Nom : ……. Prénom : …….

TP SIN

Bluetooth

Support : Carte Arduino et Shield BLE

Support : Carte arduino Méga et shield BLE

Pré requis (l'élève doit savoir):

- Savoir utiliser un ordinateur
- Avoir réaliser le TP sur les moteurs à courant continu (ETT)
- Savoir programmer en C++ sur C++ Builder

Programme

Objectif terminal :

L'élève doit être capable de programmer une carte arduino méga

Matériels :

- Logiciel C++ Builder version X8 mini
- Carte Arduino méga
- Potentiomètre pour simuler capteur
- Moteur courant continu
- Module de puissance moteur
- Un bouton
- 1 led
- Module BLE shield v2.1
- Téléphone Android (Bluetooth V4.0 mini)

Nom : ……. Prénom : …….

1. **<u>Travail demandé</u>**

- Réaliser le branchement suivant après avoir installé le module BLE sur la carte Arduino :

Led sur entrée sortie digital 22

Potentiomètre entée sortie analogique A0

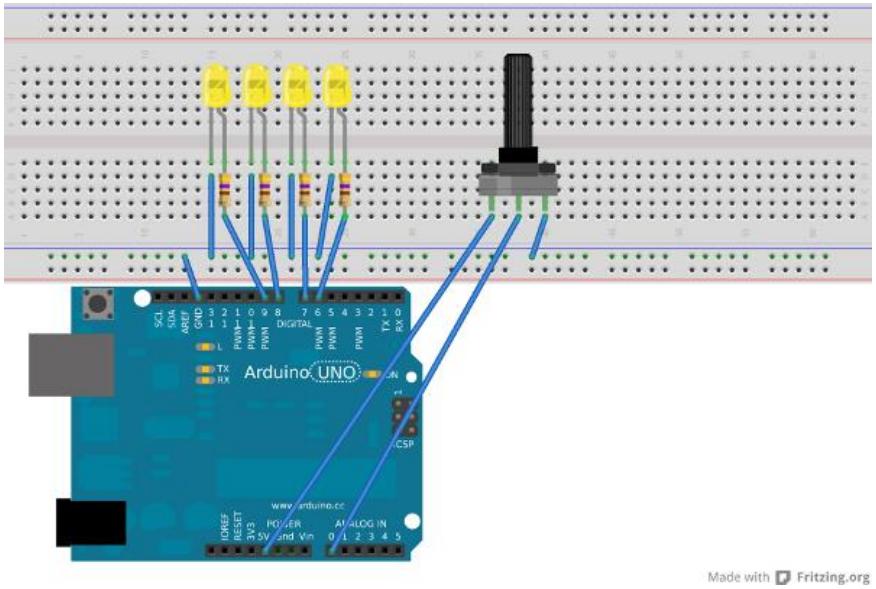
Bouton sur entrée digital 24

- Réaliser le dessin sur Fritzing et vous collez sur le TP

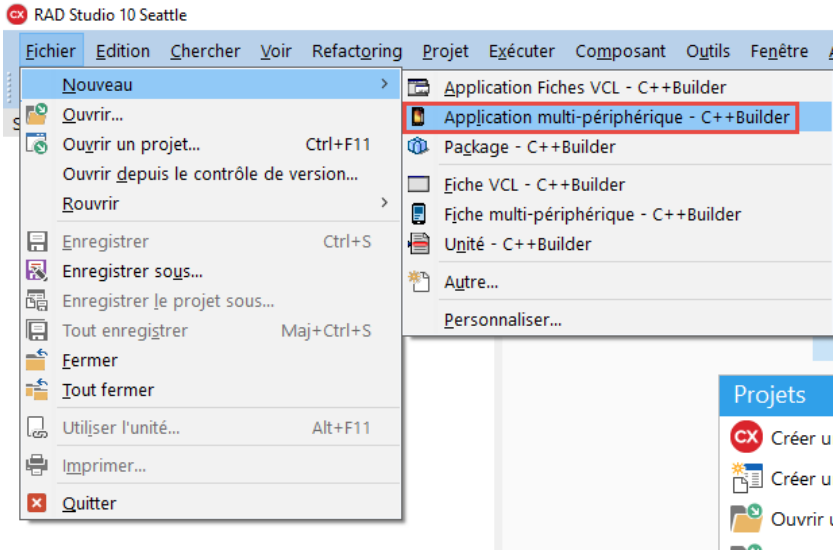
Nom : ……. Prénom : …….

http://fritzing.org/download/

Exemple branchement :

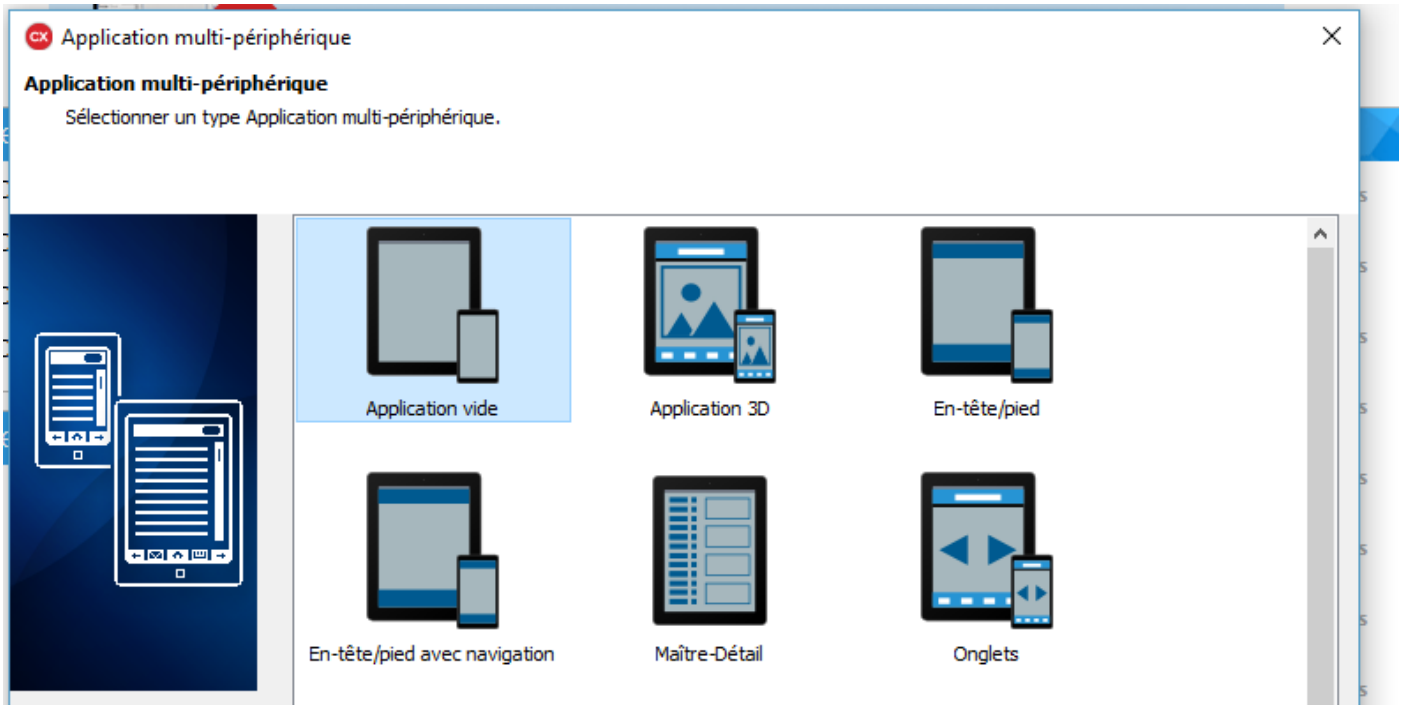


- Démarrer C++ Builder et créer nouvelle application multi-périphérique

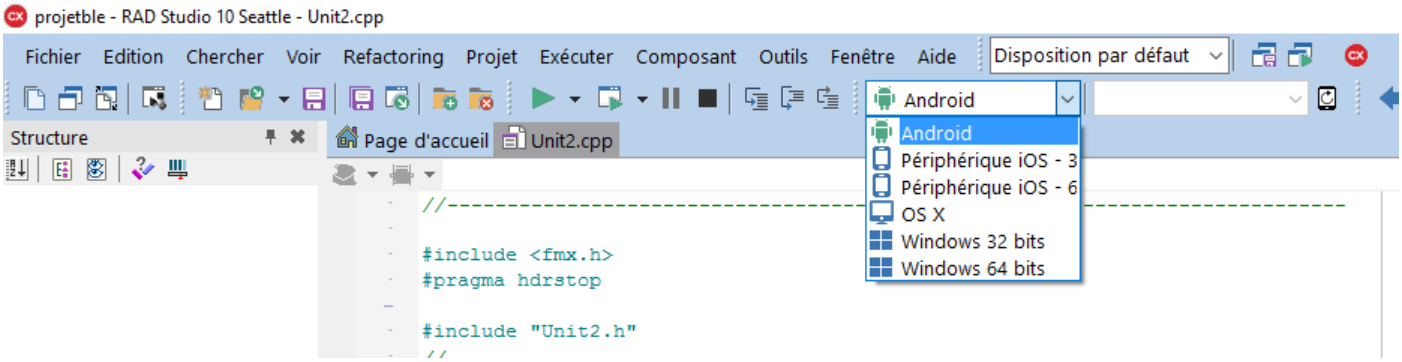


Nom : ……. Prénom : …….

