Premier Programme : Prise en main du logiciel

1. Lancez B4A. Vous devriez obtenir l'écran suivant :

2. La première chose à faire est d'enregistrer notre programme, pour se faire, cliquez sur le menu *File-Save,* je vous recommande de créer un dossier pour chaque programme. Créez donc un dossier nommé [programme1] puis donnez un nom à votre projet par exemple "programme n1"

Développement directement sur le SmartPhone

B4A offre la possibilité de concevoir vos programmes en utilisant un outil de virtualisation de SmartPhone Android. Cela permet donc de ne pas avoir besoin de matériel pour commencer. Hélas cette fonctionnalité est extrêmement « gourmande » en ressource machine, et à moins de disposer d'ordinateur de dernière génération (i7 quadruple-coeur et mémoire vive importante) rend l'utilisation de la virtualisation très lourde.

Vous allez donc travailler directement avec les SmartPhones pour la conception de vos programmes.

Deux solutions s'offrent à vous :

Utiliser un câble USB entre le SmartPhone et le PC. Dans ce cas le SmartPhone doit être configuré en mode USB-débogage et vous devez disposer des drivers pour le SmartPhone (ce qui n'est pas toujours simple) ;

Installer une petite application dans le SmartPhone que l'on trouve ici :

http://www.basic4ppc.com/android/files/b4a_bridge.apk

Et utiliser la commande *Tools – B4A Bridge – Connect Wireless*.

La procédure complète est détaillée ici :

http://www.basic4ppc.com/forum/basic4android-getting-started-tutorials/7978-b4a-bridge-new-way-connect-your-device.html#post45042

C'est cette dernière possibilité que vous allez mettre en oeuvre.

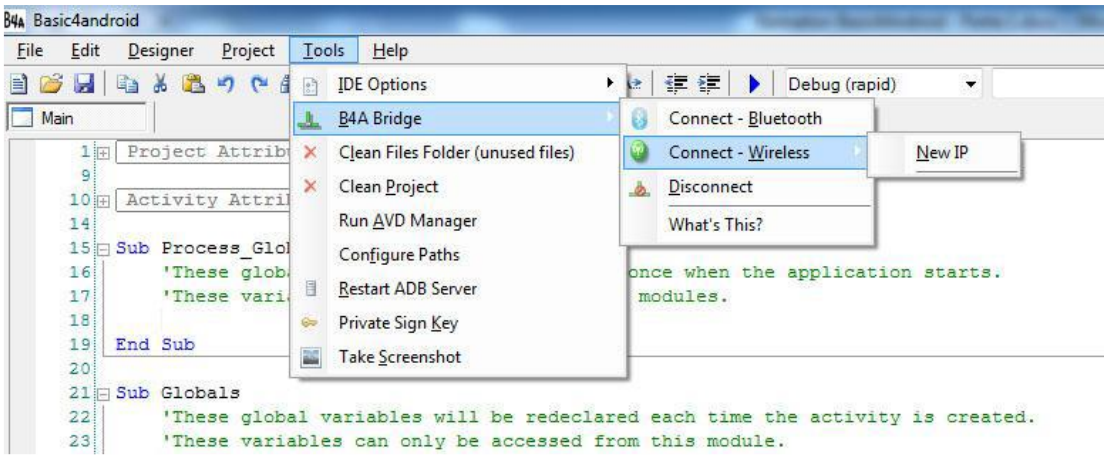
Sur le SmartPhone :

3. Lancez l'application *B4A-Bridge* sur le SmartPhone (elle doit normalement être installée).

4. Repérez l'adresse IP du SmartPhone (192.168…….).
5. Appuyez sur le bouton *Start – Wireless*

Sur Basic4Android :

