Spout Dll

A library for texture sharing between applications.

spout.zeal.co

Reference Manual

Version 1.01

June 2014

1. Revisions

Version 1.00 - May 2014

- Initial release.

Version 1.01 - June 2014

- New functions CloseSpout, GetSenderCount, GetSenderNameInfo.
- Documented restoration of user fbo binding for texture sharing functions.
- InitReceiver returns false if no senders running.

Programmer notes

Each Spout receiver or sender is a a single instance. Multiple receivers or senders are not supported. Also a receiver and sender in the same application is not supported.

InitReceiver / ReleaseReceiver or InitSender / ReleaseSender can be called multiple times within the application if necessary. However, it is important to call CloseSpout at program termination.

InitReceiver returns false immediately if no senders are running. The overhead of this call is very low and the function can be called repeatedly until a sender is started.

GetSenderNames is only compatible with Visual Studio C++ compilers. The alternative method is to enumerate senders with GetSenderCount and GetSenderNameInfo.

Texture sharing functions use an fbo internally. If an application calls these functions with an fbo bound, the binding will be lost and has to be restored after the call.

2. Introduction

The Spout developer dll is a C++ dll that allows programmers to embed Spout texture sharing functions into their own application.

It is provided as both 32 bit and 64 bit versions.

An OpenGL context must be established before any of these functions can be called.

3. Using the dll

The process is simple :

For a sender.

1) Setup - Initialize a sender (InitSender)
2) Draw - Send a texture (SendTexture)
3) Close - Release the sender (ReleaseSender)
4) Exit – Close Spout on program termination (CloseSpout)

For a receiver.

1) Setup - Initialize a receiver (InitReceiver)
2) Draw - receive a texture (ReceiveTexture) and draw it
3) Close - Release the receiver (ReleaseReceiver)
4) Exit – Close Spout on program termination (CloseSpout)

3.1 Receiver functions

There are additional functions required for a receiver.

When intializing

Initialize with the sender name that the receiver should connect to. That sender has to be running for connection to succeed.

For success

Check the texture compatibility flag returned. If not texture share compatible it will have initialized in memoryshare mode, so tests for sender name or selection of multiple senders are not relevant.

Check the sender name returned. If it is different, the requested sender was not detected and connection was made to the current "active" sender. i.e. the first one started or the last selected by the user.

Check the width and height returned. Adjust the receiving texture dimensions to this size.

For failure

No further functions are possible, so release the sender.

When receiving a texture

For success just draw the received texture.

For failure

1) Width and height are returned zero for texture read failure. This should not happen but if it does, no further functions are possible.

2) New width and height are returned if the sender has changed size or a different sender has been selected by the user. The local texture must then be resized.

3.2 User sender selection

SelectSenderDialog

Activates a modal dialog with a list of names for the currently running senders and allows the user to choose one, or cancel.

Once the user has selected a new sender, ReceiveTetxure will return false, but with the new sender name, width and height, so that the local texture can be adjusted.

NOTE: not compatible with some host applications.

SelectSenderPanel

Activates an executable program "SpoutPanel.exe" that provides the same functionality as the modal dialog. Since this is a separate executable program it has the advantage that it has no effect on the dll or the host application. However, SpoutPanel.exe must be in the same folder as the host.

3.3 Utility functions

TextureShareCompatible

Returns true if the hardware supports texture sharing. It is not necessary to use this before the initialization functions because these also return a texture share compatibility flag. However, it is useful to determine compatibility at the outset without having to initialize a sender or receiver.

GetSenderNames

Returns a set of name strings. This is useful if you want to create your own sender list for user selection.

NOTE: only compatible with Visual Studio C++.

To use it, establish a string set and pass a pointer to the set to this function. The set is filled with the current list of sender names. Note that each name is a string, but Spout uses a 256 character array.

For example :

```
std::set<string> senders; // empty set to receive the name strings
std::set<string>::iterator iter; // iterator for the set
string namestring; // Name string to iterate through the set
char name[256]; // Spout name character array

if(GetSenderNames(&senders)) {
    // list the names
    if(senders.size() > 0) {
        for(iter = senders.begin(); iter != senders.end(); iter++) {
            namestring = *iter; // the sender name string
            strcpy_s(name, 256, namestring.c_str()); // the character name
        }
    }
}
```

GetSenderInfo

Returns the width, height and DirectX shared texture handle of a sender.

GetSenderCount

Returns the number of Spout senders.

GetSenderNameInfo

Returns the name, width, height and texture share handle of a sender in the list of senders.

