the pressed key is for testing purpose.
39: Set "_down_" and inputted value of event key to keyEventGathering.

testKeyPress

This function is keypress event handler.
46: Check menuKeyHandler() function and if menu button is executed, just return.
47: If menu button is not executed and only key inputs are received, set "_press_" and inputted value of event key to keyEventGathering.

testKeyUp

This function is a handler that is executed when keyup event occurs.
52: Set "_up_" and inputted value of event key to keyEventGathering
54-57: If hasMenuExecuted is true, initialize to false.
57-59: If hasMenuExecuted is false, call checkEachResult() function.

Sample Code

```
01 :   function initTest()
02 :   {
03 :          addEventHandler(document.body, "keydown", testKeyDown);
04 :          addEventHandler(document.body, "keyup", testKeyUp);
05 :          addEventHandler(document.body, "keypress", testKeyPress);
06 :          initTestProgress();
07 :   }
08 :
09 :   function backHandler()
10 :   {
11 :          window.location.replace("../menu_commonjavascriptapi.html");
```

```
12 :   }
13 :
14 :   function resetHandler()
15 :   {
16 :          initTestProgress();
17 :   }
18 :
19 :   var backToYellow = false;
20 :   var resetToGreen = false;
21 :   var hasMenuExecuted = false;
22 :
23 :   function menuKeyHandler(event)
24 :   {
25 :          var userInput = getKeyCode(event);
26 :          switch(userInput)
27 :          {
28 :                 case     VK_BACK     :     if(!backToYellow){backHandler();
   hasMenuExecuted = true;} break;
29 :                 case VK_RED : case 82 : if(!resetToGreen){resetHandler();
   hasMenuExecuted = true;} break;
30 :                 case VK_GREEN : case 71 : if(resetToGreen){resetHandler();
   hasMenuExecuted = true;} break;
31 :                 case VK_YELLOW : case 89 : if(backToYellow){backHandler();
   hasMenuExecuted = true;} break;
32 :          }
33 :   }
34 :
35 :   var keyEventGathering = "";
36 :
37 :   function testKeyDown(event)
38 :   {
39 :          keyEventGathering += "_down_" + getKeyCode(event);
40 :          jsLog.lgevent('javascript: Key Event Handling : KeyDown, Key : ' +
   getKeyName(event));
41 :   }
42 :
43 :   function testKeyPress(event)
44 :   {
45 :          jsLog.lgevent('javascript: Key Event Handling : KeyPress, Key : '
   + getKeyName(event));
46 :          if(menuKeyHandler(event, true) == "menuExecuted"){return;}
47 :          keyEventGathering += "_press_" + getKeyCode(event);
48 :   }
49 :
50 :   function testKeyUp(event)
51 :   {
52 :          keyEventGathering += "_up_" + getKeyCode(event);
53 :          jsLog.lgevent('javascript: Key Event Handling : KeyUp, Key : ' +
   getKeyName(event));
54 :          if(hasMenuExecuted){
55 :                 hasMenuExecuted = false;
56 :                 keyEventGathering = "";
57 :          }else{
58 :                 checkEachResult();
59 :          }
60 :   }
```

2.3 Displaying Text

The following code displays test keys on screen with text and change the color of text and background according to the proceeding status of the test.

setSelectedKey

Receives selectedKeyIndex and changes the background color of key which will be tested next.
03-10: The function is executed by number of the size of keysToTestArray.
05-07: Change the background color of key text of received selectedKeyIndex which is the key index of the next test.
07-09: Background colors of other key texts are set with default value.

initTestProgress

Set key text and class for testing.
13: Declare and initialize currentKeyIndex.
17: Initialize currentKeyIndex to 0.
18-21: Using For loop, set class to each key by number keysToTestArray size.
22: Call setSelectedKey() function and initialize to test the first key.
23: Display the description on screen.

setKeysOnScreen

Displays key text on screen which is assigned to keysToTestArray.