6. Sur B4A, lancez la commande *Tools – B4A Bridge – Connect Wireless* dans B4A. TP

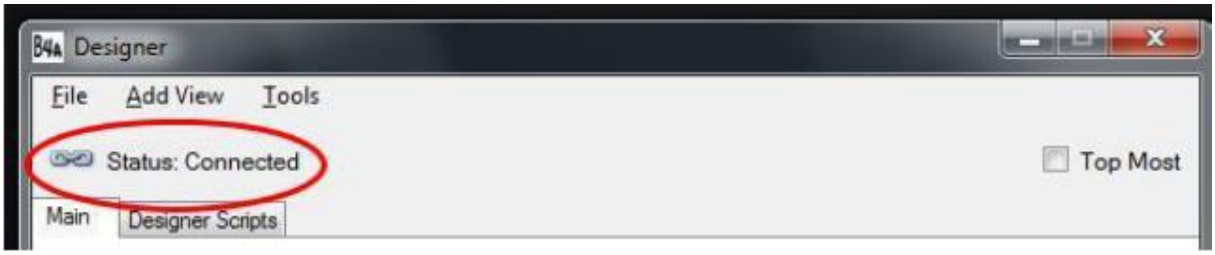


7. Saisissez l'adresse IP relevée précédemment.

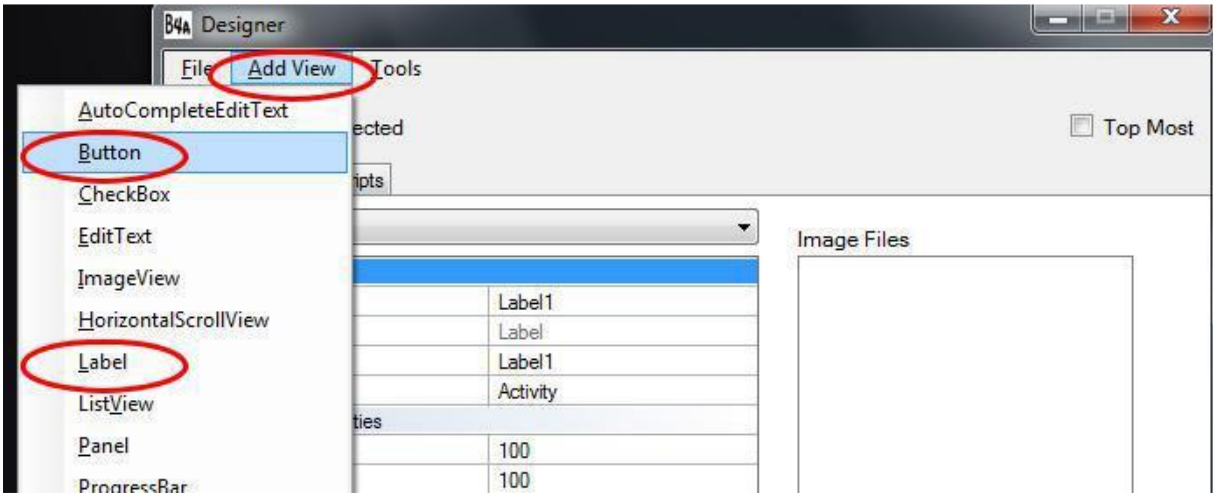
Remarque : si sur le SmartPhone c'est la première fois que le Bridge est installé, celui-ci va vous demander d'installer des applications (au moins le Designer). Validez ces installations.

Vous allez d'abord « dessiner » l'écran (la feuille dans la terminologie B4A) de votre application Android en utilisant le Designer de B4A. Celui-ci va vous permettre de positionner les objets avec lesquels l'utilisateur de votre programme aura une interaction possible.

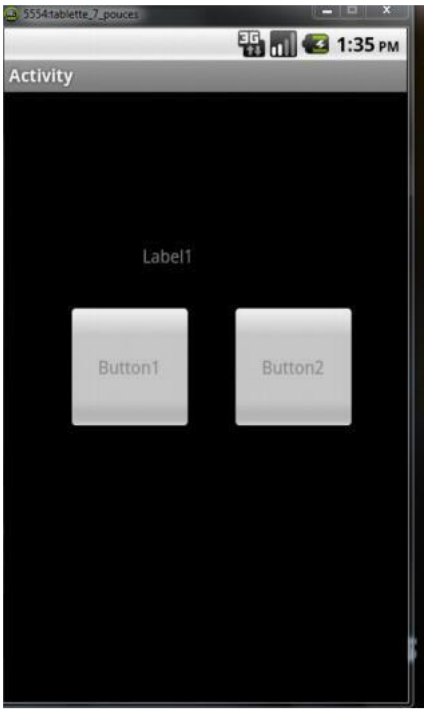
8. Lancez le Designer (menu *Designer*) et connectez-le avec votre SmartPhone en cliquant sur l'icône (ou menu *Tools-Connect to Device*)



9. Vous allez placer trois objets sur votre feuille : deux boutons (Button) et une zone de texte (Label) : menu *AddView – Button* et *Addview-Label*