4. Spout library files

Spout.h	*header file*
Spout32.dll	*32 bit dll*
Spout32.lib	*32 bit import library*
Spout64.dll	*64 bit dll*
Spout64.lib	*64 bit import library*

Declared in your include file :

```
#include "Spout.h"
using namespace Spout;
```

5. Functions

InitSender

```
bool InitSender(char *name,
                unsigned int width,
                unsigned int height,
                bool& bTextureShare,
                bool bMemoryShare = false);
```

Initialise a Spout sender.

name
Character array containing the name of the sender to Initialise. Minimum dimension 256 characters.

width
Width of texture to be sent.

height
Height of texture to be sent.

bTextureShare
Returns texture share compatible (true) or not (false).

bMemoryShare
Optional - set to true to force memoryshare mode.

Returns true for success or false for initialisation failure.

InitReceiver

```
bool InitReceiver (char *name,
                   unsigned int& width,
                   unsigned int& height,
                   bool& bTextureShare,
                   bool bMemoryShare = false);
```

Initialise a Spout receiver.

name
Character array containing the name of the sender to attempt connection to. Minimum dimension 256 characters.

width
Returns width of sender texture.

height
Returns height of sender texture.

bTextureShare
Returns texture share compatible (true) or not (false).

bMemoryShare
Optional - set to true to force memoryshare mode.

Returns true for success
  Sender name is returned different if the active sender has been used.
Returns false if no senders are running or for initialisation failure.

ReleaseSender

```
bool ReleaseSender();
```

Release a Spout sender.

ReleaseReceiver

```
bool ReleaseReceiver();
```

Release a Spout receiver.

CloseSpout

```
bool CloseSpout();
```

Release all Spout resources on program termination.

SendTexture

```
bool SendTexture(GLuint TextureID,
                 GLuint TextureTarget,
                 unsigned int width,
                 unsigned int height,
                 bool bInvert=true);
```

Send a texture. A sender must have been initialised.

bInvert
Optional - DirectX and OpenGL texture Y coordinates are reversed, therefore a received shared OpenGL texture has to be flipped vertically. This is the default. To bypass this inversion, set the bInvert flag to false.

NOTE :
If a host calls SendTexture with a framebuffer object actively bound, it must restore that binding after the call, because the texture transfer makes use of an FBO for intermediate processing. The same applies to ReceiveTexture.

ReceiveTexture

```
bool ReceiveTexture(GLuint TextureID,
                    GLuint TextureTarget,
                    unsigned int &width,
                    unsigned int &height);
```

Receive a texture. A receiver must have been initialised.

width, height - dimensions of the sender texture.
These variables are changed by Spout if the sender dimensions change.
The caller is responsible for changing the receiving texture size.

Returns :

true - all OK.

false -

1) width and height are zero for texture read failure.
2) width and height are returned changed changed for sender change
The local texture then has to be resized.

TextureShareCompatible

```
bool TextureShareCompatible();
```

Returns texture share compatibility (true) or not (false).

Optional because InitSender and InitReceiver both return a flag indicating whether texture share is compatible or not.

Useful to determine compatibility at the outset.

SelectSenderDialog

```
bool SelectSenderDialog();
```

Used by a receiver to activate a modal dialog for selecting a sender.
NOTE: not compatible with some host applications, use SelectSenderPanel instead.

SelectSenderPanel

```
bool SelectSenderPanel();
```

Used by a receiver to activate an executable program “SpoutPanel.exe” for selecting a sender. SpoutPanel.exe must be in the folder of the host program.

GetSenderNames

```
bool GetSenderNames(std::set<std::string> *sendernames);
```

Returns a set of sender names.
NOTE: only compatible with Visual Studio C++.

GetSenderInfo

```
bool GetSenderInfo(char *name,
                   unsigned int &width,
                   unsigned int &height,
                   HANDLE &dxTextureHandle);
```

Returns the width, height and DirectX shared texture handle of a sender.

GetSenderCount

```
int GetSenderCount();
```

Returns the number of Spout senders running. Use this in conjunction with GetSenderNameInfo below to retrieve the details of a sender in the list.

GetSenderNameInfo

```
bool GetSenderNameInfo(int index,
                       char* sendername,
                       int size,
                       unsigned int &width,
                       unsigned int &height,
                       HANDLE &dxSharehandle);
```

Returns the name, width, height and texture share handle of a sender.

Index
the number of the sender is the list

sendername
the sender name returned

size
the maximum number of bytes of the name to be returned

width
the width of the sender texture

height
the height of the sender texture

dxSharehandle
the share handle or the sender shared directX texture.