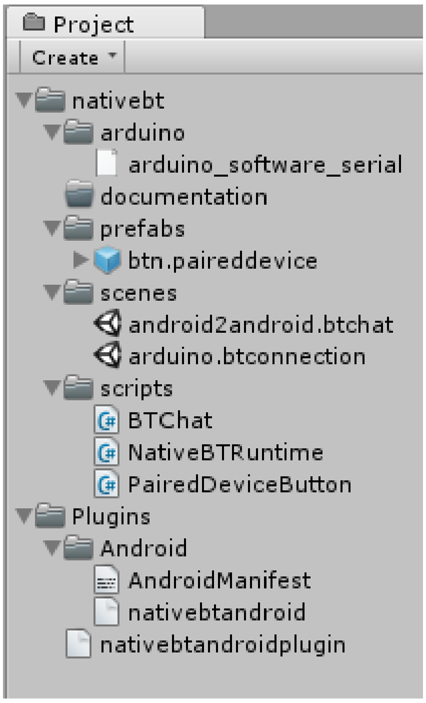
Table of Contents

Introduction

First of all thank you for downloading NativeBT. This asset will save you a lot of time implementing a working Bluetooth connection to another device. NativeBT currently works only with Android smartphones! In the future it will support IOS as well.
If you like the asset, please rate it on the Unity Asset Store (https://www.assetstore.unity3d.com/#!/content/75782)! Thank you in advance!

You can only test this Asset on a real Android smartphone. Don't start this Asset on PC → You will get errors.

The asset hierarchy



```
Project
Create ▾
▼ nativebt
  ▼ arduino
      arduino_software_serial
    documentation
  ▼ prefabs
    ▶ btn.paireddevice
  ▼ scenes
      android2android.btchat
      arduino.btconnection
  ▼ scripts
      BTChat
      NativeBTRuntime
      PairedDeviceButton
▼ Plugins
  ▼ Android
      AndroidManifest
      nativebtandroid
    nativebtandroidplugin
```

After you have imported Native BT you should see 2 main folders. It's important that the „Plugins“ folder is outside of the nativebt folder! Just leave it under the root „Assets“ folder of you project hierarchy, otherwise the plugins won't be loaded by Unity. So now we go into detail.

nativebt

This folder contains all demo files. So it is just for demonstration how this Asset works. If you create a new project you don't need that folder. You just have to import the Plugins folder.

Arduino

This folder contains the Arduino script to communicate throw an HC05 Bluetooth Module. Upload this script to your Arduino (I have used an Arduino Nano for testing) and start the Serial Monitor in the Arduino IDE. Pair the

Arduino with your Android smartphone and connect them with the demo scene (arduino.btconnection). Write something in the command line of the Serial Monitor and hit enter. The message will be displayed on your Android smartphone. I will tell you step by step in the “Examples” section of this readme.

prefabs

This folder contains the button prefab for paired devices.

scenes

Contains 2 scenes that can be used out of the box to connect to another Android device or to connect to an Arduino via a HC05 Bluetooth Module.

scripts

All necessary Scripts for the demoscenes.

PairedDeviceButton:
Holds the Bluetooth device for each list item.

NativeBTRuntime:
Holds a static reference to the main Bluetooth class and the Bluetooth Wrapper class.

