

Nom :

Prénom :

TP SIN

Etude liaison bluetooth Beacon

Pré requis (l'élève doit savoir):

Savoir utiliser un ordinateur

Réaliser un programme sur C++ Builder

Programme

Objectif terminal:

L'élève doit être capable de se connecter à une balise Beacon

Matériel

- Ordinateur
- Carte arduino yun
- Module bluetooth V4.0 USB
- Putty : <http://www.putty.org/>
- Module Beacon Social Retail

<http://www.ibeaconstore.fr/>



A propos du Beacon Social Retail

Très léger avec ses 21 grammes pile comprise, le Beacon Social Retail se faufile partout.

Notre Beacon convient aussi bien aux professionnels puisqu'il reste un boîtier programmable. Vous pouvez donc paramétrer les UDID à votre convenance en fonction des besoins de vos clients et protéger l'accès par un mot de passe.

Contenu du coffret :

- ▶ 1 borne iBeacon
- ▶ 1 pile CR2477
- ▶ Bande adhésive de fixation

L'utilisation du Beacon Social Retail n'a de limite que celle de votre imagination !

L'application fournie avec note Beacon (compatible à partir de iOS 7 et Android 4.3) permet de localiser tout objet où personne dans un rayon de 50 à 90 mètres autour de vous.

N'hésitez pas à prendre contact avec nous pour des quantités supérieures à 50 pièces car nous pouvons en usine, réaliser pour vous les opérations de programmation telles que les informations Major, Minor, Device Name...

Travail demandé

- Deux possibilités :

Soit vous utiliser les modules Beacons, ou l'Arduino Yun

- Paramétrage Beacon

Nom :

Prénom :

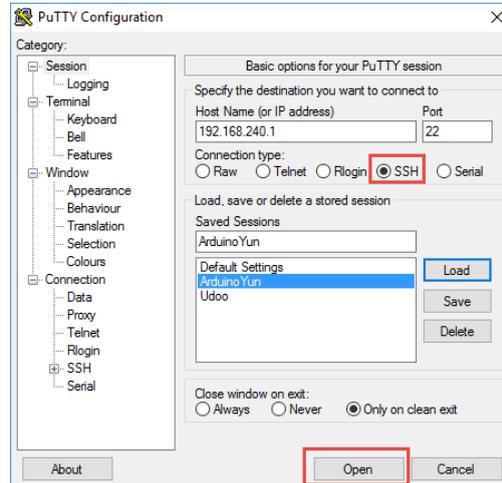
<http://www.digitalsocialretail.com/getstarted/>

- Paramétrage de l'Arduino Yun

Lien : <http://fibasile.github.io/arduino-yun-ibeacon.html>

- Configuration IBeacon à partir de la ligne de commande

Démarrer le logiciel Putty et connectez-vous à la carte Arduino.



Tout d'abord, nous vérifions si le périphérique est reconnu au port USB :

```
root@Arduino:~# lsusb
```

Le résultat se présente comme cela :

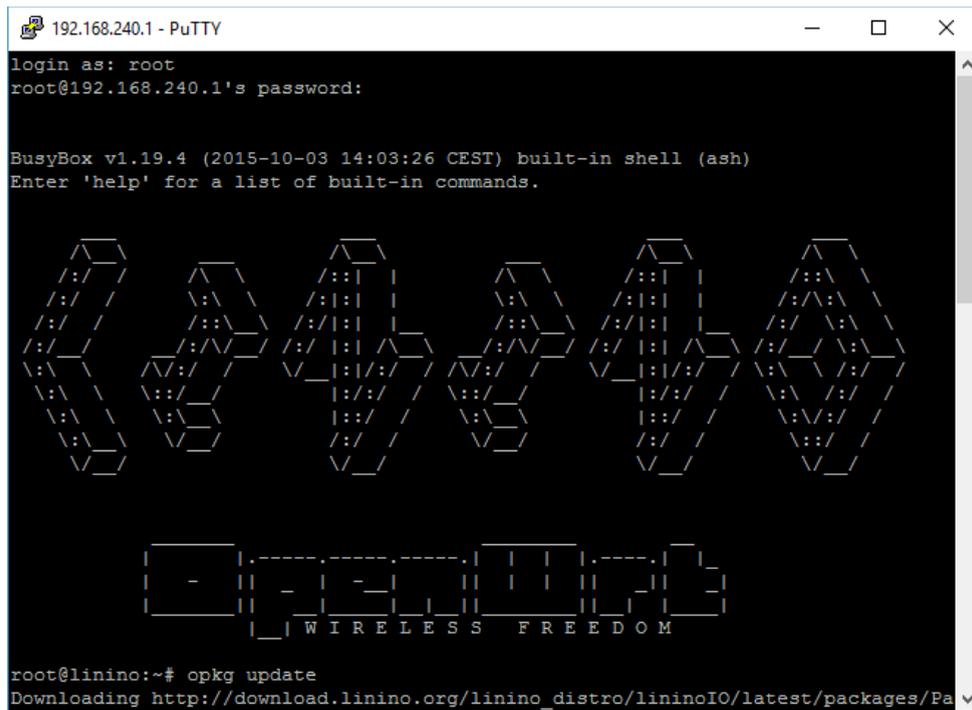
```
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 002: ID 058f:6254 Alcor Micro Corp. USB Hub
Bus 001 Device 005: ID 0a12:0001 Cambridge Silicon Radio, Ltd Bluetooth Dongle (HCI mode)
Bus 001 Device 004: ID 058f:6366 Alcor Micro Corp. Multi Flash Reader
```

Maintenant nous avons besoin d'installer le package bluetooth

```
root@Arduino:~# opkg update
root@Arduino:~# opkg install kmod-bluetooth
```

Nom :

Prénom :



```
192.168.240.1 - PuTTY
login as: root
root@192.168.240.1's password:

BusyBox v1.19.4 (2015-10-03 14:03:26 CEST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

root@linino:~# opkg update
Downloading http://download.linino.org/linino_distro/lininoIO/latest/packages/Pa
```

Puis installer le package suivant

```
root@Arduino:~# opkg install http://fibasile.github.io/images/arduino-yun-ibeacon/bluez_5.13-1_ar71xx.ipk
```

Tester la liaison :

```
root@Arduino:~# hciconfig
hci0: Type: BR/EDR Bus: USB
      BD Address: 00:1A:7D:DA:71:09 ACL MTU: 310:10 SCO MTU: 64:8
      UP RUNNING
      RX bytes:1002 acl:0 sco:0 events:44 errors:0
      TX bytes:233 acl:0 sco:0 commands:44 errors:0
```

Testons l'interface hci0:

```
root@Arduino:~# hciconfig hci0 up
root@Arduino:~# hciconfig
hci0: Type: BR/EDR Bus: USB
      BD Address: 00:1A:7D:DA:71:09 ACL MTU: 310:10 SCO MTU: 64:8
      UP RUNNING
      RX bytes:1002 acl:0 sco:0 events:44 errors:0
      TX bytes:233 acl:0 sco:0 commands:44 errors:0
```

Nom :

Prénom :

Nous allons rentrer les paramètres du Beacon:

```
root@Arduino:~# hcitool -i hci0 cmd 0x08 0x0008 1E 02 01 1A 1A FF 4C 00 02 15 <UUID> <MAJOR> <MINOR> <POWER> > 00
```

Nous allons d'abord générer un UUID, pour cela on va installer le package suivant :

```
root@Arduino:~# opkg install uuidgen
root@Arduino:~# uuidgen
d28c3c18-945a-49b5-809f-d5c09ba095e6
```

Une fois obtenu un UUID, nous allons l'écrire sur la carte avec un Major et Minor égale à 0 :

```
root@Arduino:~# hciconfig hci0 noleadv
root@Arduino:~# hcitool -i hci0 cmd 0x08 0x0008 1E 02 01 1A 1A FF 4C 00 02 15 D2 8C 3C 18 94 5A 49 B5 80 9F D5 C0 9
B A0 95 E6 00 00 00 00 C9 00
< HCI Command: ocf 0x08, ocf 0x0008, plen 32
1E 02 01 1A 1A FF 4C 00 02 15 D2 8C 3C 18 94 5A 49 B5 80 9F
D5 C0 9B A0 95 E6 00 00 00 00 C9 00
> HCI Event: 0x0e plen 4
01 08 20 00
```

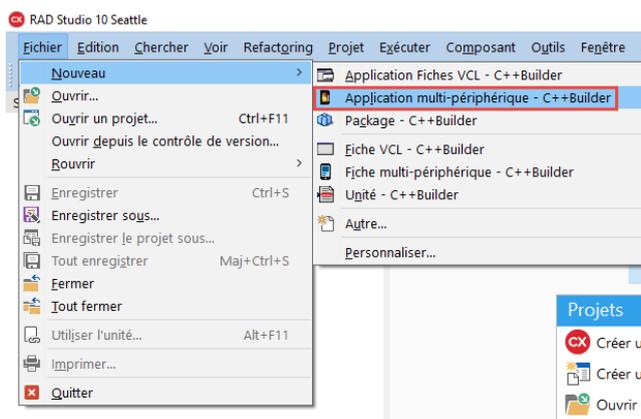
On lance la balise

```
root@Arduino:~# hciconfig hci0 leadv
```

Pour l'arrêter.

```
root@Arduino:~# hciconfig hci0 noscan
```

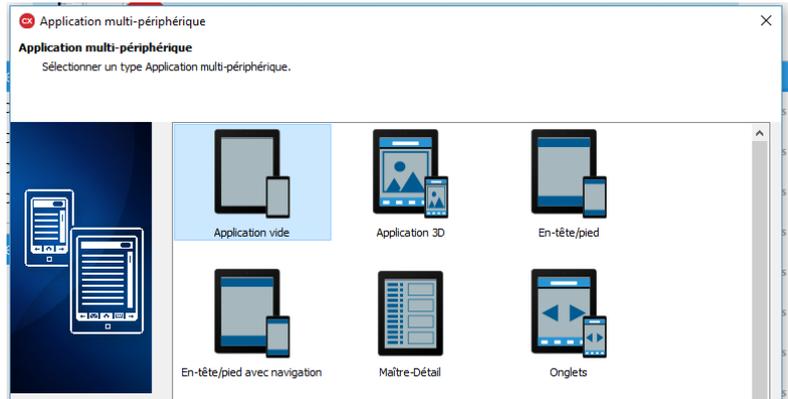
- Création de l'application Android
- Démarrer C++ builder et créer nouvelle application multi-périphérique



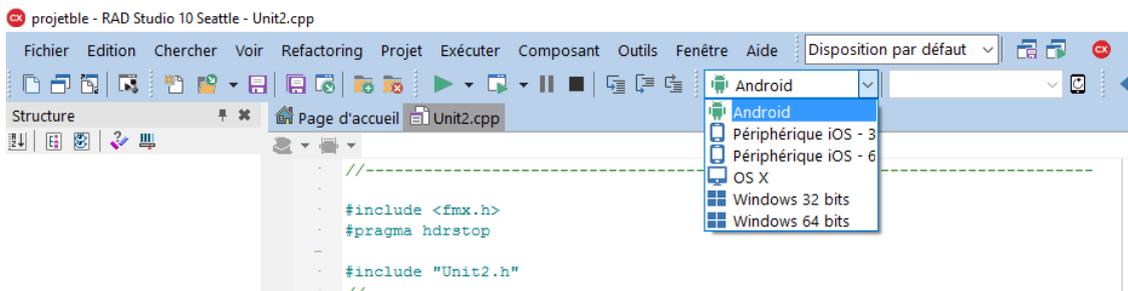
Nom :

Prénom :