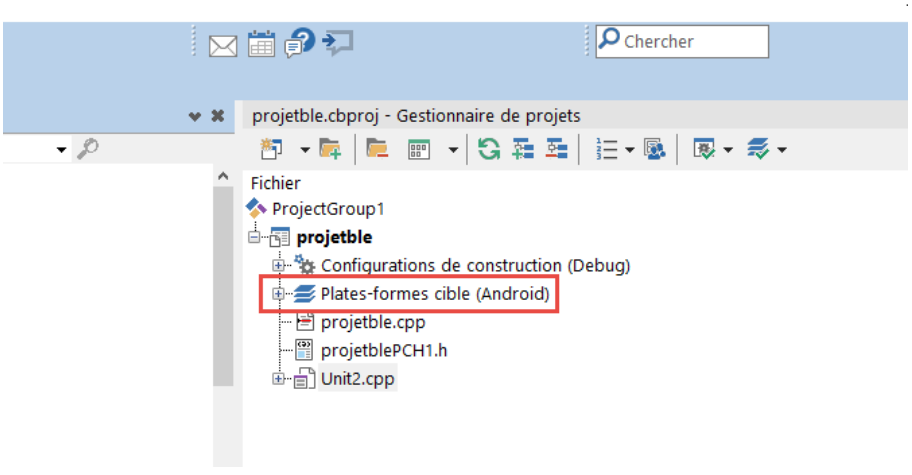
- Puis sélectionner application vide



- Sélectionner affichage android

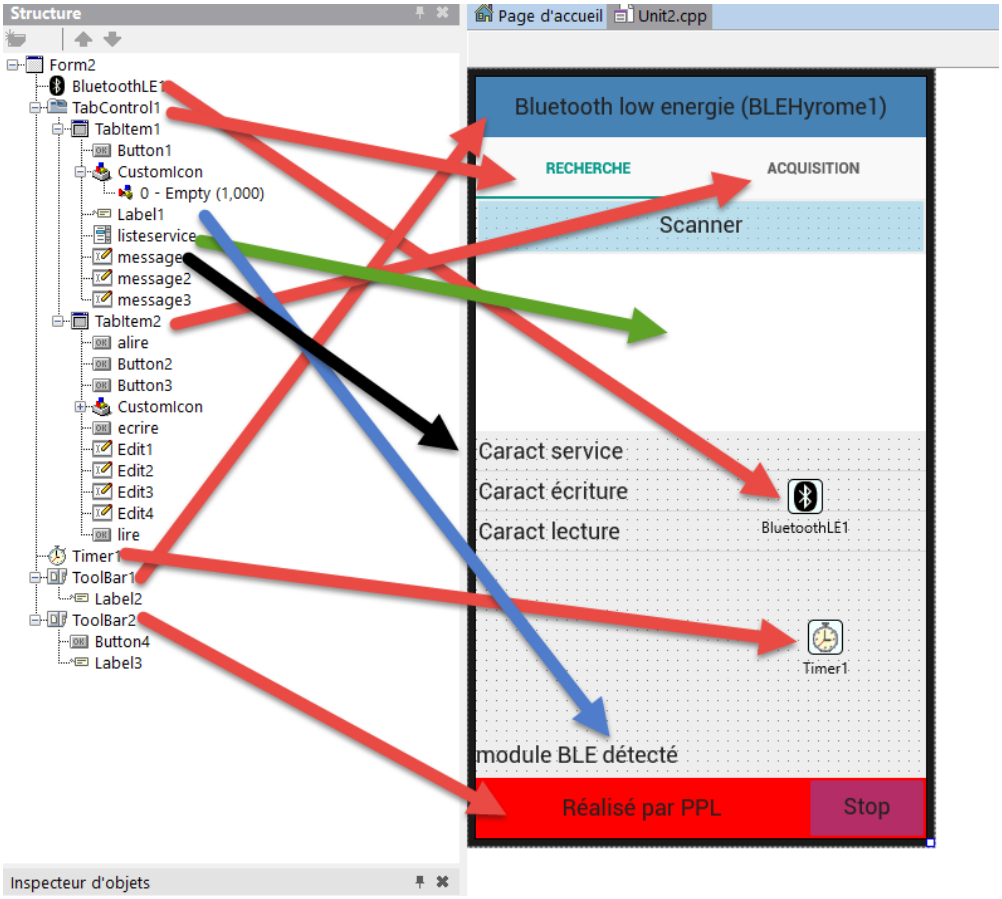


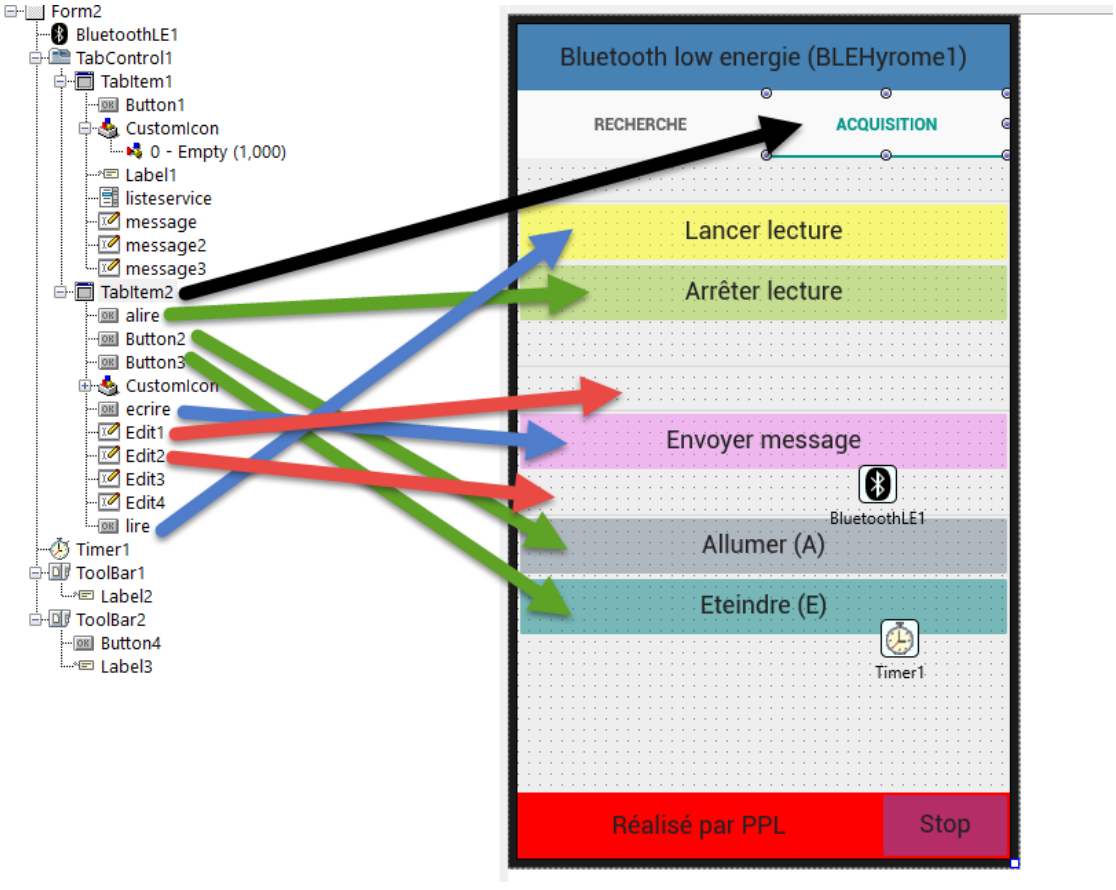
- Sélectionner plates-formes cible (Android)



Nom : ……. Prénom : …….