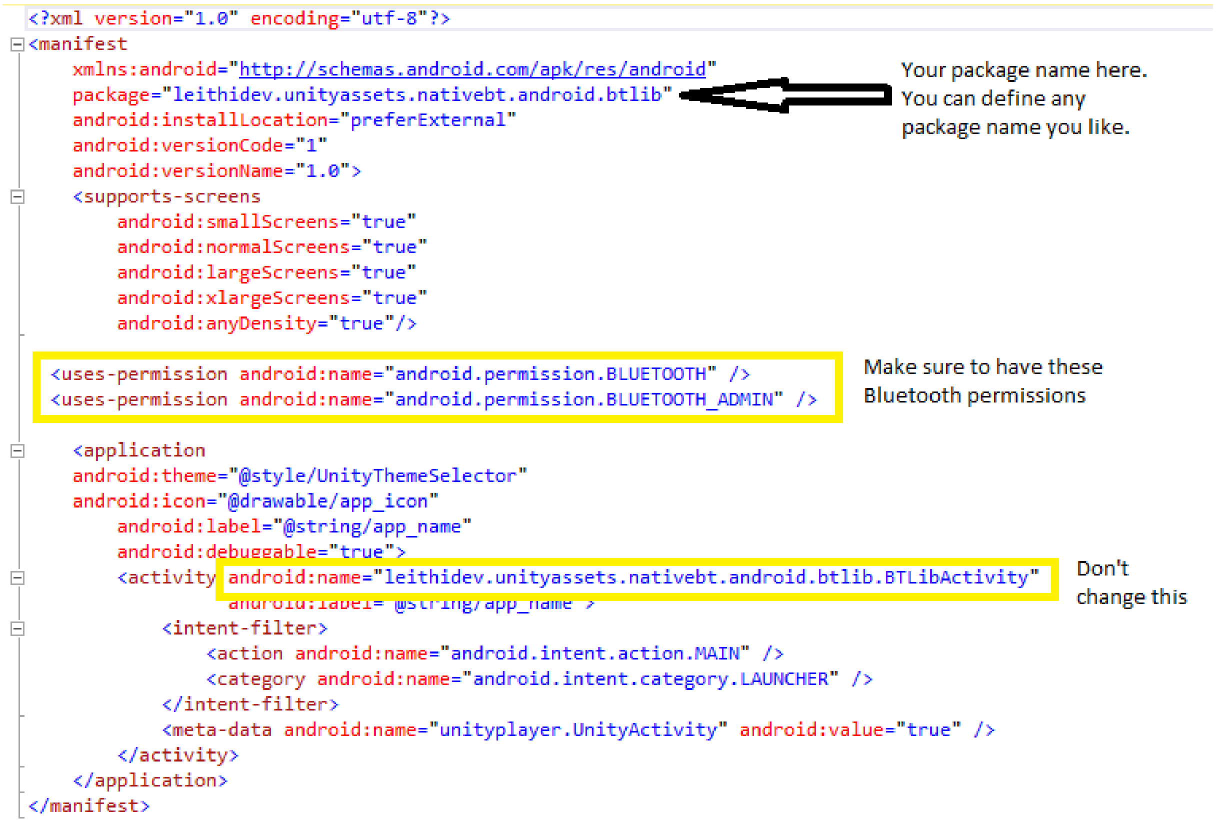
BTChat:
Handles the connection (connect / disconnect / connected) to a Bluetooth device and displays some information and sent / received messages.

Plugins

This folder contains the actual NativeBT libraries. These are important for your project.
Copy this Plugins folder in the root of your Assets folder as it is otherwise Unity won’t find the plugins.

Android

This Folder contains the Android Manifest and the Android library.
To the manifest:



```
<?xml version="1.0" encoding="utf-8"?>
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    package="leithidev.unityassets.nativebt.android.btlib"
    android:installLocation="preferExternal"
    android:versionCode="1"
    android:versionName="1.0">
    <supports-screens
        android:smallScreens="true"
        android:normalScreens="true"
        android:largeScreens="true"
        android:xlargeScreens="true"
        android:anyDensity="true"/>

  <uses-permission android:name="android.permission.BLUETOOTH" />
  <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />

    <application
    android:theme="@style/UnityThemeSelector"
    android:icon="@drawable/app_icon"
        android:label="@string/app_name"
        android:debuggable="true">
        <activity android:name="leithidev.unityassets.nativebt.android.btlib.BTLibActivity"
                  android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
            <meta-data android:name="unityplayer.UnityActivity" android:value="true" />
        </activity>
    </application>
</manifest>
```

Your package name here.
You can define any
package name you like.

Make sure to have these
Bluetooth permissions

Don't
change this

You can adjust the Android Manifest as you like or need.
You have to make sure that you have set the Bluetooth permissions and the correct activity!