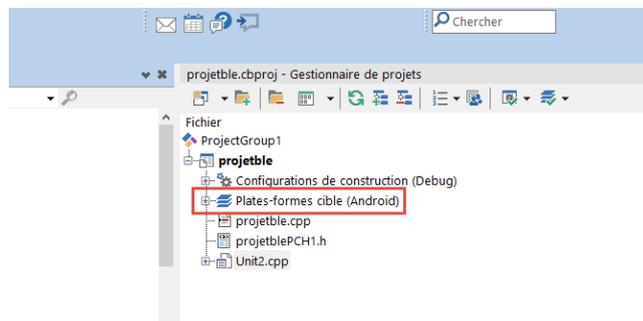
- Puis sélectionner application vide



- Sélectionner affichage android



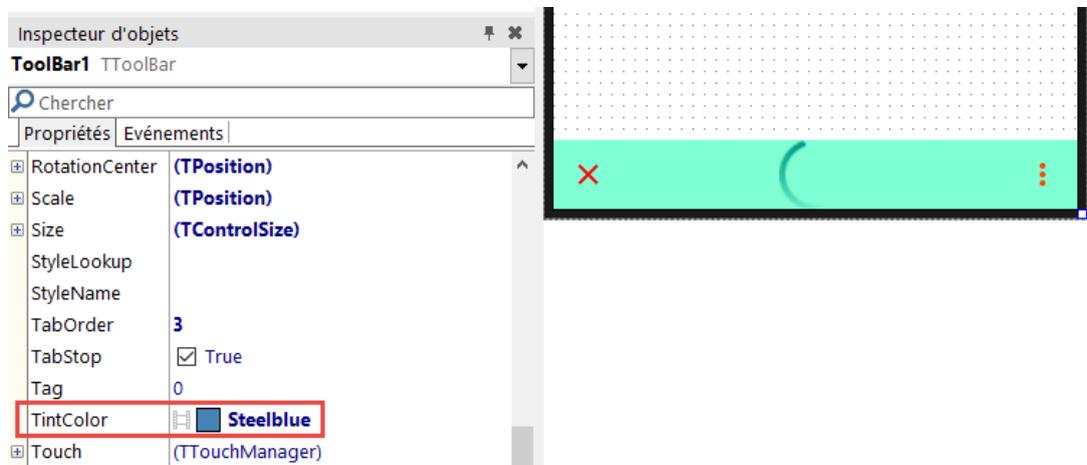
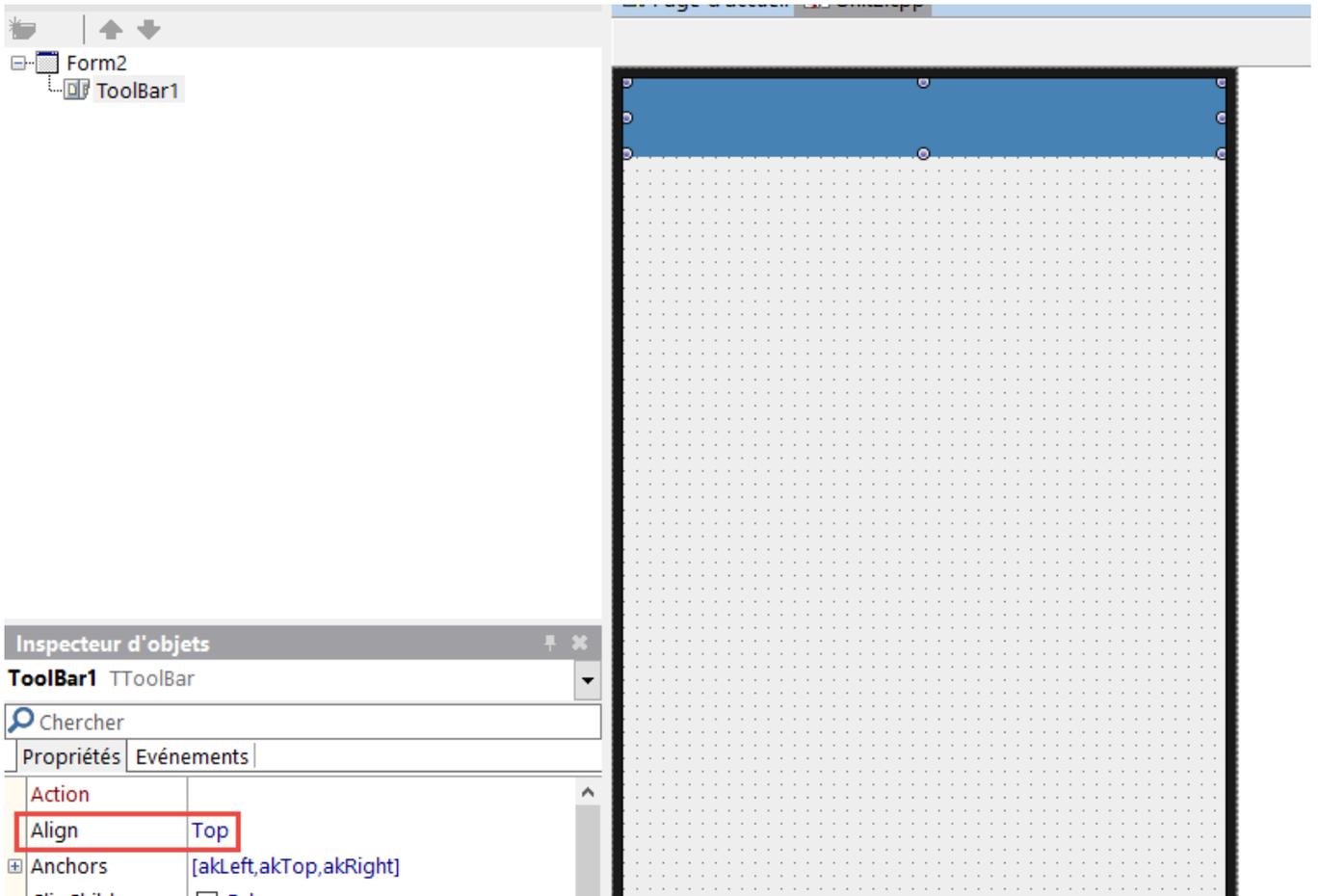
- Sélectionner plates-formes cible (Android)



- On va créer la présentation suivante
 - Installer un premier ToolBar

Nom :

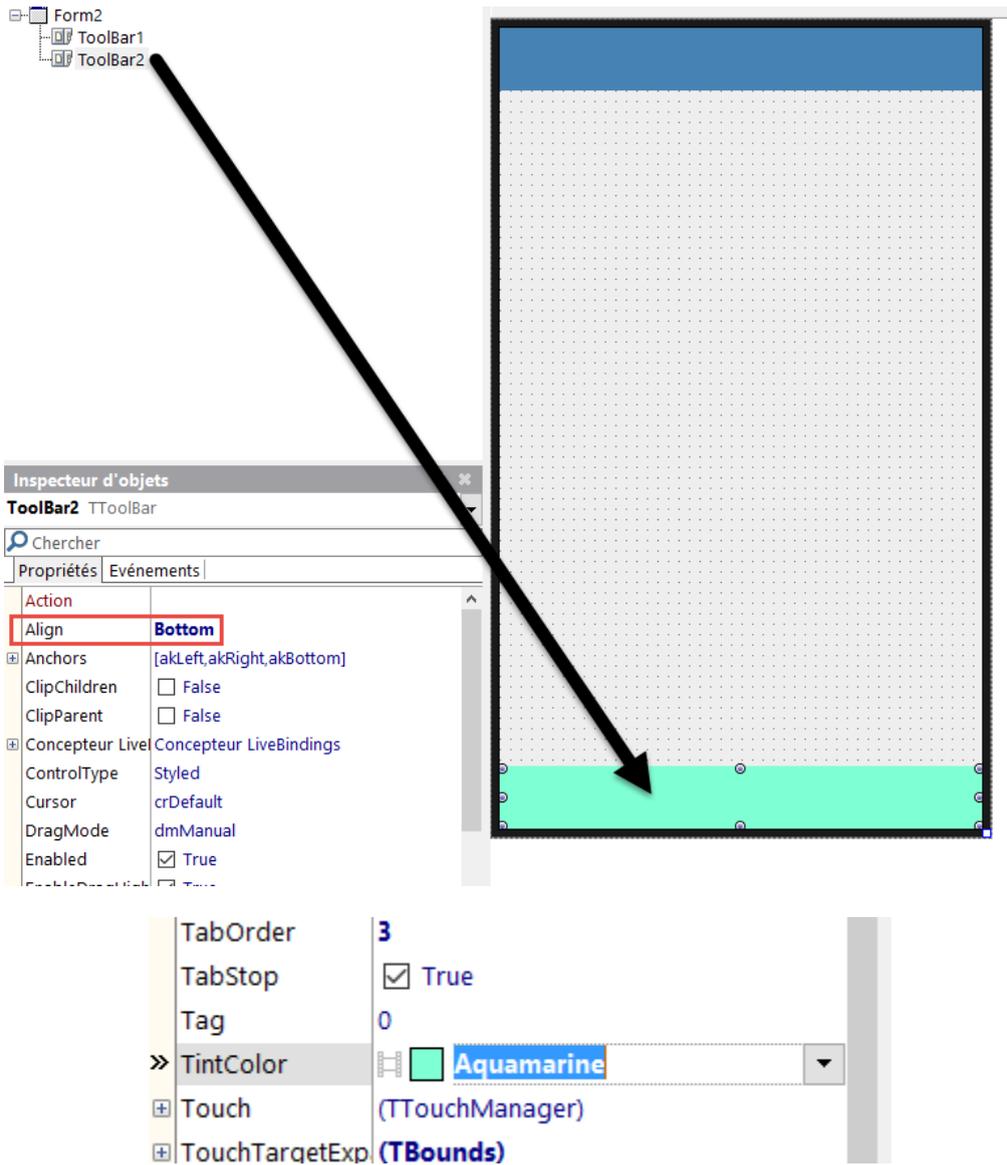
Prénom :



- Installer un deuxième ToolBar

Nom :

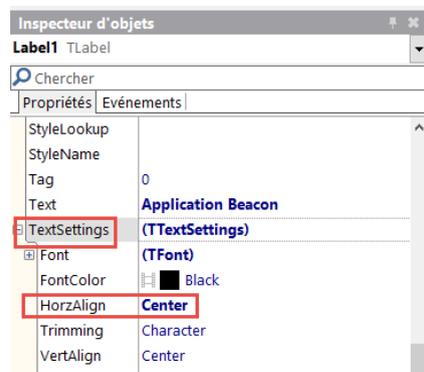
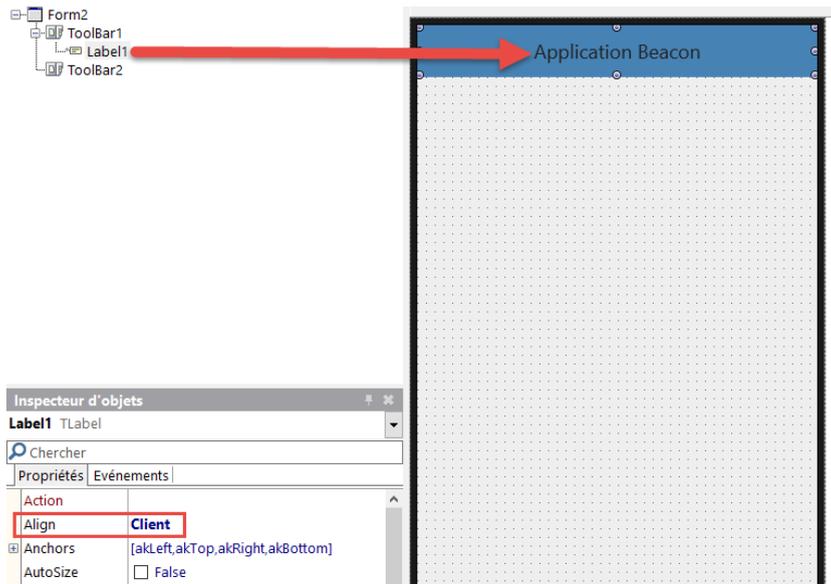
Prénom :



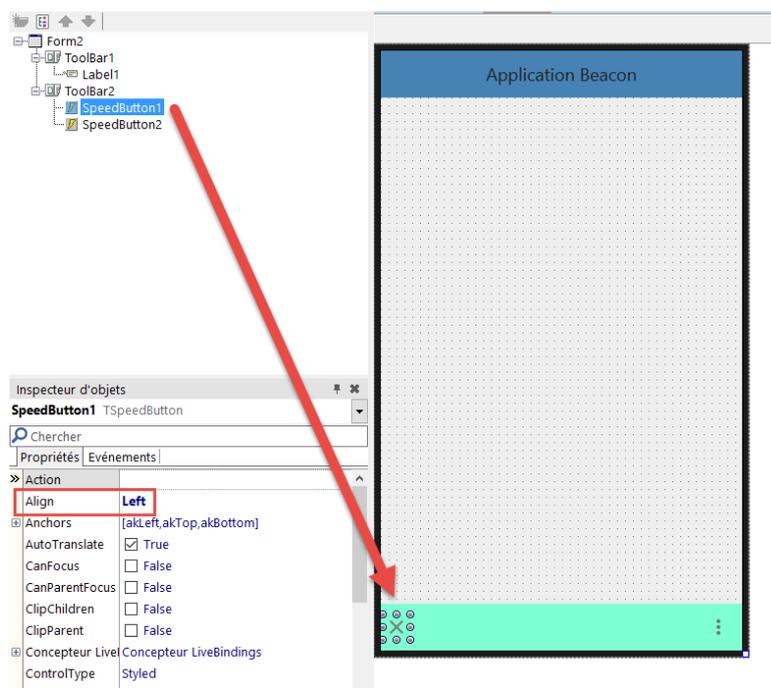
- Installer un Label dans le premier ToolBar

Nom :

Prénom :

