视沃科技

大牛直播 SDK(V2)

Unity3D 调用说明

官网：http://daniulive.com

Github：https://github.com/daniulive/SmarterStreaming

目 录

1. Android 播放端 SDK 说明

1.1 demo 说明

- SmartU3dAndroidPlayer：大牛直播 SDK Unity3D Android RTMP/RTSP 直播播放端工程。

1.2 功能说明

标准接口：

- 音频：AAC/G.711/speex；
- 视频：H.264；
- 播放协议：RTMP/RTSP；
- 支持 RTSP TCP/UDP 模式切换；
- 支持纯音频、纯视频、音视频播放；
- 支持秒开模式；
- 音视频多种 render 机制；
- 支持 buffer 设置；
- 真正靠谱的超低延迟；
- 支持多实例播放；
- 支持播放 url 快速切换；
- 断网自动重连，支持视频追赶；
- 支持视频 video 实时旋转。

增值接口：

- 同时支持 rtsp、rtmp 播放；
- 播放过程中，实时静音、取消静音；
- 播放端回调 YUV，供 unity3d 调用完成绘制；
- 实时快照；
- 实时录像。

1.3 集成说明

- Unity3D 接口和调用 demo，参见：SmartPlayerAndroidMono.cs
- SmartU3dAndroidPlayer\Assets\Plugins\Android\libs 下相关库到工程：

- Smartavengine.jar 加入到工程；
- smartplayerunity3d.jar 加入工程；
- libs\arm64-v8a 和 SmartPlayer\libs\armeabi 下 libSmartPlayer.so。

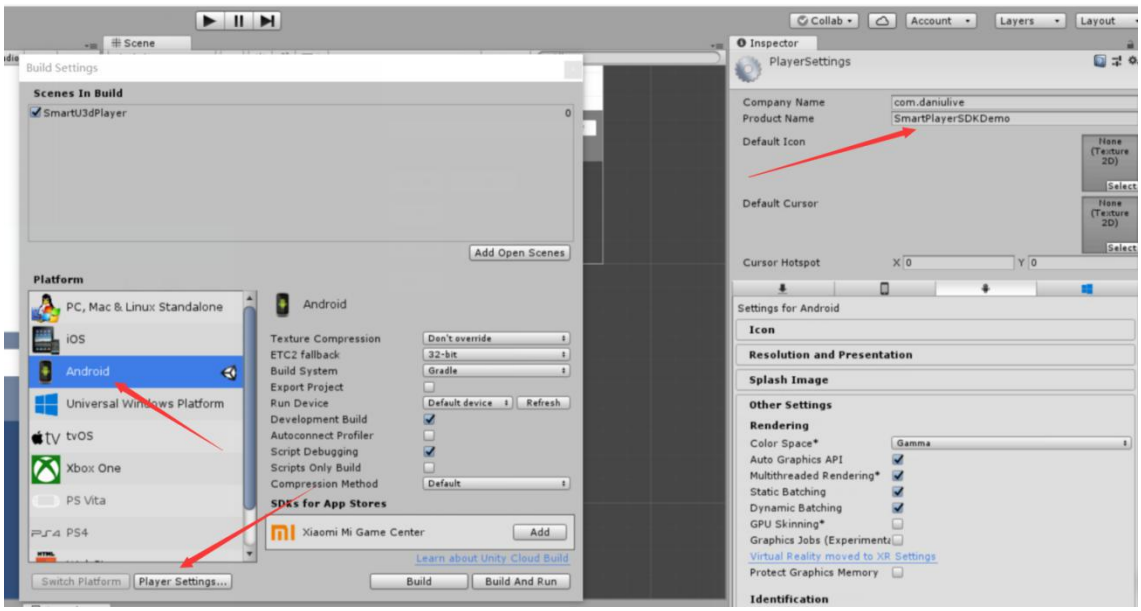
➢ 在 SmartU3dAndroidPlayer\Assets\Plugins\Android\AndroidManifest.xml 配置相关权限：