How to use NativeBT

All necessary classes are in the leithidev.unityassets.nativebt.android.* namespace.
There are 2 important classes in the core namespace:

AndroidNativeBTHandler

You need an instance of this class to get a AndroidNativeBTWrapper class.
You also have to set the delimeter of the sent string. Most times this is the NewLine sign “\n”. But you can use whatever delimeter you want.
E.q.: If you want to end a sent message at “.” you have to set the delimeter to “.”
“First message from a Bluetooth device.Second message from a Bluetooth device.”

This would trigger 2 times the BTMessageReceived event. That means the Bluetooth connection detects 2 messages. Events will be explained later.

```
this._btHandler = new AndroidNativeBTHandler();
this._btHandler.SetDelimeter(System.Environment.NewLine);
```

AndroidNativeBTWrapper

This class is a property of the AndroidNativeBTHandler class.
You can access an instance of that class by simply getting the property BTWrapper of the AndroidNativeBTHandler instance.

```
this._btHandler.BTWrapper
```

All Bluetooth functions are available in this class.

So you only need 3 lines of code to setup NativeBT.
Now you can connect to any Bluetooth device by simply calling the Connect method of the BTWrapper instance.

Main Funtions

Finding Bluetooth devices

You can find Bluetooth devices in 3 ways.

1) By knowing the address of the remote device and get a LWBluetoothDevice instance.
   BTWrapper.GetBTAdapter().GetRemoteDevice(<address>).
2) By looking up your paired devices and iterate over the returned list.
   BTWrapper.GetPairedDevices()
3) By searching for available unpaired devices.
   BTWrapper.StartDiscoverDevices()

Option 1 and 2 are synchronous methods. This means the will return you LWBluetoothDevice objects instantly.
Option 3 is a asynchronous method. After you have started the method it will search for available nearby devices.
Each time a device is found the event:

```
OnBTDeviceFound(LWBluetoothDevice btDevice)
```

will be triggered and contains the LWBluetoothDevice instance.

Pairing devices

After you have found some devices by starting the method BTWrapper.StartDiscoverDevices() you can call
`BTWrapper.CreateBond(<LWBluetoothDevice>)`
to start the pairing request and the event:
`OnBTPairingRequest(LWBluetoothDevice btDevice)` will be triggered. (This requires Android API Level 19!
If both devices accept the event:
OnBTPaired(LWBluetoothDevice btDevice) will be triggered.
If something fails or one of the 2 devices do not accept the event:
OnBTPairingFailed(LWBluetoothDevice btDevice) will be triggered.

Connecting to a Bluetooth device

After you have found a Bluetooth device (LWBluetoothDevice) you can call the method:
`BTWrapper.Connect(<LWBluetoothDevice>, <UUID>);`
The UUID must be the same the Host is listening on.
You can get all UUIDS from the remote device by calling the method:
`BTWrapper.GetUUIDS(<LWBluetoothDevice>)`
This method requires Android API Level 15!

Disconnecting

BTWrapper.Disconnect()

Listening on a BluetoothSocket

BTWrapper.Listen(<secure (true/false)>, <UUID>);

Events

Here comes the advantage of the asset….Events.
Here is a list of all events that can be listened on. I think the events are self-explanatory.

```
OnBTAdapterDiscoveryCanceled(IList<LWBluetoothDevice> foundBTDevices);
OnBTAdapterDiscoveryFinished(IList<LWBluetoothDevice> foundBTDevices);
OnBTAdapterDiscoveryStarted();
OnBTBondStateChanged(int currentState, int previousState);
OnBTDeviceConnected(LWBluetoothDevice btDevice);
OnBTDeviceConnecting(LWBluetoothDevice btDevice);
OnBTDeviceConnectingFailed(LWBluetoothDevice btDevice);
OnBTDeviceDisconnected(LWBluetoothDevice btDevice);
OnBTDeviceFound(LWBluetoothDevice btDevice);
OnBTDeviceListening(string uuid);
OnBTDisabled();
OnBTEnableApproved();
OnBTEnableCanceled();
OnBTEnabled();
OnBTMessageReceived(string msg);
OnBTMessageSent(string msg);
OnBTPaired(LWBluetoothDevice btDevice);
OnBTPairingFailed(LWBluetoothDevice btDevice);
OnBTPairingRequest(LWBluetoothDevice btDevice);
```

Events are used in the following way:
Get an instance of the class AndroidNativeBTHandler and get the property BTEventsHandler. All available events are accessable there.

