Voice-Command Home Electronic Appliances via Bluetooth and NFC on Windows Phone 8

Windows Phone 8 comes with NFC, Bluetooth and Speech Recognition access capability. These things could be applied to many great services on earth. In this article, I will show you how I use them to develop a **Voice-Command Home Electronical Appliances Control via Bluetooth and NFC**

Introduction

You might be curious how those things work together. Let me explain. Actually this project couldn't work alone with Windows Phone 8 device but it needs a Bluetooth Home Eletronical Appliances Controller to work with. However I don't know where to buy one since I made it myself. If you have some skill on hardware development, believe me that it is not so hard to build one =)

And this is the duty of Bluetooth, NFC and Speech Recognition in this system:

- **Bluetooth** was used to connect and control the Electronic Appliance via RFCOMM over Bluetooth.
- **NFC** was used to initiate the Bluetooth connection by connect to the Bluetooth device with address stored inside NFC Tag.
- **Speech Recognition** and **Speech Synthesizer** was used to interact with user using voice.

Check this demonstration for clearer image of this project.

The media player is loading...

Prepare phone

Before implement the whole system, you need to prepare something listed below:

- Go to phone settings and turn on Bluetooth
- Go to phone settings and enable Tap&Send
- Pair the Bluetooth Controller with your phone (in order to be able to establish connection)
- Enable Speech Recognition System

Prepare NFC tag by writing Bluetooth Address

Since NFC is used to initiate the Bluetooth Connection, so Bluetooth Address is needed to be stored inside the tag in this pattern

(C) Copyright Nokia 2014. All rights reserved.

```
xx:xx:xx:xx:xx:xx
```

with message type **Windows.HomePair**

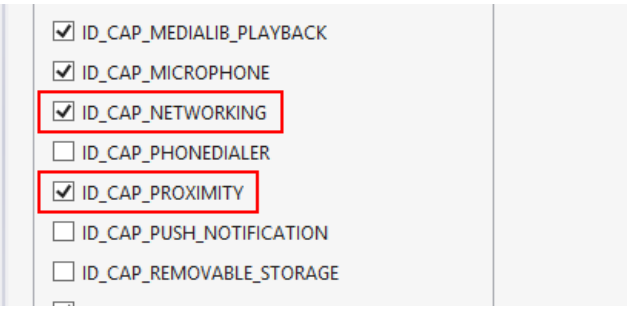
If you don't have any tool to write the tag, you could write it via your Windows Phone device with these codes.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Navigation;
using Microsoft.Phone.Controls;
using Microsoft.Phone.Shell;
using Windows.Storage.Streams;
using Windows.Networking.Proximity;

namespace HomePair
{
    public partial class NFCTagWriter : PhoneApplicationPage
    {
        public NFCTagWriter()
        {
            InitializeComponent();

            if (ProximityDevice.GetDefault() != null)
                mStatusText.Text = "Status: NFC Present";
            else
                mStatusText.Text = "Status: Your phone has no NFC or NFC is disabled";
        }

        private void BtnTagWrite_Clicked(object sender, RoutedEventArgs e)
        {
            var dataWriter = new DataWriter() { UnicodeEncoding = UnicodeEncoding.Utf8
};
            dataWriter.WriteString("00:00:00:00:00:00");


ProximityDevice.GetDefault().PublishBinaryMessage("Windows:WriteTag.HomePair",
dataWriter.DetachBuffer(), MesssageTransmitted);
        }

        private void MesssageTransmitted(ProximityDevice sender, long messageId)
        {
            Deployment.Current.Dispatcher.BeginInvoke(() =>
            {
                MessageBox.Show("The message is written");
            });
        }

    }
}
```

Read NFC Tag and initiate the Bluetooth Connection

(c) Copyright Nokia 2014. All rights reserved.

Please make sure that you have already checked the **ID_CAP_NETWORKING** and **ID_CAP_PROXIMITY** to enable NFC and Bluetooth usage on your application.



To wait for the tag, just simple call this single line of code.