Sample Code

```
01 :   function setSelectedKey(selectedKeyIndex)
02 :   {
03 :          for(var i = 0 ; i < keysToTestArray.length ; i++)
04 :          {
05 :                 if(selectedKeyIndex == i){
06 :
       document.getElementById(keysToTestArray[i][0]).style.backgroundColor    =
   "#FF8000";
07 :                 }else{
08 :
       document.getElementById(keysToTestArray[i][0]).style.backgroundColor    =
   "";
09 :                 }
10 :          }
11 :   }
12 :
13 :   var currentKeyIndex = 0;
14 :
15 :   function initTestProgress()
16 :   {
17 :          currentKeyIndex = 0;
18 :          for(var i = 0 ; i < keysToTestArray.length ; i++)
19 :          {
20 :
       document.getElementById(keysToTestArray[i][0]).className="eachTestGuide";
21 :          }
22 :          setSelectedKey(0);
23 :          setInnerTextById("keydownuppress_test_description",  "Press  OK-Key
   to test Key-Events and Key-Code.");
24 :   }
25 :
26 :   function setKeysOnScreen()
27 :   {
28 :          for(var i = 0 ; i < keysToTestArray.length ; i++)
29 :          {
30 :                 setInnerTextById(keysToTestArray[i][0],
   keysToTestArray[i][1]);
31 :          }
32 :   }
```

2.4 Executing Test

Use **checkEachResult** function to test each key event and display in order.

03: Return if currentKeyIndex is bigger than the size of keysToTestArray
05: Declare and initialize keyGatheringToCompare.
07-16: Set the string to keyGatheringToCompare to compare if the correct key is pressed when key event is occurred according to the keys of keysToTestArray[currentKeyIndex].
18-20: If the value of keyEventGathering is same with the value of keyGatheringToCompare, the key text is displayed with green.
21-22: If it is not same, display with red.
24: Initialize keyEventGathering.
26-28: If currentKeyIndex is Red button, set resetToGreen to false, make button executable, and display button text with red.
31-34: If currentKeyIndex is Back button, set backToYellow to false, make button executable, and display button text with white.
36: Increase the value of currentKeyIndex by 1.
37: If the value of currentKeyIndex is smaller than the size of keysToTestArray execute the following codes.
38: Call setSelectedKey function with currentKeyIndex parameter.
40-42: If the value of keysToTestArray is “PORTAL” and the device does not supprt “PORTAL” key, display the corresponding text.
43-45: Otherwise, display the corresponding text.
47-50: If the value of keysToTestArray is Red and it is executed from test menu, set the key text in menu to change into green.
52-55: If the value of keysToTestArray is Back and it is executed in test key, set the key text in menu to change into yellow.
56-59: If the value of currentKeyIndex is bigger than the size of keysToTestArray, display the corresponding text.

Sample Code

```
01 :   function checkEachResult()
02 :   {
03 :         if(currentKeyIndex >= keysToTestArray.length){return;}
04 :
05 :     var keyGatheringToCompare = "";
06 :
07 :         if(keysToTestArray[currentKeyIndex][3] == "d_p_u"){
08 :               keyGatheringToCompare         =         "_down_"          +
   keysToTestArray[currentKeyIndex][2]
09 :                                                     +    "_press_"    +
   keysToTestArray[currentKeyIndex][2]
10 :                                                     +      "_up_"      +
   keysToTestArray[currentKeyIndex][2];
11 :         }else if(keysToTestArray[currentKeyIndex][3] == "d_u"){
12 :               keyGatheringToCompare         =         "_down_"          +
   keysToTestArray[currentKeyIndex][2]
13 :                                                     +      "_up_"      +
   keysToTestArray[currentKeyIndex][2];
14 :         }else{
15 :               keyGatheringToCompare = "";
16 :         }
17 :
18 :         if(keyEventGathering == keyGatheringToCompare){
19 :
      document.getElementById(keysToTestArray[currentKeyIndex][0]).className
   += " greenColor";
20 :         }else{
21 :
      document.getElementById(keysToTestArray[currentKeyIndex][0]).className
   += " redColor";
22 :         }
23 :
24 :         keyEventGathering = "";
25 :
26 :         if(keysToTestArray[currentKeyIndex][1] == "RED"){
27 :               resetToGreen = false;
28 :               document.getElementById("btn_reset").className            =
```

```
   "buttonDescription redColor";
29 :          }
30 :
31 :          if(keysToTestArray[currentKeyIndex][1] == "BACK"){
32 :                backToYellow = false;
33 :                document.getElementById("btn_back").className             =
   "buttonDescription whiteColor";
34 :          }
35 :
36 :          currentKeyIndex++;
37 :          if(currentKeyIndex < keysToTestArray.length){
38 :                setSelectedKey(currentKeyIndex);
39 :
40 :                var device = document.getElementById("device");
41 :                If((keysToTestArray[currentKeyIndex][1]=="PORTAL")   &&   !
   device.supportPortalKey){
42 :                      setInnerTextById("keydownuppress_test_description",
   "Press " + "RESET" + "-Key to test Key-Events and Key-Code.");

43 :                }else{
44 :                      setInnerTextById("keydownuppress_test_description",
   "Press " + keysToTestArray[currentKeyIndex][1] + "-Key to test Key-Events
   and Key-Code.");
45 :                }
46 :
47 :                if(keysToTestArray[currentKeyIndex][1] == "RED"){
48 :                      resetToGreen = true;
49 :                document.getElementById("btn_reset").className            =
   "buttonDescription yellowColor";
50 :                }
51 :
52 :                if(keysToTestArray[currentKeyIndex][1] == "BACK"){
53 :                      backToYellow = true;
54 :                      document.getElementById("btn_back").className       =
   "buttonDescription yellowColor";
55 :                }
56 :          }else{
57 :                setSelectedKey(-1);
58 :                setInnerTextById("keydownuppress_test_description",     "No
   more test is left on this page.");
59 :          }
60 :   }
```

