RAPPORT PJI

Sujet 99 : *Tutoriels multi-écrans pour des applications Android*

Fouad EL ATI & Hajar BOUHALLAF
Master 1 Informatique – Lille1
Année universitaire 2017 – 2018

-

Auteur: Gérard PALIGOT

Responsable informatique : Cédric DUMOULIN

Formation : Formations en Informatique de Lille (FIL) - Batiment M3, Université Lille 1, Avenue Paul Langevin, 59650 Villeneuve-d'Ascq

Sommaire

I. Introduction

Dans le cadre de notre projet de PJI, nous nous intéressons à développer des tutoriels multi-écrans pour des applications Android. La plupart des personnes (des personnes âgées ou des enfants) ont des difficultés à manipuler correctement une application quelconque. Pour faire face à cette problématique, les tutoriels qui existent dans les applications permettent aux utilisateurs de mieux comprendre le fonctionnement de l'application choisie et de rendre la manipulation de ces application plus pratique et plus fluide.

Voici le lien de notre projet (les codes) est disponible sur .

Nous allons donc mettre en place des tutoriels dans une application quelconque. De plus, ces tutoriels sont compatibles et valables pour toutes les applications.

Nous utilisons Android version Android 8.0 (Oreo) et la librairie ShowcaseView conçu par Axel Curran : .

Ce projet reste pour nous important voire essentiel pour chacun de nos projets professionnels : obtenir des compétences de la programmation sous Android et nous souhaitons découvrir encore davantage.

Dans un premier temps, nous allons comprendre l'utilisation du ShowcaseView. Puis nous étudierons les différentes étapes de réalisation avec certaines difficultés rencontrées et les solutions apportées. Dernièrement, nous verrons la partie d'architecture logicielle avant de dresser un bilan de celui-ci.

II. Explication du besoin

L’objectif est de créer une application contenant des tutoriels multi-écrans. Un tutoriel comporte plusieurs écrans où chaque écran possède une démonstration c’est à dire dans chaque écran, il y a d'enchaînement des ciblages pour tous les composants par exemple montrer et expliquer à quoi servent les boutons.

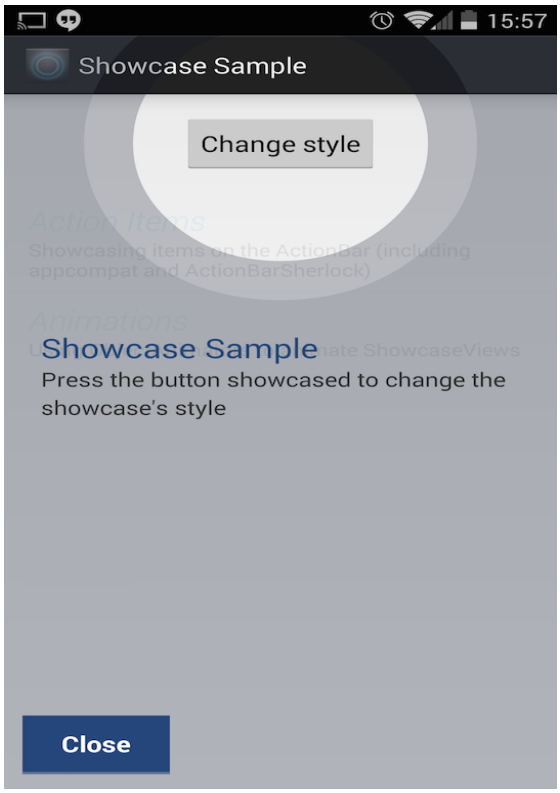
Pour cela, il faudra utiliser la bibliothèque ShowcaseView permettant de cibler tous les composants d’un écran courant.

III. Initiation ShowcaseView

La bibliothèque ShowcaseView (SVC) est une bibliothèque représentant des fonctionnalités permettant de mettre en évidence des parties spécifiques d’applications à l’utilisateur avec une superposition distinctive et attrayante. Cette bibliothèque est idéale aussi pour signaler les points d'intérêts pour les utilisateurs, les gestes ou les éléments obscurs mais utiles. Elle est compatible avec les versions à partir de l’API LEVEL 11+. Pour créer ShowcaseView, on procède comme suit :

```
new ShowcaseView.Builder(this)
    .setTarget(new ActionViewTarget(this, ActionViewTarget.Type.HOME))
    .setContentTitle("Showcase Sample")
    .setContentText("Press the button showcased to change the showcase’s style")
    .build();
```

Le résultat est le suivant :



- *ShowcaseView.Builder(this)* : on utilise le Builder pattern pour instancier la ShowcaseView depuis l'activité où on souhaite afficher la vue en paramètre. Dans notre cas, c'est l'activité courante.