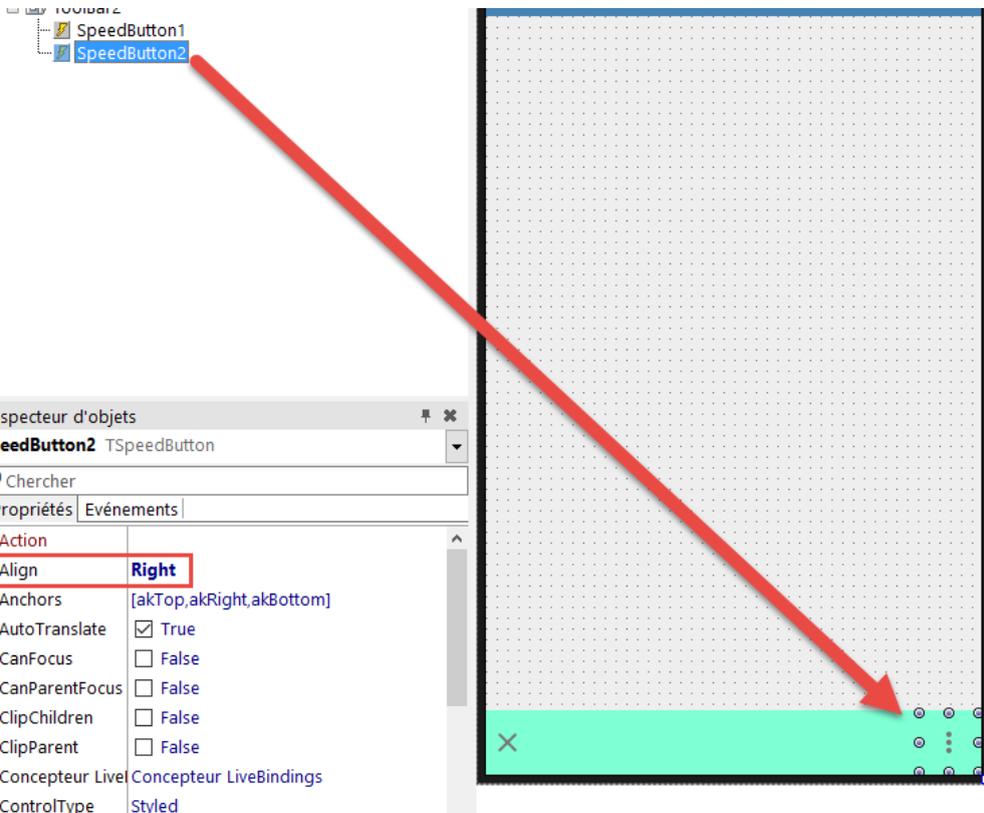
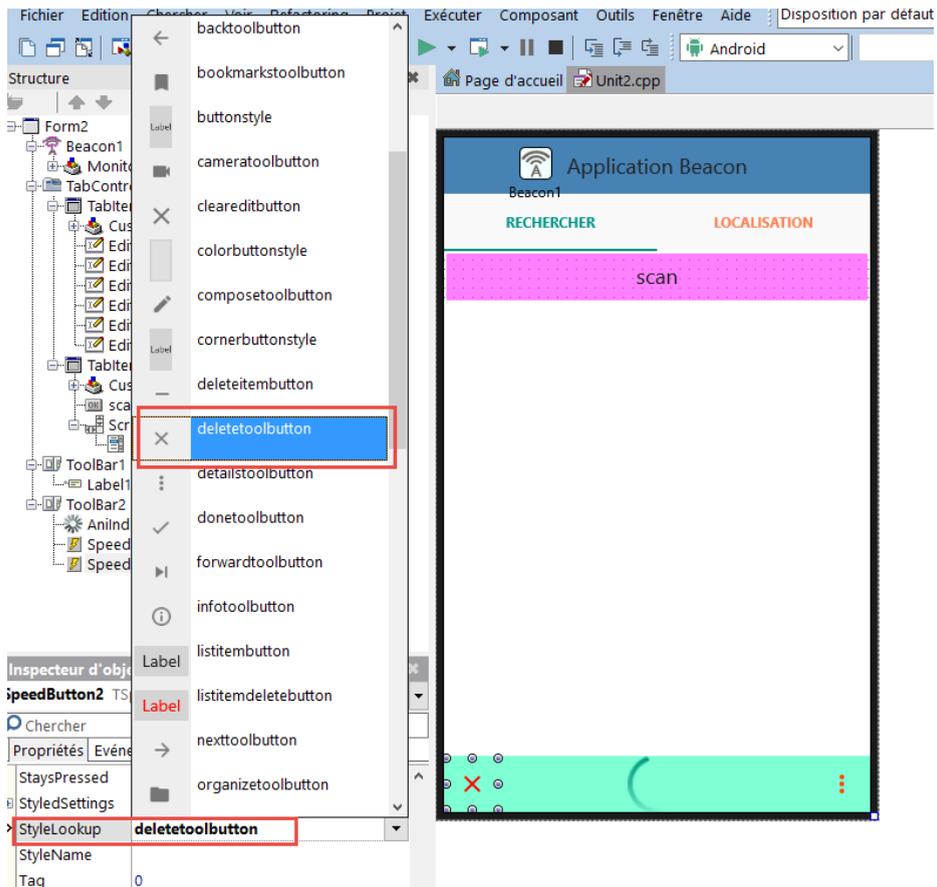


- Dans le deuxième toolBar, rajouter deux speedButton



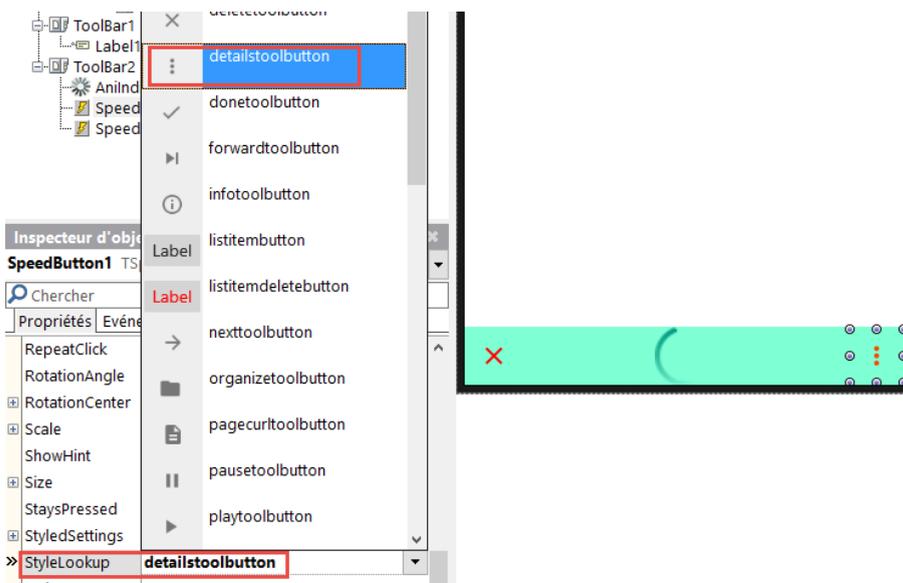
Nom :

Prénom :

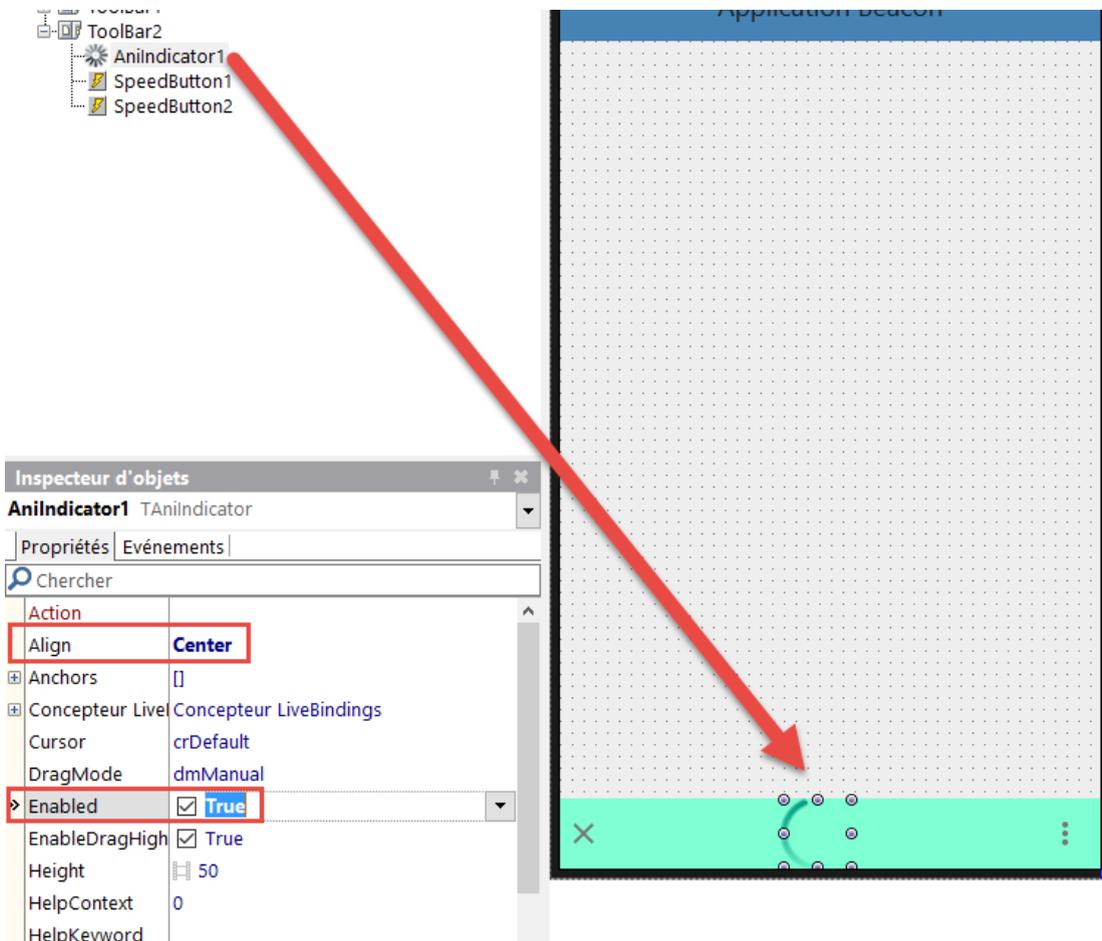


Nom :

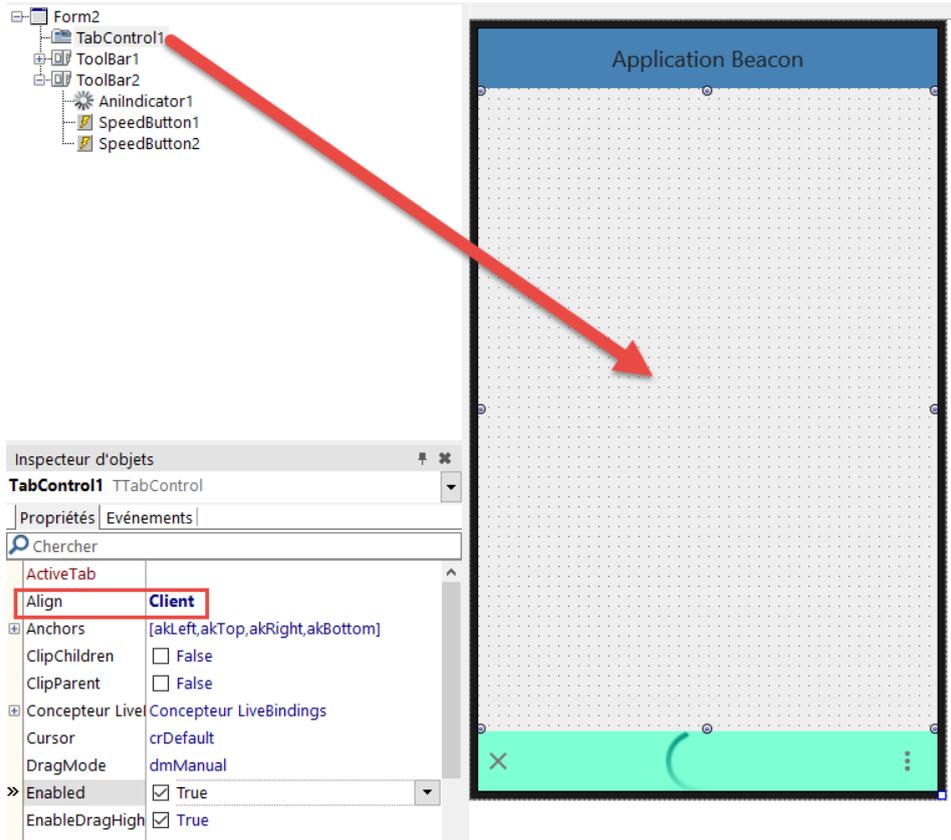
Prénom :