2.5 Source Code of keydownuppress.html

Source code of keydownuppress.html is as follows:

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

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

<script>

	var keysToTestArray = [["div_01", "OK", 13, "d_p_u"]
					,["div_02", "PLAY", 415, "d_u"]
					,["div_03", "PAUSE", 19, "d_u"]
					,["div_04", "STOP", 413, "d_u"]
					,["div_05", "REWIND", 412, "d_u"]
					,["div_06", "FAST_FWD", 417, "d_u"]
					,["div_07", "PAGE_UP", 33, "d_u"]
					,["div_08", "PAGE_DOWN", 34, "d_u"]
					,["div_09", "LEFT", 37, "d_u"]
					,["div_10", "RIGHT", 39, "d_u"]
					,["div_11", "UP", 38, "d_u"]
					,["div_12", "DOWN", 40, "d_u"]
					,["div_13", "NUMBER 0", 48, "d_p_u"]
					,["div_14", "NUMBER 1", 49, "d_p_u"]
					,["div_15", "NUMBER 2", 50, "d_p_u"]
					,["div_16", "NUMBER 3", 51, "d_p_u"]
					,["div_17", "NUMBER 4", 52, "d_p_u"]
					,["div_18", "NUMBER 5", 53, "d_p_u"]
					,["div_19", "NUMBER 6", 54, "d_p_u"]
					,["div_20", "NUMBER 7", 55, "d_p_u"]
					,["div_21", "NUMBER 8", 56, "d_p_u"]
					,["div_22", "NUMBER 9", 57, "d_p_u"]
					,["div_23", "RED", 403, "d_u"]
					,["div_24", "GREEN", 404, "d_u"]
					,["div_25", "YELLOW", 405, "d_u"]
					,["div_26", "BLUE", 406, "d_u"]
					,["div_27", "INFO", 457, "d_u"]
					,["div_28", "BACK", 461, "d_u"]
					,["div_29", "PORTAL", 1000, "d_u"]];

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

		//common initialize function
		commonInitialize();
		requestSourceCode();
		setPageID("Key Down/Up/Press");
		jsLog.initLG();

		//add onclick event handler
		addEventHandler(document.getElementById("btn_back"), "click",
backHandler);
		addEventHandler(document.getElementById("btn_reset"), "click",
resetHandler);
```

```
		//add event handler for menu
		addEventHandler(document.body, "keydown", menuKeyHandler);

		//set keys to test
		setKeysOnScreen();

		//add event handler to test
		setTimeout(initTest,700);

		var device = document.getElementById("device");
		if(! device.supportPortalKey)
		{
			document.getElementById("div_29").style.visibility =
"hidden";
		}
	}

	function initTest()
	{
		addEventHandler(document.body, "keydown", testKeyDown);
		addEventHandler(document.body, "keyup", testKeyUp);
		addEventHandler(document.body, "keypress", testKeyPress);
		initTestProgress();
	}

	function backHandler()
	{
		window.location.replace("../menu_commonjavascriptapi.html");
	}

	function resetHandler()
	{
		initTestProgress();
	}

	var backToYellow = false;
	var resetToGreen = false;
	var hasMenuExecuted = false;

