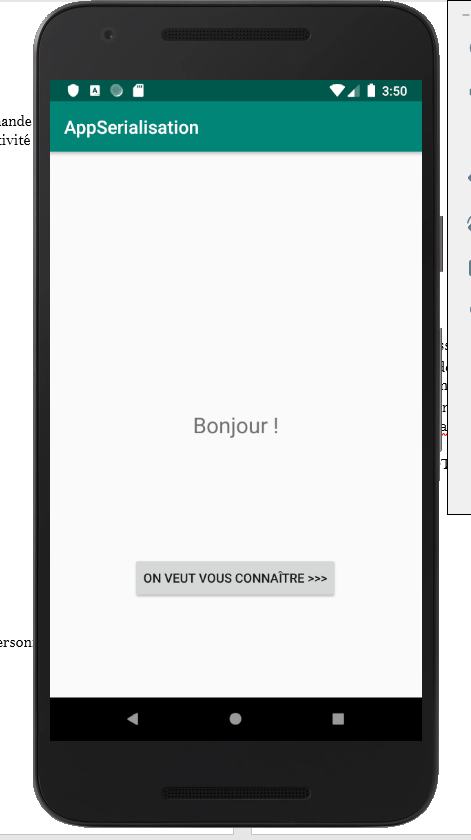
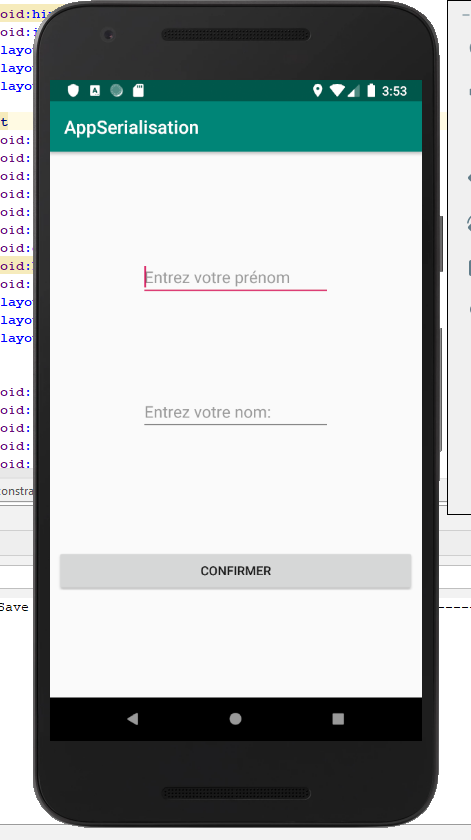
# Annexe 4 – startActivityForResult

Nouveaux éléments :

* startActivityForResult ( regarder projet ImportanceBufferMP3 Annexe 2 pour vous donner une idée )
* utiliser la sérialisation à une autre sauce

# A)Exercice :

Une application simple où on salue l’usager dans une Activity; on lui demande d’entrer des informations sur lui dans une seconde activité afin de revenir sur l’activité initiale avec une salutation personnalisée :

Quand on clique sur confirmer, on revient sur la première activité en personnalisant le message d’accueil avec les entrées texte:



* Codez ce projet en KOTLIN avec les deux activités et la gestion des événements clic ( vous pouvez le faire avec une expression lambda codée à partir de la fonction de haut niveau qui ajoute l’écouteur à la source )
* Coder une petite classe Utilisateur dans son propre fichier contenant comme attributs un prénom et un nom ; il s’agira des informations qui transiteront d’une activité à l’autre.
* Pas besoin de singleton car on ne fera que transférer un objet Utilisateur d’une activité à l’autre
* Étant donné qu’on est en présence d’un « boomerang » ou d’une activité qu’on démarre pour obtenir un résultat devant retourner à l’activité initiale, on va reprendre la technique « startActivityForResult » qu’on a utilisée à l’annexe 2. Suivez les indications suivantes ( en vous inspirant du travail de l’annexe 2 pour la 1ère partie du moins ) :

## technique ( startActivityForResult en KOTLIN )

**Activité de départ ( AccueilActivity ) :**

* + Déclarer un objet ActivityResultLauncher<Intent>
  + Le créer à l'aide de la méthode registerForActivityResult contenant 2 paramètres :
    - Un objet de type ActivityResultContracts.StartActivityForResult()
    - Un objet d'une classe mettant en œuvre l'interface ActivityResultCallback<ActivityResult>
  + Lancer un intent avec la méthode launch sur notre objet ActivityResultLauncher<Intent> lorsqu’on clique sur le bouton

**Activité de destination ( IdentificationActivity )**

* créer un Intent de retour sans destination (Intent() )
* gérer le clic du bouton qui permettra de transmettre un objet Utilisateur à l’activité de départ ( retour du boomerang )
* putExtra( « identifiant », objet Utilisateur ); moyen pour transmettre des infos d’une activité à l’autre par un Intent…vous vous rappelez ?
  + comment passer un objet Utilisateur comme extra ??
  + Comment rendre une classe / objet sérialisable ??
* setResult(code de retour , nom de l’intent de retour )
* finish()

