Reading joysticks under Windows

Note : The information in this application note is the result of our understanding of the use of the mmsystem library.
It is not an absolute truth and does not commit us in any way!

The joystick reading under Windows uses a standard dll :

- 'winmm.dll' on recent versions
- 'mmsystem.dll' on older versions (XP).

Most useful functions :

The list is not exhaustive.

- JoySetCapture : Capture of a joystick
- JoyReleaseCapture : Release a joystick
- joyGetDevCaps : Return the characteristics of the joystick
- joyGetNumDevs : Number of connected joysticks
- joyGetPos : Returns the positions of 3 axes : X, Y, Z and buttons
- joyGetPosEx : Returns the positions of 7 axes : X, Y, Z, R, U, V, POV and buttons
- the other functions are to be discovered on the site of Microsoft

For all the details, just do a search on the internet with one of these functions..

Delphi ou Lazarus

Under Delphi or Lazarus, you have to include the MMSYSTEM unit..

Langage C

The procedures to query any joystick under windows are described on the microsoft site :

https://learn.microsoft.com/en-us/windows/win32/api/joystickapi/

Number of joysticks

Up to 16 joysticks can be used at the same time.
The number of connected joysticks is obtained by calling the function
UINT joyGetNumDevs();

Windows 10 / XP	Valid values for uJoyID range from 0 (JOYSTICKID1) to 15
Windows NT 4.0	Valid values are limited to JOYSTICKID1 and JOYSTICKID2.

Warning : the dll does not differentiate Joystick interfaces from HID interfaces. A composite device could use several identifiers.

Steps to follow

- **Find the joystick ID** : By default, Windows does not give any information about which joystick corresponds to which ID. This is not a problem with a single joystick, which is the case most of the time. So uJoyID = 0.
  But if we have several joysticks, we don't know in which order they are assigned or if this order will be permanent !
  See our example in Lazarus :
  Find my joystick among other joysticks
- **Capture the joystick** : JoySetCapture
- **Acquire moves and buttons** :
  Loop or use a timer to acquire the joystick data : joyGetPos or joyGetPosEx
- **Release the joystick** : JoyReleaseCapture

Find my joystick among other joysticks

We can make a query of all the joysticks present with the **joyGetDevCaps** function to find a joystick

- with a particular **VID** and **PID**
- but also with a particular number of **axes** or **buttons**, which also allows to eliminate HID interfaces

With this method, we can't distinguish 2 identical joysticks !

```
program TryMyJoystick;
// Example in Lazarus free pascal

uses MMSYSTEM, SysUtils, Crt;
var uJoyID:Integer;

function FindMyJoystick(MyVID, MyPID: word; const MyAxes: integer = -1;
  const MyButtons: integer = -1): integer;
var
  c,JoyVID, JOYPID, joyNumber,JoyAxes, JoyButtons: word;
  uJoyID: integer;
  InfosCaps : TJOYCAPS ;
begin
  // Get Number of connected joystick
  JoyNumber := joyGetNumDevs;
  // Initialize uJoyID
  uJoyID := -1;

  // check every joystick (remember Joystick #1's ID = 0, etc.)
  c := 0;
  while (c < JoyNumber - 1) do
  begin
    // try to read information from joyGetDevCaps
    if joyGetDevCaps(c, @InfosCaps, SizeOf(InfosCaps)) = JOYERR_NOERROR then
    begin
      JoyVID := InfosCaps.wMid; // Get Vendor ID
      JoyPID := InfosCaps.wPid; // Get Product ID
      JoyAxes := InfosCaps.wNumAxes; // Get Axes Quantity
      JoyButtons := InfosCaps.wNumButtons; // Get Buttons Quantity
      // is it my Joystick ?
      // Check if VID and PID are identical
      if (JoyVID = MyVID) and (JoyPID = MyPID)
        // Optionnally check number of Axes is identical
        and ((MyAxes < 0) or (JoyAxes = MyAxes))
        // Optionnally check number of Buttons is identical
        and ((MyButtons < 0) or (JoyButtons = MyButtons))
      then uJoyID := c;  // Select This Joystick
    end;
    Inc(c); // Try Next Joystick
  end;

  Result := uJoyID;
end;

begin
  uJoyID:=FindMyJoystick($25c7,$0119,3,12); // VID , PID, 3 Axes, 12 Buttons
  writeln (format('Joystick #%d',[uJoyID]));
  Readln;
end.
```

Acquisition of a joystick

```
program TryMyJoystick;
// Example in Lazarus free pascal

uses MMSYSTEM, SysUtils, Crt;

var uJoyID:Integer;
  MyJoy: TJoyInfoEx;
  JoyHandle: THandle;

function FindMyJoystick(MyVID, MyPID: word; const MyAxes: integer = -1;
...

procedure ReadJoystick(uJoyID: integer);
  var
    Buttons: integer;
    AxeX, AxeY, AxeZ: integer;
    ErrorResult: MMResult;
  begin
    if (uJoyID <> -1)
    then ErrorResult := joyGetPosEx(uJoyID, @MyJoy)
    else ErrorResult := 255;

    if ErrorResult = JOYERR_NOERROR then
    begin
      AxeX := MyJoy.wxpos; // Read X position
      AxeY := MyJoy.wypos; // Read Y Position
      AxeZ := MyJoy.wzpos; // Read Z Position
      Buttons := MyJoy.wButtons; // Read Buttons
      writeln(format('X=%d, Y=%d, Z=%d, Buttons:%d ', [AxeX, AxeY, AxeZ, Buttons]));
    end;
  end;

  procedure CaptureJoystick(uJoyID: integer);
  begin
    case JoySetCapture(JoyHandle, uJoyID, 0, False) of
      JOYERR_NOERROR:    Writeln('JOYERR_NOERROR  ');
      MMSYSERR_NODRIVER: Writeln('MMSYSERR_NODRIVER');
      JOYERR_PARMS:      Writeln('JOYERR_PARMS    ');
      JOYERR_NOCANDO:    Writeln('JOYERR_NOCANDO  ');
      JOYERR_UNPLUGGED:  Writeln('JOYERR_UNPLUGGED');
      else
        Writeln('???');
    end;
  end;

begin
  uJoyID := FindMyJoystick($25c7, $0119, 3, 12); // VID , PID, 3 Axes, 12 Buttons
  writeln(format('Joystick #%d', [uJoyID]));
  if uJoyID >= 0 then
  begin
    MyJoy.dwSize := 65535;
    MyJoy.dwFlags := JOY_RETURNX OR JOY_RETURNY OR JOY_RETURNZ OR JOY_RETURNBUTTONS;
    // MyJoy.dwFlags := JOY_RETURNALL;
    CaptureJoystick(uJoyID);
    repeat
      ReadJoystick(uJoyID);
      delay(50); // pause 50 ms
    until keypressed;
    JoyReleaseCapture(uJoyID);
  end;
end.
```