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" >
</uses-permission>
<uses-permission android:name="android.permission.INTERNET" ></uses-permission>
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS" />
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS" />
```

➢ 如需集成到自己系统测试，请用大牛直播 SDK 的 app name(不然集成提示 license failed)，正式授权版按照授权 app name 正常使用即可：

➢ 如何改 app-name：

Unity3D 模式下：File-->Build Settings-->Android-->Player Settings:



1.4 调用时序(V2)

1. 【最先调用】NT_U3D_Init：player 初始化，目前预留；

2. 【获得 player 句柄】NT_U3D_Open，设置上下文信息，返回 player 句柄；

3. 【设置 GameObject】NT_U3D_Set_Game_Object，注册 Game Object，用于消息传递；

4. 【设置硬解码】NT_U3D_SetVideoDecoderMode，设置是否用硬解码播放，如硬解码不支持，自动适配到软解码；

5. 【audio 输出类型】NT_U3D_SetAudioOutputType()，如果 use_audiotrack 设置为 0，将会自动选择输出设备，如果设置为 1，使用 audiotrack 模式；

6. 【缓冲设置】NT_U3D_SetBuffer，设置播放端缓存数据 buffer，以毫秒(ms)为单位，如超低延迟模式下，不需 buffer 数据，设置为 0；

7. 【RTSP TCP/UDP 设置】NT_U3D_SetRTSPTcpMode，设置 TCP/UDP 播放模式，**注意：此接口仅用于 RTSP；**

8. **【实时静音-可实时调用】**NT_U3D_SetMute，设置播放过程中，实时静音/取消静音；

9. 【快速启动】NT_U3D_SetFastStartup，Set fast startup(快速启动)，设置快速启动后，如果 CDN 缓存 GOP，daniulive player 可快速出帧；

10. 【低延迟模式】NT_U3D_SetPlayerLowLatencyMode，**针对类似于直播娃娃机等期待超低延迟的使用场景，超低延迟播放模式下，延迟甚至可达到 200~400ms；**

11. **【视频显示角度设置-可实时调用】**NT_U3D_SetRotation，**针对类似于安防摄像头或其他设备出来的图像倒置现象，支持视频播放 view 顺时针旋转，当前支持 0 度，90 度, 180 度, 270 度 旋转，注意除了 0 度之外， 其他角度都会额外消耗性能；**

12. 【下载速度回调设置】NT_U3D_SetReportDownloadSpeed，设置下载速度上报，默认不上报下载速度；

13. 【快照设置】NT_U3D_SetSaveImageFlag()，设置是否需要在播放或录像过程中快照；

14. **【快照-录像或播放后，可随时调用】**NT_U3D_SaveCurImage，播放过程中，根据设置路径和文件名，实时快照；

15. **【快速切换 url-可实时调用】**NT_U3D_SwitchPlaybackUrl，快速切换播放 url，快速切换时，只换播放 source 部分，适用于不同数据流之间，快速切换（如娃娃机双摄像头切换或高低分辨率流切换）；

16. 【录像设置】NT_U3D_CreateFileDirectory，创建文件路径；

17. 【录像设置】NT_U3D_SetRecorderDirectory，设置文件路径；

18. 【录像设置】NT_U3D_SetRecorderFileMaxSize，设置每个录像文件最大 size，以兆（M）为单位，范围(5M~500M)；

19. 【设置播放或录像 URL】NT_U3D_SetUrl，设置播放/录像 url；

20. **【播放】**NT_U3D_StartPlay，开始播放；

21. **【播放】**NT_U3D_GetVideoFrame，获取底层回调的 YUV 数据；

22. **【播放】**NT_U3D_StopPlay，停止播放；

23. **【录像】**NT_U3D_StartRecorder，开始录像；

24. **【录像】**NT_U3D_StopRecorder，停止录像；

25. **【关闭】**NT_U3D_Close，关闭播放器实例；

26. **【最后调用】**NT_U3D_UnInit，UnInit Player，最后调用。

1.5 Event 回调

```
/// <summary>
/// android 传递过来 code
/// </summary>
/// <param name="code"></param>
```