10. Organisez les éléments pour qu'ils prennent place comme sur l'image suivante :



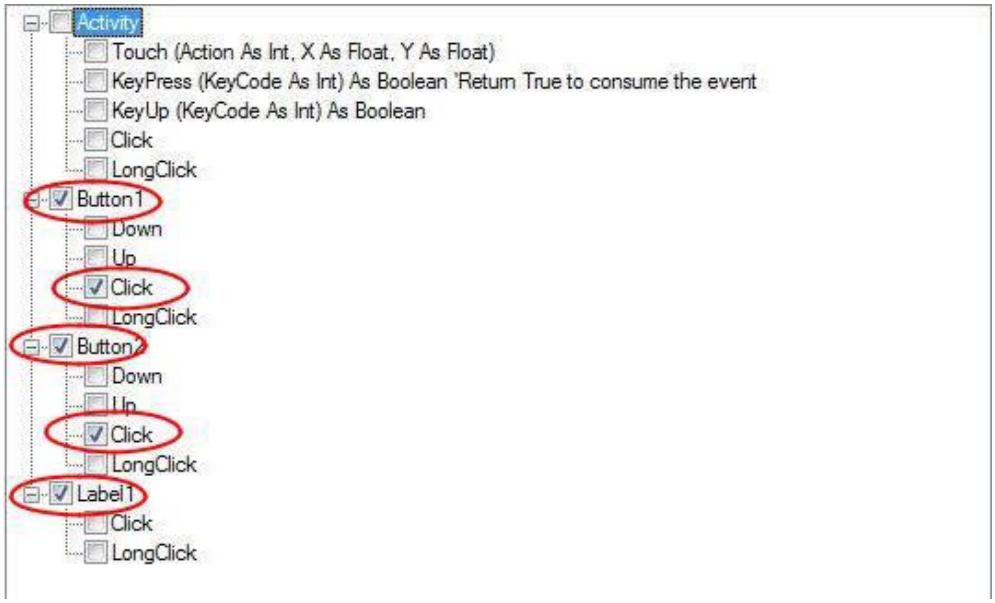
Votre projet propose donc maintenant quatre objets accessibles par le menu-déroulant

Vous n'allez pas pour le moment vous occuper des propriétés de chaque objet, mais plutôt voir comment ils s'intègrent dans votre projet.

On désire que si l'utilisateur clique sur Button1 le texte de Label1 soit « Bonjour » ; s'il clique sur Button2 le texte soit « Au revoir ». Rien d'extraordinaire mais cela va vous permettre de changer les propriétés d'un objet suite à un évènement sur un autre objet.

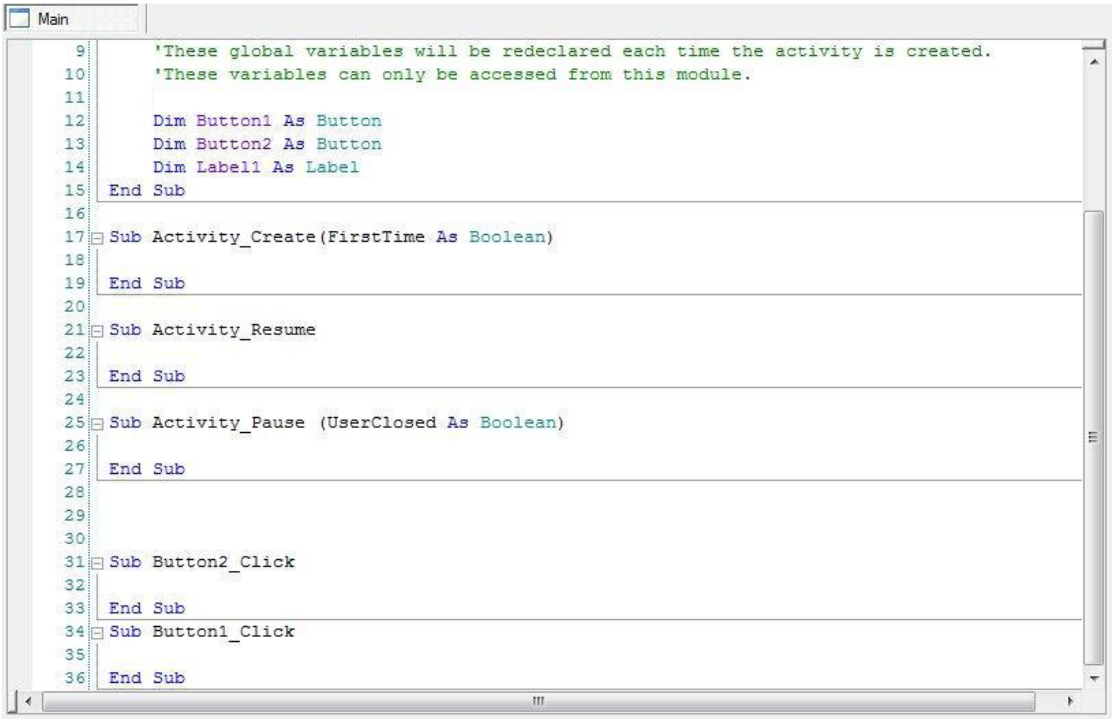
On va maintenant automatiquement générer les lignes de commande correspondant aux évènements que l'on désire traiter.

11. Pour cela utilisez le menu *Tools-Generate Members*. Ouvrez toutes les arborescences et cochez les cases comme sur l'image suivante



Les sélections devant Button1, Button2 et Label1 vont permettre de définir les objets dans votre programme. Vous n'en avez pas forcément besoin ; dans votre cas seul Label1 aurait pu être coché. La sélection des Click sous Button1 et Button2 va automatiquement créer les fonctions associées à cet événement dans votre programme.