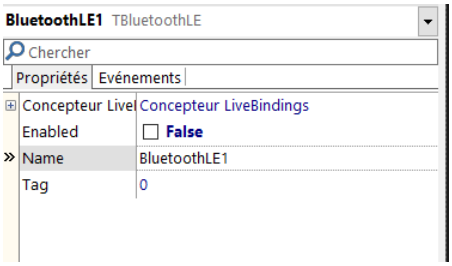
- Créer la présentation suivante

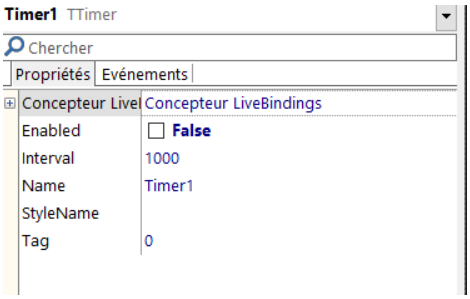




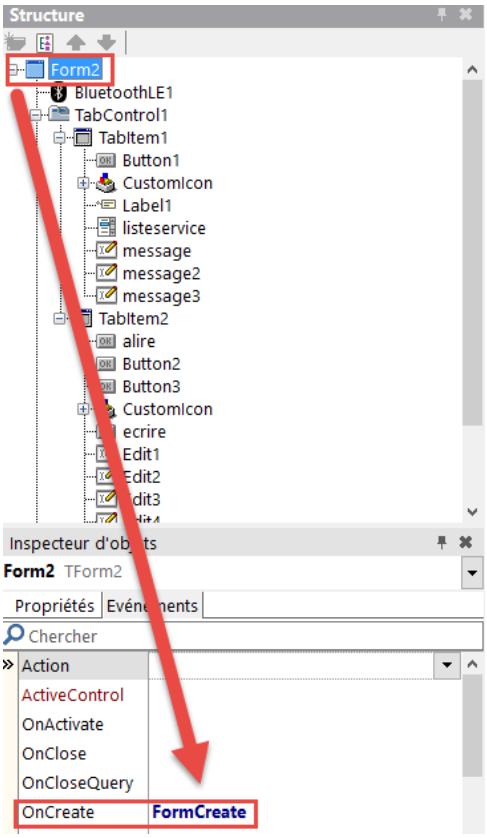
Nom : ……. Prénom : …….

- Rentrer les propriétés suivantes





- Créer la fonction suivante



- Ecrire dans le fichier .cpp le programme suivant

```
void __fastcall TForm2::FormCreate(TObject *Sender)
{
this->TabItem2->Enabled=false;  //rend non actif au démarrage le TabItem2
}
//---------------------------------------------------------------------------
```

Nom : ……. Prénom : …….

- Ecrire dans le fichier .h sous private. Les informations suivantes

```
private:    // Déclarations utilisateur
TBluetoothLEManager * FBLEManager;
    bool FServicesDiscovered;
 void __fastcall DoScan(void);
    int i,j,k,index;
    UnicodeString adresse;
    UnicodeString nomcaracteristique;
        UnicodeString UUIcaracteristique;
    TBluetoothGattCharacteristic * AChar;
    TBluetoothGattCharacteristicList *ACharList;
        TBluetoothGattDescriptorList *Descriptor;
        int CurrentService;
    int CurrentCharacteristic;
 TBluetoothLEDevice*    FBLEDevice;
TBluetoothGattService*FBLEGattService;
TBluetoothGattService*FBLEGattServiceL;
    TBluetoothGattCharacteristic*FBLEGattChar;
        TBluetoothGattCharacteristic*FBLEGattCharL;
```

A partir de la version 10.3 vous devez insérer dans votre programme Android des permissions. Pour cela vous devez rajouter les informations suivantes

http://docwiki.embarcadero.com/RADStudio/Rio/en/Android_Permission_Model

https://developer.android.com/reference/android/Manifest.permission

```
#ifndef Unit2H
#define Unit2H
//---------------------------------------------------
#include <System.Classes.hpp>
#include <System.Permissions.hpp>
#include <FMX.Controls.hpp>
#include <FMX.Forms.hpp>
#include <System.Bluetooth.Components.hpp>
#include <System.Bluetooth.hpp>
```

Dans le fichier .cpp

```
#include <fmx.h>
#ifdef __ANDROID__
    #include <Androidapi.Helpers.hpp>
    #include <Androidapi.JNI.Os.hpp>
#endif
#include <FMX.DialogService.hpp>
#pragma hdrstop

#include "Unit2.h"
//---------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.fmx"
TForm2 *Form2;
```

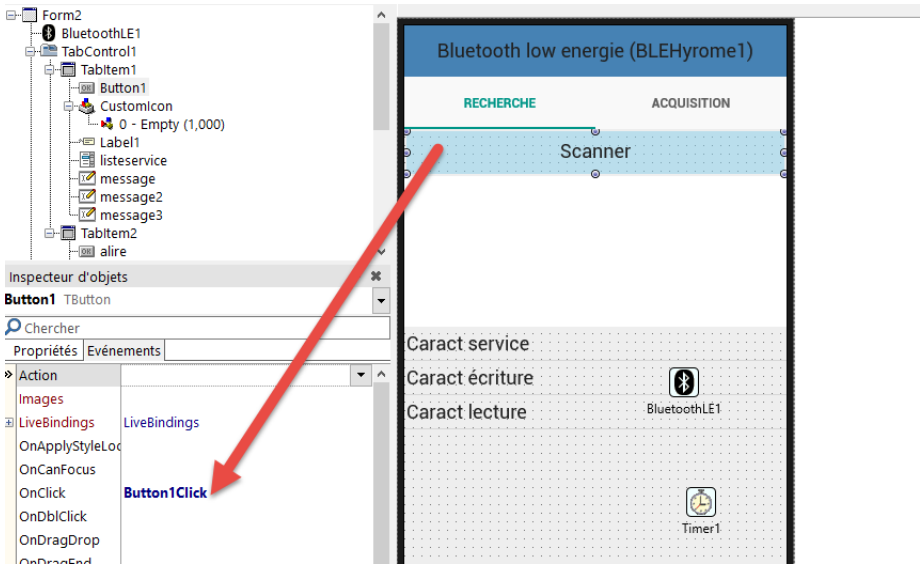
Remarque :

DoScan() est une fonction qu'on va créer pour permettre de rechercher les périphériques BLE

Le reste, ce sont des variables qui vont nous servir pour le programme.

Nom : ……. Prénom : …….

- Sélectionner le bouton scanner et créer la fonction Button1Click



- Dans le fichier .cpp grâce la fonction Button1Click, on va lancer la recherche des périphériques
- Ecrire les éléments suivants

```
void __fastcall TForm2::Button1Click(TObject *Sender)
{
this->listeservice->Items->Clear;    // effacement de la liste
 BluetoothLE1->Enabled=true;         // lancement du module bluetooth
   this->message->Text="Message";    // écrire le message "message"
     this->TabItem2->Enabled=false;  //rend inactif le TabItem2
 DoScan();                           //lancement de la fonction
}
//---------------------------------------------------------------------------
```