```
public void onNTSmartEvent(string param)
{
    if (!param.Contains(","))
    {
        Debug.Log("[onNTSmartEvent] android 传递参数错误");
        return;
    }
   string[] strs = param.Split(',');
   string player_handle =strs[0];
   string code = strs[1];
   string param1 = strs[2];
   string param2 = strs[3];
   string param3 = strs[4];
   string param4 = strs[5];
   Debug.Log("[onNTSmartEvent] code: 0x" + Convert.ToString(Convert.ToInt32(code), 16));
    switch (Convert.ToInt32(code))
    {
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_STARTED:
            Debug.Log("开始。。");
            break;
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_CONNECTING:
            Debug.Log("连接中。。");
            break;
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_CONNECTION_FAILED:
            Debug.Log("连接失败。。");
            break;
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_CONNECTED:
            Debug.Log("连接成功。。");
            break;
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_DISCONNECTED:
            Debug.Log("连接断开。。");
            break;
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_STOP:
            Debug.Log("停止播放。。");
            break;
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_RESOLUTION_INFO:
            Debug.Log("分辨率信息: width: " + Convert.ToInt32(param1) + ", height: " + Convert.ToInt32(param2));
            break;
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_NO_MEDIADATA_RECEIVED:
            Debug.Log("收不到媒体数据，可能是 url 错误。。");
            break;
```

```
            case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_SWITCH_URL:
                Debug.Log("切换播放 URL。。");
                break;

            case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_CAPTURE_IMAGE:
                Debug.Log("快照: " + param1 + " 路径：" + param3);