	function menuKeyHandler(event)
	{
		var userInput = getKeyCode(event);
		switch(userInput)
		{
			case VK_BACK : if(!backToYellow){backHandler();
hasMenuExecuted = true;} break;
			case VK_RED : case 82 : if(!resetToGreen){resetHandler();
hasMenuExecuted = true;} break;
			case VK_GREEN : case 71 : if(resetToGreen){resetHandler();
hasMenuExecuted = true;} break;
			case VK_YELLOW : case 89 : if(backToYellow){backHandler();
hasMenuExecuted = true;} break;
		}
	}

	var keyEventGathering = "";

	function testKeyDown(event)
	{
		keyEventGathering += "_down_" + getKeyCode(event);
		jsLog.lgevent('javascript: Key Event Handling : KeyDown, Key : '
+ getKeyName(event));
	}

	function testKeyPress(event)
	{
		jsLog.lgevent('javascript: Key Event Handling : KeyPress, Key :
' + getKeyName(event));
		if(menuKeyHandler(event, true) == "menuExecuted"){return;}
		keyEventGathering += "_press_" + getKeyCode(event);
	}
```

```
	function testKeyUp(event)
	{
		keyEventGathering += "_up_" + getKeyCode(event);
		jsLog.lgevent('javascript: Key Event Handling : KeyUp, Key : ' +
getKeyName(event));
		if(hasMenuExecuted){
			hasMenuExecuted = false;
			keyEventGathering = "";
		}else{
			checkEachResult();
		}
	}

	function setSelectedKey(selectedKeyIndex)
	{
		for(var i = 0 ; i < keysToTestArray.length ; i++)
		{
			if(selectedKeyIndex == i){

	document.getElementById(keysToTestArray[i][0]).style.backgroundColor =
"#FF8000";
			}else{

	document.getElementById(keysToTestArray[i][0]).style.backgroundColor =
"";
			}
		}
	}

	var currentKeyIndex = 0;

	function initTestProgress()
	{
		currentKeyIndex = 0;
		for(var i = 0 ; i < keysToTestArray.length ; i++)
		{
			document.getElementById(keysToTestArray[i][0]).className =
"eachTestGuide";
		}
		setSelectedKey(0);
		setInnerTextById("keydownuppress_test_description", "Press OK-
Key to test Key-Events and Key-Code.");
	 }

	//set keys to test
	function setKeysOnScreen()
	{
		for(var i = 0 ; i < keysToTestArray.length ; i++)
		{
			setInnerTextById(keysToTestArray[i][0],
keysToTestArray[i][1]);
		}
	}

	function checkEachResult()
	{
		if(currentKeyIndex >= keysToTestArray.length){return;}
		var keyGatheringToCompare = "";

		if(keysToTestArray[currentKeyIndex][3] == "d_p_u"){
			keyGatheringToCompare = "_down_" +
keysToTestArray[currentKeyIndex][2]
							+ "_press_" +
keysToTestArray[currentKeyIndex][2]
							+ "_up_" +
keysToTestArray[currentKeyIndex][2];
		}else if(keysToTestArray[currentKeyIndex][3] == "d_u"){
			keyGatheringToCompare = "_down_" +
keysToTestArray[currentKeyIndex][2]
							+ "_up_" +
```

```
keysToTestArray[currentKeyIndex][2];
		}else{
			keyGatheringToCompare = "";
		}

		if(keyEventGathering == keyGatheringToCompare){

	document.getElementById(keysToTestArray[currentKeyIndex][0]).className
+= " greenColor";
		}else{

	document.getElementById(keysToTestArray[currentKeyIndex][0]).className
+= " redColor";
		}

		keyEventGathering = "";
		//in case just tested red key
		if(keysToTestArray[currentKeyIndex][1] == "RED"){
			resetToGreen = false;
			document.getElementById("btn_reset").className =
"buttonDescription redColor";
		}
		//in case just tested back key
		if(keysToTestArray[currentKeyIndex][1] == "BACK"){
			backToYellow = false;
			document.getElementById("btn_back").className =
"buttonDescription whiteColor";
		}

		currentKeyIndex++;
		if(currentKeyIndex < keysToTestArray.length){
			setSelectedKey(currentKeyIndex);
			var device = document.getElementById("device");
			if((keysToTestArray[currentKeyIndex][1]=="PORTAL")
&& !device.supportPortalKey){
				if(! device.supportPortalKey){

	setInnerTextById("keydownuppress_test_description", "Press " + "RESET"
+ "-Key to test Key-Events and Key-Code.");
				}
			}else{
				setInnerTextById("keydownuppress_test_description",
"Press " + keysToTestArray[currentKeyIndex][1] + "-Key to test Key-Events and
Key-Code.");
			}