- *.setTarget(new ActionViewTarget(this, ActionViewTarget.Type.HOME))* : permet de spécifier le Target(bouton) où on veut mettre la vue.

- .*setContentTitle("Showcase Sample")*: permet de définir le titre pour pouvoir l'afficher ensuite lorsque le ShowcaseView est créé.

- .setContentText("Press the button showcased to change the showcase's style") : permet de définir le texte accompagné du titre crée du ShowcaseView.

- .build(): permet de créer un ShowcaseView à partir du Builder.

IV. Différents étapes de réalisation

Pour mettre en place notre projet applicatif, nous avons suivi les différentes étapes dans l'ordre chronologique :

1. Création des écrans où pour chaque écran contient des boutons

Comme première étape de notre projet, nous avons commencé par créer quatre différentes activités : A , B , C et D. Chaque activité est un écran où chaque écran contient des boutons de manipulation permettant d'accéder à l'activité l'une des autres activités.

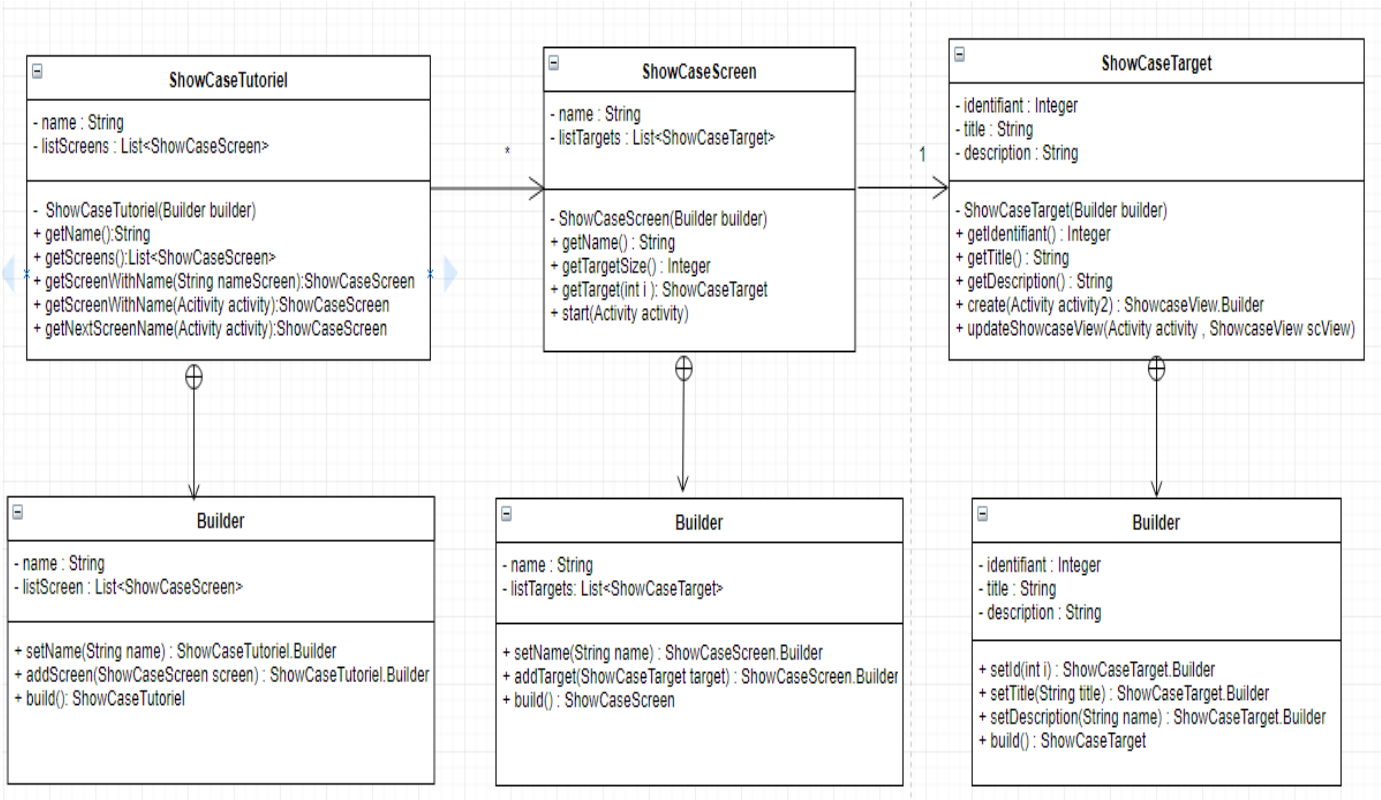
Afin de bien nous familiariser la bibliothèque ShowcaseView , nous avons créé, dans chaque activité, un objet ShowCaseView permettant de manipuler ses composants ou ses targets (boutons).