                if (Convert.ToInt32(param1) == 0)
                {
                    Debug.Log("截取快照成功。.");
                }
                else
                {
                    Debug.Log("截取快照失败。.");
                }
                break;
            case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_RECORDER_START_NEW_FILE:
                Debug.Log("[record]开始一个新的录像文件 : " + param3);
                break;
            case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_ONE_RECORDER_FILE_FINISHED:
                Debug.Log("[record]已生成一个录像文件 : " + param3);
                break;
            case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_START_BUFFERING:
                Debug.Log("Start_Buffering");
                break;
            case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_BUFFERING:
                Debug.Log("Buffering: " + Convert.ToInt32(param1));
                break;
            case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_STOP_BUFFERING:
                Debug.Log("Stop_Buffering");
                break;
            case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_DOWNLOAD_SPEED:
                Debug.Log("download_speed:" + param1 + "Byte/s" + ", "
                        + (Convert.ToInt32(param1) * 8 / 1000) + "kbps" + ", " + (Convert.ToInt32(param1) / 1024)
                        + "KB/s");
                break;
        }
    }
```

2. iOS 播放端 SDK 说明

2.1 demo 说明

- SmartU3diOSPlayer：大牛直播 SDK Unity3D iOS RTMP/RTSP 直播播放端工程。

2.2 功能说明

标准接口：

- 音频：AAC/G.711/speex；
- 视频：H.264；
- 播放协议：RTMP/RTSP；
- 支持 RTSP TCP/UDP 模式切换；
- 支持纯音频、纯视频、音视频播放；
- 支持秒开模式；
- 音视频多种 render 机制；
- 支持 buffer 设置；
- 真正靠谱的超低延迟；
- 支持多实例播放；
- 支持播放 url 快速切换；
- 断网自动重连，支持视频追赶；
- 支持视频 video 实时旋转。

增值接口：

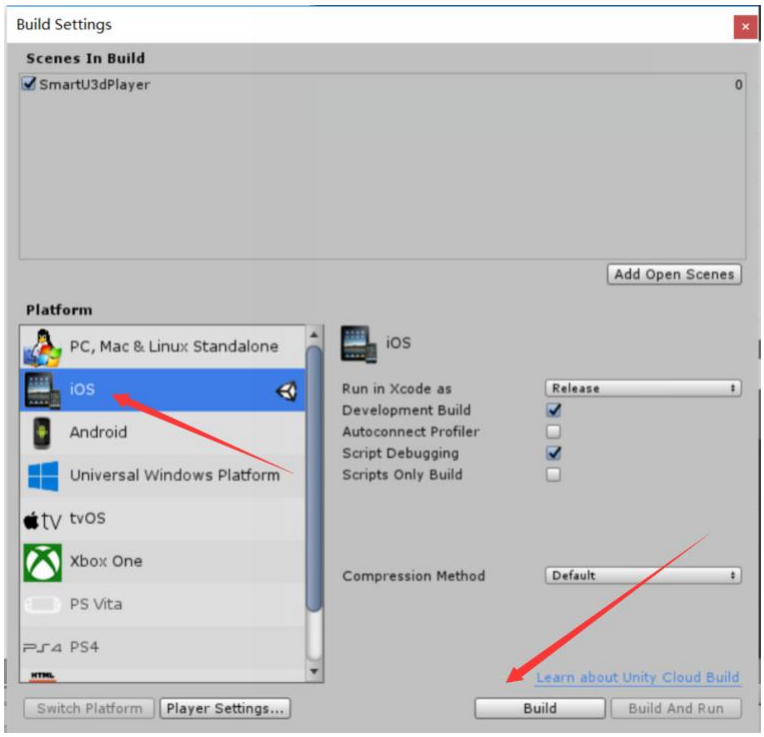
- 同时支持 rtsp、rtmp 播放；
- 播放过程中，实时静音、取消静音；
- 播放端回调 YUV，供 unity3d 调用完成绘制；
- 实时快照；
- 实时录像。

2.3 集成说明

拷贝以下文件，到 Assets-->Plugins-->iOS 目录：

相关头文件和调用说明，参见：SmartPlayeriOSMono.cs

Unity3D 工程下，File-->Build Settings，Platform 选择 iOS，然后点击 build，设置目录，生成 xcode 工程：



生成后的 xcode 工程，添加以下依赖库：

- 相关库：libSmartPlayerSDK.a
- 引入以下依赖 framework
  - libbz.tbd
  - Libbz2.tbd
  - libiconv.tbd
  - libstdc++.tbd
  - Libc++.tbd
  - Accelerate.framework
  - AssetsLibrary.framework
  - AudioToolBox.framework
  - AVFoundation.framework
  - CoreMedia.framework
  - Foundation.framework
  - GLKit.framework
  - OpenGLES.framework
  - UIKit.framework
  - VideoToolBox.framework

- 如需集成到自己系统测试，请用大牛直播的 app name：

  Info.plist-->右键 Open As-->Source Code

  添加或者编辑

  <key>CFBundleName</key>

  <string>SmartiOSPlayer</string>

- 快照添加到“照片”权限：

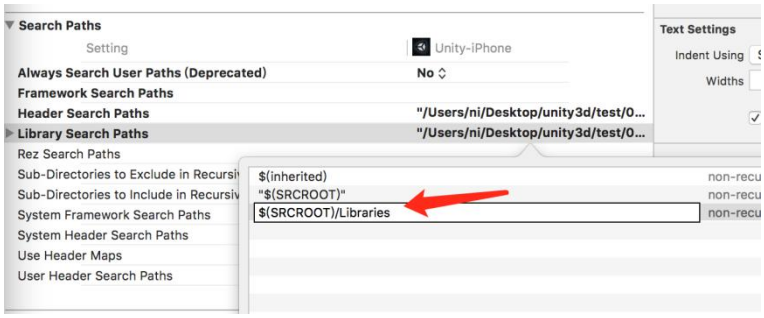
  Info.plist-->右键 Open As-->Source Code

  添加

  <key>NSPhotoLibraryUsageDescription</key>

  <string>1</string>

- 导出后的 xcode 工程，如编译不过，参考以下设置：



2.4 调用时序(V2)

1. **【最先调用】**NT_U3D_Init：player 初始化，目前预留；

2. **【获得 player 句柄】**NT_U3D_Open，设置上下文信息，返回 player 句柄；

3. 【设置 GameObject】NT_U3D_Set_Game_Object，注册 Game Object，用于消息传递；

4. 【设置硬解码】NT_U3D_SetVideoDecoderMode，设置是否用硬解码播放，如硬解码不支持，自动适配到软解码；

5. 【缓冲设置】NT_U3D_SetBuffer，设置播放端缓存数据 buffer，以毫秒(ms)为单位，如超低延迟模式下，不需 buffer 数据，设置为 0；

6. 【RTSP TCP/UDP 设置】NT_U3D_SetRTSPTcpMode，设置 TCP/UDP 播放模式，**注意：此接口仅用于 RTSP；**

7. **【实时静音-可实时调用】**NT_U3D_SetMute，设置播放过程中，实时静音/取消静音；

8. 【快速启动】NT_U3D_SetFastStartup，Set fast startup(快速启动)，设置快速启动后，如果 CDN 缓存 GOP，daniulive player 可快速出帧；

9. 【低延迟模式】NT_U3D_SetPlayerLowLatencyMode，**针对类似于直播娃娃机等期待超低延迟的使用场景，超低延迟播放模式下，延迟甚至可达到 200~400ms；**

10. **【视频显示角度设置-可实时调用】**NT_U3D_SetRotation，**针对类似于安防摄像头或其他设备出来的图像倒置现象，支持视频播放 view 顺时针旋转，当前支持 0 度，90 度, 180 度, 270 度 旋转，注意除了 0 度之外， 其他角度都会额外消耗性能；**

11. 【下载速度回调设置】NT_U3D_SetReportDownloadSpeed，设置下载速度上报，默认不上报下载速度；

12. 【快照设置】NT_U3D_SetSaveImageFlag()，设置是否需要在播放或录像过程中快照；

13. **【快照-录像或播放后，可随时调用】**NT_U3D_SaveCurImage，播放过程中，根据设置路径和文件名，实时快照；

14. **【快速切换 url-可实时调用】**NT_U3D_SwitchPlaybackUrl，快速切换播放 url，快速切换时，只换播放 source 部分，适用于不同数据流之间，快速切换（如娃娃机双摄像头切换或高低分辨率流切换）；

15. 【录像设置】NT_U3D_CreateFileDirectory，创建文件路径，注意：iOS 只提供接口，未提供具体实现；

16. 【录像设置】NT_U3D_SetRecorderDirectory，设置文件路径；

17. 【录像设置】NT_U3D_SetRecorderFileMaxSize，设置每个录像文件最大 size，以兆（M）为单位，范围(5M~500M)；

18. 【设置播放或录像 URL】NT_U3D_SetUrl，设置播放/录像 url；

19. **【播放】**NT_U3D_StartPlay，开始播放；

20. **【播放】**NT_U3D_GetVideoFrame，获取底层回调的 YUV 数据；

21. **【播放】**NT_U3D_StopPlay，停止播放；

22. **【录像】**NT_U3D_StartRecorder，开始录像；

23. **【录像】**NT_U3D_StopRecorder，停止录像；

24. **【关闭】**NT_U3D_Close，关闭播放器实例；

25. **【最后调用】**NT_U3D_UnInit，UnInit Player，最后调用。

2.5 Event 回调

