Dolby® Game Audio Unity Plug-in Quick Start Guide

Issue 1

Confidential Information

Dolby Laboratories Licensing Corporation

Corporate Headquarters

Dolby Laboratories, Inc.
Dolby Laboratories Licensing Corporation
100 Potrero Avenue
San Francisco, CA 94103-4813 USA
Telephone 415-558-0200
Fax 415-863-1373
www.dolby.com

European Licensing Liaison Office

Dolby International AB
Apollo Building, 3E
Herikerbergweg 1-35
1101 CN Amsterdam Zuidoost
The Netherlands
Telephone 31-20-651-1800
Fax 31-20-651-1801

Asia

Dolby Japan K.K.
NBF Higashi-Ginza Square 3F
13–14 Tsukiji 1-Chome, Chuo-ku
Tokyo 104-0045 Japan
Telephone 81-3-3524-7300
Fax 81-3-3524-7389
www.dolby.co.jp

Dolby Laboratories Hong Kong Limited
Unit 5407, Central Plaza
18 Harbour Road
Wanchai, Hong Kong
Telephone 852-2519-0888
Fax 852-2519-8988

Dolby Laboratories International Services (Shanghai) Co., Ltd.
05–07a, Floor 18
The Center
989 Chang Le Road
Shanghai 200031 China
Telephone 86-21-6113-3456
Fax 86-21-6113-3400
www.dolby.com.cn

Issue 1
S13/27235

The Dolby Game Audio Unity Plug-in

Dolby® audio processing technology improves audio quality, which leads to a more enjoyable listening experience for movies, music, and video games. This technology has been integrated into a number of popular portable devices, such as the Amazon™ Kindle™ Fire™ HD. To make Dolby audio processing technology available for mobile game use, Dolby has developed a plug-in for the Unity engine.

This document describes how to integrate and use the Dolby Game Audio Unity plug-in. It also provides C# script and JavaScript that you can use for reference.

1 Integrating the Dolby Game Audio Unity Plug-in

The Dolby Game Audio Unity plug-in library consists of these files:

- libDSPlugin.so
- DSJavaPlugin.jar

To integrate the plug-in with your application, do the following:

1. Copy libDSPlugin.so and DSJavaPlugin.jar to Assets/Plugins/Android.
2. To call functions, add interface function script code to your application. The examples in this section show the syntax that you must use.

C# Script Example

```
…
[DllImport(“DSPlugin”)]
Private static extern bool isAvailable();
[DllImport(“DSPlugin”)]
public static extern int initialize();
…
```

Java Script Example

```
…
@DllImport (“DSPlugin”)
Static private function isAvailable() : Boolean {};
@DllImport (“DSPlugin”)
static private function initialize() : int {};
…
```

For more information on using the Google™ Android™ plug-in, refer to:

- http://docs.unity3d.com/Documentation/Manual/PluginsForAndroid.html
- http://docs.unity3d.com/Documentation/Manual/Plugins.html

2 Using the Dolby Game Audio Unity Plug-in

To avoid unexpected bugs and to make the best use of the Dolby Game Audio Unity plug-in, do the following:

- Check whether the target device for your application has Dolby audio processing technology available. Ensure that your application calls the `isAvailable()` application program interface (API) function before calling any other functions in the library.
- To avoid changing the system-wide Dolby audio processing configuration, call the `suspendSession()` and `restartSession()` API functions at the appropriate times. Refer to http://developer.android.com/guide/components/activities.html to learn about the activity Android application component. Then, review *Dolby Game Audio Unity Plug-in Interface Specification* for information on when these functions should be used.
- When the game is launched, call `initialize()` to initialize Dolby audio processing.
- When the game is terminated, call `release()` to reclaim the memory that Dolby audio processing used when the game was active.

Note: Dolby audio processing is switched on automatically after the Dolby plug-in is loaded, even if it was in off status before being loaded. When the game is launched, the Movie profile is set as default. The game developer can change the profile by using the corresponding API function.

3 Application Workflow

Dolby Laboratories has provided C# and JavaScript source code to show you how to use the Dolby Game Audio Unity plug-in. The code samples in this section highlight the workflow between your application and the plug-in functions.

3.1 C# Workflow

The following steps show the work flow between a C# application and the Dolby Game Audio Unity plug-in. The complete C# source code is available in CsDSControl.cs.