```
ProximityDevice.GetDefault().SubscribeForMessage("Windows.HomePair",
HomePairTagDetected);
```

and implement the callback function **HomePairTagDetected** which will return the data inside via **message.DataAsString**

```
        private void HomePairTagDetected(ProximityDevice sender, ProximityMessage
message)
        {
            // buffer represents bluetooth address
            var buffer = message.DataAsString;

            Deployment.Current.Dispatcher.BeginInvoke(() =>
            {
  // Do something with the Bluetooth Address here
            });
        }
```

And this is the class for Bluetooth Connection which will allow you to connect to Bluetooth device with specific Bluetooth Address and establish the Serial connection to communicate between phone and controller.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using Windows.Networking.Proximity;
using Windows.Networking.Sockets;
using Windows.Storage.Streams;

namespace HomePair
{
    public delegate void BluetoothHomeCommandConnectedCallback();

    public class BluetoothHomeCommandManager
    {
        public static string HomeCommandBluetoothAddress = "00:00:00:00:00:00";

        bool isConnected = false;
        bool isConnecting = false;
```


(c) Copyright Nokia 2014. All rights reserved.

```
        private static BluetoothHomeCommandManager instance;

        public static BluetoothHomeCommandManager Instance
        {
            get
            {
                if (instance == null)
                    instance = new BluetoothHomeCommandManager();
                return instance;
            }
            set
            {
                instance = value;
            }
        }

        DataWriter writer;
        DataReader reader;
        public StreamSocket streamSocket { get; set; }

        public int currentValue { get; set; }

        public async void connectBluetooth(BluetoothHomeCommandConnectedCallback
callback)
        {
            if (isConnected || isConnecting)
                return;

            isConnecting = true;

            try
            {
                PeerFinder.AllowBluetooth = true;
                PeerFinder.Start();

                PeerFinder.AlternateIdentities["Bluetooth:Paired"] = "";
                var peers = await PeerFinder.FindAllPeersAsync();

                if (peers.Count == 0)
                {
                    System.Diagnostics.Debug.WriteLine("Peer not found.");
                }
                else
                {
                    System.Diagnostics.Debug.WriteLine("Looking for " +
BluetoothHomeCommandManager.HomeCommandBluetoothAddress);

                    PeerInformation selectedPeer = null;
                    foreach (PeerInformation peer in peers)
                    {
                        System.Diagnostics.Debug.WriteLine(peer.DisplayName + ":" +
peer.HostName.RawName);
                        if (peer.HostName.RawName.Equals("(" +
BluetoothHomeCommandManager.HomeCommandBluetoothAddress + ")"))
                            selectedPeer = peer;
                    }
```


(c) Copyright Nokia 2014. All rights reserved.

```
                    if (selectedPeer == null)
                    {
                        MessageBox.Show("Not found. Please make sure that you have
already paired the device.");
                        isConnecting = false;
                        return;
                    }

                    streamSocket = new StreamSocket();
                    await streamSocket.ConnectAsync(selectedPeer.HostName, "1");

                    reader = new DataReader(streamSocket.InputStream);
                    reader.InputStreamOptions = InputStreamOptions.Partial;
                    reader.LoadAsync(1);
                    //currentValue = reader.ReadByte();

                    writer = new DataWriter(streamSocket.OutputStream);

                    isConnected = true;

                    callback();
                }
            }
            catch (Exception ex)
            {
                if ((uint)ex.HResult == 0x8007048F)
                {
                    MessageBox.Show("Bluetooth is turned off");
                }
            }

            isConnecting = false;
        }

        public bool disconnect()
        {
            if (isConnecting)
                return false;
            if (!isConnected)
                return true;

            if (streamSocket != null)
                streamSocket.Dispose();

            isConnected = false;

            return true;
        }

        public async void sendData(int data)
        {
            try
            {
                if (streamSocket == null)
                    return;
```


(c) Copyright Nokia 2014. All rights reserved.

```
                writer.WriteByte(90);
                writer.WriteByte((byte)data);
                writer.WriteByte(10);
                uint byteCount = await writer.StoreAsync();
            }
            catch (Exception e)
            {
            }
        }

    }
}
```