			//in case next test is red key
			if(keysToTestArray[currentKeyIndex][1] == "RED"){
				resetToGreen = true;
			document.getElementById("btn_reset").className =
"buttonDescription yellowColor";
			}
			//in case next testis back key
			if(keysToTestArray[currentKeyIndex][1] == "BACK"){
				backToYellow = true;
				document.getElementById("btn_back").className =
"buttonDescription yellowColor";
			}
		}else{
			setSelectedKey(-1);
			setInnerTextById("keydownuppress_test_description", "No
more test is left on this page.");
		}
	}

	function getKeyName(event)
	{
		var key = getKeyCode(event);
		for(var i = 0 ; i < keysToTestArray.length ; i++)
		{
			if(keysToTestArray[i][2] == key)
```

```
                        {
                                return keysToTestArray[i][1];
                        }
                }

                return "";
        }

</script>
</head>

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

<!-- title -->
<div class='SuiteTitle' >LG Smart TV SDK | Web Open API Tutorial</div>
<object type="application/x-netcast-info"
                id="device">
</object>
<!-- navigation -->

<div class='SuiteNavigation'>
        <div style="float:left;">File :
/commonjavascriptapi/app/keyrepeat.html</div>
</div>
<div class='SuiteTitleLine'> </div>
<!-- test contents -->
<div class='ContentArea'>
        <div class='ApiListTitleArea'>Web Open API List</div>
        <div class='ApiListArea'>
                <div class='MethodTitleArea'>
                        Methods
                        <div class='MethodListArea'>

                        </div>
                </div>
                <div class='PropertyTitleArea'>
                        Properties
                        <div class='PropertyListArea'>

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

                        </div>
                </div>
        </div>
        <div class='ViewTitleArea'>
                <div id='tabViewArea' class='SelectedViewArea'
style='float:left;' onclick="showView();">View</div>
                <div id='tabCodeArea' class='UnselectedViewArea'
style='float:right;' onclick='showCode();'>Source</div>
        </div>
        <div id='view'>
                <div class = 'ViewArea'>
                        <table width="880px" border="0" cellpadding="0"
cellspacing="0">
                                <tr height="50px">
                                <td width=200px align=left><div
class='eachTestGuide' id="div_01"></div></td>
                                <td width=200px align=left><div
class='eachTestGuide' id="div_02"></div></td>
                                <td width=200px align=left><div
class='eachTestGuide' id="div_03"></div></td>
                                <td width=200px align=left><div
class='eachTestGuide' id="div_04"></div></td>
                                <td width=200px align=left><div
class='eachTestGuide' id="div_05"></div></td>
                                </tr>
```

```
                            <tr height="50px">
                            <td width=200px align=left><div
class='eachTestGuide' id="div_06"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_07"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_08"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_09"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_10"></div></td>
                            </tr>
                            <tr height="50px">
                            <td width=200px align=left><div
class='eachTestGuide' id="div_11"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_12"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_13"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_14"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_15"></div></td>
                            </tr>
                            <tr height="50px">
                            <td width=200px align=left><div
class='eachTestGuide' id="div_16"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_17"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_18"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_19"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_20"></div></td>
                            </tr>
                            <tr height="50px">
                            <td width=200px align=left><div
class='eachTestGuide' id="div_21"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_22"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_23"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_24"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_25"></div></td>
                            </tr>
                            <tr height="50px">
                            <td width=200px align=left><div
class='eachTestGuide' id="div_26"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_27"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_28"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_29"></div></td>
                            <td width=200px align=left><div
class='eachTestGuide' id="div_30"></div></td>
                            </tr>
                        </table>
                </div>
        </div>
        <div   scrolling="AUTO" style="visibility: hidden" id='codeview'>
                <textarea class="SourceCodeArea" value=""
id='sourcecode'></textarea>
        </div>
</div>

<!-- description -->
<div id='keydownuppress_test_description' class='SuiteDescription'
```

```
style="position: absolute; left: 40px; top:630px"></div>

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

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

      <!-- red key description -->
      <div id='btn_reset' class='buttonDescription redColor'>RESET</div>

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

</body>
</html>
```