2. Création d'une liste de tutoriels

Dans cette étape, nous avons construit une liste des tutoriels où chaque tutoriel lancera un ensemble d'activités bien défini. Le problème que nous rencontrons est qu'à chaque fois, nous devons instancier un ShowcaseView par activité. De plus, imaginons que nous souhaitons créer 1000 tutoriels, la tâche du travail sera lourde (instancier plus de 1000 ShowcaseView pour les activités) et nous considérons que cela n'est pas une bonne idée.

3. Builder API

Afin de mieux gérer la création des tutoriels et de faire face à la problématique que nous rencontrons, nous avons décidé de créer une API. Nous proposons une solution de cette problématique comme le diagramme de classes associé :



Le diagramme de classe ci-dessus regroupe l'ensemble des classes suivantes :

- ShowCaseTutoriel : Cette classe permet de construire un tutoriel en utilisant le Builder Pattern contenant une liste d'écrans ShowCaseScreen (écrans) c'est à dire une liste d'écrans (activités).

- ShowCaseScreen : cette classe permet de construire un écran en utilisant le Builder Pattern contenant une liste de targets ShowCaseTarget (des composants ou des boutons) qui seront prochainement ciblé.

- ShowCaseTarget : cette classe permet de définir la cible d'un composant que nous allons ensuite l'afficher dans la vue (avec du ShowcaseView).

- ShowCaseTutorielSingleton : Cette classe nous permet d'utiliser l'objet ShowCaseTutoriel dynamiquement c'est à dire qu'elle nous permet d'accéder les contenus d'un objet ShowCaseTutorial dans n'importe quelle activité (écran). De plus, elle nous permet aussi de déterminer l'ensemble des activités (écrans) que nous allons les afficher et l'ensemble des composants que nous allons les cibler prochainement dans chaque activité (écran).

4. Lancement d’une activité avec le tutoriel

Dans cette partie , nous avons créer un tutoriel avec une seule interface en utilisant l’API de l’étape précédente plus précisément en instanciant un objet de la classe ShowCaseTutoriel et en utilisant la méthode du Builder, addScreen.

5. Lancement d’une activité avec le tuto pour une seule target

En créant le tutoriel avec une seule interface, on utilise la classe ShowCaseTarget permettant de créer la vue et de manipuler le bouton.

6. Lancement d’une activité avec le tuto pour de multiple targets

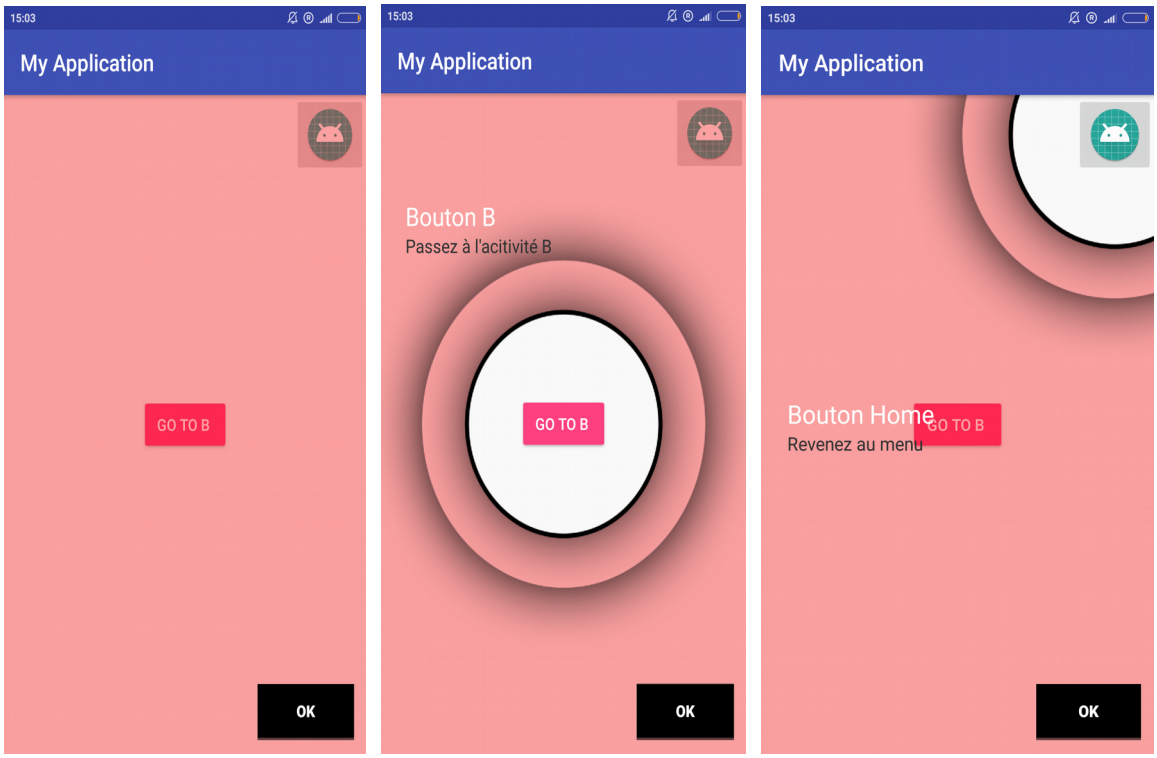
En utilisant l’étape précédente , on a pu afficher la vue et manipuler plusieurs composants de l’activité courante.