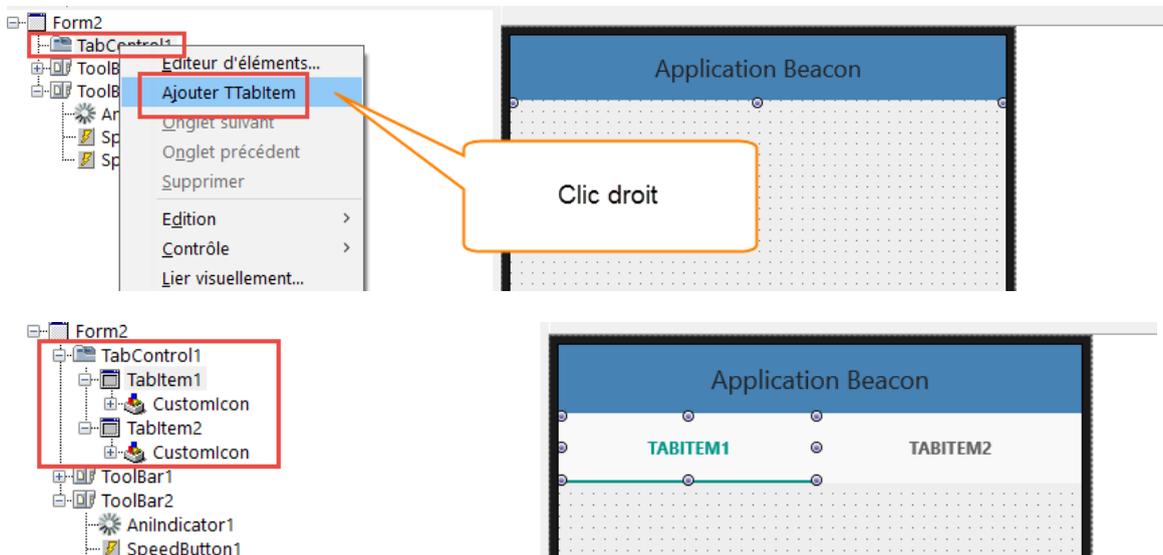
- Toujours dans le deuxième ToolBar, rajouter un Anilndicator



- Rajouter un TabControl au centre du téléphone



- Au niveau du TabControl, insérer deux TabItem



Nom :

Prénom :

Form2

- TabControl1
 - TabItem1
 - CustomIcon
 - TabItem2
 - CustomIcon
 - ToolBar1
 - ToolBar2
 - AnIndicator1
 - SpeedButton1
 - SpeedButton2

Inspecteur d'objets

TabItem1 TTabItem

Propriétés | Événements

Chercher

Padding (TBounds)

ParentShowHint True

PopupMenu

Scale (TPosition)

ShowHint False

Size (TControlSize)

StyledSettings [Family,Size,Style,FontColor]

StyleLookup

StyleName

TabOrder 0

TabStop True

Tag 0

Text **rechercher**

Application Beacon

RECHERCHER LOCALISATION

TabControl1

- TabItem1
 - CustomIcon
 - TabItem2
 - CustomIcon
- ToolBar1
- ToolBar2
 - AnIndicator1
 - SpeedButton1
 - SpeedButton2

Inspecteur d'objets

TabItem2 TTabItem

Propriétés | Événements

Chercher

StyledSettings [Family,Size,Style]

StyleLookup

StyleName

TabOrder 0

TabStop True

Tag 0

Text **localisation**

TextSettings (TTextSettings)

Font (TFont)

FontColor **Coral**

HorzAlign Center

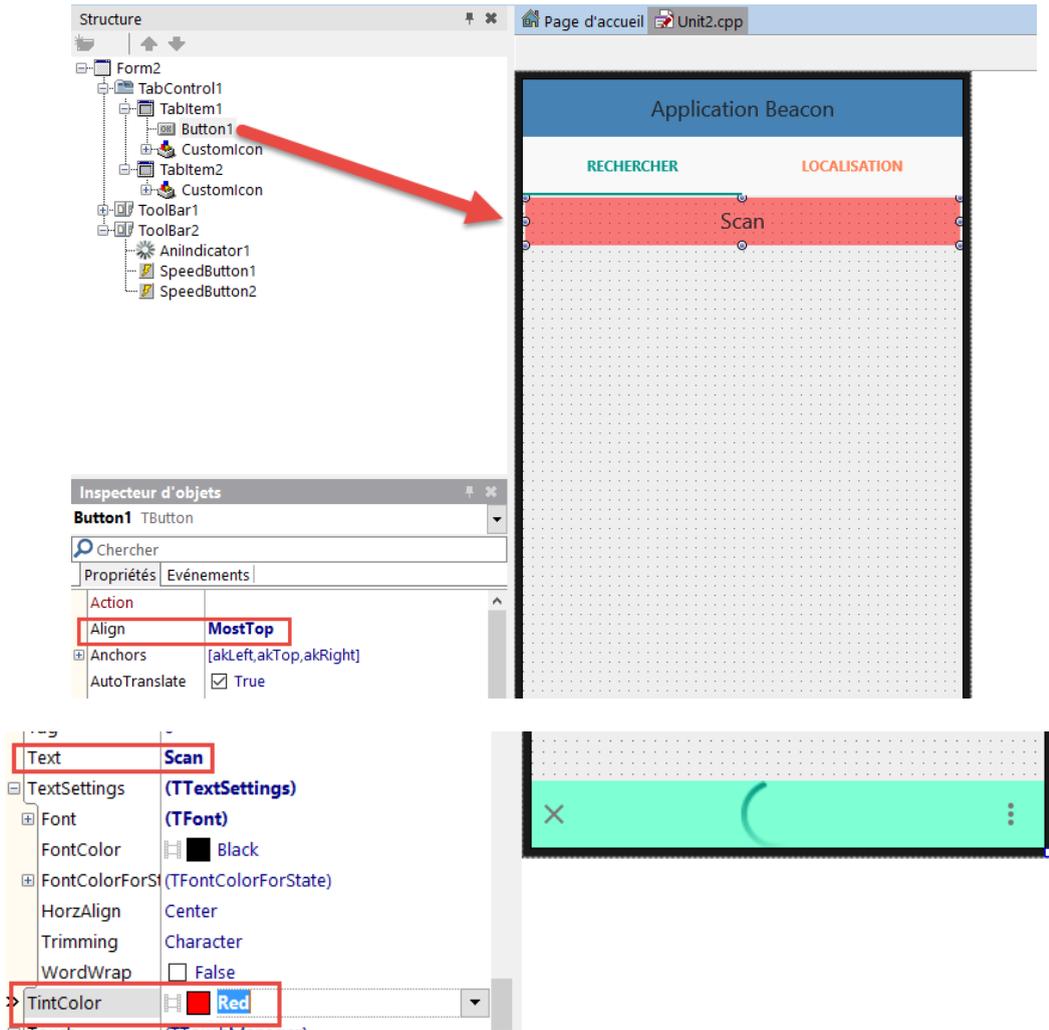
Application Beacon

RECHERCHER LOCALISATION

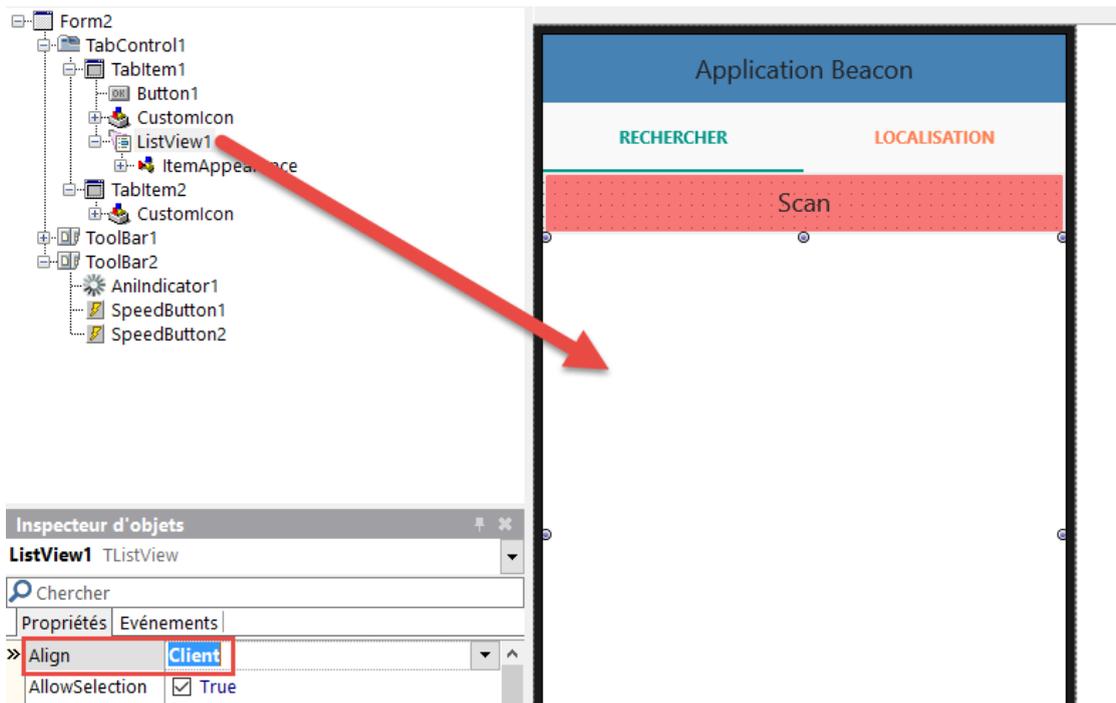
Nom :

Prénom :

- Dans le premier TabItem rajouter un bouton



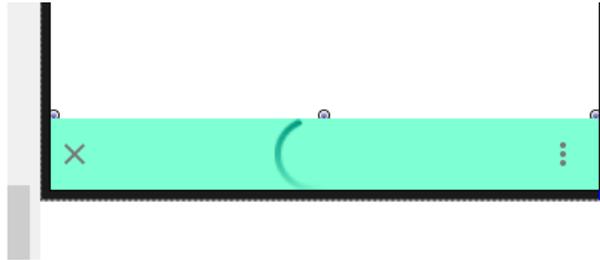
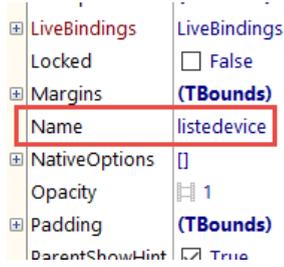
- Puis rajouter un ListView



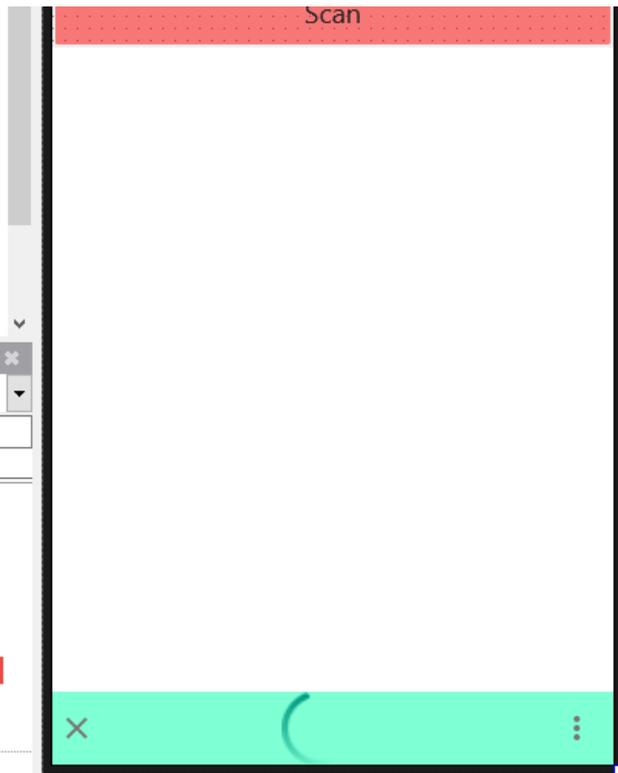
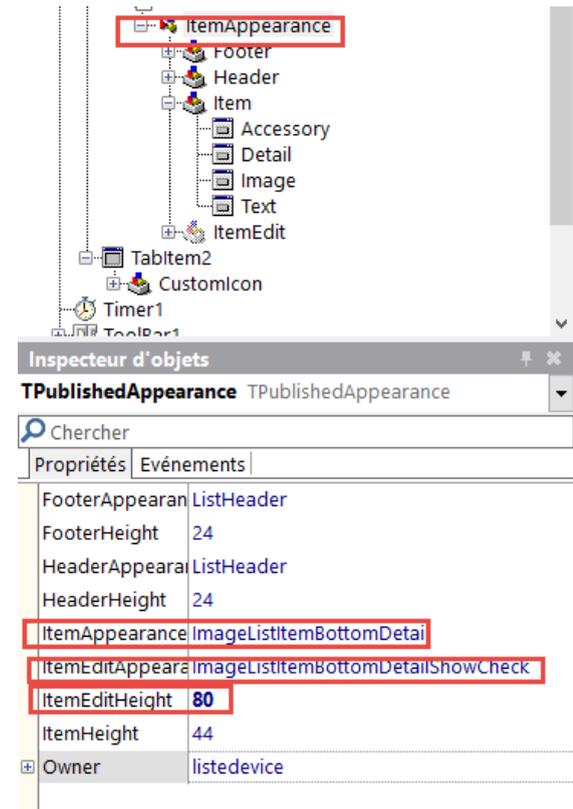
Nom :