```
 /// <summary>
/// android 传递过来 code
/// </summary>
/// <param name="code"></param>
public void onNTSmartEvent(string param)
{
    if (!param.Contains(","))
    {
        Debug.Log("[onNTSmartEvent] android 传递参数错误");
        return;
    }
   string[] strs = param.Split(',');
   string player_handle =strs[0];
   string code = strs[1];
   string param1 = strs[2];
   string param2 = strs[3];
   string param3 = strs[4];
   string param4 = strs[5];
```

```
Debug.Log("[onNTSmartEvent] code: 0x" + Convert.ToString(Convert.ToInt32(code), 16));
 switch (Convert.ToInt32(code))
 {
     case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_STARTED:
         Debug.Log("开始。。");
         break;
     case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_CONNECTING:
         Debug.Log("连接中。。");
         break;
     case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_CONNECTION_FAILED:
         Debug.Log("连接失败。。");
         break;
     case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_CONNECTED:
         Debug.Log("连接成功。。");
         break;
     case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_DISCONNECTED:
         Debug.Log("连接断开。。");
         break;
     case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_STOP:
         Debug.Log("停止播放。。");
         break;
     case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_RESOLUTION_INFO:
         Debug.Log("分辨率信息: width: " + Convert.ToInt32(param1) + ", height: " + Convert.ToInt32(param2));
         break;
     case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_NO_MEDIADATA_RECEIVED:
         Debug.Log("收不到媒体数据，可能是 url 错误。。");
         break;
     case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_SWITCH_URL:
         Debug.Log("切换播放 URL。。");
         break;

     case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_CAPTURE_IMAGE:
         Debug.Log("快照: " + param1 + " 路径: " + param3);

         if (Convert.ToInt32(param1) == 0)
         {
             Debug.Log("截取快照成功。.");
         }
         else
         {
             Debug.Log("截取快照失败。.");
```