E.q.:

```
BTHandler.BTEventsHandler.BTMessageReceived += OnMessageReceived;

private void OnMessageReceived(string msg)
{
  this._chatText.text += "\n" + this._connectedTo.GetName() + ": " + msg;
}
```

Examples

Both examples uses the same core features.
The important Script is BTChat. It uses the static wrapper class NatveBTRuntime to access NativeBT's core features.
The difference of the examples:

- Android2Android BT Chat is HOST – Client oriented
  1 of the Android Smartphones creates a Bluetooth Server Socket with a specified UUID ("c1db6770-a359-11e6-80f5-76304dec7eb7").
  The other Android Smartphones tries to connect with this specified UUID to the Bluetooth Server Socket. At this point both devices are connected and both sides get a BluetoothSocket to communicate with each other.

- Arduino2Android BT Chat is only Client oriented
  The HC-05 Bluetooth Module will be the HOST and creates a Bluetooth Server Socket with its specified UUID ("00001101-0000-1000-8000-00805F9B34FB") automatically (as it starts of).
  The Android Smartphones tries to connect with this specified UUID to the Bluetooth Server Socket. At this point both devices are connected and both sides get a BluetoothSocket to communicate with each other.

Android2Android BT Chat

What do you need:

- 2 Android Smartphones with Android 2.3.3 at least in Debugging Mode
- 1 USB Cable to connect them to the PC

First of all move the "Plugins" folder into the root assets folder of your project.
Switch your build settings to "Android"
Make sure your Android Player Settings are set to Android 2.3.3.
Make sure you have set any bundle identifier.
Make sure your Android Smartphones are paired.

Now connect both Android Smartphones to your PC and build your apk. You can use "Build and run" to automatically play the scene on your android. So you don't have to put the apk on your Android Smartphone and install it manually.

Open both applications on your Smartphones.

Host-Smartphone:

1. Click "Listen"
   Your Smartphone creates a BluetoothServerSocket and listens on the UUID from the BTChat Inspector ("c1db6770-a359-11e6-80f5-76304dec7eb7")

Client-Smartphone:

2. Click "Paired Devices"
   Your Smartphone will list all paired the devices. Attention!! This does not mean you can connect to all of them. It is only a list of paired devices!!
3. Click on your paired devices you want to connect to (the HOST-Smartphone).
4. If the connection is successful, the connected device will be displayed in the top left corner.

That’s all, you can now send messages by simply enter some text into the TextField in the bottom left corner and click “Send”.
You can also disconnect if you want and create another connection.

Arduino2Android BT Chat

What do you need:

- 1 Android Smartphone with Android 2.3.3 at least in Debugging Mode
- 1 Arduino with a BluetoothModule
- Arduino IDE to send messages to the Arduino and send it from the Arduino to your Android Smartphone
- 1 USB Cable to connect it to the PC

First of all move the “Plugins” folder into the root assets folder of your project.
Switch your build settings to “Android”
Make sure your Android Player Settings are set to Android 2.3.3.
Make sure you have set any bundle identifier.
Make sure your Android Smartphone and the BluetoothModule are paired.
Make sure you have put the “arduino_software_serial” code to your Arduino.

1. Power your Arduino.
2. The BluetoothModule should blink fast.
3. Open the application on your Android Smartphone.
4. Click “Paired Devices”
   Your Smartphone will list all paired the devices. Attention!! This does not mean you can connect to all of them. It is only a list of paired devices!!
5. Click on your paired devices you want to connect to (the BluetoothModule of your Arduino).
6. If the connection is successful, the connected device will be displayed in the top left corner.

Now open the SerialMonitor of your ArduinoIDE. Put in some text and hit enter. The message will be displayed on your Android Smartphone.