Prénom :

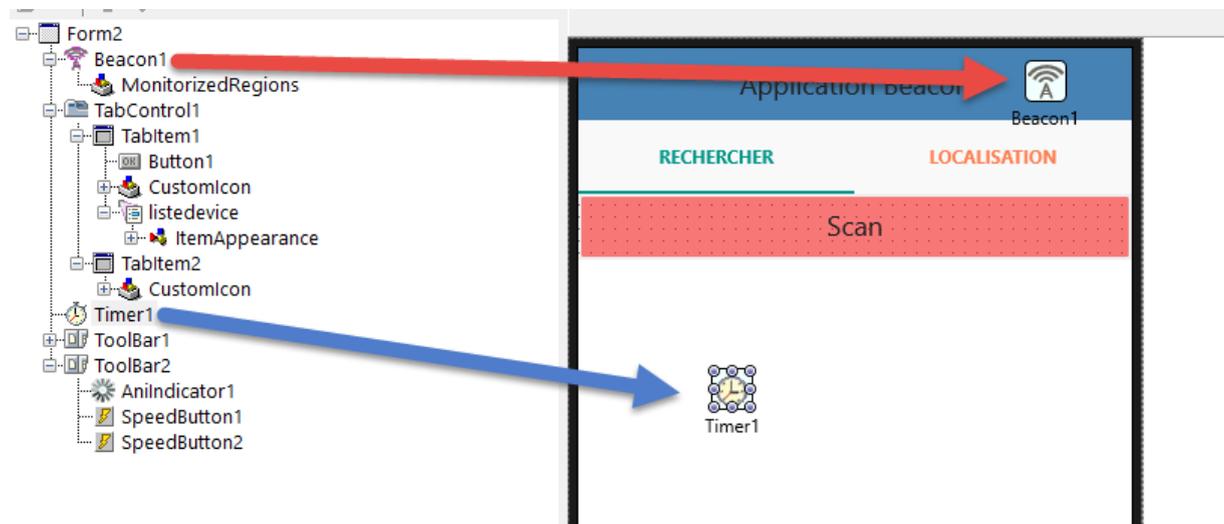
- Changer le nom



- On va modifier l'apparence du listview



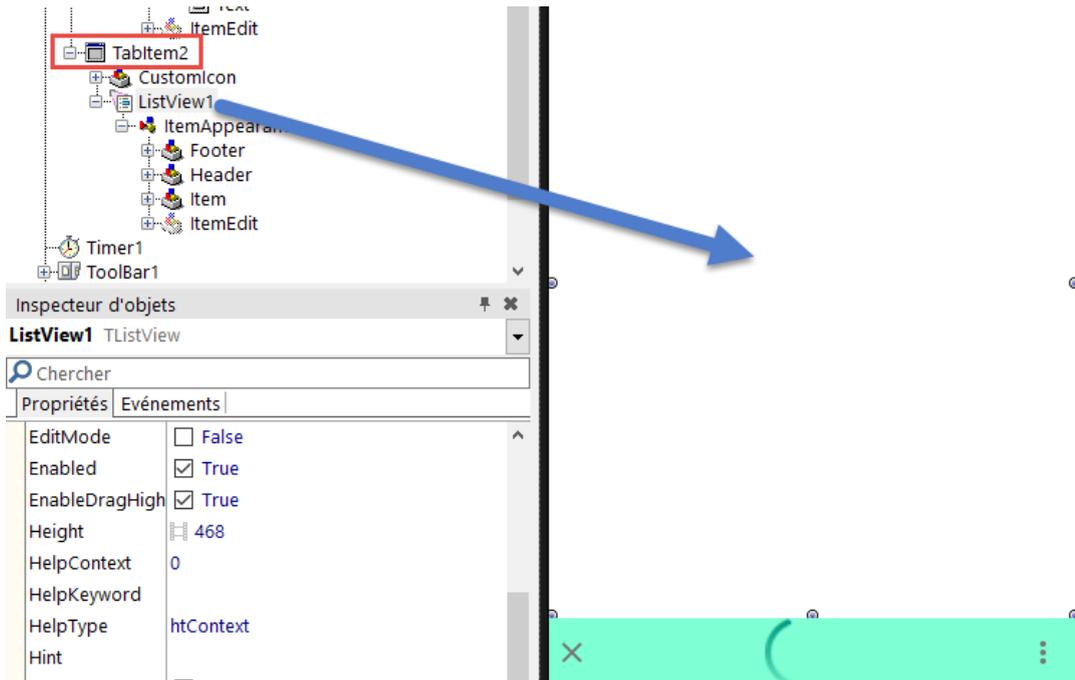
- Rajouter un TBeacon et un TTimer



Nom :

Prénom :

- Dans le deuxième TabItem, on va rajouter un autre ListView en le paramétrant comme le premier



- Paramètres du Beacon

Mode d'emploi des beacons

L'API RTL des beacons vous permet d'obtenir la distance entre le périphérique qui utilise votre application et les beacons environnants. L'API RTL des beacons fournit des informations uniquement pour les beacons se trouvant dans la région définie. Une région est un groupe de beacons. Vous pouvez spécifier une région avec ses propriétés UUID, MajorID, MinorID (pour iBeacons et AltBeacons) ou avec ses identifiants Namespace et Instance (pour Eddystone).

Obtention d'une instance de TBeaconManager

Pour utiliser l'API RTL des beacons, vous devez inclure l'unité System.Beacon dans votre application, puis appeler TBeaconManager.GetBeaconManager pour le TBeaconScanMode qui sera utilisé pour obtenir une instance de TBeaconManager.

Pour iBeacons :

```
BeaconManager := TBeaconManager.GetBeaconManager(TBeaconScanMode.Standard);
```

Pour AltBeacons :

```
BeaconManager := TBeaconManager.GetBeaconManager(TBeaconScanMode.Alternative);
```

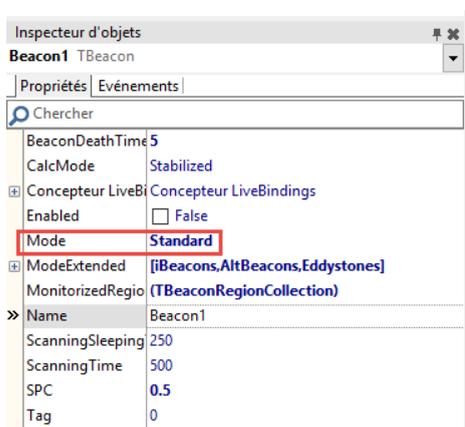
Pour Eddystone :

```
BeaconManager := TBeaconManager.GetBeaconManager(TBeaconScanMode.Eddystone);
```

Pour Extended :

```
BeaconManager := TBeaconManager.GetBeaconManager(TBeaconScanMode.Extended);
```

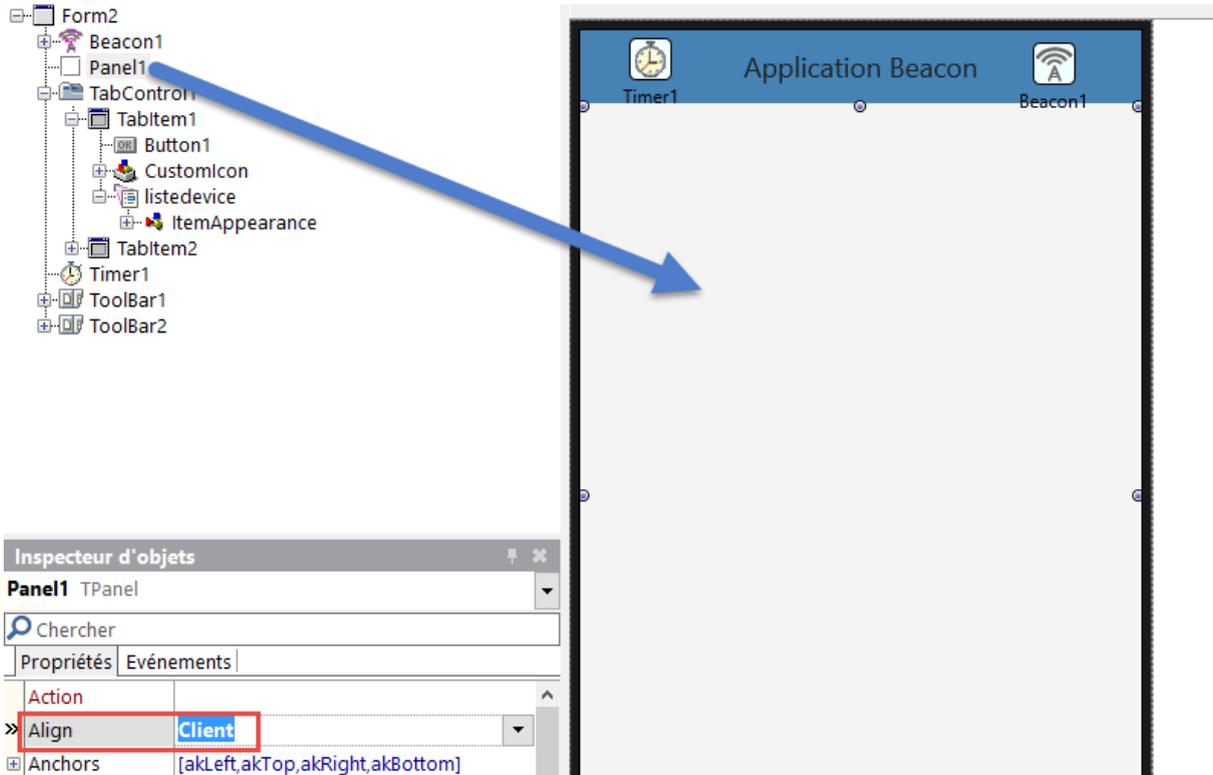
La propriété ScanMode identifie le type de beacon que le système doit rechercher : mode standard (Standard), mode alternatif (Alternative), mode Eddystone (Eddystone) ou mode étendu (Extended). Le mode Extended vous permet de rechercher plusieurs types de beacons simultanément. TBeaconManager est la classe principale de l'implémentation beacon de l'API RTL des beacons. TBeaconManager est responsable du recensement des régions et gère les événements pour obtenir les informations à partir des beacons. L'instance TBeaconManager doit être créée pour le mode Standard, le mode Alternative ou le mode Eddystone.



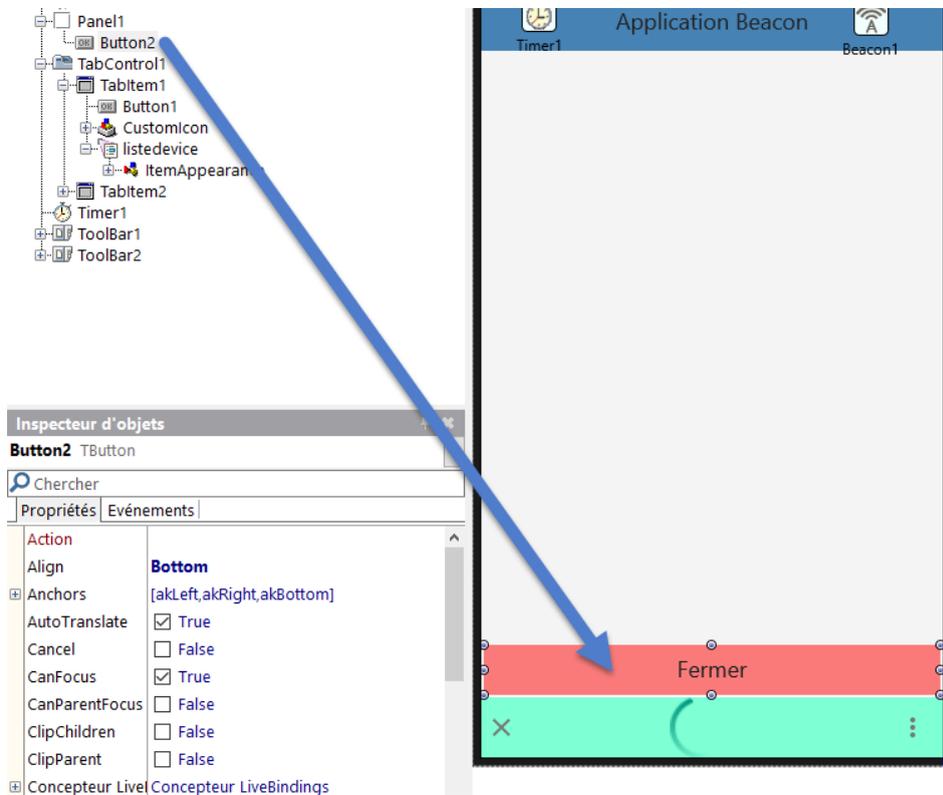
Nom :

Prénom :