12. Cliquez sur *Generate Members* puis *Close*.
13. Fermez le Designer et sauvegardez votre feuille sous le nom « feuille1 »
14. Regardez le résultat dans votre programme



Maintenant la chose est simple à comprendre : si l'utilisateur clique sur Button1 alors ce sont les lignes de code placées dans la fonction Sub Button1_click qui seront exécutées, et réciproquement pour Button2.

On va donc modifier ces lignes pour que la propriété Text de Label1 soit modifiée suite au clique sur un des boutons.

15. Pour cela complétez les lignes de codes suivantes :

```
29
30
31  Sub Button2_Click
32      Label1.Text="Au revoir"
33  End Sub
34  Sub Button1_Click
35      Label1.Text="Bonjour"
36  End Sub
```

Il ne nous reste plus qu'à définir ce que doit faire votre programme au démarrage du logiciel (un peu notre Void Main(void)…).Cela se passe sous la fonction :

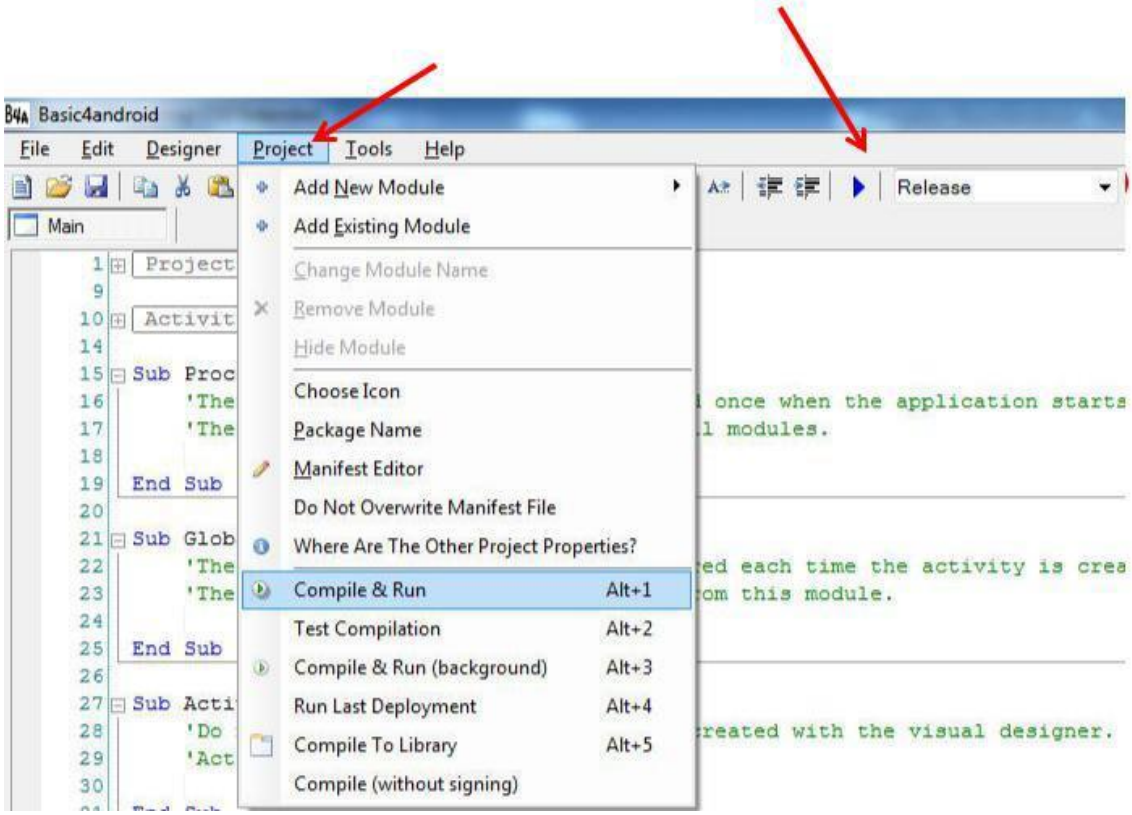
Sub Activity_Create (FirstTime as Boolean).

La seule chose à effectuer est de charger votre feuille pour permettre à l'utilisateur d'avoir accès aux deux boutons. C'est ici que l'objet Activity intervient.

16. Saisissez la ligne de commande suivante

```
16
17  Sub Activity_Create(FirstTime As Boolean)
18      Activity.LoadLayout("feuille1")
19  End Sub
20
```

17. La simulation du programme se lance :
    - soit en utilisant l'icône avec la flèche Run en bleu à condition d'être en mode Release,
    - soit par le menu *Project – Compile and Run*.

