LG Electronics

Web App Development Quick Start Guide (WebAppSample_HTML5 Video Player)

Version 1.0.0 – January 2013

LGDEV-088

Home Entertainment Company
LG Electronics, Inc.

Copyright

About This Document

Revision History

Document Version	Date	Comment
1.0.0	2013, January 10	Initial Release

Purpose

This document is a quick guide intended to describe how developers develop and test web applications using the WebAppSample_HTML5 Video Player example.

Reference Documents

Refer to the following documents:
- Developer Library in LG Developer (http://developer.lge.com) website

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
CSS	Cascading Style Sheet
SDK	Software Development Kit

Contents

Figures

Tables

1 Introduction

This chapter describes WebAppSample_HTML5 Video Player Sample and its structure.

1.1 Overview of WebAppSample_HTML5 Video Player

A web application that is developed based on general HTML standards has the following structure for its operation. If you separate each component independently, it is easy to manage application sources. Developers need to implement each component as necessary.

- **HTML: Defines application structure**
- **JavaScript: Defines application operation**
- **CSS: Defines application design elements**

In this guide, we are going to learn how to develop and test a HTML5 video player web application. HTML5 Video Player Sample uses HTML5 <video> element for displaying the video. This Sample includes the basic functionalities of the video player like play, pause, forward and backward. The seek bar or the status bar comes by default.

This sample also provides RCU (Remote Control Unit) key navigation for control elements inside the sample. It will always focus the currently selected control/button.

The operation of video player Sample application is shown below.

[Figure 1] WebAppSample_HTML5 Video Player

1.2 Structure and Resources of Sample Project

The file structure of WebAppSample_HTML5 Video Player that is provided as a sample with this guide is shown below. You can download the sample from **[Resource Center > Smart TV > SDK & Tools > Tools & Samples]** menu in LG Developer (**http://developer.lge.com**) website.

You can either copy this sample source into the WebContent folder of the project that is created in LG IDE or enter the code directly by referring to the description in this guide.

[Figure 2] Structure and Resources of Sample Project

[Table 1] Structure of Sample Project

Folder		File	Description
WebAppSample _HTML5 Video Player		index.html	Main screen file of an application
	css	player.css	Design elements of an application
	js	keycode.js	Key code definitions and value assignments
		videoplayer.js	Video player playback control functionalities.

2 Setting Up the Project

Although it is okay to use a general editor to develop a web application, you can easily develop, debug, and test the application if you use the LG IDE. This chapter describes how to set up a project using the LG IDE.

Note

- The description of this section is based on **LG Smart TV SDK V2.0.0 or higher** version.

- For detailed information on how to start using LG IDE, refer to **Developing > Using SDK** section in Developer Library in LG Developer (**http://developer.lge.com**) website.

1. Open LG IDE and click **[File > New > LG Web Project].**

2. In the 'New LG Web Project' window, give a project name and select 'SDK Version for Web Application'. Then, click **[Finish].**

3. Add JavaScript, CSS, and image files.

Create each folder for JavaScript, CSS, or image files that will be used in this example in the WebContent folder. Although folders are not mandatory, it will be helpful to manage resources using folders when considering resources increase in the future.

Select and right-click the WebContent node and select **[File]** and then enter a file name including its extension in the file name field to add JavaScript or a CSS file.

4. Now, a basic foundation is prepared to implement a sample example.

3 Implementing the Code

This chapter describes the description and expected results of WebAppSample_HTML5 Video Player example.

3.1 Descriptions on Source Codes
3.2 Expected Result

Note

Refer to **Developing > API** section in Developer Library in LG Developer (**http://developer.lge.com**) website for detailed information on how to use Web APIs.

3.1 Descriptions on Source Codes

This section describes the core part of the sample source codes. The entire source codes are provided in the attached source zip file.