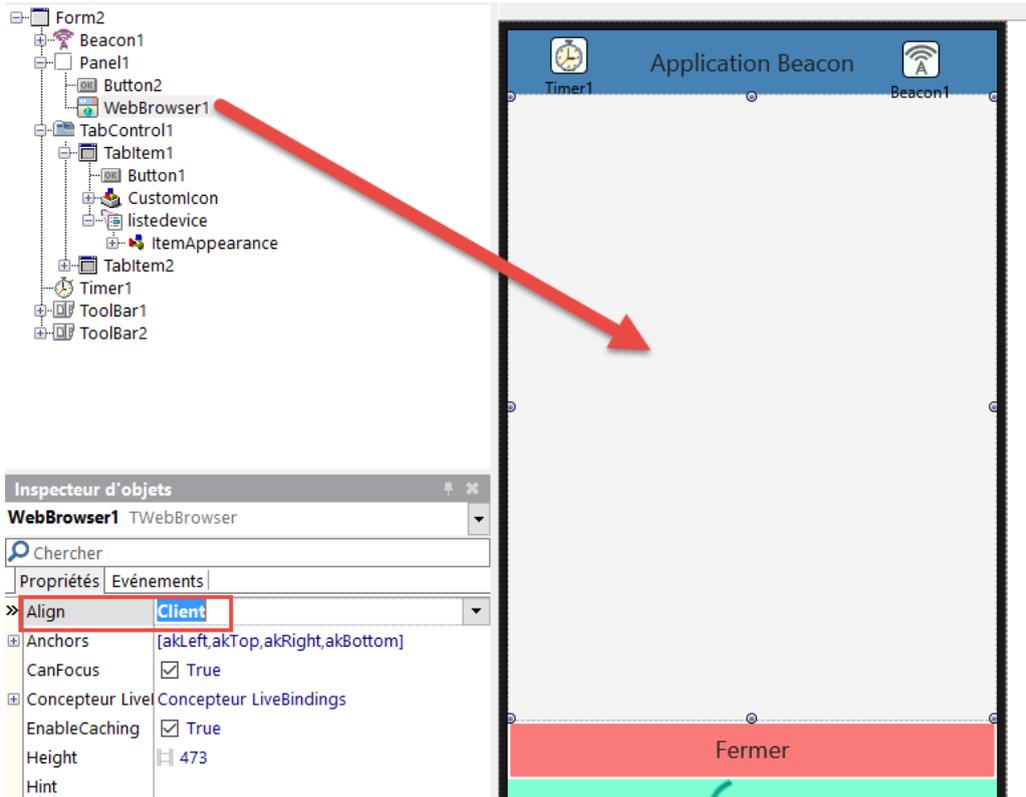
- Rajouter un Panel



- Puis dans le Panel rajouter un bouton de couleur rouge avec comme texte « Fermer »



- Puis toujours dans le Panel, rajouter un Webbrowser

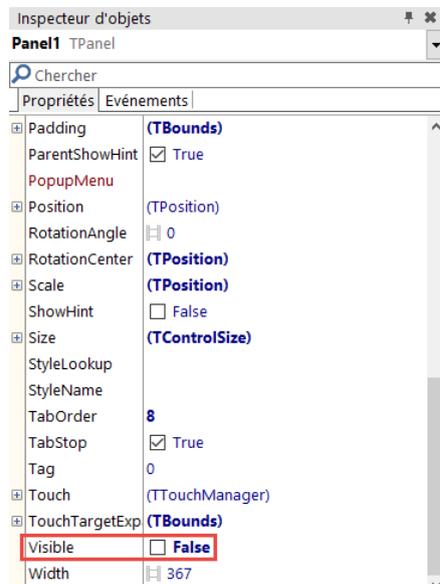


- Modifier le nom

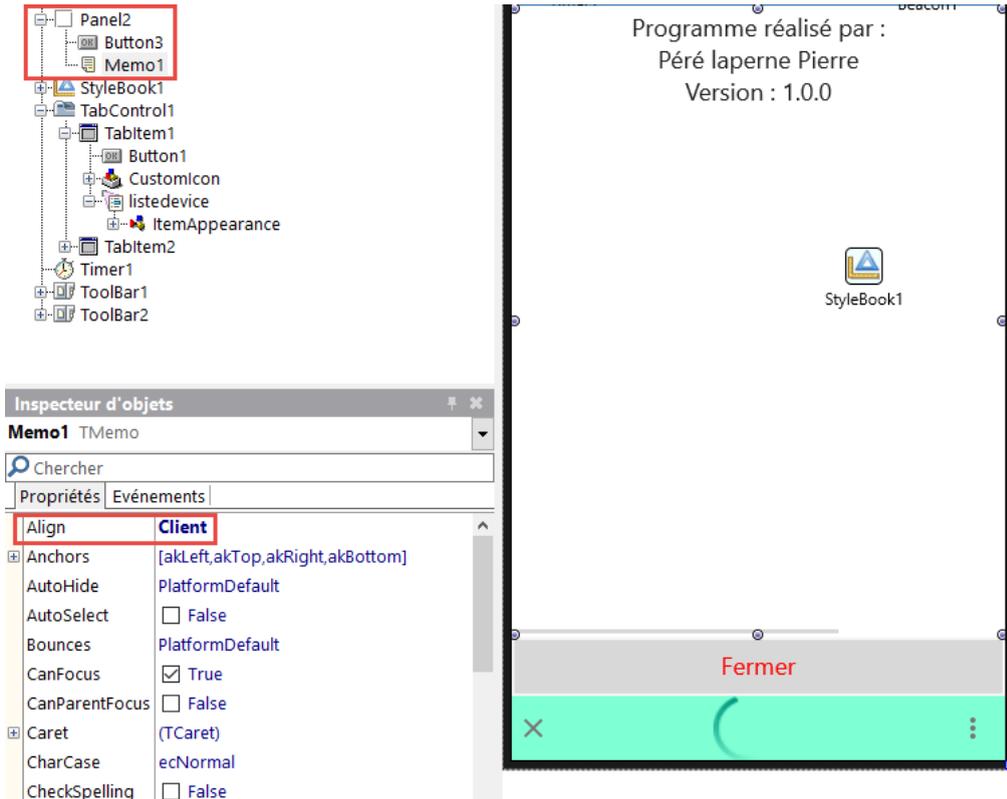
Hint	
LiveBindings	LiveBindings
Margins	(TBounds)
Name	web
Position	(TPosition)
Size	(TControlSize)
StyleName	



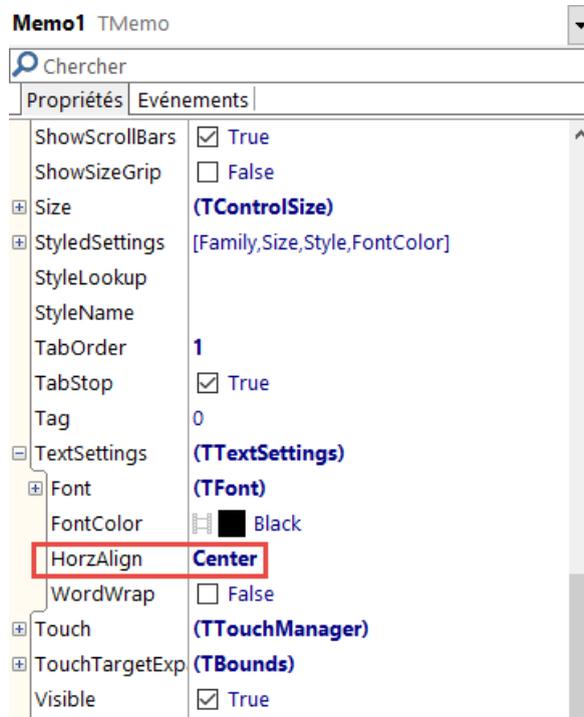
- Rendez le invisible



- **Rajouter un deuxième Panel avec les mêmes paramètres précédents**



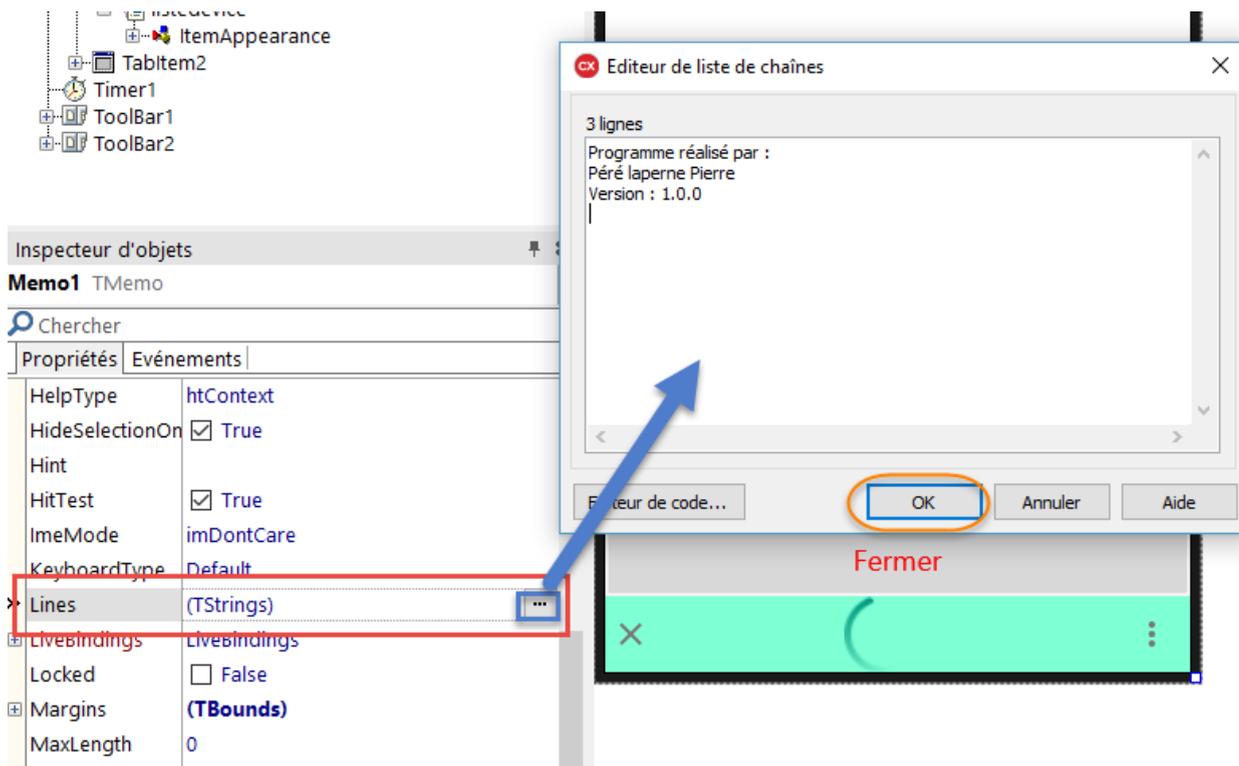
- **Centrer le texte du Memo**



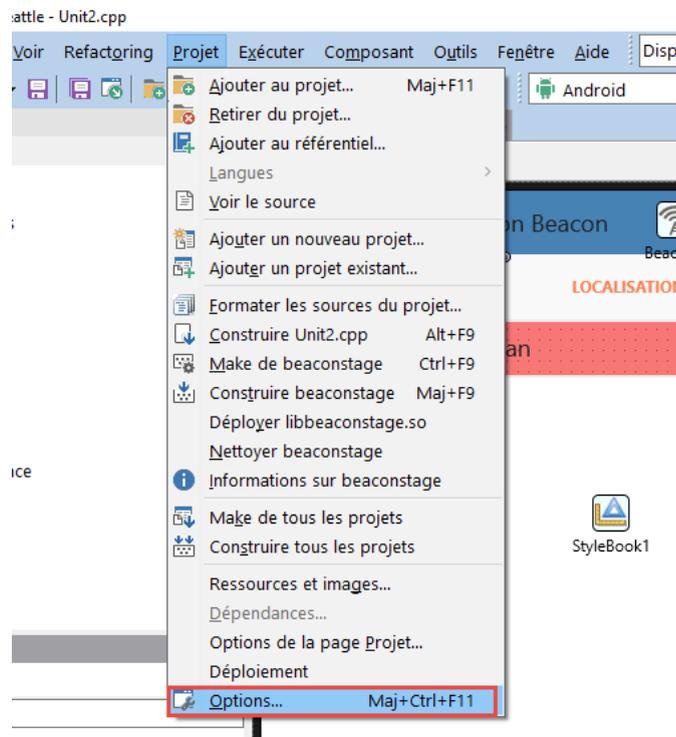
Nom :

Prénom :