7. Lancement de plusieurs activités avec le tuto contenant plusieurs targets

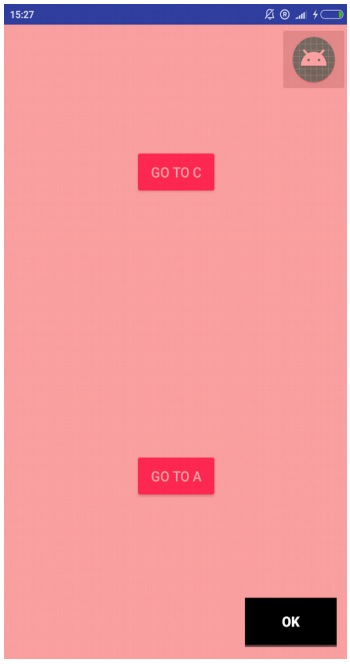
En se basant sur les étapes précédentes , on a pu afficher un tutoriel avec plusieurs activités et plusieurs targets (cibles) en utilisant toutes les classes et les méthodes de l’API.

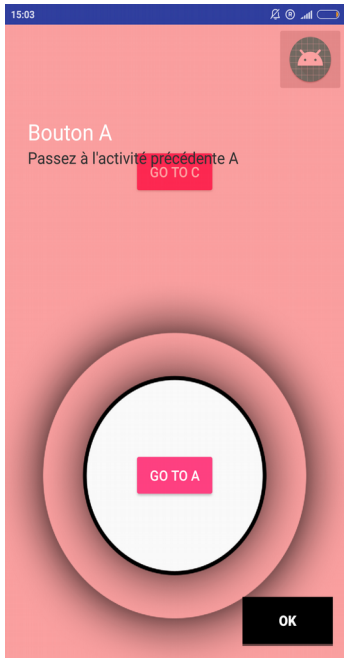
VI. Démonstration

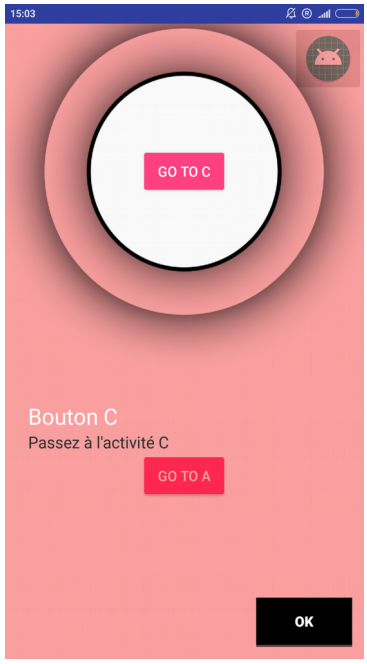
Dans cette partie , nous allons effectuer une petite démonstration de l'application créée afin de comprendre le fonctionnement. La vue commence à s'afficher à partir de l'activité A puis B, ensuite C et ainsi D . En arrivant à la dernière activité, on retourne au menu principal ou en faisant un back à chaque fois jusqu'à qu'on arrive au menu principal.