**Activité de départ ( AccueilActivity )**

* notre intent de retour ( boomerang ) revient dans la méthode onActivityResult de la classe interne mettant en œuvre l'interface ActivityResultCallback<ActivityResult>
* vérifier si le code de retour est OK
* récupérer l'intent puis l'extra à partir du paramètre de type ActivityResult pour les afficher dans la salutation de départ

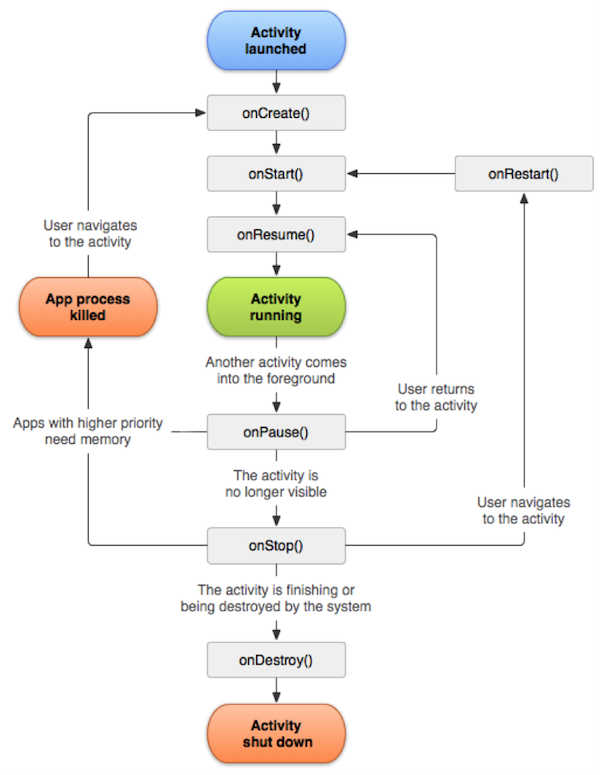
Ca fonctionne ? oui

Tournez l’émulateur que se passe-t-il avec le nom de l’usager ? il disparait

Pourquoi ? pcq lorsqu’on tourne le téléphone le cycle de vie de l’activité est refait au complet (onCreate)

Cliquez ou appuyez ici pour entrer du texte.

# B)cycle de vie



onRestoreInstanceState est appelé

Tourner le téléphone

retour d'avoir changé la langue du téléphone, …

Retour du bouton Recents, du bouton Home, "retour du boomerang"

onSaveInstanceState est appelé

Recoit un appel

Ferme L’app

Attention ! onDestroy n’est pas tjrs appelé selon la doc d’Android

<https://developer.android.com/guide/components/activities/activity-lifecycle>

<https://medium.com/@theabhishekavi/android-activity-lifecycle-9bc7de812dff>

**A-Solution : Si on veut garder les infos durant la session ou le processus courant : savedInstanceState**

* La méthode **protected void** onSaveInstanceState(Bundle infos) est appelée juste avant le onStop dans le cycle de vie d’une Activité; de cette façon, on peut sauvegarder des infos importantes lorsque le cycle de vie est refait au complet pcq on a tourné le téléphone, entre autres. Cependant, lorsque le processus de l'activité est complètement terminé, on perd les infos présentes dans le Bundle
* Autres occasions où ça peut être nécessaire : changer un paramètre sur le téléphone, le système Android a besoin de ressources urgemment
* Permet donc de conserver les infos quand on demeure dans la même session / le même processus
* Utilisé pour conserver un peu d'informations
* On va ajouter au Bundle en paramètre notre Utilisateur en extra. Un Bundle est un ensemble d'informations "extras"

Méthodes : putSerializable : une info au Bundle

getSerializable : récupérer une info du Bundle

* Vu que la rotation du téléphone entraîne qu’on repasse dans le onCreate, on va récupérer notre extra dans le paramètre savedInstanceState de lamethode OnCreate

Ca fonctionne ?

Finalement, est-ce qu’on doit passer à travers toutes ces étapes pour chaque app où l’usager tournera le téléphone ou changera la langue de son téléphone

Bien sûr que non ! : Truc : Modifier le fichier manifest pour dire que l’activité ne se redémarre pas si on tourne le téléphone ( orientation, autres situations possibles .. )

**android:configChanges="orientation|screenSize|keyboard|keyboardHidden"**

**B- Qu’arrive-t-il si on ferme l’application sur le téléphone ?? on pers tout**

Solution : Pratiquer le processus de sérialisation pour conserver le nom de l’utilisateur même s’il ferme l’application

Où sérialiser ? onStop MainActivity

Où désérialiser ? onCreate MainActivity