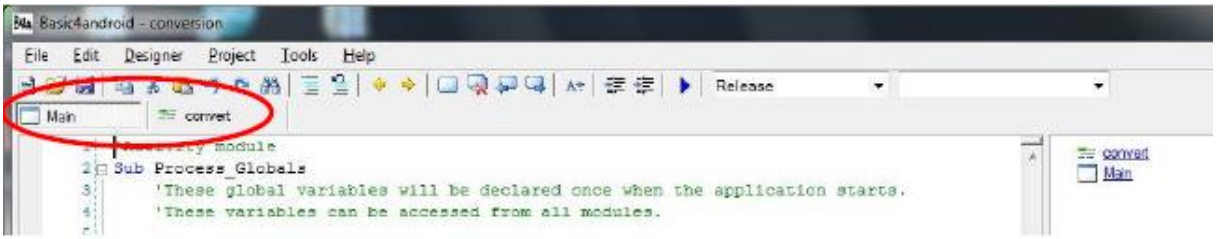
Mise en Situation : Conversion binaire - décimale

On se propose de développer une petite application Android permettant de convertir un mot binaire en décimal et réciproquement. La conversion n'étant pas forcément en elle-même la partie la plus simple de votre programme, on va utiliser un module Basic4Android réalisant ces opérations.

Programmation sous B4A

1- Lancez B4A. La première chose à faire est d'enregistrer votre programme, pour se faire, cliquez sur le menu ***File-Save.***

2- Créez un dossier nommé [conversion] puis donnez un nom à votre projet par exemple "conversion_v1"



3- Vous allez tout de suite insérer dans votre projet le module de conversion. Pour cela utilisez le menu ***Projet – Add Existing Module*** et sélectionnez le fichier **convert.bas** qui se trouve sur le serveur dans les ressources. Votre projet contient maintenant deux onglets, permettant de passer du module principal (Main) au secondaire (convert) :

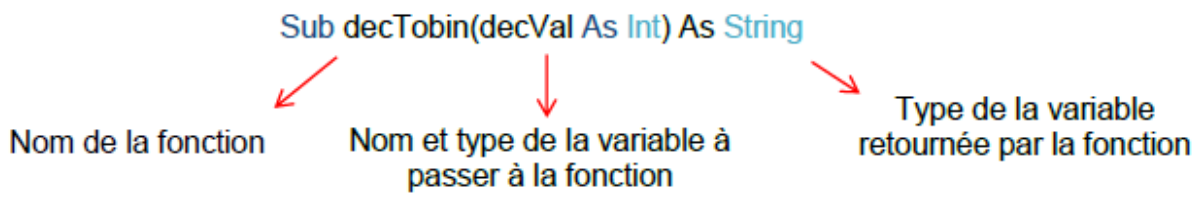
4- Cliquez sur l'onglet « **convert** » pour faire apparaitre le code (voir page suivante).

On voit ici que le module **convert** est composé de deux fonctions. Les fonctions commencent par le mot clef **Sub** et se terminent par **End Sub**. Une fonction est une partie du programme qui est exécutée car associée à un évènement (par exemple l'appui sur un bouton), ou bien qui est appelée par une autre fonction en tapant simplement son nom.

Ici nos deux fonctions ne sont associées à aucun évènement, elles devront donc être appelées.

La première fonction se nomme **decTobin**. Elle a pour rôle de réaliser la conversion d'une valeur décimal en binaire. Elle doit être appelée en lui passant une valeur de type **int** (donc variant de -2^{31} à $2^{31}-1$; voir tableau sur les variables en fin de document) qui sera stockée dans la variable **decVal,** et qui sera la valeur que l'on désire convertir en binaire. La fonction retournera une valeur de type **String** (chaine de caractères), résultat de la conversion en binaire.

Cela est visible dans l'écriture de la fonction :