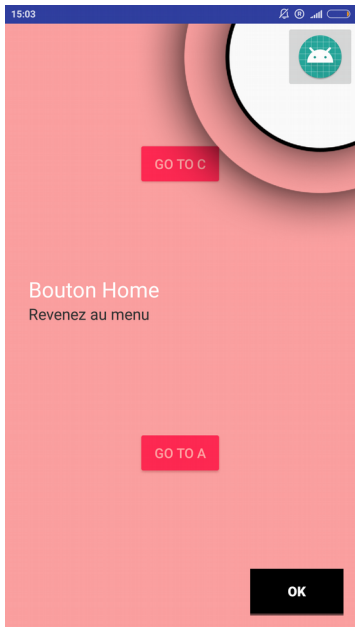


Activity A

1) 

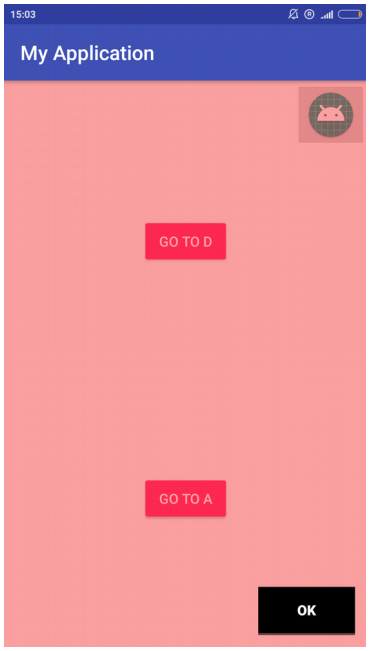
2) 

3) 

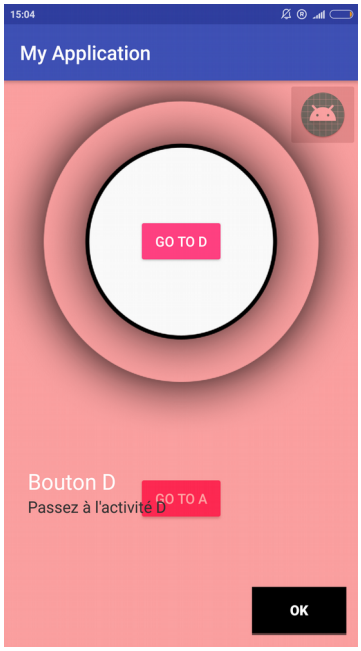
4) 

Activity B

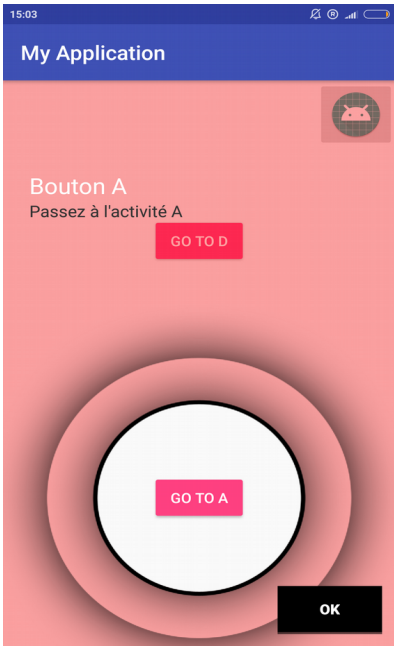
1)



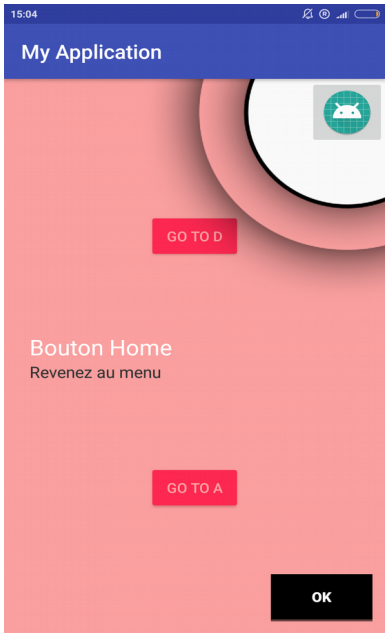
2)