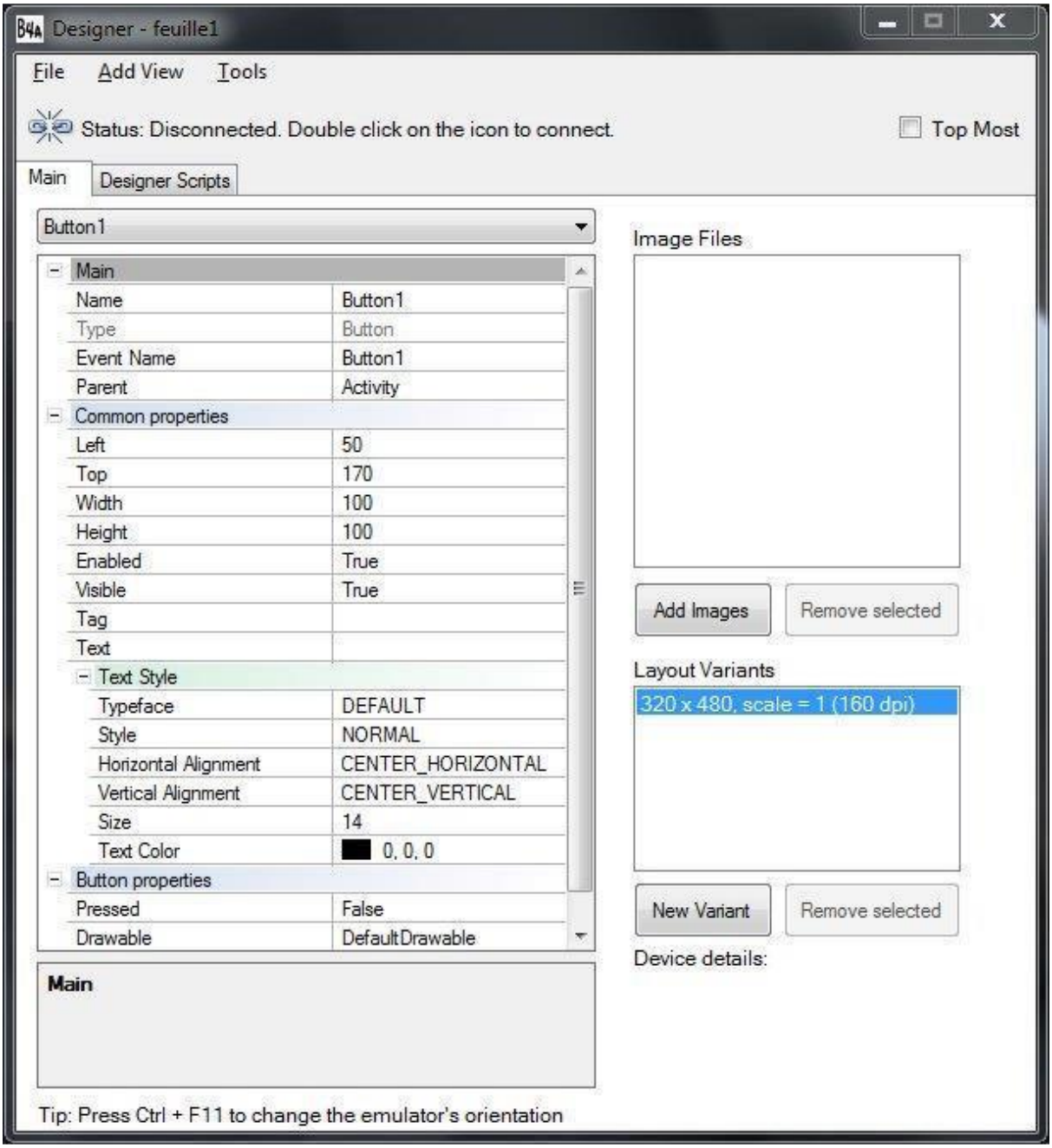


Modification n°1

On va maintenant réaliser deux modifications sur les propriétés de vos objets : une sur le Designer et une dans votre programme ; afin que sur les boutons on puisse lire « Arrivé

» et « Départ » et que le texte du Label apparaissent en rouge pour le message « Bonjour » et en bleu pour « Au revoir ».

18. Ouvrez le Designer et dans la liste déroulante choisissez Button1 pour faire apparaître ses propriétés :



19. C'est la propriété Text qui nous intéresse ici. Saisissez « Arrivé » et constatez le résultat sur votre SmartPhone. Si cela vous plait changez aussi d'autres propriétés du texte (taille, couleur ….).
20. Réalisez la même opération sur le Button2
21. Fermez le Designer en sauvegardant les modifications (il n'est pas nécessaire dans ce cas d'utiliser la commande *Generate Members* du Designer afin de modifier votre programme).

On peut dire pour clarifier les choses que vous venez de modifier de façon statique les propriétés de vos objets.

Il faut maintenant modifier la couleur du texte du Label1. Cette modification, contrairement aux précédentes doit s'effectuer pendant l'exécution de votre programme, et doit prendre deux valeurs différentes.

22. Modifiez votre programme de la façon suivante :

23. Lancez la simulation.

On peut dire pour clarifier les choses que vous venez de modifier de façon dynamique les propriétés de votre objet.

