Course 3D_XNA: 3D-Computer Graphics with XNA
Chapter C4: Wii Balance Board

The Wii Balance Board

Brian Peek wrote an excellent managed library for **Nintendos Wii controller "Wiimote"**: `WiimoteLib.dll`, which allows to connect a Wiimote or a balance board to C# applications. The main remaining problem is to get connected via **Bluetooth** because Bluetooth is a shaky technology that sucks.

How to connect the balance board to your Windows 7 PC:
Plug a Bluetooth adapter into an USB-Port:
Open the battery compartment of the balance board. Insert 4 very good AA batteries. The battery compartment contains a tiny push SYNC nub.

Method 1 = temporary Bluetooth pairing method:
Drawback: This method must be repeated whenever the PC (e.g. after sleep mode) or the balance board (f. i. after pausing) lost their Bluetooth connection.

1.1. `Start → Control Panel → Devices and Printers`.
Remove any Nintendo Device.
1.2. Push the tiny SYNC nub inside the balance board's battery compartment and hold it.
1.3. `Add a device → Nintendo RVL-WBC-01` should appear as shown on the right.
Select it. `Next → Pair without using a code`.
1.4. Keep pushing the SYNC button until the `Nintendo RVL-WBC-01` is fully connected.
Wait until both messages "`Device driver software installed successfully`" and "`Bluetooth HID device is installed successfully`" appear and vanish (can take up to 2 minutes).

Method 2 = permanent Bluetooth pairing method:
Drawback: This method requires an external utility `WiiPair.exe` (written by Richard Lynch) that must be downloaded from **http://www.richlynch.com/code/wiipair** and installed on your computer. `WiiPair.exe` is a console program that syncs the Bluetooth connection outside of the `Start → Control Panel → Devices and Printers → Add a device` mechanism and it prevents the computer and the balance board from disconnecting automatically.

2.1. `Start → Control Panel → Devices and Printers`. Remove any Nintendo device.
2.2. Start `WiiPair.exe` and the tiny SYNC nub inside the balance board's battery compartment and hold it.
The blue LED of the balance board must blink.
2.3. `Nintendo RVL-WBC-01` should appear in `Control Panel → Devices and Printers` as shown in Method 1.
2.4. `Push` the SYNC nub (the blue LED of the balance board must blink) until the `Nintendo RVL-WBC-01` is fully connected and
`until` the `wiipair.exe`-console quits with the line: "`1 Wii devices paired`".
Wait until both messages "`Device driver software installed successfully`" and "`Bluetooth HID device is installed successfully`" appear and vanish (can take up to 2 minutes).
2.5. Caution: Even method 2 doesn't connect forever.
It must be repeated after any PC boot/awake or after having switched off the balance board.

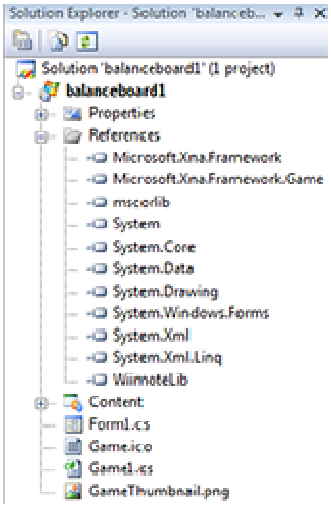
Caution: You cannot install the balance board in parallel, otherwise balanceboard1 will try to start the Wiimote instead of the balance board.

3. Test the balance board bluetooth connection:
3.1. Ensure having installed at least the **Microsoft XNA Framework Redistributable 3.1**.
3.2. Create a new folder C:\temp\BalanceBoardTest.
3.3. Download and store **http://www.miszalok.de/C_3D_XNA/C4_Controller/balanceboard1.zip** and extract the ZIP-file.
3.4. Start balanceboard1.exe.