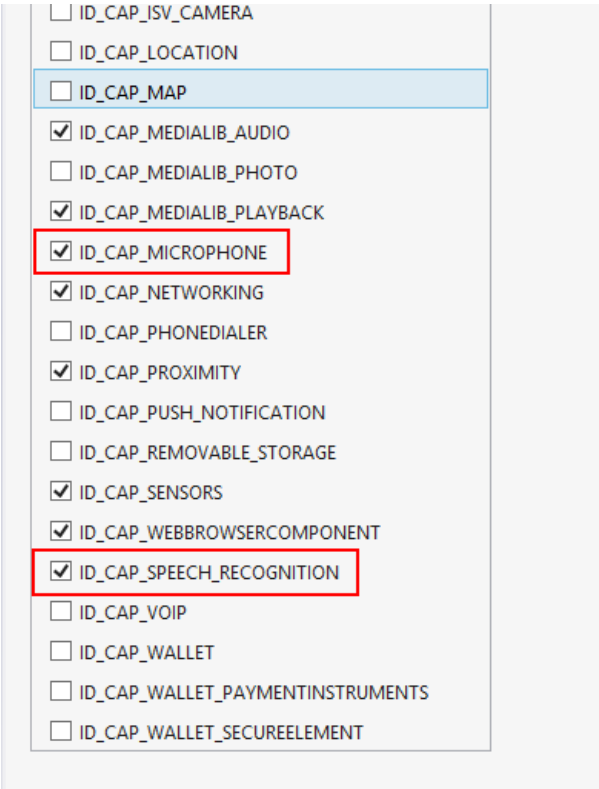
Back to NFC part, modify it like below to establish Bluetooth Connection and navigate to next page, **HomeControlPage.xaml**.

```
        private void HomePairTagDetected(ProximityDevice sender, ProximityMessage
message)
        {
            // buffer represents bluetooth address
            var buffer = message.DataAsString;

            BluetoothHomeCommandManager.HomeCommandBluetoothAddress = buffer;

            Deployment.Current.Dispatcher.BeginInvoke(() =>
            {

BluetoothHomeCommandManager.Instance.connectBluetooth(BluetoothHomeCommandConnected);
            });
        }

        private void BluetoothHomeCommandConnected()
        {
            NavigationService.Navigate(new Uri("/HomeControlPage.xaml",
UriKind.Relative));
        }
```

Implement Speech Recognition and Speech Synthesizer

Make sure that **ID_CAP_MICROPHONE** and **ID_CAP_SPEECH_RECOGNITION** are checked in Capabilities list.

(c) Copyright Nokia 2014. All rights reserved.



Implement Speech Synthesizer

Create **SpeechSysthesizer** instance

```
        SpeechSynthesizer synthesizer = new SpeechSynthesizer();
```

and simple call this line to let your phone speak.

```
  await synthesizer.SpeakTextAsync("Hello");
```

Implement Speech Recognition

Create **SpeechRecognizer** instance

```
  SpeechRecognizer recognizer = new SpeechRecognizer();
```

Add word/sentense to Speech Recognizer dictionary

```
  this.recognizer.Grammars.AddGrammarFromList("Commands",
      new List<string>() {
   "Turn on tne light",
   "It is so hot"
      }
  );
```

Implement Speech Recognition loop as async method.

```
        private async void startSpeechRecognition()
        {
            while (recoEnabled)
            {
                try
```


(C) Copyright Nokia 2014. All rights reserved.

```
                {
                    // Perform speech recognition.
                    this.recoOperation = recognizer.RecognizeAsync();
                    var recoResult = await this.recoOperation;

                    // Check the confidence level of the speech recognition attempt.
                    if ((int)recoResult.TextConfidence <
(int)SpeechRecognitionConfidence.Medium)
                    {

                    }
                    else
                    {
   // work with recoResult.Text
   // for example
   // if (recoResult.Text.Equals("Turn on tne light"))
   // {
   // BluetoothHomeCommandManager.Instance.sendData(10);
   // }
                    }
                }
                catch (System.Threading.Tasks.TaskCanceledException)
                {
                }
                catch (Exception err)
                {
                    // Handle the speech privacy policy error.
                    const int privacyPolicyHResult = unchecked((int)0x80045509);

                    if (err.HResult == privacyPolicyHResult)
                    {
                        MessageBox.Show("You must accept the speech privacy policy to
continue.");
                        this.recoEnabled = false;
                    }
                    else
                    {
                        //err.Message;
                    }
                }
            }
        }
```

and call it after SpeechRecognizer and SpeechSystensizer instances are created

```
            Deployment.Current.Dispatcher.BeginInvoke(() =>
            {
                recoEnabled = true;
                startSpeechRecognition();
            });
```

If you wish to terminate the loop, just set **recoEnabled** to false.

Now just combine each part together and implement UI. The system will be completed ! In case you didn't see it yet, don't forget to check out the video demonstration above.

(c) Copyright Nokia 2014. All rights reserved.

Code Snippets

You could download source code of this project from File:HomePair.zip

(c) Copyright Nokia 2014. All rights reserved.