Votre programme est opérationnel, mais on peut regretter que vos objets portent des noms aussi peut évocateur de leur fonction. Effectivement, si votre programme est bien plus conséquent, comment pourrez-vous vous rappeler que le Button1 est le bouton « Arrivé » ?

Pour cela B4A permet de nommer les objets.

24. Relancez le Designer et modifiez la propriété Name de Buton1 en

saisissant « BP_arrive ». Constatez que la propriété Event Name vient aussi de changer. Cette propriété correspond au nom donné par B4A dans votre programme aux fonctions associées à l'objet (Sub BP_arrive_Click par exemple). Vous pouvez très bien imaginer que votre objet porte un nom mais que ses fonctions associées en portent un autre. Néanmoins je ne vous le conseille pas.

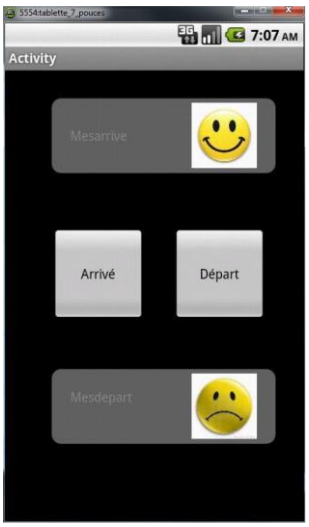
25. Modifiez les autres objets pour que Button2 devienne « BP_Depart » et Label1 devienne « Message ».
26. Ayant changé les noms des objets, il faut à nouveau demander au Designer de lier vos « nouveaux » objets avec votre programme : commande *Generate Members*
27. Fermez le Designer et constatez le résultat sur votre programme. B4A a bien généré les lignes liées aux modifications dans le Designer mais il n'a pas substitué nos anciens noms par nos nouveaux. Il faut donc le faire à la main, même si les outils du menu *Edit* rendent les choses assez simples.
28. Modifiez votre programme pour le rendre à nouveau opérationnel en nommant les deux boutons BParrive et BPdepart.

Modification n°2

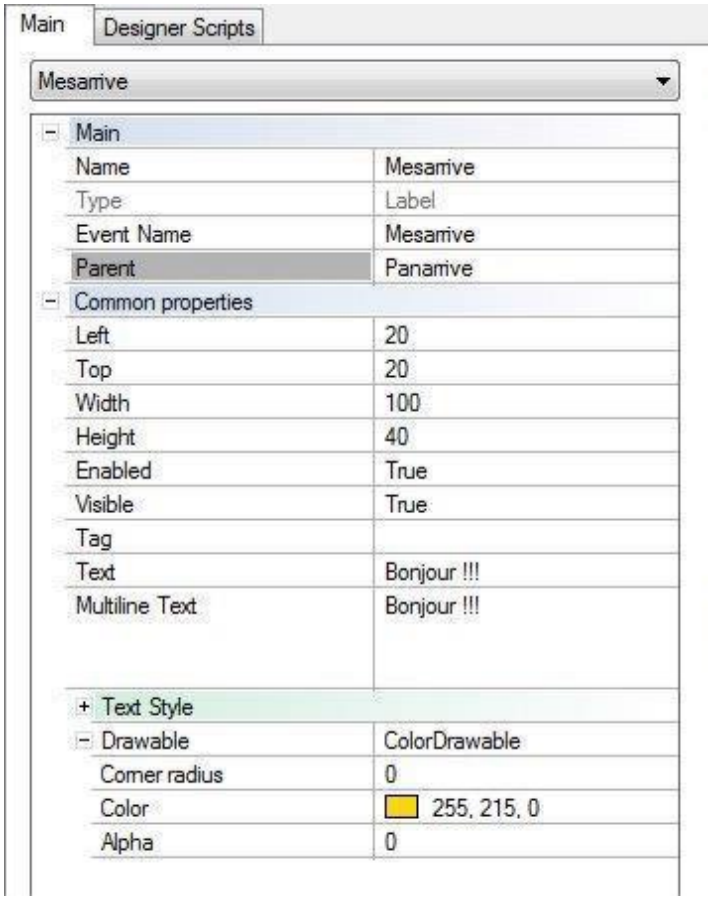
Vous allez encore modifier votre programme pour voir l'intérêt que peut avoir la notion d'appartenance dans le Designer.

Supposons que l'on veuille agrémenter votre projet d'une image accompagnant le

« Bonjour » (un smiley souriant) et d'une associée au « Au revoir » (un smiley triste). Par ailleurs les deux textes et images devront apparaître à deux endroits différents de notre écran (en haut pour l'arrivée, en bas pour le départ).