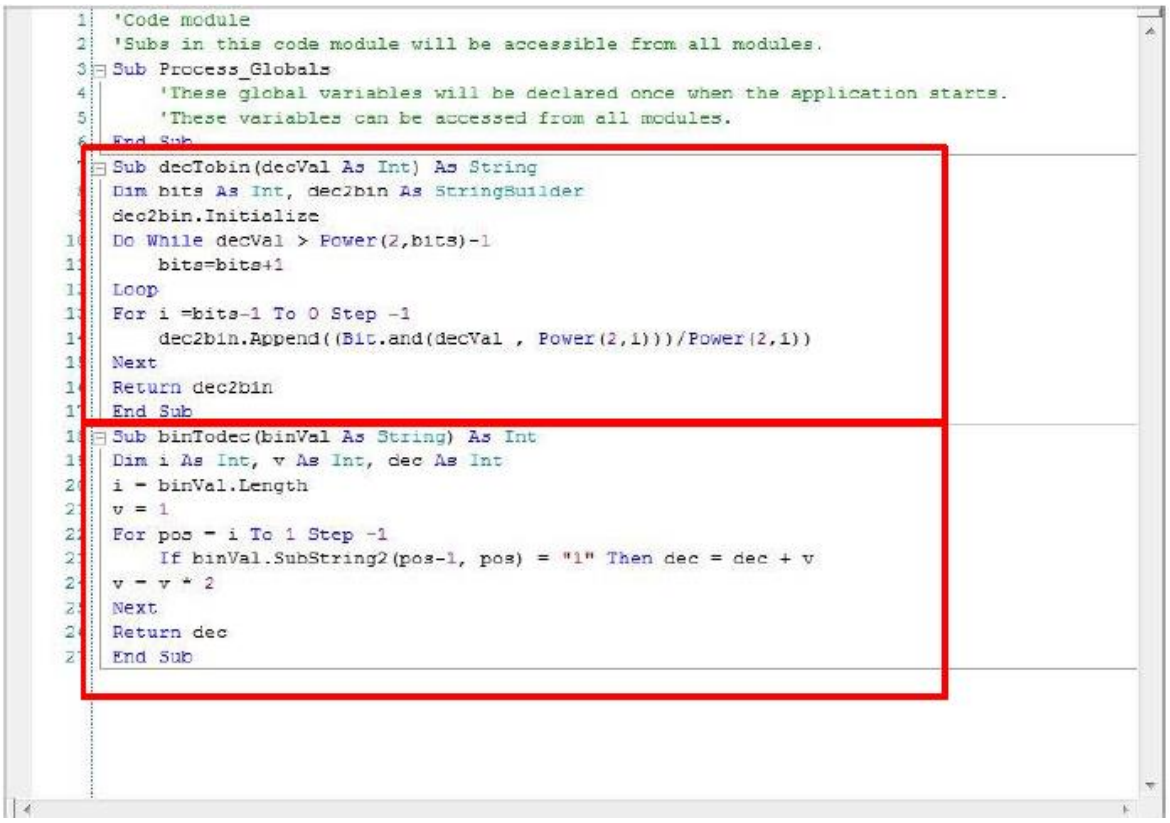


Ainsi si dans votre programme principal vous tapez la ligne suivante :

resultat=convert.decTobin (55) ;

Cela aura pour effet d'appeler la fonction **decTobin** définie dans le module **convert** en lui passant la valeur **55**. La fonction **decTobin** réalisant la conversion décimal en binaire, la variable **resultat** (qui doit être de type string) prendra comme valeur « **110111** ».

La deuxième fonction **binTodec** réalise la conversion dans l'autre sens. Il faut l'appeler en lui passant une chaine de caractères correspondant à l'écriture binaire du nombre que l'on veut convertir. La fonction retourne une variable de type **Int**, résultat de la conversion.



```
'Code module
'Subs in this code module will be accessible from all modules.
Sub Process_Globals
    'These global variables will be declared once when the application starts.
    'These variables can be accessed from all modules.
End Sub
Sub decTobin(decVal As Int) As String
Dim bits As Int, dec2bin As StringBuilder
dec2bin.Initialize
Do While decVal > Power(2,bits)-1
    bits=bits+1
Loop
For i =bits-1 To 0 Step -1
    dec2bin.Append((Bit.and(decVal , Power(2,i)))/Power(2,i))
Next
Return dec2bin
End Sub
Sub binTodec(binVal As String) As Int
Dim i As Int, v As Int, dec As Int
i = binVal.Length
v = 1
For pos = i To 1 Step -1
    If binVal.SubString2(pos-1, pos) = "1" Then dec = dec + v
v = v * 2
Next
Return dec
End Sub
```

5- Construisez, en utilisant le Designer, la feuille (l'IHM) de votre programme. Vous devrez y placer au moins les objets suivants :

- Pour la conversion décimal/binaire :
    - Un bouton (objet ***Button***) permettant de lancer la conversion du nombre saisi par l'utilisateur ;
    - Une zone de texte (objet ***EditText***) permettant la saisie du nombre décimal à convertir ;
    - Un ***Label*** dans lequel l'utilisateur pourra lire le résultat de la conversion
    - Les trois objets appartiendront à un objet ***Panel*** ;

- Pour la conversion binaire/décimal :
    - Un bouton (objet ***Button***) permettant de lancer la conversion du mot binaire saisi par l'utilisateur ;
    - Une zone de texte (objet ***EditText***) permettant la saisie du mot binaire à convertir ;
    - Un ***Label*** dans lequel l'utilisateur pourra lire le résultat de la conversion
    - Les trois objets appartiendront à un objet ***Panel*** ;

6- Générez les membres et construisez votre programme **main**.

Modification de l'application

Vous allez maintenant faire évoluer votre application pour pouvoir réaliser la conversion en hexadécimal. Ainsi l'utilisateur pourra :

- Saisir une valeur en décimal et obtenir sa conversion en binaire et en hexadécimal ;
- Saisir une valeur en binaire et obtenir sa conversion en décimal et en hexadécimal.

Pour le moment la conversion depuis une valeur en hexadécimal ne sera pas demandée.

Lorsque l'on cherche à développer une application, et que l'on ne sait pas trop comment faire, il est judicieux de se rendre sur le site du constructeur et d'y faire des recherches.

Voici le lien direct http://www.basic4ppc.com/index.html mais sachez que B4A est référencé par les moteurs de recherches et qu'en tapant B4A dans Google vous trouverez le lien.

Sur le site plusieurs zones sont intéressantes : les forums (**Online Community**) dont un en langue française ; la **documentation wiki** (wikipedia) accessible par exemple depuis les forums ; la zone **Documentation** avec plusieurs cours ; la zone **Search** vous permettant d'effectuer des recherches sur la globalité du site.

7- Modifiez maintenant votre programme (une petite recherche sur le site avec un indice…. la fonction Bit).

Types de variables dans B4A

B4A	Type	min value	max value
Boolean	boolean	False	True
Byte	integer 8 bits	-2^7 -128	$2^7 - 1$ 127
Short	integer 16 bits	-2^{15} - 32768	$2^{15}-1$ 32767
Int	integer 32 bits	-2^{31} -2147483648	$2^{31}-1$ 2147483647
Long	long integer 64 bits	-2^{63} -9223372036854775808	$2^{63}-1$ 9223372036854775807
Float	floating point number 32 bits	-2^{-149} 1.4E-45	$(2-2^{-23}) * 2^{127}$ 3.4028235 E 38
Double	double precision number 64 bits	-2^{-1074} 2.2250738585072014 E -308	$(2-2^{-52}) * 2^{1023}$ 1.7976931348623157 E 308
Char	character		
String	array of characters		