Pour la version 10.3

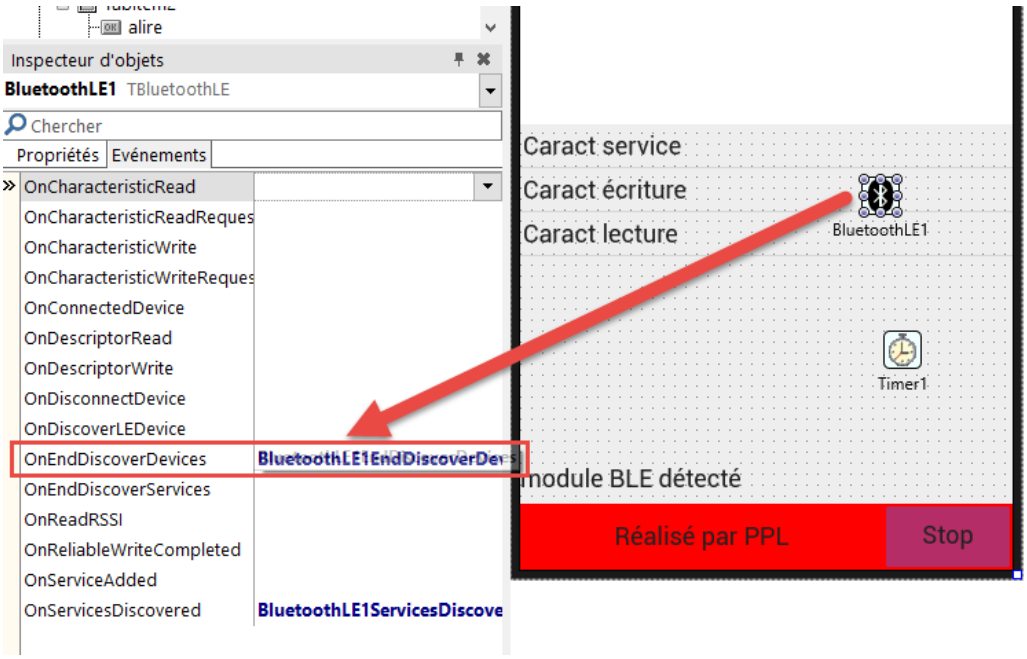
```
void __fastcall TForm2::Button1Click(TObject *Sender)
{
#ifdef __ANDROID__

        DynamicArray<String> permissions;
        permissions.Length = 1;
        permissions[0] = JStringToString(TJManifest_permission::JavaClass->ACCESS_FINE_LOCATION);

        PermissionsService()->RequestPermissions(permissions,
            [this](const DynamicArray<String> APermissions, const DynamicArray<TPermissionStatus> AGrantResults)
            {
                if ((AGrantResults.Length == 1) and(AGrantResults[0] == TPermissionStatus::Granted))
                {
                    this->listeservice->Items->Clear();    // effacement de la liste
 BluetoothLE1->Enabled=true;         // lancement du module bluetooth
   this->message->Text="Message";    // écrire le message "message"
     this->TabItem2->Enabled=false;  //rend inactif le TabItem2
 DoScan();
                }
                else
                {
      BluetoothLE1->Enabled=false;        // lancement du module bluetooth
                    ShowMessage("Location permission not granted");
                }
            });

#endif
                        //lancement de la fonction
}
//---------------------------------------------------------------------------
```

Nom : ……. Prénom : …….

- Maintenant on va créer la fonction DoScan

```
//
void __fastcall TForm2::DoScan(void)
{
BluetoothLE1->DiscoverDevices(4000);  //lance la recherche des périphériques pendant 4 secondes

}
```

- Créer la fonction suivante dans le module bluetooth



- Une fois la procédure terminée, l'événement OnEndDiscoverDevices est déclenché. Quand vous avez découvert le périphérique, vous pouvez commencer à obtenir les services et les caractéristiques du profil standard.
- Ecrire dans la nouvelle procédure les éléments suivant :

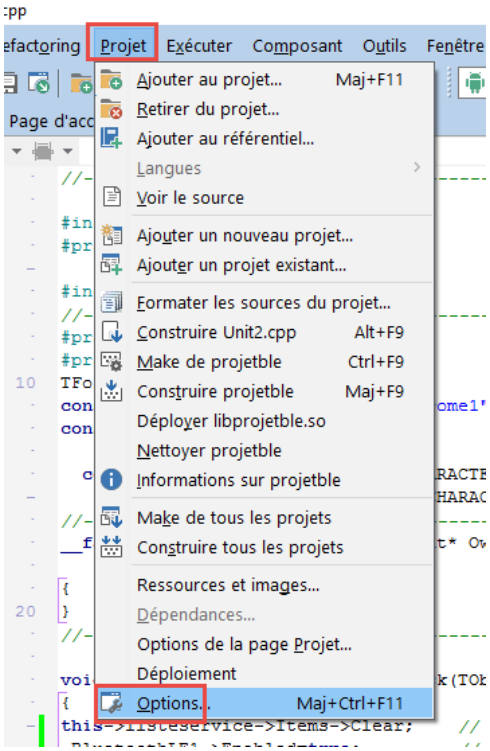
```
void __fastcall TForm2::BluetoothLE1EndDiscoverDevices(TObject * const Sender, TBluetoothLEDeviceList * const ADeviceList)

{
 TBluetoothLEDevice*Device;
if (ADeviceList->Count==0) {  // on teste si la liste des périphériques est vide
    message->Text = "Aucun module bluetooth BLE trouvé";
    return;
}

  FBLEDevice = NULL;  // on vide la variable qui va nous servir à récupérer le paramètre du périphrique choisi
 listeservice->Items->Clear;   // on vide l'affichage de la liste
 listeservice->Items->Add("Liste élément:"); // on rajoute cette ligne
  for( i=0;i < ADeviceList->Count;i++)    // on réalise une boucle à fin d'afficher les périphériques sur la liste
 {
  listeservice->Items->Add(ADeviceList->Items[i]->DeviceName);
  if (ADeviceList->Items[i]->DeviceName==HRDeviceName) {  // on contrôle que le périphérique correspond à celui désiré
    FBLEDevice = ADeviceList->Items[i];
   listeservice->Items->Add("Elément trouvé");
   Label1->Text= HRDeviceName;   // on affiche le nom de l'élément
  }
 }
 if (FBLEDevice==NULL) {
 return;
 }
 if (FBLEDevice->DiscoverServices()==false) {   //dans le cas ou aucun service n'avez été détecté, on lance la recherche des services pour le périphérique choisi
 FBLEDevice->DiscoverServices();
 }
}
//---------------------------------------------------------------------------
```

- Avant de tester le programme, on va d'abord rentrer les paramètres du périphérique, du service concerné et des caractéristiques qui vont permettre de lire et écrire des données

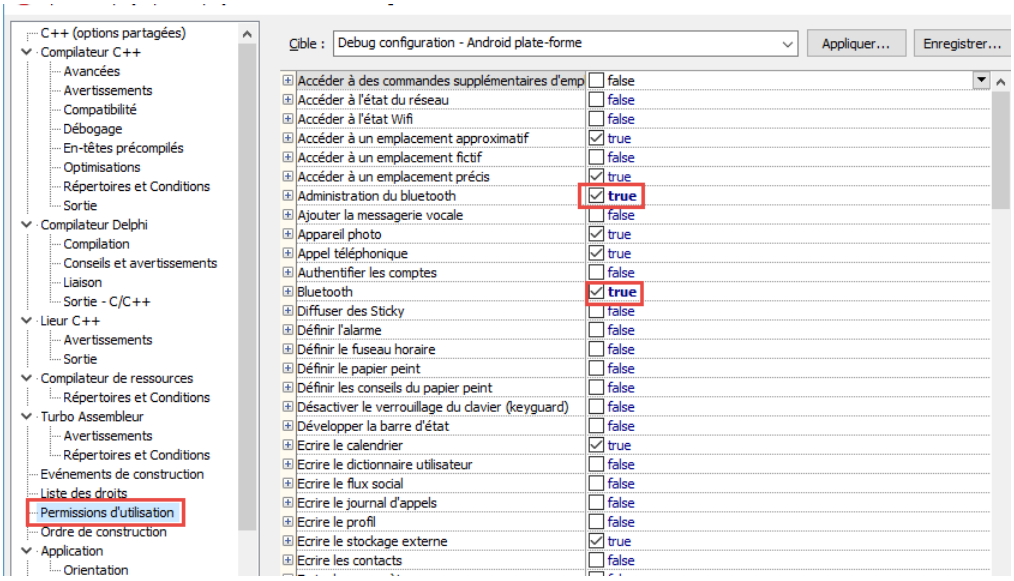
Nom : …….. Prénom : ……..

```
#include <fmx.h>
#pragma hdrstop

#include "Unit2.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.fmx"
TForm2 *Form2;
const String HRDeviceName = "BLEHyrome1";
const TBluetoothUUID    HSERVICE          = StringToGUID("{713D0000-503E-4C75-BA94-3148F18D941E}");

  const TBluetoothUUID   HMSoft_CHARACTERISTIC  = StringToGUID("{713D0003-503E-4C75-BA94-3148F18D941E}");
    const TBluetoothUUID   HMSoft_CHARACTERISTIC1  = StringToGUID("{713D0002-503E-4C75-BA94-3148F18D941E}");
```

- Une dernière chose importante, les autorisations pour Android, pour Windows ce n'est pas nécessaire.
- Sélectionner projet-option

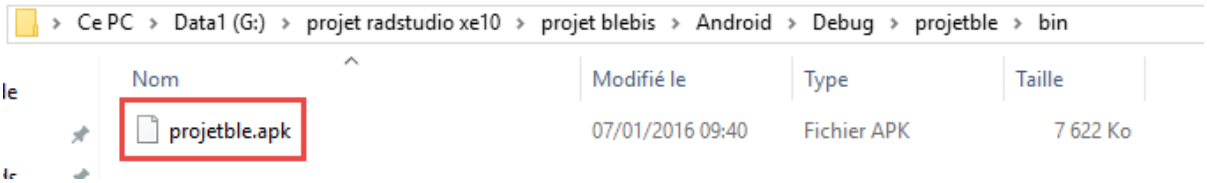


- Puis dans « Permissions d'utilisation » sélectionner les deux permissions suivantes

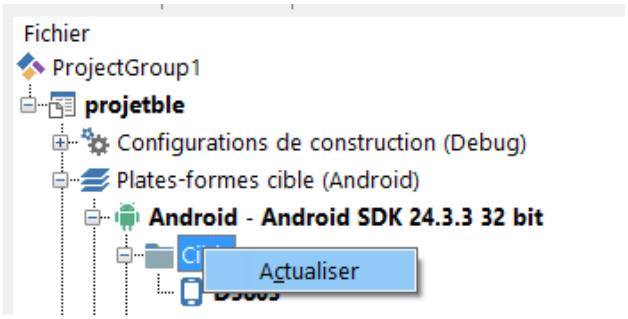


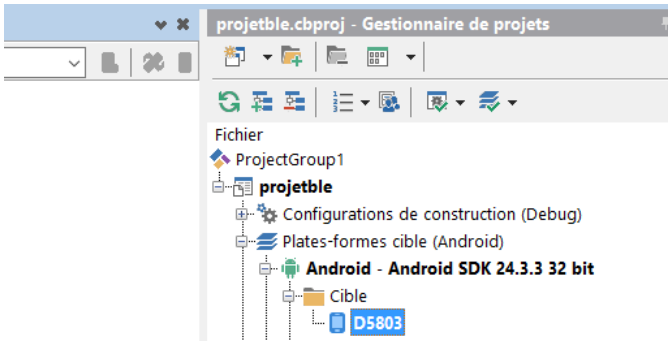
Nom : ……. Prénom : …….

- Maintenant trois solutions s'offrent à vous pour compiler et tester l'application.
  - Soit vous lancez la compilation, puis vous récupérez le fichier. apk et vous le lancer sur votre téléphone

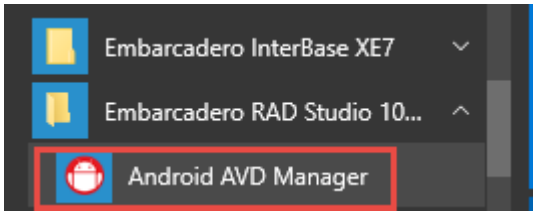


  - Soit vous branchez votre téléphone en mode développeur, et choisi dans cible votre téléphone, puis lancer la compilation. Penser à réactualiser la cible.

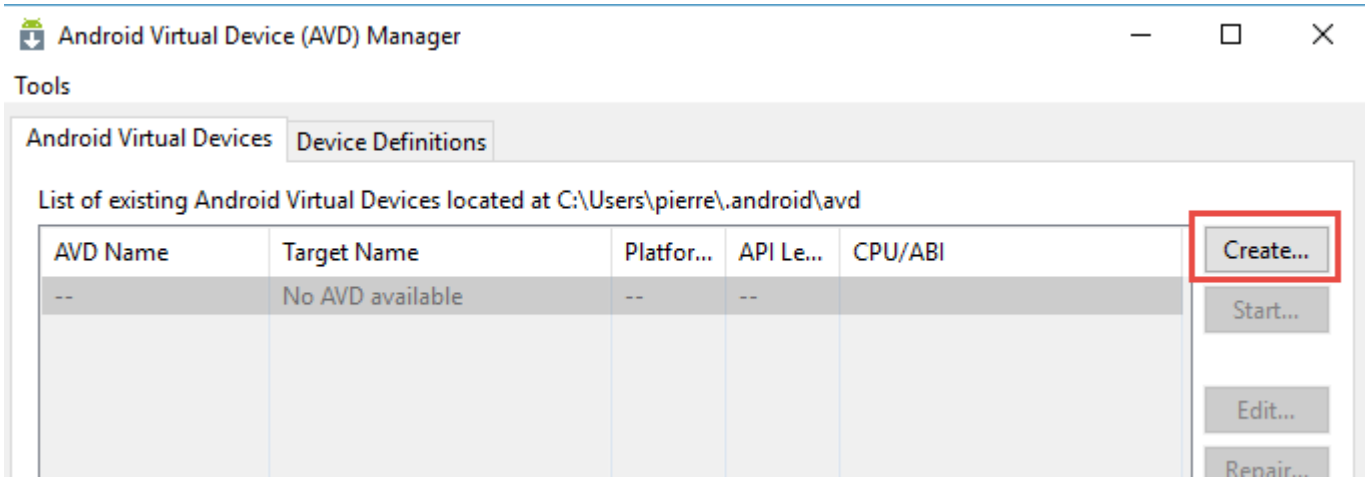




  - Dernière solution, installer un émulateur Android, puis lancer la compilation en le sélectionnant comme cible



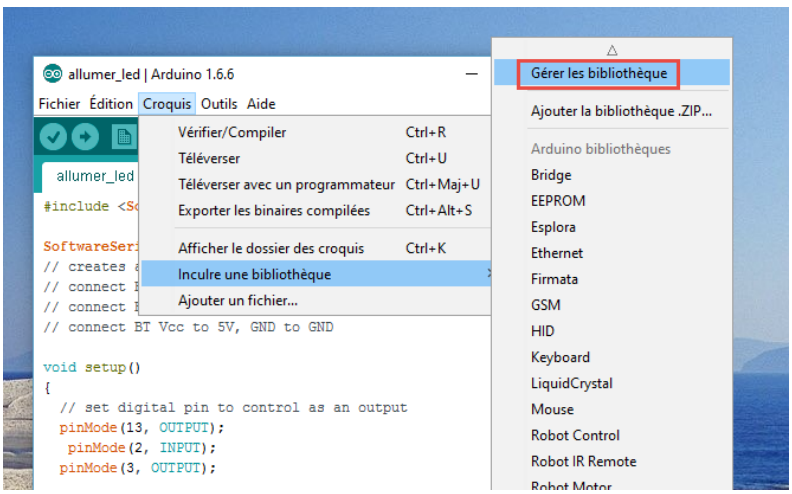
Nom : ……. Prénom : …….



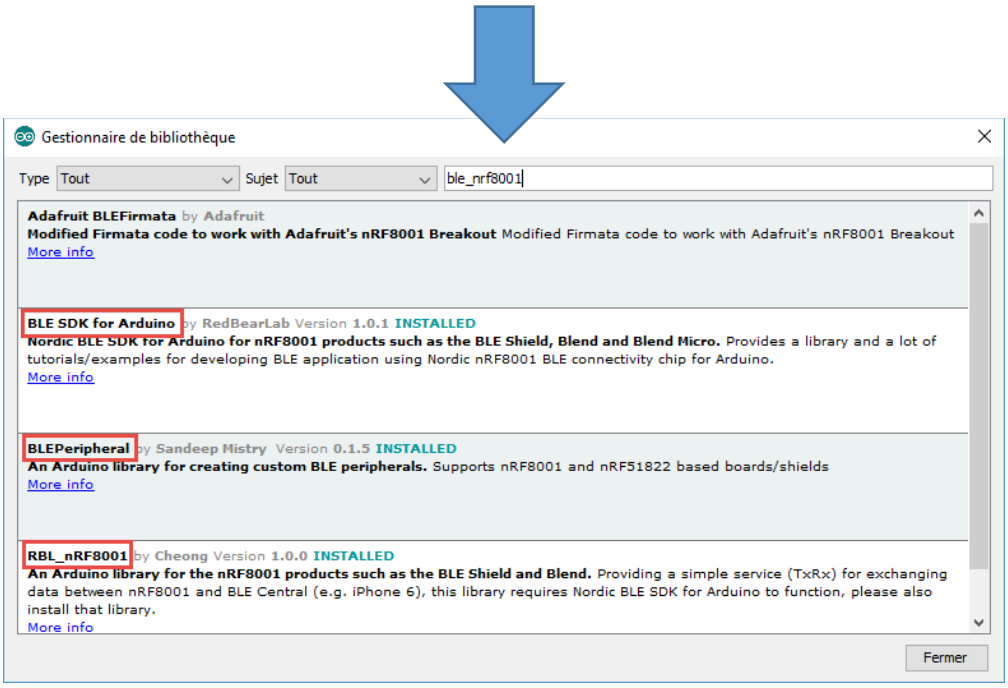
- Maintenant passons à la carte Méga.
- Installer le shield BLE sur la carte
- Lancer le programme Arduino

https://www.arduino.cc/en/Main/Software#

- Installer la bibliothèque RBL_nRF8001.h



Nom : ……. Prénom : …….



- Créer le nouveau programme suivant

```
#include <SPI.h>

#include <EEPROM.h>

#include <boards.h>

#include <RBL_nRF8001.h>

#define ANALOG_IN_PIN A0

int pot;


char* texte = "Text";

const int buttonPin = 24;

int buttonState=0;

int lastbuttonState=0;

int i;

String str;

void setup()

{
```

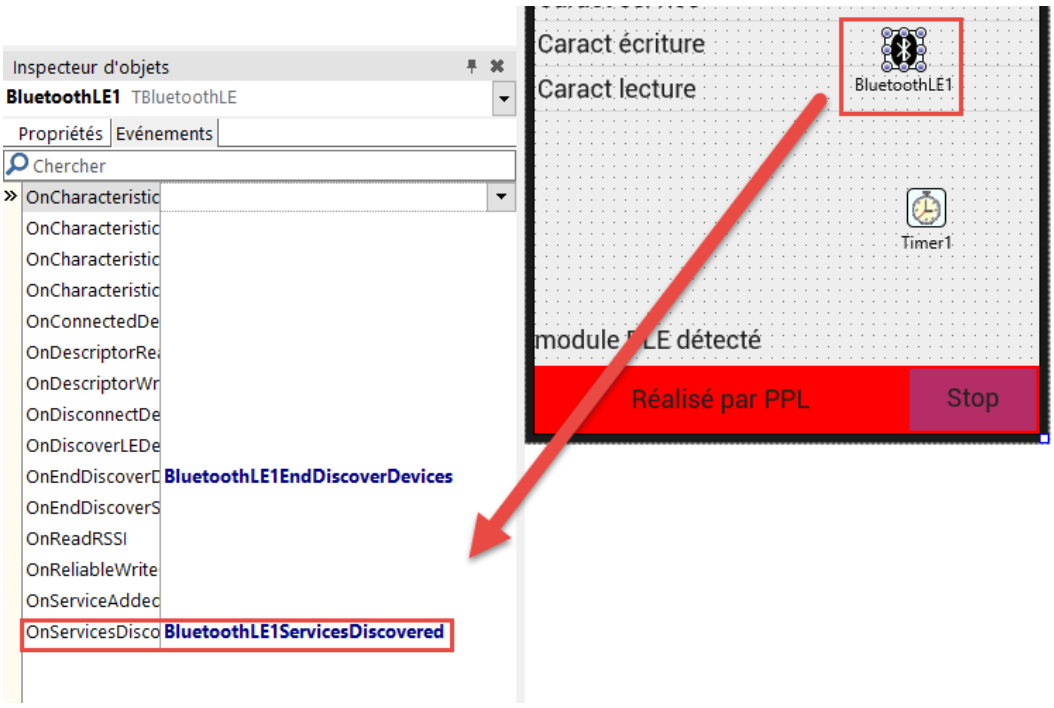
```
 ble_begin();
 Serial.begin(9600);
 ble_set_name("BLEHyrome7");
// ble_set_name("longboard");
texte="Arduino-connecte";
 for(int i=0;i<16;i++)
{
 ble_write(texte[i]);
}
 pinMode(10, OUTPUT);
 pinMode(buttonPin, INPUT);
 digitalWrite(10, LOW);
}
void loop()
{
 if ( ble_available() )

 {
 byte cmd;
 cmd = ble_read();
 Serial.write(cmd);



 // Parse data here
 switch (cmd)
 {
 case 'A': // query protocol version
 {
 Serial.println("allumer");
```

```
texte="Allume";
for(int i=0;i<6;i++)
{
ble_write(texte[i]);
}
digitalWrite(10, HIGH);

}
break;

case 'E': // query protocol version
{
Serial.println("eteindre");
texte="Eteint";
for(int i=0;i<6;i++)
{

ble_write(texte[i]);
}
digitalWrite(10, LOW);
}
break;
}

}
ble_do_events();
}
```

- Compiler le programme
- Tester l'ensemble
- Expliquer les lignes de programme Arduino ci-dessus

Nom : ……. Prénom : …….

- Dans la suite du TP, nous allons détecter les services et les caractéristiques du module BLE
- Lorsqu'on utilise la méthode DiscoverServices pour découvrir les services d'un périphérique particulier, une fois les services découverts, l'événement OnServicesDiscovered est déclenché.
- Créer la fonction suivante sur C++ builder



- Ouvrir le fichier .cpp et écrire les lignes suivantes

Nom : ……. Prénom : …….

```
void __fastcall TForm2::BluetoothLE1ServicesDiscovered(TObject * const Sender, TBluetoothGattServiceList * const AServiceList)

{
FBLEGattChar=NULL;  //vide la variable
 listeservice->Items->Clear();   //efface les lignes dans la liste Box
  TBluetoothGattService*Service;   //Crée la variable pour le service

 i=0;
 j=0;
 k=0;
for( i=0;i < AServiceList->Count;i++) {   //lance une boucle pour détecter les services
     listeservice->Items->Add("Services:");
     listeservice->Items->Add(AServiceList->Items[i]->UUIDName);   //récupère le nom du service trouvé
         listeservice->Items->Add(GUIDToString(AServiceList->Items[i]->UUID)); // récupère la caractéristique UUID du service
        TBluetoothGattCharacteristicList *ACharList = AServiceList->Items[i]->Characteristics; //récupère la liste des caractéristiques du service dans une variable
      if ( GUIDToString(AServiceList->Items[i]->UUID)=="{713D0000-503E-4C75-BA94-3148F18D941E}") {  //Contrôle si c'est le service recherché
     this->message->Text="Service trouvé";
     this->TabItem2->Enabled=true;   //rend actif le TabItem2 puisque le service est trouvé
     FBLEGattService =AServiceList->Items[i];   //enregistre le service dans la variable pour la lecture et l'écriture
      FBLEGattServiceL =AServiceList->Items[i];
      }
     }
      if (FBLEGattService ==NULL) {
      this->message->Text="Aucun service trouvé";
      return;
      }
```

```
BluetoothLE1->GetCharacteristics(FBLEGattService);   //lance la recherche des caractéristiques du service voulu
 FBLEGattChar = BluetoothLE1->GetCharacteristic(FBLEGattService, HMSoft_CHARACTERISTIC ); //pour obtenir les informations d'une caractéristique particulière
 if (FBLEGattChar==NULL) {
 message2->Text="Aucune caractéristique trouvée";
 Edit1->Text="Aucune caractéristique";
 }
  if (FBLEGattChar!=NULL) {
 message2->Text="Caractéristique trouvée : "+ GUIDToString(FBLEGattChar->UUID) ;
 Edit1->Text=GUIDToString(FBLEGattChar->UUID);
 if((FBLEGattChar->Properties.Contains(TBluetoothProperty::Write)) || (FBLEGattChar->Properties.Contains(TBluetoothProperty::WriteNoResponse))
        || (FBLEGattChar->Properties.Contains(TBluetoothProperty::SignedWrite)))    //teste les propriétés de la caractéristique
 {
     Edit2->Text="Ecriture possible";
 }
 else
 {
     Edit2->Text="Ecriture impossible";
 }
  }
```

```
    FBLEGattCharL = BluetoothLE1->GetCharacteristic(FBLEGattService, HMSoft_CHARACTERISTIC1 );
    if (FBLEGattCharL==NULL) {
    message3->Text="Aucune caractéristique trouvée";
    Edit3->Text="Aucune caractéristique";
    }
     if (FBLEGattCharL!=NULL) {
    message3->Text="Caractéristique trouvée : "+ GUIDToString(FBLEGattCharL->UUID) ;
    Edit3->Text=GUIDToString(FBLEGattCharL->UUID);
    if(FBLEGattCharL->Properties.Contains(TBluetoothProperty::Notify))
    {
        Edit4->Text="Lecture possible";
    }
    else
    {
        Edit4->Text="Lecture impossible";
    }
     }

      }
//---------------------------------------------------------------------------
```

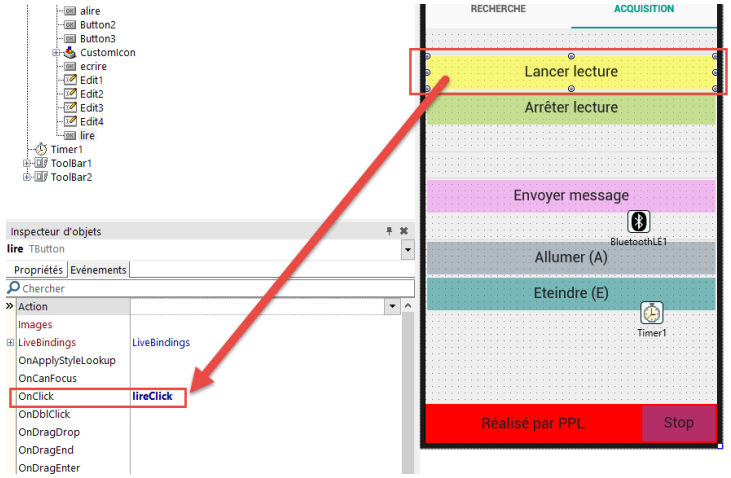
- Tester le programme

- Quand on a obtenu la caractéristique en utilisant GetCharacteristic, on peut lire ou écrire (le cas échéant) les informations contenues. Le gestionnaire d'événement OnCharacteristicRead est déclenché après la lecture de la caractéristique.

Pour obtenir la valeur en cours d'une caractéristique, on utilise la méthode ReadCharacteristic.
Utilisez WriteCharacteristic pour écrire si cette option est disponible sur le serveur.

- Réaliser la fonction suivante pour lancer la lecture

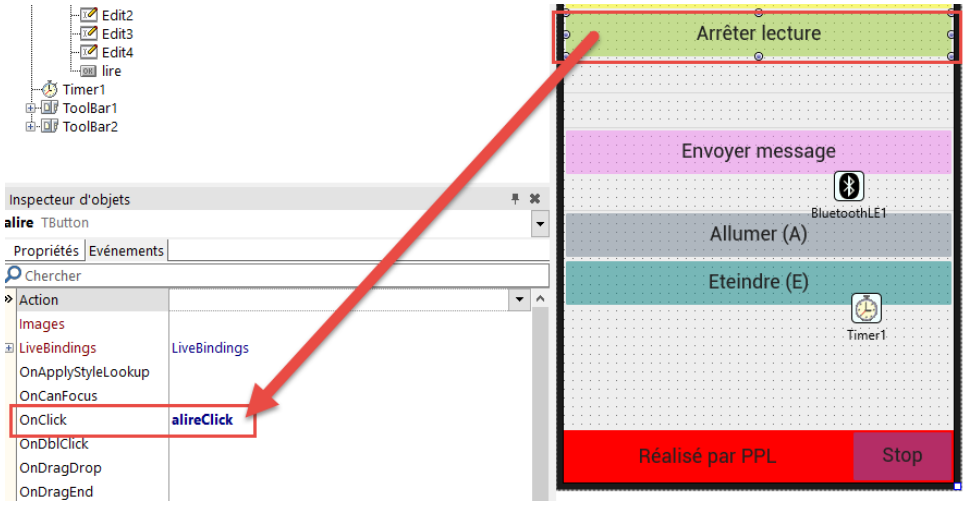
Nom : ……. Prénom : …….



- Ecrire la ligne de programme suivante

```
void __fastcall TForm2::lireClick(TObject *Sender)
{
this->Timer1->Enabled=true;       // lance le timer pour la lecture
}
//---------------------------------------------------------------------------
```

- Réaliser la fonction suivante pour arrêter la lecture

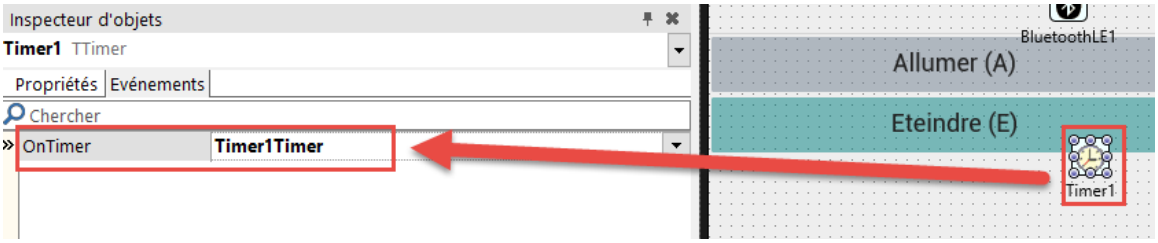


- Ecrire la ligne de programme suivante

Nom : ……. Prénom : …….

```
void __fastcall TForm2::alireClick(TObject *Sender)
{
 this->Timer1->Enabled=false;  //arrête la lecture
}
//---------------------------------------------------------------------------
```

- Réaliser la fonction suivante pour réaliser la lecture

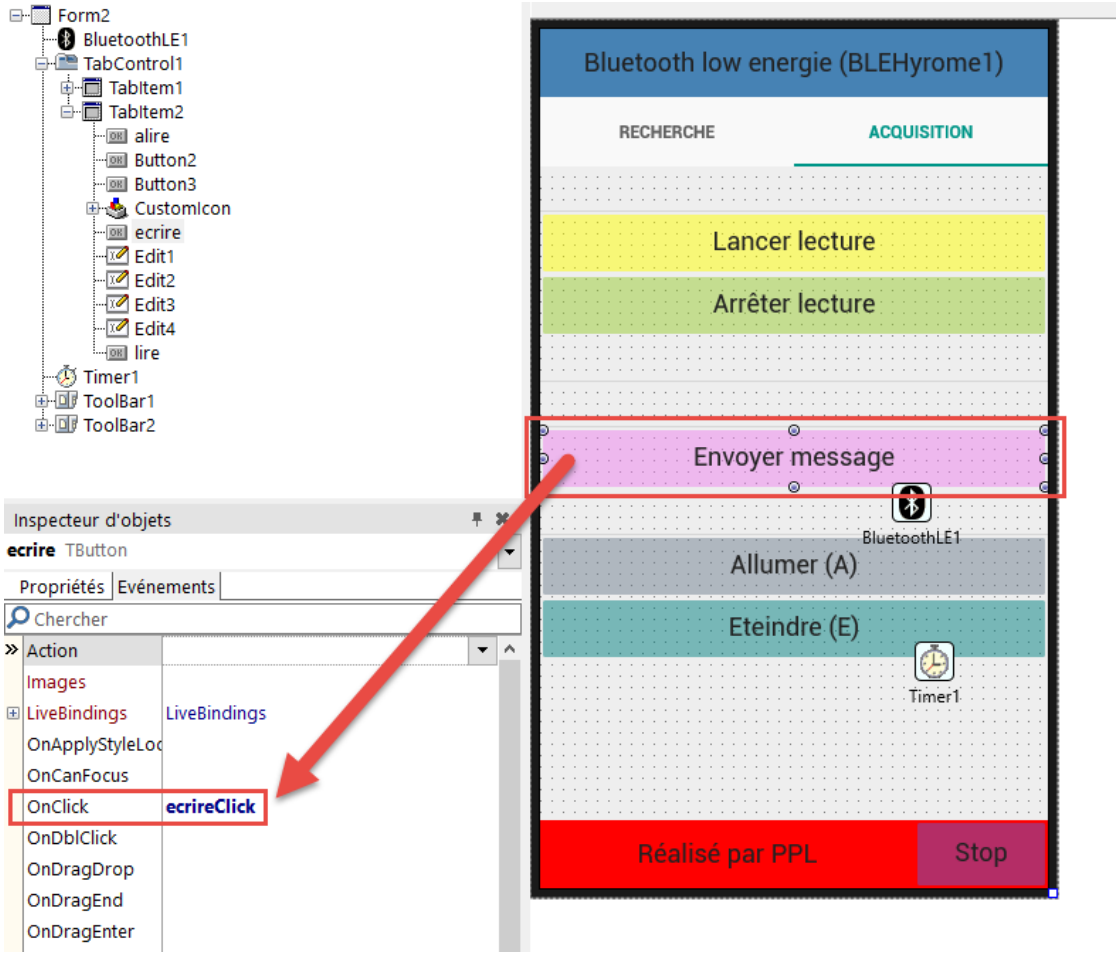


- Ecrire la ligne de programme suivante

```
void __fastcall TForm2::Timer1Timer(TObject *Sender)
{
if(FBLEDevice != NULL) {
        BluetoothLE1->GetCharacteristics(FBLEGattServiceL);   //obtient la liste des caractéristiques du service voulu
        FBLEGattCharL = BluetoothLE1->GetCharacteristic(FBLEGattServiceL, HMSoft_CHARACTERISTIC1 );// obtient la caractéristique correspondante
          BluetoothLE1->SubscribeToCharacteristic(FBLEDevice, FBLEGattCharL); //obtient l'information quand elle change
       if( FBLEGattCharL->Properties.Contains(TBluetoothProperty::Notify))  //contrôle la propriété
        {


            FBLEDevice->ReadCharacteristic(FBLEGattCharL);  //lit la valeur
        Edit4->Text=FBLEGattCharL->GetValueAsString();  //affiche la valeur
        }
        else {
            ShowMessage("This characteristic doesn''t allow Read");
        }
    }
    else {
        ShowMessage( " is not available");
    }
}
//---------------------------------------------------------------------------
```

- Réaliser la fonction suivante pour envoyer du texte

Nom : ……. Prénom : …….