```
            }
            break;
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_RECORDER_START_NEW_FILE:
            Debug.Log("[record]开始一个新的录像文件 : " + param3);
            break;
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_ONE_RECORDER_FILE_FINISHED:
            Debug.Log("[record]已生成一个录像文件 : " + param3);
            break;
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_START_BUFFERING:
            Debug.Log("Start_Buffering");
            break;
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_BUFFERING:
            Debug.Log("Buffering: " + Convert.ToInt32(param1));
            break;
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_STOP_BUFFERING:
            Debug.Log("Stop_Buffering");
            break;
        case EVENTID.EVENT_DANIULIVE_ERC_PLAYER_DOWNLOAD_SPEED:
            Debug.Log("download_speed:" + param1 + "Byte/s" + ", "
                    + (Convert.ToInt32(param1) * 8 / 1000) + "kbps" + ", " + (Convert.ToInt32(param1) / 1024)
                    + "KB/s");
            break;
    }
}
```

3. Windows 播放端 SDK 说明

3.1 demo 说明

➢ SmartU3dWinPlayer：大牛直播 SDK Unity3D Windows RTMP/RTSP 直播播放端工程。Demo 界面如下：

3.2 功能说明

标准接口：

- ✧ 音频：AAC/G.711/speex；
- ✧ 视频：H.264；
- ✧ 播放协议：RTMP/RTSP；
- ✧ 支持 RTSP TCP/UDP 模式切换；
- ✧ 支持纯音频、纯视频、音视频播放；
- ✧ 支持秒开模式；
- ✧ 音视频多种 render 机制；
- ✧ 支持 buffer 设置；
- ✧ 真正靠谱的超低延迟；
- ✧ 支持多实例播放；

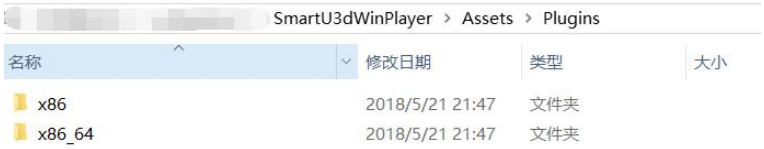
- ✧ 支持播放 url 快速切换；
- ✧ 断网自动重连，支持视频追赶；
- ✧ 支持视频 video 实时旋转、水平反转、垂直反转。

增值接口：

- ✧ 同时支持 rtsp、rtmp 播放；
- ✧ 播放过程中，实时静音、取消静音；
- ✧ 播放端回调 YUV，供 unity3d 调用完成绘制；
- ✧ 实时快照；
- ✧ 实时录像。

3.3 集成说明

- ➢ Unity3D 接口和调用 demo，参见：SmartPlayerWindowsMono.cs
- ➢ SmartU3dAndroidPlayer\Assets\Plugins\x86 和 SmartU3dAndroidPlayer\Assets\Plugins\x86_64 下相关库到工程：



  - ■ nt_base_code_define.cs 加入到工程；
  - ■ smart_player_define.cs 加入工程；
  - ■ smart_player_sdk.cs 加入工程。
- ➢ 如需集成到自己系统测试，请用大牛直播 SDK 的 app name(不然集成提示 license failed)，正式授权版按照授权 app name 正常使用即可，windows 测试 app name：SmartPlayer：

1.4 调用时序(V2)

Windows 调用时序，可参照 Windows C#调用 SDK 调用说明(参见“视沃科技-Windows-SDK 集成说明.pdf”)，此处不再赘述。

调用流程：

1. 开始播放的时候，调用 OpenPlayer()，完成 PlayerInit 和 PlayerOpen 操作，并设置 eventcallback 和 videoframecallback，并完成相关参数配置，设置 YUV 回调输出，调用 PlayerStart，完成播放端的启动：

```
public void Play()
{
    if (is_running)
```

```
{
    Debug.Log("已经在播放。。");
    return;
}
lock (frame_lock_)
{
    cur_video_frame_ = null;
}
//获取输入框的 url
string url = input_url_.text.Trim();

if (!url.StartsWith("rtmp://") && !url.StartsWith("rtsp://"))
{
    videoUrl = "rtmp://live.hkstv.hk.lxdns.com/live/hks";      //url 错误的话，demo 默认播放 hks rtmp 流做演示
}
else
{
    videoUrl = url;
}
OpenPlayer();
if (player_handle_ == IntPtr.Zero)
    return;
//设置播放 URL
NTSmartPlayerSDK.NT_SP_SetURL(player_handle_, videoUrl);
/* ++ 播放前参数配置可加在此处 ++ */
NTSmartPlayerSDK.NT_SP_SetBuffer(player_handle_, play_buffer_time_);        //设置 buffer time
int is_using_tcp = 1;        //TCP 模式
NTSmartPlayerSDK.NT_SP_SetRTSPTcpMode(player_handle_, is_using_tcp);
NTSmartPlayerSDK.NT_SP_SetMute(player_handle_, is_mute_ ? 1 : 0);                    //是否启动播放的时候静音
int is_fast_startup = 1;
NTSmartPlayerSDK.NT_SP_SetFastStartup(player_handle_, is_fast_startup);              //设置快速启动模式

NTSmartPlayerSDK.NT_SP_SetLowLatencyMode(player_handle_, is_low_latency_ ? 1 : 0);    //设置是否启用低延迟模式

//设置旋转角度(设置 0， 90， 180， 270 度有效，其他值无效)
NTSmartPlayerSDK.NT_SP_SetRotation(player_handle_, rotate_degrees);

// 设置上传下载报速度
int is_report = 0;
int report_interval = 1;
NTSmartPlayerSDK.NT_SP_SetReportDownloadSpeed(player_handle_, is_report, report_interval);
```

```
        /* -- 播放前参数配置可加在此处 -- */
        //video frame callback (YUV/RGB)
        video_frame_call_back_ = new SP_SDKVideoFrameCallBack(NT_SP_SetVideoFrameCallBack);
        NTSmartPlayerSDK.NT_SP_SetVideoFrameCallBack(player_handle_,
(Int32)NT.NTSmartPlayerDefine.NT_SP_E_VIDEO_FRAME_FORMAT.NT_SP_E_VIDEO_FRAME_FROMAT_I420, window_handle_, video_frame_call_back_);
        UInt32 flag = NTSmartPlayerSDK.NT_SP_StartPlay(player_handle_);
        if (flag == DANIULIVE_RETURN_OK)
        {
            is_need_get_frame_ = true;
            Debug.Log("播放成功");
        }
        else
        {
            is_need_get_frame_ = false;
            Debug.LogError("播放失败");
        }
        is_running = true;
    }

    private void OpenPlayer()
    {
        window_handle_ = IntPtr.Zero;
        UInt32 isInited = NT.NTSmartPlayerSDK.NT_SP_Init(0, IntPtr.Zero);
        if (isInited != 0)
        {
            Debug.LogError("调用 NT_SP_Init 失败.." + isInited.ToString());
            return;
        }
        if (player_handle_ == IntPtr.Zero)
        {
            player_handle_ = new IntPtr();
            UInt32 ret_open = NTSmartPlayerSDK.NT_SP_Open(out player_handle_, window_handle_, 0, IntPtr.Zero);
            if (ret_open != 0)
            {
                player_handle_ = IntPtr.Zero;
                Debug.LogError("调用 NT_SP_Open 失败..");
                return;
            }
        }
        event_call_back_ = new SP_SDKEventCallBack(NT_SP_SDKEventCallBack);
        NTSmartPlayerSDK.NT_SP_SetEventCallBack(player_handle_, window_handle_, event_call_back_);
```

```
        sdk_video_frame_call_back_ = new SetVideoFrameCallBack(SDKVideoFrameCallBack);
        sdk_event_call_back_ = new SetEventCallBack(SDKEventCallBack);
    }
```

2. Video frame 实时处理并绘制：

开始播放后，daniulive 直播播放端 SDK 回调 yuv 数据及相关信息，unity3d 获取到数据信息后，调用 InitYUVTexture()，完成初始化工作，调用 UpdateYUVTexture()实现数据实时刷新，当数据信息发生变化时，会二次调用 InitYUVTexture()，完成初始化。

3. 停止播放：

