Using the G-key SDK with C#

Contents

Overview

The Logitech Gaming G-key SDK enables to get the current state of G-keys and extra mouse buttons for supported Logitech gaming mice and keyboards. It's built as a C++ DLL, but it can be easily integrated in a C# assembly, using P/Invoke and function marshaling. It only works when the Logitech Gaming Software is running (8.30 or later).

Please refer to the Logitech SDK's Doc\LogitechGamingGkeySDK.pdf for details on the SDK's functionality.

Making the G-key SDK work in your C# program

The following steps show how to make the Logitech SDK work with a C# program. Please adapt the steps to your game for things to work.

Steps

1. Create a SDK C# wrapper class as follows :

LogitechGSDK.cs

```
using System.Collections;
using System.Runtime.InteropServices;
using System.Collections.Specialized;
using System;

public class LogitechGSDK  {
	//G-KEY SDK

	public const int LOGITECH_MAX_MOUSE_BUTTONS = 20;
	public const int LOGITECH_MAX_GKEYS = 29;
	public const int LOGITECH_MAX_M_STATES = 3;


	[StructLayout(LayoutKind.Sequential, Pack=2)]
	public struct GkeyCode
	{
		public ushort complete;
		// index of the G key or mouse button, for example, 6 for G6 or Button 6
	public int keyIdx{
			get{
				return complete & 255;
			}
		}
		// key up or down, 1 is down, 0 is up
		public int keyDown{
			get{
				return (complete >> 8) & 1;
			}
		}
		// mState (1, 2 or 3 for M1, M2 and M3)
		public int mState{
			get{
				return (complete >> 9) & 3;
			}
		}
```

```
            // indicate if the Event comes from a mouse, 1 is yes, 0 is no.
            public int mouse{
                  get{
                        return (complete >> 11) & 15;
                  }
            }
            // reserved1
            public int reserved1{
                  get{
                        return (complete >> 15) & 1;
                  }
            }
       // reserved2
            public int reserved2{
                  get{
                        return (complete >> 16) & 131071;
                  }
            }
      }

      [UnmanagedFunctionPointer(CallingConvention.Cdecl)]
   public delegate void logiGkeyCB(GkeyCode gkeyCode,
[MarshalAs(UnmanagedType.LPWStr)]String gkeyOrButtonString, IntPtr context); // ??

      [DllImport("LogitechGkeyEnginesWrapper ", CharSet = CharSet.Unicode,
CallingConvention = CallingConvention.Cdecl)]
      public static extern int LogiGkeyInitWithoutCallback();

      [DllImport("LogitechGkeyEnginesWrapper ", CharSet = CharSet.Unicode,
CallingConvention = CallingConvention.Cdecl)]
      public static extern int LogiGkeyInitWithoutContext(logiGkeyCB gkeyCB);

      [DllImport("LogitechGkeyEnginesWrapper ", CharSet = CharSet.Unicode,
CallingConvention = CallingConvention.Cdecl)]
      public static extern int LogiGkeyIsMouseButtonPressed(int buttonNumber);

      [DllImport("LogitechGkeyEnginesWrapper ", CharSet = CharSet.Unicode,
CallingConvention = CallingConvention.Cdecl)]
      public static extern IntPtr LogiGkeyGetMouseButtonString(int buttonNumber);

      public static String LogiGkeyGetMouseButtonStr(int buttonNumber){
            String str =
Marshal.PtrToStringUni(LogiGkeyGetMouseButtonString(buttonNumber));
            return str;
      }

      [DllImport("LogitechGkeyEnginesWrapper ", CharSet = CharSet.Unicode,
CallingConvention = CallingConvention.Cdecl)]
      public static extern int LogiGkeyIsKeyboardGkeyPressed(int gkeyNumber, int
modeNumber);

      [DllImport("LogitechGkeyEnginesWrapper ")]
      private static extern IntPtr  LogiGkeyGetKeyboardGkeyString(int gkeyNumber, int
modeNumber);

      public static String LogiGkeyGetKeyboardGkeyStr(int gkeyNumber, int modeNumber){
            String str =
Marshal.PtrToStringUni(LogiGkeyGetKeyboardGkeyString(gkeyNumber, modeNumber));
```

```
                return str;
        }

        [DllImport("LogitechGkeyEnginesWrapper ", CharSet = CharSet.Unicode,
CallingConvention = CallingConvention.Cdecl)]
        public static extern void LogiGkeyShutdown();
}
```

2. Call the functions from the wrapper from your C# code as follows:

```
// Use this for initialization
void Start () {
//Value used to show the two different ways to implement g-keys support in your game
                //change it to false to try the non-callback version
        usingCallback = true; //or false, depending on your implementation
        if (usingCallback){
                LogitechGSDK.logiGkeyCB cbInstance = new
LogitechGSDK.logiGkeyCB(this.GkeySDKCallback);
                LogitechGSDK.LogiGkeyInitWithoutContext(cbInstance);
        }
        else
                LogitechGSDK.LogiGkeyInitWithoutCallback();
}

// Update is called once per frame
void Update(){
  if(!usingCallback){
     for (int index = 6; index <= LogitechGSDK.LOGITECH_MAX_MOUSE_BUTTONS; index++) {
                if (LogitechGSDK.LogiGkeyIsMouseButtonPressed(index) == 1) {
                        // Code to handle what happens on gkey pressed on mouse
                }
        }

     for (int index = 1; index <= LogitechGSDK.LOGITECH_MAX_GKEYS; index++) {
         for (int mKeyIndex = 1; mKeyIndex <= LogitechGSDK.LOGITECH_MAX_M_STATES;
mKeyIndex++) {
            if (LogitechGSDK.LogiGkeyIsKeyboardGkeyPressed(index, mKeyIndex) == 1) {
                        // Code to handle what happens on gkey pressed on keyboard/headset
            }
          }
     }
   }
}

void GkeySDKCallback(LogitechGSDK.GkeyCode gKeyCode, String gKeyOrButtonString, IntPtr
context){
        if(gKeyCode.keyDown == 0){
                if(gKeyCode.mouse == 1){
                        // Code to handle what happens on gkey released on mouse
        }
                else{
                        // Code to handle what happens on gkey released on keyboard/headset
                }
```

```
	}
	else{
		if(gKeyCode.mouse == 1){
			// Code to handle what happens on gkey pressed on mouse
		}
		else{
			// Code to handle what happens on gkey pressed on keyboard
		}

	}

	void OnDestroy () {
		//Free G-Keys SDKs before quitting the game
	LogitechGSDK.LogiGkeyShutdown();
	}
```

3. Copy Logitech SDK's Lib\GameEnginesWrapper\x86\LogitechGkeyEnginesWrapper.dll to your c# 32bit executable path
4. Copy Logitech SDK's Lib\GameEnginesWrapper\x64\LogitechGkeyEnginesWrapper.dll to your c# 64bit executable path
5. Compile and run your program

For questions/comments, email devtechsupport@logitech.com