- Ecrire la ligne de programme suivante

```
void __fastcall TForm2::ecrireClick(TObject *Sender)
{
    if(FBLEDevice != NULL) {
         BluetoothLE1->GetCharacteristics(FBLEGattService);
         FBLEGattChar = BluetoothLE1->GetCharacteristic(FBLEGattService, HMSoft_CHARACTERISTIC );
        if(( FBLEGattChar->Properties.Contains(TBluetoothProperty::Write)) || ( FBLEGattChar->Properties.Contains(TBluetoothProperty::WriteNoResponse))
            || ( FBLEGattChar->Properties.Contains(TBluetoothProperty::SignedWrite)))
        {
        AnsiString Value;
                Value=this->Edit2->Text;
        for (i = 0; i <Value.Length(); i++) {
        FBLEGattChar->SetValueAsString(Value[i]);  //récupère le texte écrit (format string)

            FBLEDevice->WriteCharacteristic(FBLEGattChar);  //envoie le texte
                                }

        }
        else {
            ShowMessage("This characteristic doesn''t allow Write");
        }
    }
    else {
        ShowMessage( " is not available");
    }
}
//---------------------------------------------------------------------------
```

- Tester le programme
  - Lancer la lecture
  - Envoyer la lettre A. Que se passe t ’il au niveau de la led et au niveau de l’application Android

  - Envoyer la lettre E. Que se passe t ’il au niveau de la led et au niveau de l’application Android

Nom : ……. Prénom : …….

- Terminer le programme Android afin que lorsqu'on appuie sur le bouton allumer(A), la led s'allume et lorsqu'on appuie sur le bouton allumer(E), la led s'éteint.
- Terminer le programme Android afin que lorsqu'on appuie sur le bouton Stop l'application se ferme correctement.
- Modifier le programme Arduino pour envoyer la valeur du potentiomètre lorsqu'on appuie sur le bouton

Remarque :

Pour cela on utilisera une variable

char buf[] = "1023";

https://www.arduino.cc/en/Reference/Char

Cette variable va permettre de récupérer la valeur du potentiomètre. Malheureusement, la valeur du potentiomètre est une valeur entière « int ». Donc il faudra transformer la variable « int » en « char ». On va donc utiliser la fonction sprintf()

Exemple :

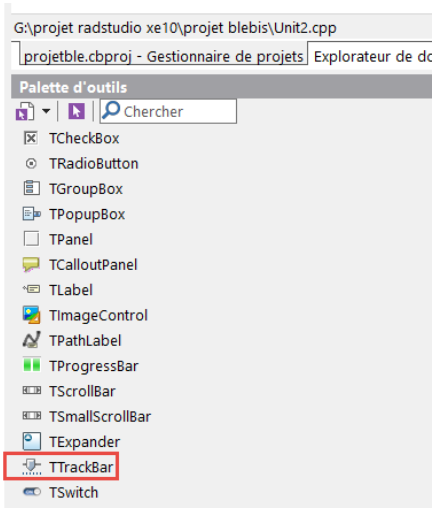
```
float temperature = 15.2;
char message[20]; //taille max de 20 caractère pour le message, pour l'exemple
sprintf(message, "%f", temperature); //comme un printf mais dans une chaine !
```

On utiliser %d pour une valeur entière

Voir vidéo

http://www.sti2dsinhyrome.fr/video%20tp%20BLE%20.html

- Rajouter un Slider sur le programme Android et modifier le programme Arduino afin de commander sur la carte Arduino la vitesse de rotation d'un moteur à courant continu.



Nom : ……. Prénom : …….

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Programme Arduino

```
#include <SPI.h>

#include <EEPROM.h>

#include <boards.h>

#include <RBL_nRF8001.h>



#define DIGITAL_OUT_PIN1    10 //commande led

#define DIGITAL_OUT_PIN2    11 //commande moteur

String texte;

String texte1;

int valeur;

void setup()

{
```

```
  // Default pins set to 9 and 8 for REQN and RDYN

  // Set your REQN and RDYN here before ble_begin() if you need

  //ble_set_pins(3, 2);


  // Set your BLE Shield name here, max. length 10

   ble_set_name("BLEHyrome1");


  // Init. and start BLE library.

  ble_begin();


  // Enable serial debug

  Serial.begin(9600);


  pinMode(DIGITAL_OUT_PIN1, OUTPUT);

   pinMode(DIGITAL_OUT_PIN2, OUTPUT);


}


void loop()

{


  // If data is ready

  while(ble_available())

  {

    // read out command and data

    byte data0 = ble_read();

    byte data1 = ble_read();

    byte data2 = ble_read();

     texte=data1;
```

```
    texte1=data2;


     Serial.write(data0);
     Serial.write(data1);
     Serial.write(data2);



  if (data0 == 65)  // commande led
  {
    Serial.println("ok");
    if (data1 ==49)
    {
      digitalWrite(DIGITAL_OUT_PIN1, HIGH);
    }
   if (data1 ==48)
    {
      digitalWrite(DIGITAL_OUT_PIN1, LOW);
    }
  }
 if (data0 == 70 && data2==255)  // commande moteur
  {


int myStringLength = texte.length()+1;


 char myChar[myStringLength];
 texte.toCharArray(myChar,myStringLength);


 int valeur = atoi(myChar);
```

```
    valeur=(valeur-48)*255/10;
        Serial.println(valeur);
    analogWrite(DIGITAL_OUT_PIN2, valeur);

  }
  if (data0 == 70 && data2!=255)  // Commande moteur
  {

int myStringLength = texte.length()+1;

 char myChar[myStringLength];
 texte.toCharArray(myChar,myStringLength);

 int valeur = atoi(myChar);
 int myStringLength1 = texte1.length()+1;

 char myChar1[myStringLength1];
 texte1.toCharArray(myChar1,myStringLength1);

 int valeur1 = atoi(myChar1);
    valeur=((valeur-48)*10+(valeur1-48))*255/10;
        Serial.println(valeur);
           analogWrite(DIGITAL_OUT_PIN2, valeur);

  }
 }
 // Allow BLE Shield to send/receive data
 ble_do_events();
}
```

Nom : ……. Prénom : …….

- Réaliser le branchement sur fritzing avec transistor de puissance, diode de roue libre etc

Programme C++ Builder

Envoyer « A1 » ou « A0 » pour allumer ou éteindre la lampe et envoyer « F » suivi de la valeur du Slider de 0 à 10 pour faire varier la vitesse du moteur

Attention pour cette partie, il faudra envoyer le mot complet et non lettre par lettre

```
if(FBLEDevice != NULL) {
     BluetoothLE1->GetCharacteristics(FBLEGattService);
     FBLEGattChar = BluetoothLE1->GetCharacteristic(FBLEGattService, HMSoft_CHARACTERISTIC );
    if(( FBLEGattChar->Properties.Contains(TBluetoothProperty::Write)) || ( FBLEGattChar->Properties.Contains(TBluetoo
        || ( FBLEGattChar->Properties.Contains(TBluetoothProperty::SignedWrite)))
    {
    AnsiString Value;
            Value=this->Edit2->Text;

    FBLEGattChar->SetValueAsString(this->Edit2->Text);  //récupère le texte écrit (format string)

        FBLEDevice->WriteCharacteristic(FBLEGattChar);  //envoie le texte


    }
    else {
        ShowMessage("This characteristic doesn''t allow Write");
    }
}
else {
    ShowMessage( " is not available");
}
```

- En vous aidant du tableau ci-dessus expliquer le programme Arduino

Nom : ……. Prénom : …….

- Modifier le programme Arduino afin d'envoyer la valeur de commande du moteur (0 à 255) au téléphone chaque fois que le Shield BLE reçoit un message du téléphone pour faire varier la vitesse.
- Rajouter votre programme au TP

Voir vidéo

http://www.sti2dsinhyrome.fr/video tp BLE commande moteur.html

- Réaliser le montage et le tester