```
public void Close()
{
   ClosePlayer();
 }
  private void ClosePlayer()
  {
      is_need_get_frame_ = false;
      is_need_init_texture_ = false;
      if (player_handle_ == IntPtr.Zero)
      {
          return;
      }
      UInt32 flag = NTSmartPlayerSDK.NT_SP_StopPlay(player_handle_);
      if (flag == DANIULIVE_RETURN_OK)
       {
           Debug.Log("停止成功");
       }
       else
       {
           Debug.LogError("停止失败");
       }
       NTSmartPlayerSDK.NT_SP_UnInit();
       player_handle_ = IntPtr.Zero;
       is_running = false;
}
```

3.5 Event 回调

```
 private void SDKEventCallBack(UInt32 event_id,
       Int64 param1,
```

```
        Int64 param2,
        UInt64 param3,
        [MarshalAs(UnmanagedType.LPStr)] String param4,
        [MarshalAs(UnmanagedType.LPStr)] String param5,
        IntPtr param6)
    {
        if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_PLAYBACK_REACH_EOS == event_id)
        {
            //本地 flv 文件播放之用
            return;
        }
        else if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_RECORDER_REACH_EOS == event_id)
        {
            NTSmartPlayerSDK.NT_SP_StopRecorder(player_handle_);
            return;
        }
        else if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_PULLSTREAM_REACH_EOS == event_id)
        {
            if (player_handle_ != IntPtr.Zero)
            {
                NTSmartPlayerSDK.NT_SP_StopPullStream(player_handle_);
            }
            return;
        }
        if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_CONNECTING == event_id
            || (UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_CONNECTION_FAILED == event_id
            || (UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_CONNECTED == event_id
            || (UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_DISCONNECTED == event_id)
        {
            connection_status_ = event_id;
        }
        if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_START_BUFFERING == event_id
            || (UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_BUFFERING == event_id
            || (UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_STOP_BUFFERING == event_id)
        {
            buffer_status_ = event_id;
            if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_BUFFERING == event_id)
            {
                buffer_percent_ = (Int32)param1;
            }
        }
```

```
if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_DOWNLOAD_SPEED == event_id)
{
    download_speed_ = (Int32)param1;
}

String t_show_str = "";
if (connection_status_ != 0)
{
    t_show_str += "链接状态: ";
    if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_CONNECTING == connection_status_)
    {
        t_show_str += "链接中";
    }
    else if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_CONNECTION_FAILED == connection_status_)
    {
        t_show_str += "链接失败";
    }
    else if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_CONNECTED == connection_status_)
    {
        t_show_str += "链接成功";
    }
    else if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_DISCONNECTED == connection_status_)
    {
        t_show_str += "链接断开";
    }
}

if (download_speed_ != -1)
{
    String ss = "  下载速度: " + (download_speed_ * 8 / 1000).ToString() + "kbps " + (download_speed_ / 1024).ToString() + "KB/s";
    t_show_str += ss;
}

if (buffer_status_ != 0)
{
    t_show_str += "  缓冲状态: ";
    if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_START_BUFFERING == buffer_status_)
    {
        t_show_str += "开始缓冲";
    }
    else if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_BUFFERING == buffer_status_)
```

```
            {
                String ss = "缓冲中 " + buffer_percent_.ToString() + "%";
                t_show_str += ss;
            }
            else if ((UInt32)NTSmartPlayerDefine.NT_SP_E_EVENT_ID.NT_SP_E_EVENT_ID_STOP_BUFFERING == buffer_status_)
            {
                t_show_str += "结束缓冲";
            }
        }
        Debug.Log(t_show_str);
    }
```

4 商务合作

公司：上海视沃信息科技有限公司
地址：上海市浦东新区张江高科技园区碧波路 635 号 传奇广场 202-2F 13-15 室

手机：130-7210-2209 或 135-6452-9354
QQ ：89030985 或 2679481035

官网：http://www.daniulive.com
Github 地址：https://github.com/daniulive/SmarterStreaming

QQ 群 1：499687479(即将满员)
QQ 群 2：294891451(推荐加入)