- **Rajouter le texte**

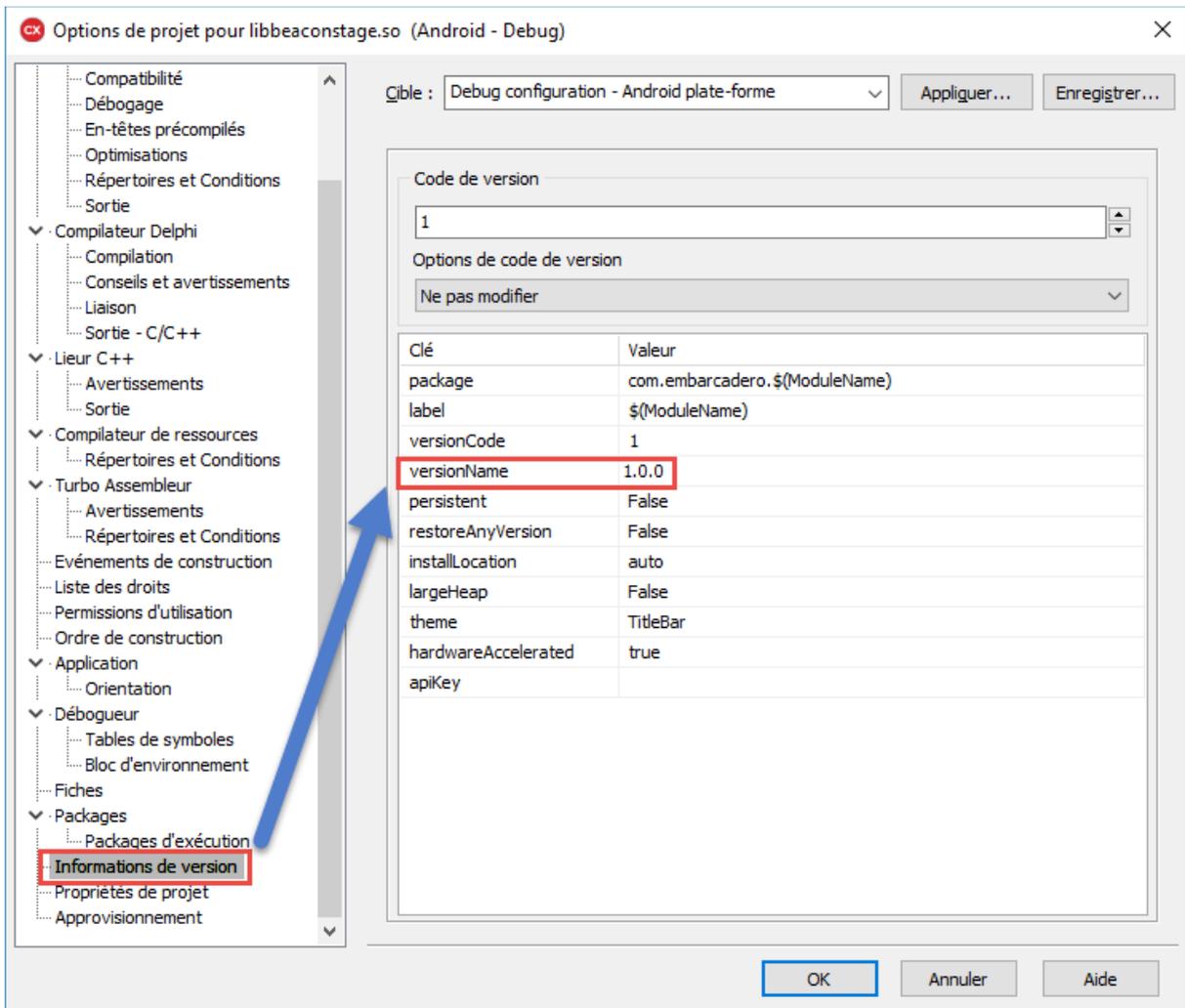


- **Pour modifier la version en fonction du texte**



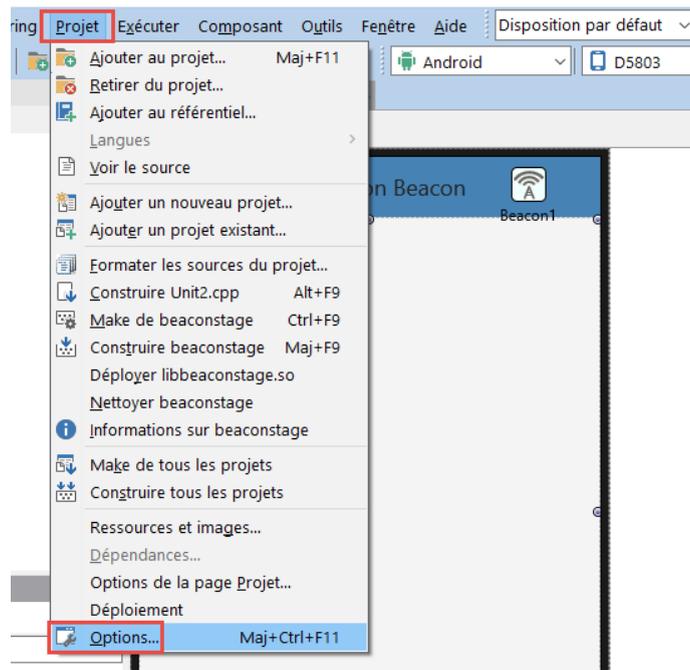
Nom :

Prénom :



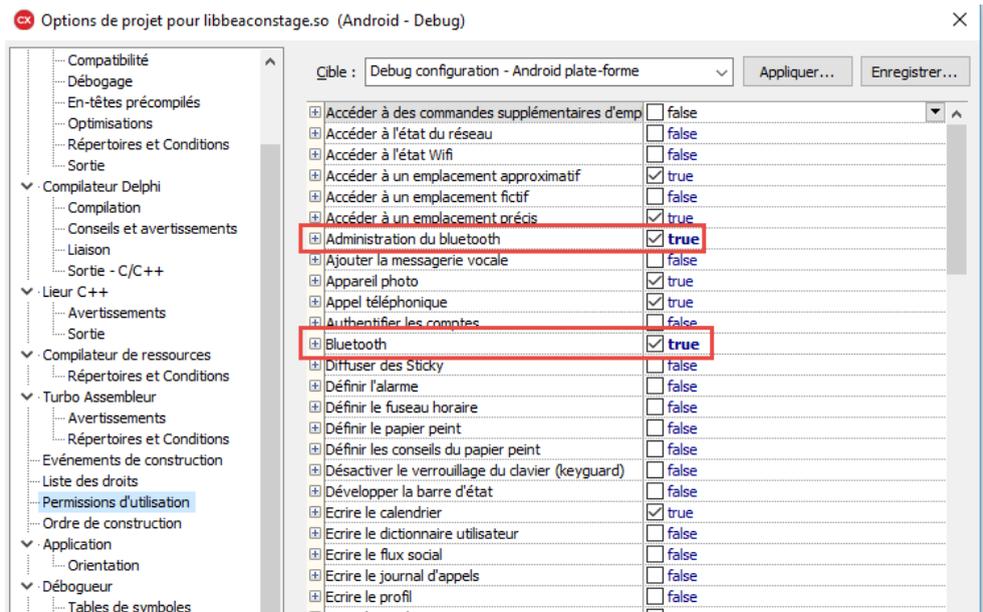
- **Programmation**

- Dans un premier temps on va rajouter des autorisations pour utiliser le Bluetooth du téléphone

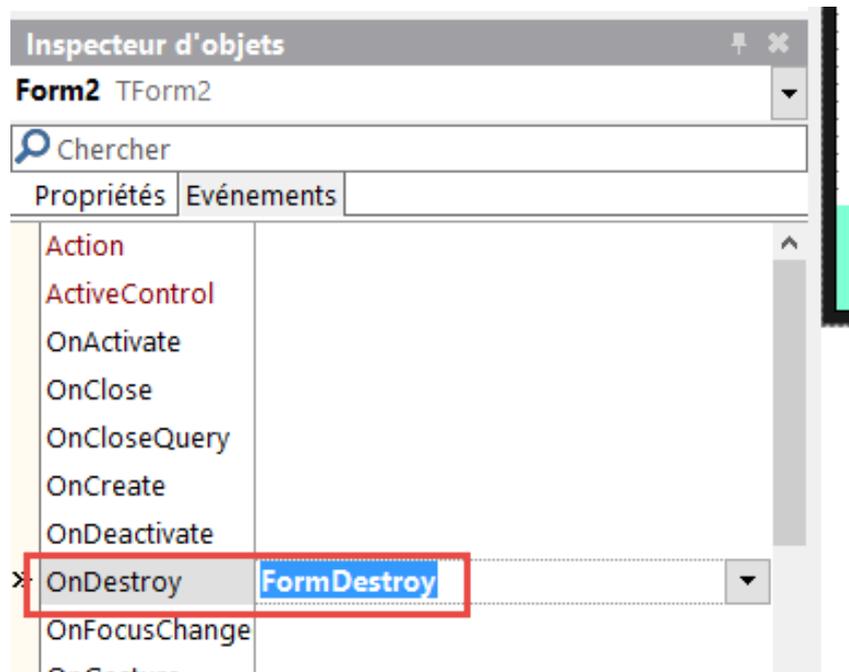


Nom :

Prénom :



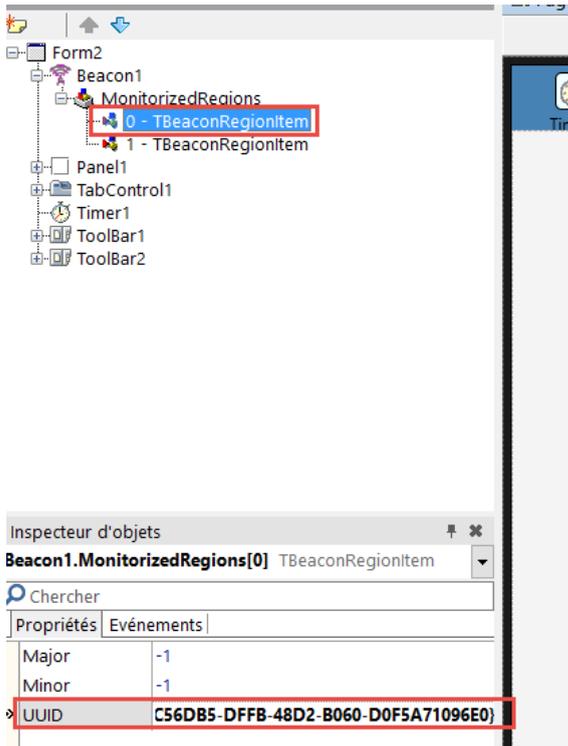
- Créer la fonction FormDestroy



```
void __fastcall TForm2::FormDestroy(TObject *Sender)
{
    this->Beacon1->Enabled=false; // arrêt recherche beacon lors de la fermeture de l'application
}
//-----
```


Nom :

Prénom :



- Créer la variable `FCurrentBeaconList` qui va permettre de récupérer la liste des Beacon

```
void __fastcall Beacon1BeaconProximity(TObject * const Sender, TBeaconProximity Proximity);  
private: // Déclarations utilisateur  
TBeaconList FCurrentBeaconList;  
50 public: // Déclarations utilisateur  
__fastcall TForm2(TComponent* Owner);  
};  
//-----  
extern PACKAGE TForm2 *Form2;  
//-----  
#endif
```

Unit2.cpp | Unit2.h | Conception | Historique

- Créer la fonction Beacon1BeaconEnter qui va permettre de détecter les Beacon

```

50 //-----
51 void __fastcall TForm2::Beacon1BeaconEnter(Object * const Sender, IBeacon * const ABeacon,
52     const TBeaconList CurrentBeaconList)
53 {
54     TListItem *LItem1;
55
56     LItem1 = listedevice->Items->Add();
57     TVarRec v[] = [{ ABeacon->Distance }; //déclare la variable pour la distance
58     LItem1->Text = " UUID: " + GUIDToString(ABeacon->GUID);
59     LItem1->Detail = " Major:" + IntToStr(ABeacon->Major)+ " Minor:" + IntToStr(ABeacon->Minor) +
60         " Rssi: " + IntToStr(ABeacon->Rssi) + Format(" Distance: %f m", v, 0);
61
62     // affiche la puissance du signal et la distance du Beacon
63
64     FCurrentBeaconList = CurrentBeaconList;
65
66 }
67 //-----
    
```

- Faire la même chose quand les beacons sortent de la zone

```

250 //-----
251 void __fastcall TForm2::Beacon1BeaconExit(Object * const Sender, IBeacon * const ABeacon,
252     const TBeaconList CurrentBeaconList)
253 {
254     TListItem *LItem1;
255
256     LItem1 = listedevice->Items->Add();
257     TVarRec v[] = [{ ABeacon->Distance }; //déclare la variable pour la distance
258     LItem1->Text = " UUID: " + GUIDToString(ABeacon->GUID);
259     LItem1->Detail = " Major:" + IntToStr(ABeacon->Major)+ " Minor:" + IntToStr(ABeacon->Minor) +
260         " Rssi: " + IntToStr(ABeacon->Rssi) + Format(" Distance: %f m", v, 0);
261
262     // affiche la puissance du signal et la distance du Beacon
263
264     FCurrentBeaconList = CurrentBeaconList;
265
266 }
267 //-----
    
```