If you obtain an error message such as "The device is not connected" or "Can't find a BalanceBoard", you have to reinstall the Bluetooth connection from scratch.
4.1. Uninstall completely all Bluetooth Radios from Control Panel → System → Device Manager.
4.2. Remove the USB-Bluetooth stick and re-insert it into the USB port.
4.3. An "Installing device driver software + Click here for status"-message will appear.
4.4. Wait (up to 2 minutes) until two new messages "Your device is ready to use" and "Device driver software installed successfully" will appear and vanish.
4.5. Goto Method 2.

If there is no way to connect to Nintendo RVL-WBC-01:
5.1. Check the batteries. (The balance board has an excessive power consumption.)
5.2. Try another USB Bluetooth stick and do not install any special Bluetooth driver.
Windows 7 default Bluetooth and **HID** drivers works best.
See: **Brian Peeks Blog**, **Accessing the Wiimote**.

Project balanceboard1



1. Main Menu after starting VS 2008:
File → New Project... → Project types: XNA GameStudio 3.1 →
Templates: Windows Game (3.1) →
Name: balanceboard1 → Location: C:\temp →
Create directory for solution: switch it off → OK.
Solution Explorer - balanceboard1: Delete the file Program.cs and the content of Game1.cs.

2. If You find no Solution Explorer-window, open it via the main menu:
View → Solution Explorer.
Inside the Solution Explorer-window click + in front of balanceboard1.
A tree opens. Look for the branch "References".
Click the + in front of References. Check if there are the references:
Microsoft.XNA.Framework and Microsoft.XNA.Framework.Game.

2.1 Download WiimoteLib v1.7 from
http://www.codeplex.com/WiimoteLib/Release/ProjectReleases.aspx
and store WiimoteLib_1.7.zip to C:\temp.
2.2 Extract this ZIP-file and copy WiimoteLib.dll twice to C:\temp\balanceboard1\bin\x86\Debug and to C:\temp\balanceboard1\bin\x86\Release.
2.3 Right click the Solution Explorer branch References → Click Add Reference... →
Select the Browse tab → Add a reference to one of the WiimoteLib.dlls using the Browse tab of the Add Reference-Dialog Box.

3. Create a new class file: balanceboard1 → Add → Windows Form... → Name: Form1.cs → Add
Click the plus-sign in front of balanceboard1 → Form1.cs
Delete the file Form1.Designer.cs underneath Form1.cs and the content of Form1.cs.