Following sources are related with the main screen (index.html)

Linking JavaScript and CSS (index.html)

Specifies the path of JavaScript and style sheet source file that will be used in an application.

```
<link rel="stylesheet" href="css/player.css"/>
<script src="js/jquery-1.8.2.min.js"></script>
<script src="js/keycode.js"></script>
<script src="js/videoplayer.js"></script>
<script src="js/jquery-ui.js"></script>
```

Designing displaying structure (index.html)

Designs the structure to be displayed on an application screen using the tag.

```
<div class="playerLayout">
Main Display Area

<video id="video" style="background-color: gray;" width="1280" height="720">
Video element declairation with width and height
</video>

<div class="playerBottom">
Bottom area of video player

<div class="playerButtonLayout">
Video player button layout area

<div class="progessBarLayout">
Progress bar and its components
</div>

<div class="buttonLayout">
Playback control buttons
</div>

<div id="ballCoverage">
<div id="progressBall" class="progressBallInitial" > </div>
</div>
Progress Ball Area

<div class="keyHelp" >
    <div class="backKey"></div> Back Button
    <div class="exitKey"></div> Exit Button
</div>
Area for Help Keys
```

Displaying the progress bar (index.html)

The status of the progress bar and the remaining time are updated as the video is being played.

Progress Bar Area is divided in five parts. First part is background, second part will show buffered position, third will show current status, fourth part is clicked area and last part will show remain time and total time.

```
<div class="progressBar">
          <div id="progressBg" class="progress progressBg"></div>
          <div id="progressBuffer" class="progress progressBuffer" ></div>
          <div id="progressBarStatus" class="progress
progressBarStatus"></div>
          <div id="progressBarClick" class="progress progressBarClick"></div>
```

```
          <div class="runningTime"> <span id="remainingTime"></span> <span
id="totalTime" ></span> </div>
        </div>
```

Displaying the running movie information (index.html)

Displays the name and type of the running video.

```
<div class="runningMovieInfo">
          <div class="runningMovieName"> </div>
          <div class="runningMovieType"></div>
</div>
```

Displaying the button layout (index.html)

Contains various buttons like stop, play, rewind, forward and option.

```
<div id="buttonLayout">
        <ul>
          <li id="stop" class="stopButton" ><img
src="images/player_btn_icon/movie_btn_icon_stop_n.png" alt="stopBtn"
class="center"/></li> Stop button

          <li id="play" class="playButton"><img
src="images/player_btn_icon/movie_btn_icon_play_n.png" alt="playBtn" class="center"
/></li> Play button

          <li id="rewind" class="rewindButton"><img
src="images/player_btn_icon/movie_btn_icon_rewind_n.png" alt="rewindBtn"
class="center" /></li> Rewind button

          <li id="forward" class="forwardButton"><img
src="images/player_btn_icon/movie_btn_icon_forward_n.png" alt="forwardBtn"
class="center"/></li> Forward button

          <li id="option" class="optionButton">
            <div class="imgTextCenter"> <img
src="images/player_btn_icon/movie_btn_icon_option_n.png" alt="optionBtn"/> <b
class="textCenter">Option</b> </div> Option button

          </li>
        </ul>
      </div>
```

Implementing to load the video player(videoplayer.js)

This has the implementation of ready function, which loads the video player. The video object is created and the playMedia()method is called.

playMedia method plays & pauses the video depending on the ready status of video.

```
$('document').ready(function() {
    allMenuObject =new Array();
    allExitObject =new Array();
    video = getVideo();

    playMedia();

});
```

Gets video element embedded inside web page

```
function getVideo() {
    return document.getElementById('video');
}
```

Implementing the play and pause functionalities(videoplayer.js)

Code for playing and pausing the video. The readyState property returns the current ready state of the audio/video. The ready state indicates if the audio/video is ready to play or not.

If ready state of video is 0, video is stopped, then loadDataSrc method will load video content and play the video.

If ready state of video is 4 and if video is in paused state, play the content, else pause the video.

```
function playMedia()
{
    stopped=false;
    if(video.readyState == 0)
    {
        loadDataSrc(true);
        video.play();
    }
    if(video.readyState == 4)
    {
        if(video.paused)
        {
            //play
            video.play();
        }
        else
        {
            //pause
            video.pause();
        }
    }
    $('#progressBall').attr('class', 'progressBall');
}
```

Implementing the forward media functionality (videoplayer.js)

Sets the current playback position in the video by 10 seconds to forward it.

```
function forwardMedia()
{
    if(video.currentTime < video.duration && !video.paused)
    {
        video.currentTime+=10;
    }
}
```

Implementing the rewind media functionality(videoplayer.js)

Sets the current playback position in the video by -10 seconds to rewind it.

```
function rewindMedia()
{
    if(video.currentTime > 0 && !video.paused)
    {
        video.currentTime-=10;
    }
}
```

Implementing the option media functionality(videoplayer.js)