1. The application must import functions from libDSPlugin.so.

```
[DllImport("DSPlugin")]
public static extern  bool isAvailable();
[DllImport("DSPlugin")]
public static extern  int initialize();
[DllImport("DSPlugin")]
public static extern  int setProfile(int profileid);
[DllImport("DSPlugin")]
public static extern  int suspendSession();
[DllImport("DSPlugin")]
public static extern  int restartSession();
[DllImport("DSPlugin")]
public static extern void release();
```

…

2. The application must check whether Dolby audio processing is available.

```
bool bAvail = isAvailable();
```

3. The application initializes Dolby audio processing if it is available.

```
int ret = initialize()
```

4. The application must call `suspendSession()` when it exits the foreground and `restartSession()` when it reenters the foreground.

   a. The application sends a `pauseStatus` of `true` to all game objects when the game pauses, and the application can invoke `suspendSession()` while the game is paused.

```
void OnApplicationPause(bool pauseStatus) {
if (pauseStatus) {
    int ret = suspendSession();
    LOGI("void OnApplicationPause(bool pauseStatus), suspendSession = " +
ret);
    }
LOGI("void OnApplicationPause(bool pauseStatus), pauseStatus = " +
pauseStatus);
}
```

   b. Using the `OnApplicationFocus()` call-back function, the application sends `focusStatus` to all game objects when the game gets or loses focus. When the value of `focusStatus` is `true`, the application can invoke the `restartSession()` function.

```
void OnApplicationFocus(bool focusStatus) {
    if (focusStatus) {
            int ret = restartSession();
            LOGI("void OnApplicationFocus(bool focusStatus), restartSession =
" + ret);
    }
    LOGI("void OnApplicationFocus(bool focusStatus), focusStatus = " +
focusStatus);
}
```

5. When the application is shutting down, it must send the result of `OnApplicationQuit()` to all game objects. Ensure that it releases the instance of `DolbyAudioProcessing` by having the application call `release()`.

```
void OnApplicationQuit() {
    release();
    LOGI("void OnApplicationQuit()");
}
```

…

3.2 JavaScript Workflow

The code samples in this section show the workflow between the JavaScript application and the Dolby Game Audio Unity plug-in. The complete JavaScript source code is available in JsDSControl.js.

1. The application must import functions from libDSPlugin.so.

```
@DllImport ("DSPlugin")
static private function isAvailable() : int {};
@DllImport ("DSPlugin")
static private function initialize() : int {};
@DllImport ("DSPlugin")
static private function setProfile(profileid : int) : int {};
@DllImport ("DSPlugin")
static private function suspendSession() : int {};
@DllImport ("DSPlugin")
static private function restartSession() : int {};
@DllImport ("DSPlugin")
static private function release() {};
...
```

2. The application must check whether Dolby audio processing is available.

```
var bAvail : boolean isAvailable();
```

3. The application initializes Dolby audio processing if Dolby audio processing is available.

```
ret = initialize();
```

4. The application must call `suspendSession()` when it exits the foreground and `restartSession()` when it reenters the foreground.

   a. The application sends a `pauseStatus` of `true` to all game objects when the game pauses, and the application can invoke `suspendSession()` while the game is paused.

```
function OnApplicationPause(pauseStatus: boolean) {
if (pauseStatus) {
    var ret = suspendSession();
    LOGI("function OnApplicationPause(pauseStatus: boolean), suspendSession
= " + ret);
    }
LOGI("function OnApplicationPause(pauseStatus: boolean), pauseStatus = " +
pauseStatus);
}
```

   b. Using the `OnApplicationFocus()` call-back function, the application sends `focusStatus` to all game objects when the player gets or loses focus. When the value of `focusStatus` is `true`, the application can invoke the `restartSession()` function.

```
function OnApplicationFocus(focusStatus: boolean) {
    if (focusStatus) {
        var ret = restartSession();
           LOGI("function OnApplicationFocus(focusStatus: boolean),
restartSession = " + ret);
    }
    LOGI("function OnApplicationFocus(focusStatus: boolean), focusStatus = "
+ focusStatus);
}
```

5. When the application is shutting down, it must send the result of `OnApplicationQuit()` to all game objects. Ensure that it releases the instance of `DolbyAudioProcessing` by having the application call `releaseDolbyAudioProcessing()`.

```
function OnApplicationQuit() {
    releaseDolbyAudioProcessing();
    LOGI("function OnApplicationQuit()");
}
...
```

4 Best Practices

- Call the `suspendSession()` function when the application leaves the foreground, and call the `restartSession()` function when it reenters the foreground. This ensures that the application correctly maintains the system-wide Dolby audio processing configuration.

Note: `suspendSession()` and `restartSession()` are paired functions. The application should invoke `suspendSession()` with the `OnApplicationPause()` call-back function when the call-back function returns `true`. The application should invoke `restartSession()` with the `OnApplicationFocus()` call-back function when that call-back function returns `true`. If `suspendSession()` and `restartSession()`are used in the wrong order, they will return failure and do nothing.

- Design the application to maintain the preferred profile ID. The Dolby Game Audio Unity plug-in does not maintain it and uses the movie profile by default. Because the Dolby audio effect on the device is global, it will be used in many games. The `suspendSession()` and `release()` APIs will restore the system-wide configuration when the game is exited.