The complete code of Game1.cs

Throw away the default content of `Game1.cs` and write the following code into the empty window:

```
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Graphics;
using WiimoteLib;
using System;

static class Program
{ static void Main() { Game1 game = new Game1(); game.Run(); }
  public class Game1 : Microsoft.Xna.Framework.Game
  { private GraphicsDeviceManager g;
    private VertexBuffer vbuffer;
    private BasicEffect  effect;
    private EffectPass   pass;
    private Form1   form = new Form1();
    private Wiimote bb   = new Wiimote();
    const int   cylFaces  = 40;
    const float cylRadius = 3f;
    VertexPositionNormalTexture[] cylinder = new VertexPositionNormalTexture[cylFaces];
    Matrix[] zoom        = new Matrix[6];
    Matrix[] shift       = new Matrix[6];
    Matrix[] shiftToZero = new Matrix[6];
    public static float[] height = { 10f, 10f, 5f, 5f, 30f, 1f };
    public static Vector2 gravity = new Vector2( 10f, 10f );
    public Game1()
    { g = new GraphicsDeviceManager( this );
      this.Exiting += new System.EventHandler( OnExit );
    }

    protected override void Initialize()
    { g.PreferredBackBufferWidth = 500;
      g.PreferredBackBufferHeight = 500;
      g.ApplyChanges();
      g.IsFullScreen = false;
      Window.AllowUserResizing = true;
      Window.Title = "XNA using Wii Balance Bord";
      try { bb.Connect();
            bb.SetLEDs(1);
          }
      catch { System.Windows.Forms.MessageBox.Show( "Can't find a BalanceBoard" ); Exit(); }
      form.Location = new System.Drawing.Point( Window.ClientBounds.Right + 8,
                                                Window.ClientBounds.Top );
      form.Size = new System.Drawing.Size( 100, Window.ClientBounds.Height );
      form.Show();
      base.Initialize();
      double arcus = MathHelper.TwoPi / cylFaces;
      for ( int i=0; i < cylFaces; i++ )
      { cylinder[i].Normal.X = (float)System.Math.Cos( i * arcus );
        cylinder[i].Normal.Z = (float)System.Math.Sin( i * arcus );
        cylinder[i].Position.X = cylRadius * cylinder[i].Normal.X;
        cylinder[i].Position.Z = cylRadius * cylinder[i].Normal.Z;
        if ( i%2 == 0 ) cylinder[i].Position.Y =  0.5f;
        else            cylinder[i].Position.Y = -0.5f;
      }
      vbuffer = new VertexBuffer( g.GraphicsDevice, typeof(VertexPositionNormalTexture),
                                  cylFaces, BufferUsage.WriteOnly );
      vbuffer.SetData< VertexPositionNormalTexture >( cylinder );
      effect = new BasicEffect( g.GraphicsDevice, null );
      effect.View = Matrix.CreateLookAt( new Vector3(3, -30, 100), Vector3.Zero, Vector3.Up );
      effect.Projection = Matrix.CreateOrthographic( 80, 80, 0.01f, 100f );
      effect.LightingEnabled = true;
      pass = effect.CurrentTechnique.Passes[0];
      BasicDirectionalLight light = effect.DirectionalLight0;
      light.Direction = new Vector3( 0f, 0f, 1f );
      light.Direction.Normalize();
      light.DiffuseColor  = Color.White.ToVector3();
      light.SpecularColor = Color.CornflowerBlue.ToVector3();
      light.Enabled = true;
    }
```

```
    protected override void Update(GameTime gameTime)
    { WiimoteState s = bb.WiimoteState;
      BalanceBoardState   bbs = s.BalanceBoardState;
      BalanceBoardSensorsF kg = bbs.SensorValuesKg;
      height[0] = MathHelper.Max( 1f, kg.TopLeft    /4 );
      height[1] = MathHelper.Max( 1f, kg.TopRight   /4 );
      height[2] = MathHelper.Max( 1f, kg.BottomLeft /4 );
      height[3] = MathHelper.Max( 1f, kg.BottomRight/4 );
      height[4] = MathHelper.Max( 1f, bbs.WeightKg  /4 );
      height[5] = 1f;
      gravity.X =     bbs.CenterOfGravity.X;
      gravity.Y = -3f*bbs.CenterOfGravity.Y;
      for ( int i=0; i < 5; i++ )
        zoom[i] = Matrix.CreateScale( 1f, height[i], 1f );
      zoom[5] = Matrix.CreateScale( 0.5f, 0.5f, 0.5f );
      shift[0] = Matrix.CreateTranslation( -20f, height[0]/2f-20, 80f );
      shift[1] = Matrix.CreateTranslation(  20f, height[1]/2f-20, 80f );
      shift[2] = Matrix.CreateTranslation( -20f, height[2]/2f-20, 20f );
      shift[3] = Matrix.CreateTranslation(  20f, height[3]/2f-20, 20f );
      shift[4] = Matrix.CreateTranslation(   0f, height[4]/2f-20, 50f );
      shift[5] = Matrix.CreateTranslation(  gravity.X, height[5]/2f-20, gravity.Y + 50f );
      for ( int i=0; i < 5; i++ )
      { try { form.trackbar[i].Value = (int)height[i]; }
        catch { form.trackbar[i].Value = 150; }
      }
      try { form.trackbar[5].Value = Convert.ToInt32( gravity.X );
            form.trackbar[6].Value = Convert.ToInt32( gravity.Y );
          } catch {}
      g.GraphicsDevice.Vertices[0].SetSource( vbuffer, 0,
                                              VertexPositionNormalTexture.SizeInBytes );
      g.GraphicsDevice.VertexDeclaration = new VertexDeclaration( g.GraphicsDevice,
                                              VertexPositionNormalTexture.VertexElements );
      g.GraphicsDevice.RenderState.CullMode = CullMode.None;
      //g.GraphicsDevice.RenderState.FillMode = FillMode.WireFrame;
      g.GraphicsDevice.RenderState.DepthBufferEnable      = false;
      g.GraphicsDevice.RenderState.DepthBufferWriteEnable = false;
    }

    protected override void Draw( GameTime gameTime )
    { g.GraphicsDevice.Clear( Color.DarkBlue );
      effect.Begin();
      for ( int i = 0; i < 6; i++ )
      { pass.Begin();
          effect.World =  zoom[i] * shift[i];
          try { g.GraphicsDevice.DrawPrimitives( PrimitiveType.TriangleStrip, 0, cylFaces-2 ); }
          catch {}
        pass.End();
      }
      effect.End();
    }
    private void OnExit( object sender, EventArgs e )
    { bb.Disconnect();
      form.Close();
    }
  } // end of class Game1
} // end of class Program
```