29. Ouvrez le designer et supprimez l'objet Label (il faut cliquer dessus pour le sélectionner et utiliser le menu *Tools- Remove Selected Views*).
30. Vous allez placer deux panneaux (panel), un en haut l'autre en bas. Nommez les respectivement Panarrive et Pandepart. Définissez leur la même taille en largeur (Width) et hauteur (Height). Choisissez une couleur qui se détache du fond de l'écran.
31. Sélectionnez False à deux propriétés de ces panneaux :
    - Visible : cela aura pour effet de ne pas rendre visible le panneau au lancement du programme ;
    - Enabled : cela interdira à l'utilisateur d'utiliser le panneau.
32. Placez maintenant un label que vous nommerez Mesarrive. Dans la propriété Text saisissez le texte à afficher « Bonjour !!!! ». Modifiez sa couleur pour qu'il s'écrive en jaune. Par défaut votre label appartient à l'objet Activity. On va changer sa parenté pour qu'il soit rattaché au panneau arrivée (Panarrive).

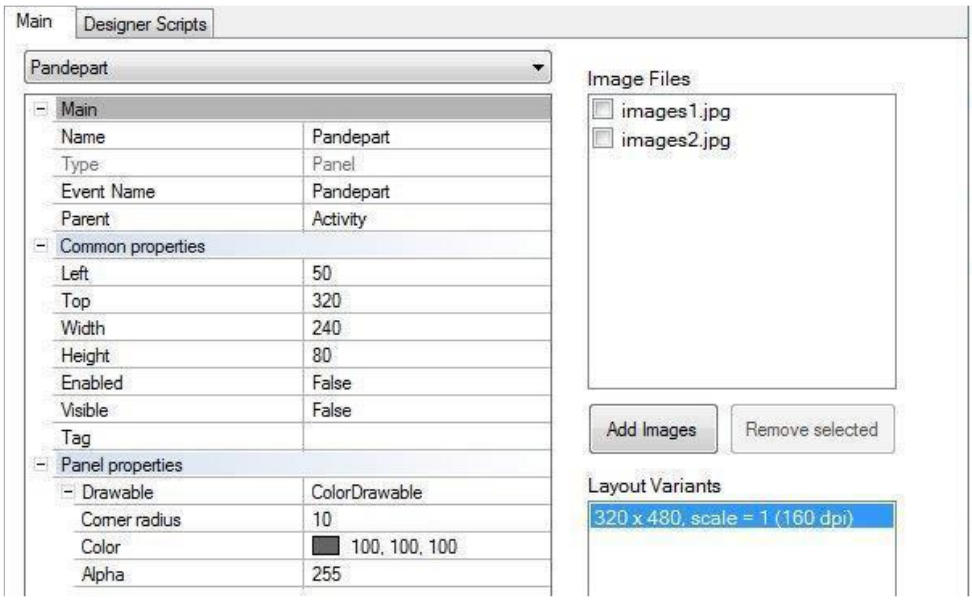


Une fois apparenté au panneau, votre label ne peut se déplacer qu'à l'intérieur de ce dernier. Par ailleurs il hérite des propriétés du panneau : si le panneau n'est pas visible, le label non plus.

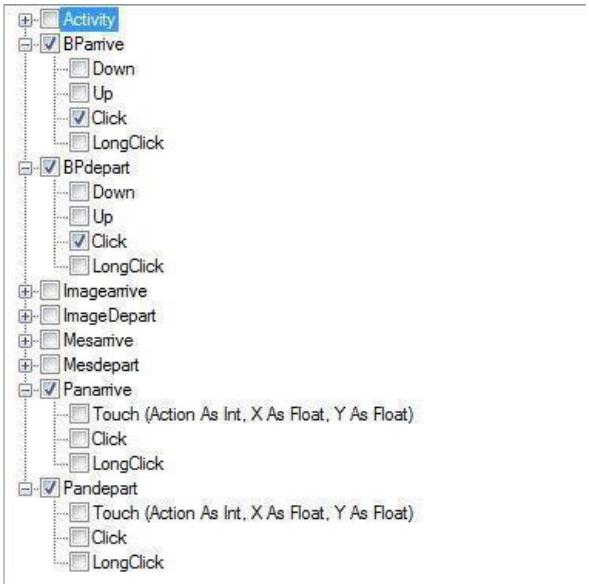
33. Réalisez la même opération pour un deuxième label « Mesdepart » parenté avec le panneau Pandepart.

Reste maintenant à placer les images. Il faut d'abord charger les fichiers images pour qu'ils puissent s'insérer dans votre projet.