Code for opening a window to setup the aspect ratio for full screen video, picture quality adjustment and audio adjustment.

```
function optionMedia()
{
    if(window.NetCastLaunchQMENU) {
          window.NetCastLaunchQMENU();
      }
}
```

Implementing to check the file extensions(videoplayer.js)

Code to check the supported formats of the video.

```
function checkFileExtensions (url) {
      for (var i = 0; i < supportedMimeTypes.length; i++) {
            if (url.lastIndexOf(supportedMimeTypes[i]) > 0) {
              currentType=url.slice(url.indexOf(".")+1,url.length);
                 return true;
            }
        }
         return false;
}
```

Processing key event (videoplayer.js)

Defines action according to the remote control key input.

NetCastBack method is used to return to my app screen on click of return button.

setFocus method is used to change the color of the button from normal to focus based on index parameter (mouse over or remote keynavigation event).

selectedButton method is used to handle mouse click event or click of Enter button.

showPlayer method is used to show the player controls.

triggerHide method is used to call the hide function in 5 seconds.

```
$(document).keydown(function(event) {
       var key = event.keycode || event.which;
       switch(key){
          case VK_BACK :if(window.NetCastBack){
                          window.NetCastBack();
                        }
                      break;
          case VK_RIGHT :
                        setFocus(1)
                        break;
          case VK_LEFT :
                        setFocus(-1)
                        break;
          case VK_UP :
                        if(rowID<4){
                          rowID++;
                        }
                        if(     video.paused  && rowID==2){
                          rowID=3;
                        }

                        if(rowID==3){
                          tempcntIndex=cntIndex;
                          cntIndex>1?cntIndex=1:"";
                        }
```

```
                        setFocus(0);
                        break;
            case VK_DOWN :
                        if(rowID>1){
                          rowID--;
                          tempcntIndex!=-1?cntIndex=tempcntIndex:"";
                          tempcntIndex!=-1;
                        }
                        if(     video.paused && rowID==2){
                          rowID=1;
                        }
                        setFocus(0);
                        break
            case VK_ENTER :
                        $(".playerBottom").is(":visible")?
selectedButton():"";
                        break;
            }
            showPlayer();
            triggerHide();
    });
});
```

Event Listeners in HTML5 Video Player (videoplayer.js)

Play

The play event is triggered when the video has been started or is no longer paused. When video starts playing, it changes the play image to pause image and gets the playing video information.

getVideoSourceInfo is used to get the source information like movie name and type of running video's.

getVideoPlayInfo used to get playing information like total time and remaining time of the currently running video.

```
video.addEventListener("play",function(){
       getVideoSourceInfo();
       getVideoPlayInfo();
       triggerHide();
       $("#play > img").remove();
       $('#play').append('<img
src="images/player btn icon/movie btn icon pause n.png" class="center" />');
       /*when mouse is clicked on progress bar, progressBall moves & video
plays from where progressBall will be pointing */
       $('.progressBarClick').click(function(e){
           if(!video.paused){
              var x = e.pageX-$(this).offset().left;
              video.currentTime=(x/progressBarWidth)*video.duration;
           }
       });
       });
```

Pause

The pause event is triggered when the video has been paused. When the video is been paused, it changes the pause image to play image.

```
video.addEventListener("pause",function(){
           $("#play > img").remove();
           $('#play').append('<img
src="images/player btn icon/movie btn icon play n.png" class="center" />');
    });
```

TimeUpdate

The timeupdate event is triggered when the current playback position has changed. It may be due to forward, rewind or dragging the progressBall.

```
video.addEventListener("timeupdate",function(){
        getVideoPlayInfo()
    });
```

Progress

The progress event is triggered when the browser is buffering the video.

```
video.addEventListener("progress", function(){
        buffering();
    });
```

Ended

The ended event is triggered when the video has ended.
When the video is ended, resetProgress method resets progress bar and time related UI.

```
video.addEventListener("ended",endPlay,false);
    function endPlay() {
        video.src=" ";
        resetProgress();
        if(!stopped){
            vidIndex<videoArray.length-1 ?vidIndex++:vidIndex=0;
            playMedia();
        }
    }
```

