3DRUDDER SDK

9/14/2016

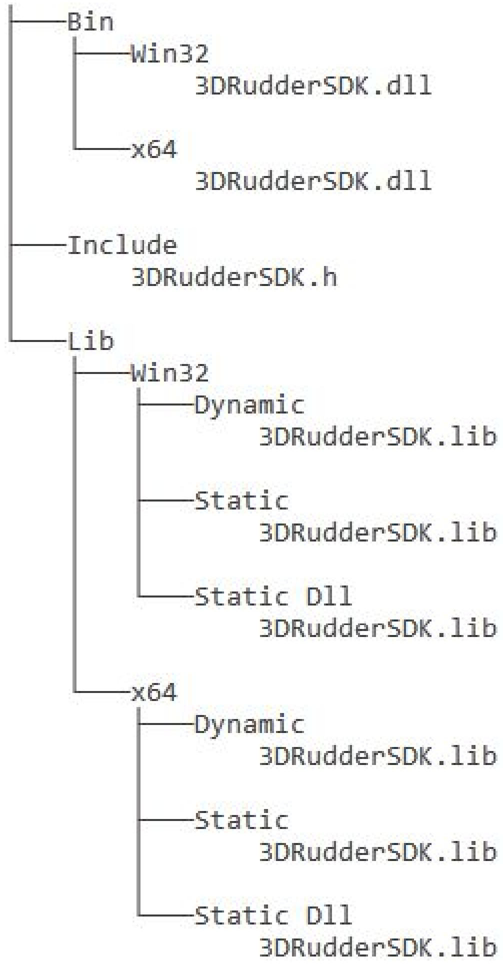
Version 0.4 for Windows

This is the pre-release of the 3DRudder SDK

3DRudder SDK

ER ...)

SDK Organization



```
├───Bin
│   ├───Win32
│   │       3DRudderSDK.dll
│   │
│   └───x64
│           3DRudderSDK.dll
│
├───Include
│       3DRudderSDK.h
│
└───Lib
    ├───Win32
    │   ├───Dynamic
    │   │       3DRudderSDK.lib
    │   │
    │   ├───Static
    │   │       3DRudderSDK.lib
    │   │
    │   └───Static Dll
    │           3DRudderSDK.lib
    │
    └───x64
        ├───Dynamic
        │       3DRudderSDK.lib
        │
        ├───Static
        │       3DRudderSDK.lib
        │
        └───Static Dll
                3DRudderSDK.lib
```

Type of library available

- With this release of the SDK we provide the static and dynamic libraries in 32 and 64 bits.
- We only provide the multi-threaded version.
- The “Static Dll” are a static library with the DLL CRT compilation (/MD).
- Visual Studio 2015 had been used to compile those libraries.

Static library usage

To use the static libraries you need to define `_3DRUDDER_SDK_STATIC` to avoid `dllimport`

SDK Usage

Include the SDK definition

```
#include "3DRudderSDK.h"
```

3DRudder NameSpace

The SDK use the namespace `ns3DRudder`

Get the SDK Class pointer

```
ns3DRudder::CSdk* pSdk=ns3DRudder::GetSDK();
```

SDK Reference

All the SDK is defined in the class `ns3DRudder::CSdk.` With this SDK it's possible to manage up to four 3DRudder `_3DRUDDER_SDK_MAX_DEVICE` define the port number.

Get the sdk version

```
unsigned short GetSDKVersion()
```

Return the SDK version of the library, it's possible to compare this version with the `_3DRUDDER_SDK_VERSION` define included in the 3DRudderSDK.h to compare if the library and the .h match. The version is a fixed point unsigned short in hexadecimal: 0x0040 mean version 0.4.

Get the number of connected 3DRudder

```
int GetNumberOfConnectedDevice()
```

Return the number of 3DRudder currently connected to the computer.

Check if a 3DRudder is connected to the port

```
bool IsDeviceConnected(int nPortNumber)
```

Return true if a 3DRudder is connected to the `nPortNumber` port.

Get the Firmware version of a 3DRudder

```
unsigned short GetFirmwareVersion(int nPortNumber)
```

Return version number of the firmware of the 3DRudder connected to the `nPortNumber` port. The version is a fixed point unsigned short in hexadecimal: 0x1152 mean version 1.1.5.2

Return 0xFFFF in case of error.

Play a sound on a 3DRudder

```
ErrorCode PlaySnd(int nPortNumber, short int nFrequency, short int nDuration)
```

It's possible to play a sound on a 3DRudder connected to the `nPortNumber` port.

`nFrequency` define the frequency of the sound in Hz (440 is a A).

`nDuration` define the duration of the sound in ms.

`ErrorCode` is the possible error code returned by this method.

Get the 3DRudder State

```
ErrorCode Get3DRudderState(int nPortNumber, State* pState)
```

This method fill the structure `State` pointed by `pState` for the 3DRudder connecter to the `nPortNumber` port.

`ErrorCode` is the possible error code returned by this method.

The structure `State`:

```
class State
{
public:
        int aX;
        int aY;
        int aZ;
        int rZ;
        Status status;
        unsigned short s1, s2, s3, s4, s5, s6;
};
```

aX is the X Axis (you can use `GetXAxis`() to access to this value)
aY is the Y Axis (you can use `GetYAxis`() to access to this value)
aZ is the Z Axis (you can use `GetZAxis`() to access to this value)
rZ is the Z Rotation (you can use `GetZRotation`() to access to this value)
s1 to **s6** are the the six sensor value (you can use `GetSensor`(int nIndex) to access to this value)
status give the current status of the 3DRudder. (you can use `GetStatus`() to access to this value)
This status could have the values :

`NoFootStayStill`:

Puts the 3DRudder on the floor, curved side below, without putting your feet on the device. The user waits for 2 seconds for the 3DRudder to boot up until 3 short beeps are heard.

`Initialisation`:

The 3DRudder initialize for about 2 seconds. Once done a long beep will be heard from the device. The 3DRudder is then operational.

`PutYourFeet`:

Put your first feet on the 3DRudder.

`PutSecondFoot`:

Put your second Foot on the 3DRudder.

`StayStill`:

The user must wait still for half a second for calibration until a last short beep is heard from the device. The 3DRudder is ready to be used.

`InUse`:

The 3DRudder is in use.

`ExtendedMode:`

The 3DRudder is in use and is fully operational with all the features enabled.

Error Code

`ns3DRudder::CSdk::ErrorCode` define the error code used by the SDK:

`Success:` No error
`NotConnected:` The 3DRudder is not connected.
`Fail:` Fail to execute the method.
`IncorrectCommand:` Incorrect internal command.
`Timeout:` Communication with the 3DRudder timeout.
`WrongSignature:` Wrong signature of the version of the Firmware.
`NotReady:` The data you try to read is not ready.