34. Pour cela utilisez la commande *Add Images* et chargez les fichiers images1.jpg et images2.jpg



35. Placez maintenant un objet Imageview ; renommez le Imagearrive ; choisissez comme parent Panarrive ; choisissez comme fichier image (Image file) le fichier images1.jpg ; modifiez la taille de l'image et sa position pour qu'il se place dans le panneau.
36. Réalisez la même chose pour le départ avec un objet ImageView que vous nommerez Imagedepart dans le panneau Pandepart.
37. Votre conception est terminée, lancez la commande *Tools-Generate Members* pour ne gérer que les choses suivantes :
    - La définition des deux panneaux ;
    - La création des deux fonctions associées au Click sur les deux boutons



38. Fermez le Designer et analysez votre programme. Il devrait ressembler à celui-là

```
Sub Globals
    'These global variables will be redeclared each time the activity is created.
    'These variables can only be accessed from this module.
    Dim Panarrive As Panel
    Dim Pandepart As Panel
    Dim BParrive As Button
    Dim BPdepart As Button
End Sub

Sub Activity_Create(FirstTime As Boolean)
    Activity.LoadLayout("feuille1")
End Sub

Sub Activity_Resume

End Sub

Sub Activity_Pause (UserClosed As Boolean)

End Sub

Sub BPdepart_Click

End Sub
Sub BParrive_Click

End Sub
```

39. Modifiez les lignes des deux fonctions de la manière suivante :

```
Sub BPdepart_Click
    Pandepart.Visible=True
    Panarrive.Visible=False
End Sub
Sub BParrive_Click
    Pandepart.Visible=False
    Panarrive.Visible=True
End Sub
```

40. Compilez votre programme puis exécutez-le.

On voit que le fait d'avoir associé les objets ImageDepart et Mesdepart au panneau Pandepart nous a ainsi permis de ne pas avoir eu à changer leurs propriétés. Ils ont hérité de celles du panneau.

Modification n°3

Vous allez encore améliorer le programme. Pour le moment le panneau Panarrive reste visible tant que l'utilisateur ne clique pas sur le bouton Départ (réciproquement pour l'arrivé).

On souhaite qu'en cas de non activité pendant un certain temps (disons 5 secondes) les deux panneaux disparaissent.

La première solution consisterait à mettre en place un délai de 5s après l'apparition du panneau, puis de l'effacer. Une boucle comptant (type For…Next) permet de réaliser cette temporisation. Mais ce n'est pas du tout la bonne solution :

- Comment être certains de la précision de la durée ;
- Pendant la boucle votre programme est bloqué.

Pour remédier à notre problème B4A propose un objet Timer. Celui-ci n'est pas disponible dans le Designer puisqu'il n'a pas d'interaction possible avec l'utilisateur. Il va donc falloir travailler directement avec le programme en VB.

Cet objet possède trois paramètres :

Initialize : initialisation du timer en définissant le nom de l'événement associé à l'objet et l'intervalle en ms où le timer déclenche un événement.

Syntaxe : nom_du_timer.Intialize(« nom_de_l'événement » as String, Intervalle as Long)
Exemple : Timer1.initialize(« Timer1 »,1000)

Interval : pour redéfinir l'intervalle en ms où le timer déclenche un événement.

Syntaxe : nom_du_timer.Interval=Intervalle as Long

Exemple : Timer1.Interval=1000

Enabled : valide ou non les fonctionnalités du timer

Syntaxe : Timer1.Enabled=True ou False

Un seul événement est associé : Tick lorsque le timer atteint son intervalle de temps

Sub Timer1_Tick

41. La première étape consiste donc à déclarer l'objet Timer. Modifiez le programme :

```
'Activity module
Sub Process_Globals
    'These global variables will be declared once when the application starts.
    'These variables can be accessed from all modules.
    Dim Montimer As Timer
End Sub
```

42. Au lancement de votre programme il faut initialiser le Timer :

```
Sub Activity_Create(FirstTime As Boolean)
    Activity.LoadLayout("feuille1")
    Montimer.Initialize("Effacement",5000)

End Sub
```

43. Ensuite, il faut créer la fonction associée à l'objet :

```
Sub Effacement_Tick
    Pandepart.Visible=False
    Panarrive.Visible=False
End Sub
```

44. Et enfin, il faut relancer pour 5 secondes le Timer après l'appui sur un des deux boutons poussoirs :

```
Sub BPdepart_Click
    Montimer.Enabled=False
    Pandepart.Visible=True
    Panarrive.Visible=False
    Montimer.Enabled=True
    End Sub
Sub BParrive_Click
    Montimer.Enabled=False
    Pandepart.Visible=False
    Panarrive.Visible=True
    Montimer.Enabled=True
```

45. Testez votre programme.

On voit que le fait d'avoir associé les objets ImageDepart et Mesdepart au panneau Pandepart vous a ainsi permis de ne pas avoir eu à changer leurs propriétés. Ils ont hérité de celles du panneau.

A vous de faire….

Proposez un programme qui sera composé de deux boutons, placés de part et d'autre de l'écran. Au lancement du programme seul le bouton de droite sera visible. Lorsque l'utilisateur appuiera dessus cela aura pour effet de le faire disparaitre et de faire apparaitre celui de gauche. La réciproque sera vraie avec le bouton gauche.