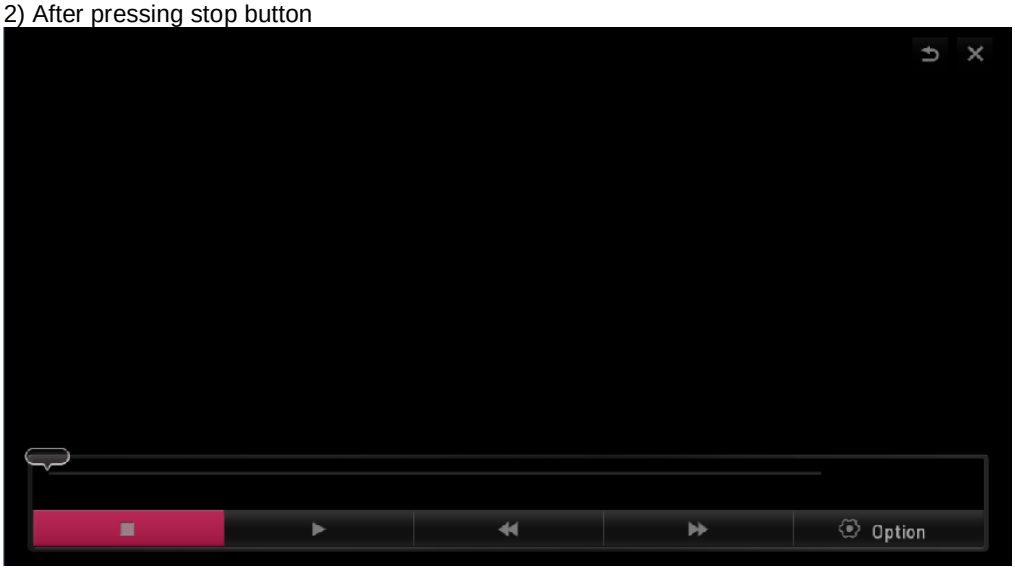
3.2 Expected Result

The following screen will be displayed when the sample application starts on the LG Smart TV Emulator 2012. The emulator version is based on SDK V2.1.0 release.

1) Initial Screen

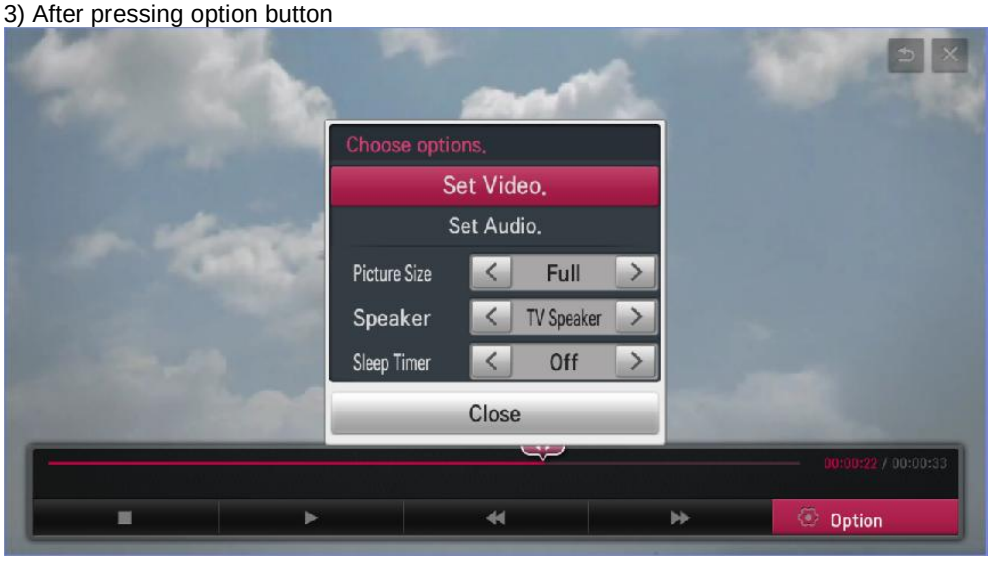
[Figure 3] WebAppSample__ HTML5 Video Player_Initial Screen

2) After pressing stop button



[Figure 4] WebAppSample__ HTML5 Video Player_Stop Screen

3) After pressing option button



[Figure 5] WebAppSample__ HTML5 Video Player_Option Screen

4 Testing the Application

This chapter briefly describes how to test a developed application.

To debug the application on emulator, refer to **Testing > Testing App on Emulator** section in Developer Library in LG Developer (**http://developer.lge.com**) website

To run and debug the application on real TV, refer to **Testing > Deploying and Testing App on Real TV** section in Developer Library in LG Developer website.