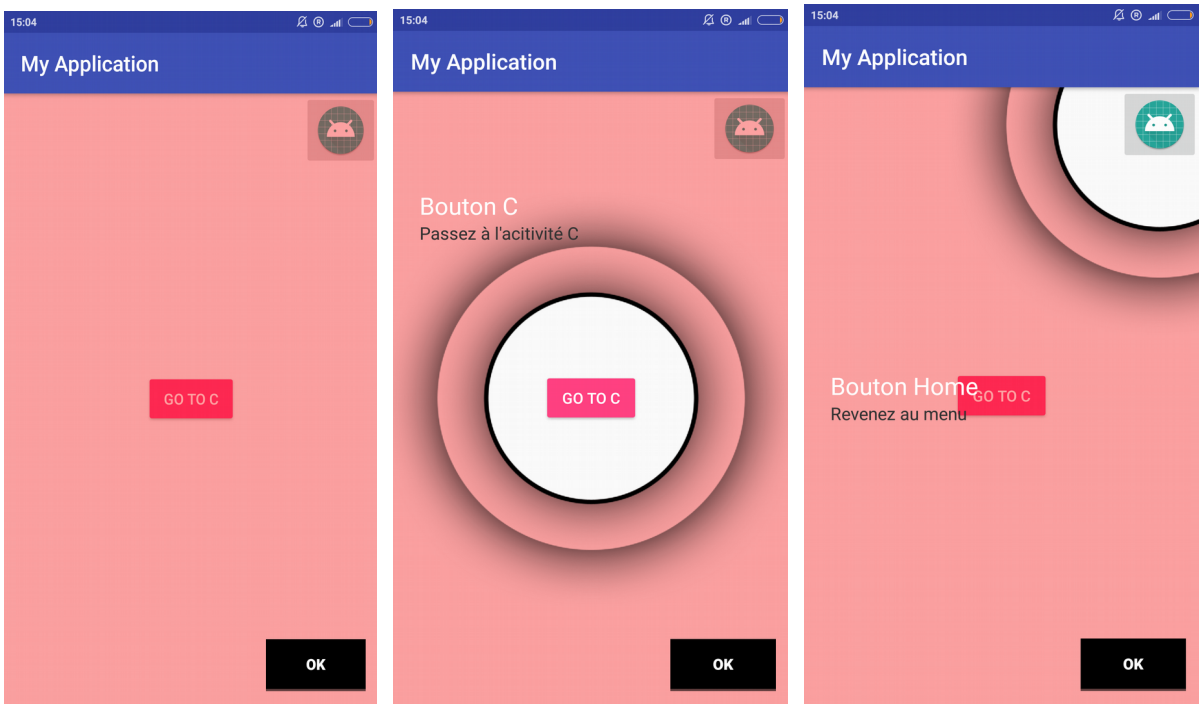
3)



4)



Activity C



Activity D

VII. Bilan

Pour tout résumer après la réalisation finale de mon application, nous sommes plutôt satisfait de nous par rapport les travails demandés. Tous les travails ont été fourni avec succès. De plus, durant notre réalisation de l'application, nous avons rencontré des difficultés à concevoir cet application au début du projet c'est à dire qu'au début du projet, nous n'avons pas encore défini correctement l'architecture logicielle comme il faut pour produire notre application.

Dernièrement, nous avons réparti notre travail en individuel et ainsi travaillé en équipe à Décathlon BTWIN Village. Nous avons constaté que travailler en équipe
est primordiale et la communication entre nous est importante afin d'assurer que l'avancement du projet se passe bien.

VIII. Conclusion

Au cours de ce travail , nous nous sommes intéressés en particulier à la programmation sous la plateforme Android , qui constitue une plateforme open-source , utilisés par des milliers de développeurs et utilisateurs mobiles , en particulier à l'utilisation du package ShowCaseView.

Pour répondre à la problématique cité dessus , nous avons développé une application Android proposant une liste de tutoriels ou chaque tutoriel permet de manipuler les activités : A, B, C et D en prenant en compte le tutoriel courant .

Pour atteindre nos objectifs, nous avons utilisé l'environnement de développement Android Studio, ainsi que la méthode UML pour modéliser notre API et poursuivre les étapes de réalisation.

Finalement , cette expérience nous a permis de comprendre le fonctionnement d'un projet informatique et comment répondre à une problématique ainsi que approfondir nos connaissances en développement mobile sous Android.

IX. Remerciements

Nous tenons à remercier Monsieur Cédric DUMOULIN et Monsieur Gérard PALIGOT qui nous ont conseillé, soutenu et ont répondu régulièrement à nos questions tout au long de notre semestre et qui ont participé de différentes façons (via leur suivi) à la réussite de mon projet.