- Maintenant on va utiliser une nouvelle fonction pour faire un effet de style suivant la proximité des Beacon en modifiant la couleur du titre du TabItem2

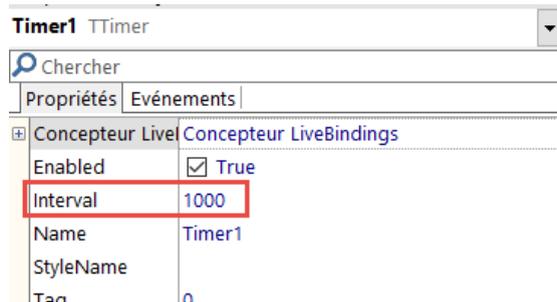
```

60 //-----
61 void __fastcall TForm2::Beacon1BeaconProximity(Object * const Sender, IBeacon * const ABeacon,
62     TBeaconProximity Proximity)
63 {
64     Integer valeur;
65
66     valeur=(ABeacon->Proximity);
67     switch (valeur)
68     {
69     case 1:
70
71         this->TabItem2->TextSettings->FontColor=(TAlphaColor) TAlphaColorRec::Aqua ;
72
73         break;
74     case 2:
75
76         this->TabItem2->TextSettings->FontColor=(TAlphaColor) TAlphaColorRec::Chartreuse;
77         break;
78     case 3:
79
80         this->TabItem2->TextSettings->FontColor=(TAlphaColor) TAlphaColorRec::Gold;
81         break;
82     };
83
84 }
85 //-----
    
```

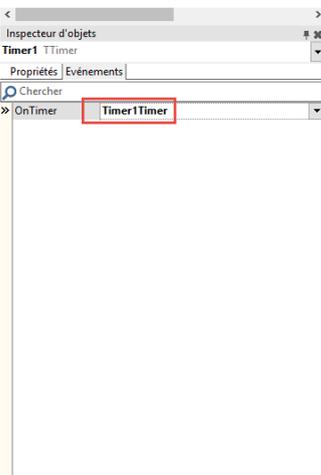
Nom :

Prénom :

- Pour terminer, suivant le Beacon à proximité, on va ouvrir une page Web
 - On va lancer une application suivant un intervalle de temps du Timer



TForm2:Timer1Timer(TObject *Sender)

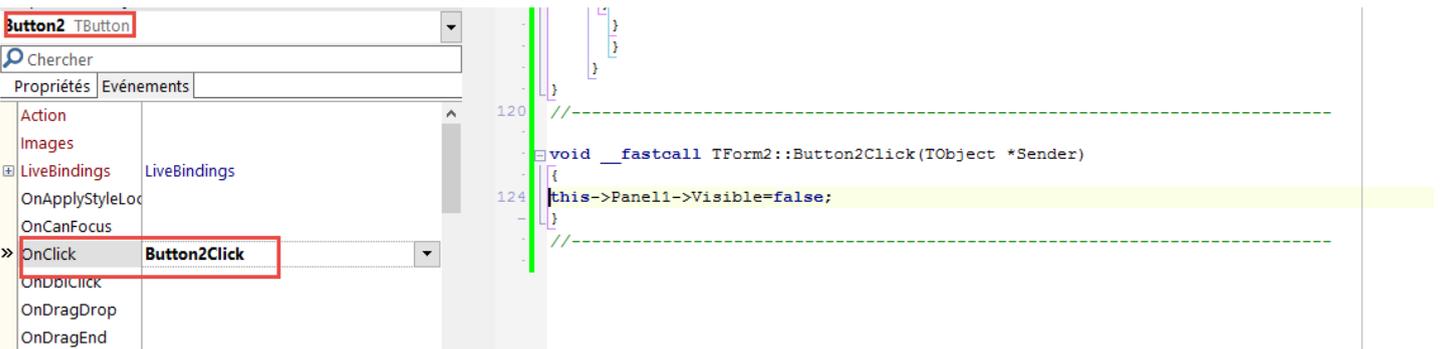


```
void __fastcall TForm2::Timer1Timer(TObject *Sender)
{
    int I;
    TListViewItem *LItem;
    listebeacon->Items->Clear();

    for (I = 0; I < FCurrentBeaconList.Length; I++) //réalise une boucle pour afficher les Beacons
    {
        if ((FCurrentBeaconList[I] != NULL) && (FCurrentBeaconList[I]->ItsAlive()))
        {
            LItem= listebeacon->Items->Add();
            LItem->Text = GUIDToString(FCurrentBeaconList[I]->GUID);
            LItem->Detail = "Major: " + IntToStr(FCurrentBeaconList[I]->Major) + " Minor: " + IntToStr(FCurrentBeaconList[I]->Minor) +
                "Rssi: " + IntToStr(FCurrentBeaconList[I]->Rssi) + " Distance: " + FloatToStr(FCurrentBeaconList[I]->Distance);

            //affichage page web
            if (FCurrentBeaconList[I]->Distance<0.5) { //réalise un test suivant la distance pour afficher la bonne page web
                this->Panel1->Visible=true;
                if (FCurrentBeaconList[I]->Minor==1) {
                    this->web->Reload(); //met à jour les pages web sur le navigateur
                    //affiche la page web correspondante dans le navigateur
                    this->web->Navigate("http://www.sti2dsinhyrome.fr/beacon/index3.php");
                }
                if (FCurrentBeaconList[I]->Minor==2) {
                    this->web->Reload(); //met à jour les pages web sur le navigateur
                    this->web->Navigate("http://www.sti2dsinhyrome.fr/beacon/index2.php");
                }
            }
        }
    }
}
```

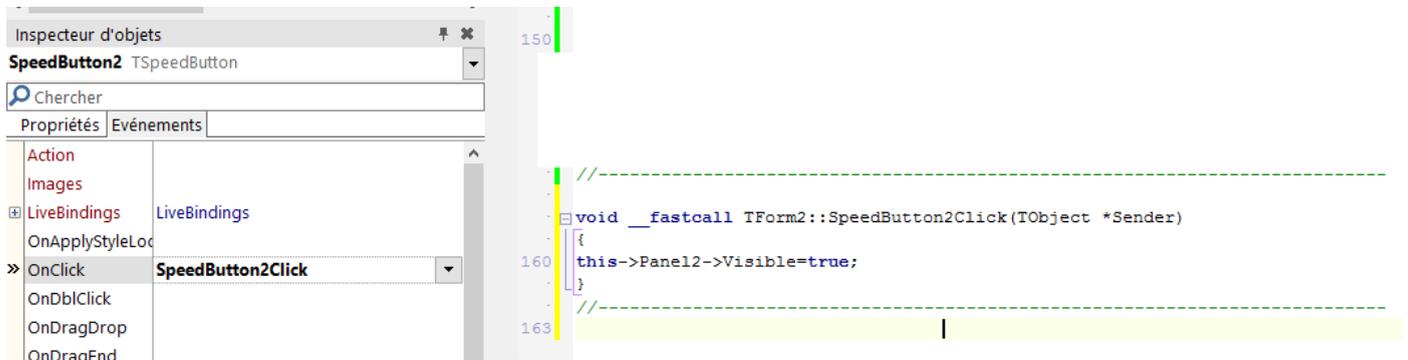
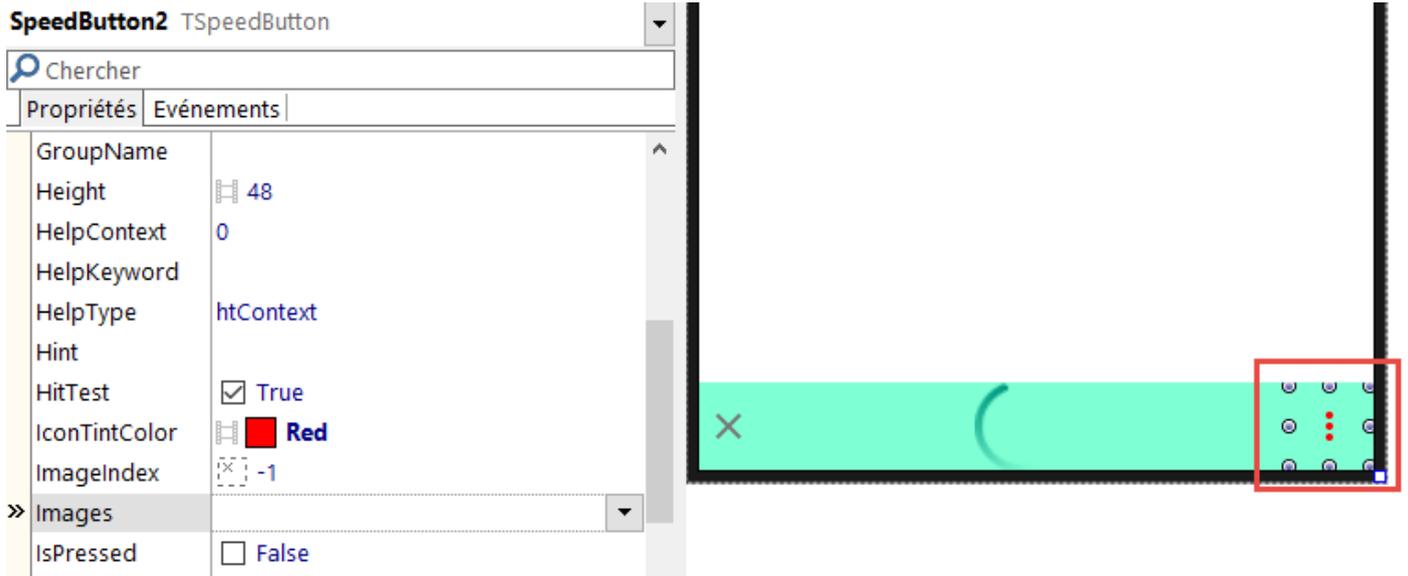
- On va créer une fonction pour fermer le Panel en actionnant le bouton « Fermer »



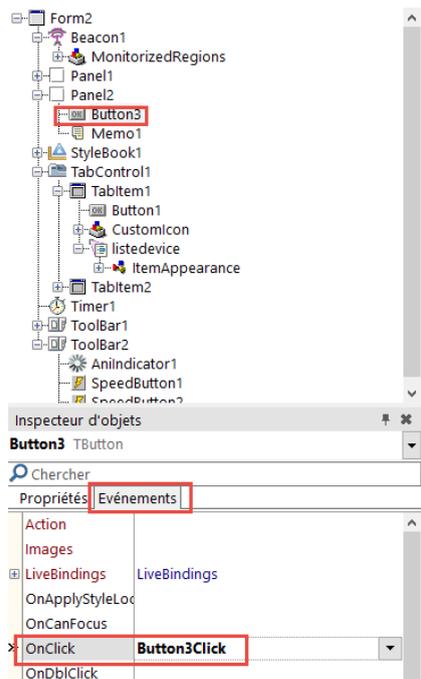
Nom :

Prénom :

- On va créer une fonction pour ouvrir le Panel2 avec le SpeedButton2



- On va utiliser le bouton « fermer » pour fermer le Panel2



```

void __fastcall TForm2::Button3Click(TObject *Sender)
{
54  this->Panel2->Visible=false;
}
//-----

```

- En fin pour fermer l'application proprement, en actionnant le speedButton avec la croix

```

void __fastcall TForm2::SpeedButton1Click(TObject *Sender)
130 this->Timer1->Enabled=false;
    this->Beacon1->StopScan();
133 this->Beacon1->Enabled=false;
    Form2->Close(); //Arrêt de l'application
}
//-----

```

- Tester l'application

- Mise en application
 - Rajouter un tabitem comme ci-dessous

- Lorsqu'on sélectionne un module Beacon dans le dossier localisation, le numéro de la salle correspondante, le major, le minor et la distance doivent apparaître dans les Edit correspondants dans le dossier distance. Plus on se rapproche plus les cercles vers le centre se colorent de couleurs différentes.

Nom :

Prénom :

Exemple :



Remarque :

Pour récupérer les éléments dans le ListView, on utilisera la fonction suivante :

```
listebeacon TListView
Propriétés Événements
Chercher
OnEditModeCha
OnEnter
OnExit
OnFilter
OnGesture
OnItemClick listebeaconItemClick
OnItemClickEx
OnItemsChange
OnKeyDown
OnKeyUp
OnMouseDown

//-----
void __fastcall TForm2::SpeedButton2Click(TObject *Sender)
{
    this->Panel2->Visible=true;
}
//-----
void __fastcall TForm2::listebeaconItemClick(TObject * const Sender, TListViewItem * const AItem)
{
    detail1=AItem->Detail;
    UUID1=AItem->Text;
}
```

Pour récupérer un caractère dans une chaîne de caractères, on indique la position du caractère :

```
minor1=detail1[16]; // on demande le 17° caractère (le premier à pour valeur 0)
```

- Vous pouvez récupérer le programme. apk ici pour le tester:

