De WII Remote Programmeren

De WII Remote of WIImote van de gelijknamige gameconsole van Nintendo kunnen we gaan inlezen en aansturen vanop de computer want de WIImote maakt gebruik van Bluetooth om zijn signalen te versturen. De WIImote wordt door de PC herkent als een standaard HID-device of Human Interface Device herkent. Een HID-device beschikt over een aantal kanalen waarover data kan worden verstuurd. Elk kanaal wordt meestal gekoppeld aan een specifieke functionaliteit/data en de data kan maar in één richting worden verstuurd. Zo zijn er kanalen voor het inlezen van de buttons, aansturen van de speaker, inlezen van beweging, inlezen van posities, aansturen van de leds en de trilfunctie. De bedoeling is om deze verschillende functionaliteiten te verkennen.

De WIImote functies

In de WIImote zijn verschillende functies aanwezig die al dan niet afzonderlijk kunnen worden ingelezen of aangestuurd.

Buttons

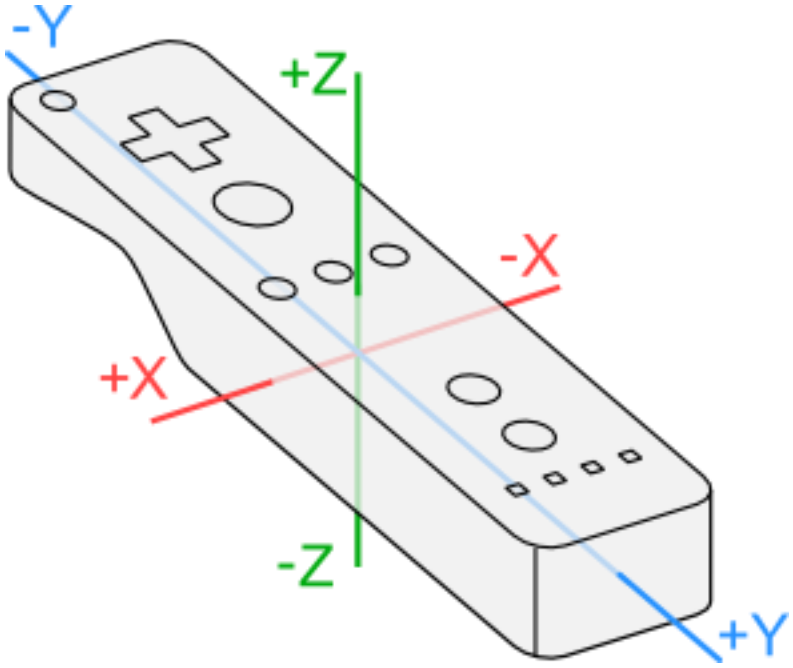
De WIImote beschikt over een elftal buttons. Wanneer we informatie opvragen van de WIImote dan wordt de status van de verschillende buttons bijna altijd meegegeven.

LED's en Rumble of trilfunctie

Er zijn een 4-tal LED's aanwezig op de WIImote. De status van deze LED's kan uiteraard worden gezet maar kan eveneens worden opgevraagd. Via de Rumble kan men de WIImote laten trillen.

Accelerometer

De accelerometer in de WIImote is er voor verantwoordelijk om de bewegingen van de controller te detecteren. Het gaat hier niet om de positie maar puur om de beweging. Dus als je de WIImote stil houdt aan het plafond of aan de grond, de accelerometer zal dezelfde waardes geven. De accelerometer kan bewegingen in de X, Y en Z-as detecteren. Hierbij is de X-as van links naar rechts, de Y-as van voor naar achter en de Z-as van boven naar onder (zie figuur).



IR-camera

Vooraan de WIImote zit een infrarood camera. Deze infrarood camera heeft als doel de positie van de WIImote te bepalen. Deze infrarood camera kan tot vier punten volgen in zijn gezichtsveld. De WIImote gaat dus op zoek naar een aantal duidelijk herkenbare warmtepunten om zich te orienteren, wanneer hij deze niet vind zal je dan ook merken dat WIImote geen correcte data wat betreft positie zal doorsturen. Om dit te behelpen kunnen we best zelf een aantal duidelijke punten voorzien. Normaal wordt bij de WII spelconsole een zogenaamde sensor bar geleverd. Deze sensor bar is dan verantwoordelijk voor het voorzien van een aantal referentiepunten. Deze sensor bar is eigenlijk niet veel meer dan twee infrarood LED-arrays. Dus deze kunnen we gemakkelijk zelf voorzien met behulp van een aantal IR-LED's zonder dat we de spelconsole nodig hebben.

Memory en Registers

In de WIImote kunnen ook gegevens worden opgeslagen of uitgelezen via ingebouwd memory en registers. Zo kunnen bvb. de MII charachters en hun data worden opgeslagen op de WIImote en overgebracht van de ene spelconsole naar de andere.

Speaker

Via de ingebouwde speaker kunnen we tenslotte geluid ook naar de WIImote sturen. Hiervoor moeten we gebruik maken van gewone PCM-encoding of ADPCM-encoding.

Andere

Daarnaast kunnen ook nog statussen en data opgevraagd worden van onder andere de batterij, extensions zoals de Nunchuck, etc.

De WIImote verbinden met de computer

Om de WIImote te gebruiken met de computer moet we een koppeling maken via USB. Hiertoe moeten we de volgende stappen volgen:

1. Ga na of de Bluetooth verbinding van uw computer actief staat. Hoe je dit moet realiseren varieert natuurlijk van computer tot computer.
2. Houd de beide toetsen 1 en 2 van de WIImote samen ingedrukt en blijf deze ingedrukt houden. De vier LED's op de WIImote moeten gedurende deze stap blijvend knipperen. Deze procedure zorgt er voor dat de WIImote kan herkend worden via Bluetooth.
3. Laat nu via het Bluetooth menu uw computer zoeken naar Bluetooth apparaten. In de lijst van gevonden apparaten kan je de WIImote herkennen aan de naam "Nintendo RVL-CNT-01".
4. Maak nu een verbinding met het gevonden apparaat. Wanneer naar een PIN-code of Passkey gevraagd worden dan moet dit leeg gelaten worden.
5. Indien gevraagd wordt welke type service of apparaat dit is kan mag je kiezen voor keyboard/mouse/HID service.
6. Nu is uw WIImote gelinkt aan de computer

Maak nu gebruik van het programma WIImoteTest om te zien of je effectief waardes van de WIImote ontvangt.

De WIImote programmeren

Zoals al eerder vermeld kunnen we de WIImote aanspreken als een HID-device op onze computer. Na initialisatie maakt HID gebruik van zogenaamde reports om zijn data over en weer te sturen. Een report een byte array met een vaste lengte en waarbij de eerste byte een code bevat die aangeeft welke data de report bevat. De lengte van een report bij de WIImote is steeds 22 bytes. De reports kunnen dan nog eens worden opgesplitst in twee grote soorten nl. Input reports en Output reports. Input reports worden verstuurd vanop de WIImote naar de computer, Output reports worden verstuurd door de computer naar de WIImote. Om te kunnen communiceren met HID-devices en dus reports te versturen en ontvangen moeten we gebruik maken van de hid.dll en kernel32.dll bibliotheken. Om de functies in deze bibliotheken gemakkelijk te kunnen aanspreken is er een speciale dll voorzien nl. WII.HID.dll die het programmeer werk vereenvoudigd. Leg in uw project een reference naar WII.HID.dll. Welke reports we kunnen versturen en ontvangen kunnen we terug vinden op het internet via http://www.wiibrew.org/wiki/Wiimote .

Initialisatie

Vooraleer we kunnen reports versturen en ontvangen moeten we de communicatie openen met onze WII remote. Hiertoe kunnen we gebruik maken van de class HIDDevice in WII.HID.dll. Deze class heeft een shared methode GetDevice die als parameters het Vendor-ID en Product-ID vraagt van het HID-device. De methode keert een HIDDevice object terug die ons toe laat om reports te versturen en te ontvangen. De WIImote heeft als Vendor-ID 0x57E en als Product-ID 0x306.

```
//Connectie openen met HID device met als vendor ID &H57E en als product ID &H306
```

```
_device = HIDDevice.GetHIDDevice(0x57E, 0x306);
```

Reports versturen

Als we een report willen versturen dan kunnen we gebruik maken van de WriteReport methode van ons HIDDevice object. Deze methode vraagt een object van het type HIDReport. We kunnen niet zelf instanties aanmaken van het type HIDReport maar ons HIDDevice object kan dit wel via de CreateReport methode. Onderstaande code maakt het report om de LED's en trilfunctie aan te sturen van de WIImote. Via de website http://www.wiibrew.org/wiki/Wiimote vinden we terug dat het report-ID voor deze operatie 0x11 is en we in de eerste data byte de waardes voor de LED's en de trilfunctie kunnen invullen. Onderstaande code doet alle player LED's branden en de WIImote gaat ook trillen.

```
//Report aanmaken
 HIDReport report = _device.CreateReport();
 // Report ID instellen op dat voor het aansturen van de player LED's
 report.ReportID = 0x11;
 //Alle LED's en trilfunctie aanzetten
 report.Data[0] = (byte)0xF1;
 //Het report versturen
 _device.WriteReport(report);
```

Met deze methodiek kunnen we nu reports versturen om de status op te vragen, de LED's en Rumble aan te sturen, Memory en Registers te lezen en schrijven, de IR camera in te stellen, de reporting mode instellen, ...

Reports ontvangen

Het inlezen van reports gebeurd asynchroon. Dit wil zeggen dat we vragen om een report te lezen maar dat we niet onmiddellijk een resultaat terug krijgen. In plaats daarvan zal een callback functie opgeroepen worden als het resultaat er is. Deze callback functie geven we door bij het aanroepen van de functie ReadReport. In onderstaand voorbeeld zie je de code voor de lees-request met het doorgeven van de callback functie. De callback functie die we doorgeven moet voldoen aan de volgende signatuur nl. ...void XXXXX(HIDReport report).

http://msdn.microsoft.com/en-us/library/d186xcf0(v=vs.110).aspx

```
// Lees request starten, OnReadReport aanroepen als het resultaat er is
device.ReadReport(OnReadReport)
```

In de functie OnReadReport kunnen we nu het resultaat verwerken. Dit doen we door eerst te kijken wat het report-ID is en dan kunnen we data verwerken met behulp van de beschrijvingen op de site http://www.wiibrew.org/wiki/Wiimote. Als we continue willen inlezen kunnen we in de callback functie opnieuw de ReadReport functie aanroepen.

```
private void OnReadReport(HIDReport report)
  {
        switch (report.ReportID)
          {
```

```
            case 0x20:
               // Status report
               break;
            case 0x21:
               // Memory en register read report
              break;
            case 0x22:
               // Acknowledge report
              break;
            case 0x30:
               // Core buttons data report
                 break;
            case 0x31:
               // Core buttons en accelerometer data report
               break;
            case 0x37:
               break;
         }
         _device.ReadReport(OnReadReport);
      }
   }
```

Deze aanpak heeft echter een nadeel. De OnReadReport functie wordt op een andere thread uitgevoerd dan de GUI thread. Hierdoor kunnen we nergens in de functie OnReadReport onze GUI updaten. Daarom kunnen we onze routine een beetje herwerken zodat we wel onze GUI kunnen updaten.

Let op deze manier van werken geldt enkel met Forms.

http://msdn.microsoft.com/en-us/library/zyzhdc6b(v=vs.110).aspx

```
   private void OnReadReport(HIDReport report)
   {
      if (this.InvokeRequired)
      {
            this.Invoke(new ReadReportCallback(OnReadReport), report);
       }
      else
      {
         switch (report.ReportID)
         {
            case 0x20:
               // Status report
               break;
            case 0x21:
               // Memory en register read report
              break;
            case 0x22:
               // Acknowledge report
              break;
            case 0x30:
               // Core buttons data report
                 break;
            case 0x31:
               // Core buttons en accelerometer data report
               break;
```

```
            case 0x37:
                break;
        }
        _device.ReadReport(OnReadReport);
    }
}
```

In deze functie verwijst Me naar het form. Test dit nu uit door een Status Request report te versturen met WriteReport en door het resultaat dan in te lezen via ReadReport. Verwerk de verschillende bits en bytes in het status report en visualiseer deze op de GUI.

Indien we willen werken met Windows moeten we gebruik maken van de klasse Dispatcher. De background thread moet via een delegate het werk doorgeven aan de dispatcher die gelinkt is met de UI thread.

http://msdn.microsoft.com/en-us/library/system.windows.threading.dispatcher(v=vs.110).aspx

```
private   void OnReadReport(HIDReport report)
    {

        if (Thread.CurrentThread != Dispatcher.Thread)
        {
            this.Dispatcher.Invoke(new ReadReportCallback(OnReadReport), report);
        }
        Else

     {
            switch (report.ReportID)
            {
```

Data reporting mode

De status van de verschillende functionaliteiten in de WIImote wordt doorgestuurd via zogenaamde data reports. De data reporting mode bepaald welke data wordt verstuurd via de data reports. Deze data reporting mode kunnen we instellen via een report met ID 0x12. Hier kunnen we in de eerste byte opgeven of we continue data reports wensen te ontvangen ook als de data in de WIImote niet gewijzigd is of als we enkel reports willen ontvangen als er iets gewijzigd is. De tweede byte bepaald welke de data we gaan ontvangen. We kunnen hier kiezen uit de volgende waardes nl. 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37 en 0x3E/0x3F. Welke data we in de afzonderlijke gevallen ontvangen kunnen we terug vinden in op de site http://www.wiibrew.org/wiki/Wiimote. De data reporting mode die we kiezen is ook bepalend voor hoe we straks onze IR-Camera moeten configureren. Enkel bij 0x33, 0x36, 0x37 en 0x3E/0x3F kunnen we data ontvangen van de IR-Camera. **Hierbij moet bij 0x33 de camera geconfigureerd worden in Extended mode, bij 0x36 en 0x37 in Basic mode en bij 0x3E/0x3F in Extended mode!** Het inlezen van de verschillende reports gebeurd via ReadReport. Uit het data report kunnen we dan de waardes halen voor de verschillende buttons, de IR-Camera en de Accelerometer. Maak gebruik van http://www.wiibrew.org/wiki/Wiimote om de verschillende reports te verwerken. Let op bepaalde bits van de accelerometer zitten verweven in de button bits!

IR-Camera configureren

Om de IR camera de configureren moeten een aantal stappen doorlopen worden. Bij het configureren moeten bepaalde waardes naar de registers in de WIImote worden geschreven. Het schrijven van deze registers of het memory kunnen we via report-ID 0x16. Hierbij moeten we het adres van het register of de geheugenlocatie meegeven, de lengte van de data en of we naar het geheugen of naar het register schrijven. Om dit te vergemakkelijken kan je gebruik maken van onderstaande functie die een array van bytes naar een welbepaald address schrijft.

```
private void WriteData(int address, byte[] data)
    {
        if ((_device != null))
        {
            int index = 0;


            while (index < data.Length)
            {
                // Bepaal hoeveel bytes er nog moeten verzonden worden
                int leftOver = data.Length - index;

                // We kunnen maximaal 16 bytes per keer verzenden dus moeten we het aantal te verzenden
bytes daarop limiteren
                int count = (leftOver > 16 ? 16 : leftOver);

                int tempAddress = address + index;

                HIDReport report = _device.CreateReport();
                report.ReportID = 0x16;
                report.Data[0] = (byte) ((tempAddress & 0x4000000) >> 0x18);
                report.Data[1] = (byte)((tempAddress & 0xff0000) >> 0x10);
                report.Data[2] = (byte)((tempAddress & 0xff00) >> 0x8);
                report.Data[3] = (byte)((tempAddress & 0xff));
                report.Data[4] = (byte)count;
                Buffer.BlockCopy(data, index, report.Data, 5, count);
                _device.WriteReport(report);
                index += 16;
            }
        }
    }
```

Wanneer we naar het geheugen schrijven moeten we nu het address van het geheugen hexadecimaal doorgeven in het formaat 0xXXXXXX. Willen we echter naar het register schrijven dan moeten we nog een extra waarde toevoegen aan het adres zodat het er als volgt uit ziet 0x4XXXXXX. De 4 in het address maakt duidelijk dat het om een register gaat.

Voor het aanzet van de camera volgen we nu de volgende procedure

1. Stuur een report met ID 0x13 waar bij bit 2 van de eerst data byte op 1 staat
2. Stuur een report met ID 0x1A waar bij bit 2 van de eerst data byte op 1 staat
3. Stuur een byte array met 1 byte met waarde 0x08 naar het **registeradres** 0xB00030

4. Stuur een byte array met de acht bytes voor het eerste sensitivity block naar **registeradres** 0xB00000 bvb [0x02, 0x00, 0x00, 0x71, 0x01, 0x00, 0x90, 0x00, 0x41] voor maximale gevoeligheid
5. Stuur een byte array met de twee bytes voor het tweede sensitivity block naar registeradres 0xB0001A bvb [0x40, 0x00] voor maximale gevoeligheid
6. Stuur een byte array met 1 byte naar het **registeradres** 0xB00033 met de mode voor de camera. De waarde voor de mode hangt af van de reporting mode die we willen. Voor reporting mode 0x36 en 0x37 kiezen we als waarde 0x01 (Basic), voor reporting mode 0x22 kiezen we 0x03 (Extended) en voor 0x3E/0x3F kiezen we 0x05 (Full)
7. Stuur een byte array met 1 byte met waarde 0x08 naar het **registeradres** 0xB00030

Wanneer we deze procedure correct is uit gevoerd kunnen we via de data reports de punten van de camera in lezen.