The complete code of Form1.cs

Replace the existing lines of `Form1.cs` by the following code:

```
using System;
using System.Drawing;
using System.Windows.Forms;

public class Form1 : System.Windows.Forms.Form
{ public const Int32 nTrackBars = 7;
  public TrackBar[] trackbar = new TrackBar[nTrackBars];
  Label[] label = new Label[nTrackBars];

  public Form1()
  { StartPosition = FormStartPosition.Manual;
    BackColor = Color.White;
    for ( int i = 0; i < nTrackBars; i++ )
    { trackbar[i] = new TrackBar(); Controls.Add( trackbar[i] );
      trackbar[i].ValueChanged += new EventHandler( trackbar_handler );
      trackbar[i].Name = "tr" + i.ToString();
      label[i] = new Label(); Controls.Add( label[i] );
      trackbar[i].AutoSize = false;
      trackbar[i].TickStyle = TickStyle.None;
      trackbar[i].Minimum =  0;
      trackbar[i].Maximum = 150;
      label[i].TextAlign = ContentAlignment.TopCenter;
    }
    trackbar[5].Minimum = -15;
    trackbar[5].Maximum =  15;
    trackbar[6].Minimum = -45;
    trackbar[6].Maximum =  45;
    label[0].Text = "front left";
    label[1].Text = "front right";
    label[2].Text = "back left";
    label[3].Text = "back right";
    label[4].Text = "Weight";
    label[5].Text = "Gravity X";
    label[6].Text = "Gravity Y";
    foreach (Control c in Controls) c.BackColor = Color.Gray;
  }
  protected override void OnResize( EventArgs e )
  { Int32 w = ClientRectangle.Width;
    Int32 h = ClientRectangle.Height / Controls.Count;
    Int32 top = 1;
    for ( int i = 0; i < Controls.Count; i++ )
    { Controls[i].Top    = top;
      Controls[i].Left   = 2;
      Controls[i].Width  = w;
      Controls[i].Height = h - 2;
      top += h;
    }
    for ( int i = 0; i < nTrackBars; i++ ) trackbar[i].Height = h;
  }
  protected void trackbar_handler( object sender, System.EventArgs e )
  { Int32 value = ((TrackBar)sender).Value;
    switch( ((TrackBar)sender).Name )
    { case "tr0":
           Program.Game1.height[0] = value;
           break;
      case "PenThickness":
           break;
      case "Brightness":
           break;
    }
  }
}
```

Click `Debug` → `Start Debugging F5`.
Step on your connected balance board and try it out.