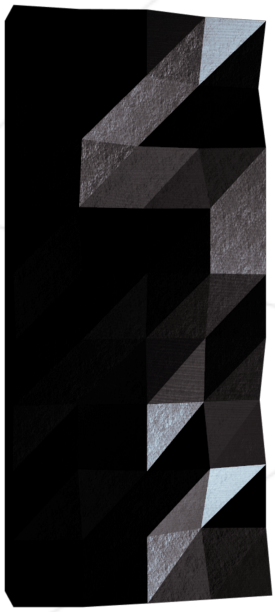


ft_hangouts

Le Grand Ambassadeur

Pierre-Elie Kesslassy pkesslas@student.42.fr

Summary: Le but de ce projet est de vous familiariser avec le système Android en réalisant une application de gestion de contacts.

Contents

Chapter I

Préambule

Pas de neige sur le mont Fuji,
ni de fleurs dans les cerisiers,
pas de gazon dans mes plates-bandes,
son règne est minéral.

Magique est le liquide,
magique est la source,
magique est le cristal,
magique est le jardin.

Le jardinier magicien
cristal, cristal, cristal oh cristal
ho ho ho ho ho ho ho ho ho ho ho ho ho ho ho ho
hohohohohohohohohohohohohohohohohohohohohohohohohohohohohohohoho
hohohohohohohohohohohohohohohohohohohohohohohoho ho ho ho ho

cristal, cristal, cristal, cristal
Tu ne peux pas le toucher.
Tu n'es pas prêt, tu peux juste le regarder,
Tes yeux sont prêts, tu peux le supporter.
Ouaaaa
Oh c'est beau, c'est beau, c'est beau

Le jardinier magicien
n'a pas besoin d'engrais dans son jardin.
brillant comme les étoiles,
brillant comme une mer d'huile,
brillant comme les dents d'un gros chien.

brillant comme les insectes,
brillant comme une saucisse,
brillant comme la peau d'un ado.

cristal oh cristal
cristal oh cristal
cristal oh cristal
cristal oh crya ha ha

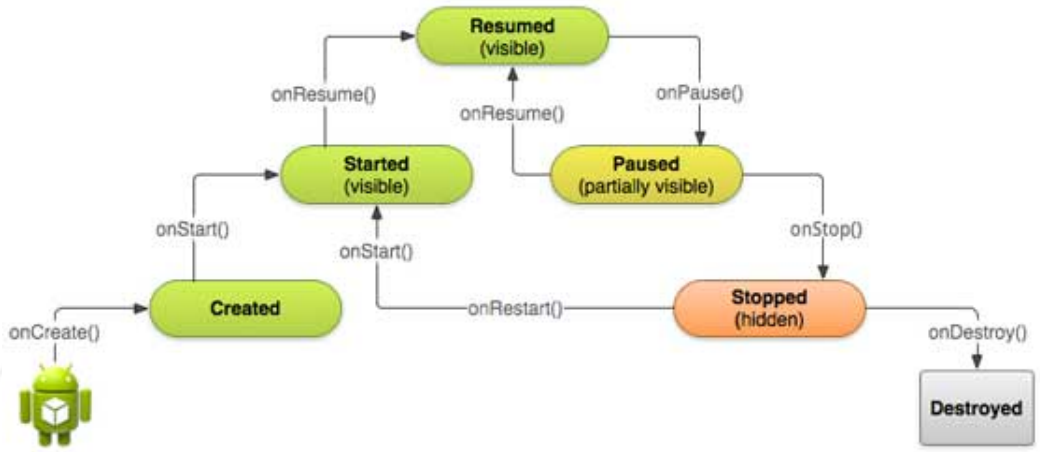
Salut C'est Cool - *Le jardinier magicien* lien.

Chapter II

Introduction

Pour ce projet, vous devrez réaliser une application Android permettant de créer un contact et de dialoguer avec par SMS.

Le but ici est de comprendre comment fonctionne une application Android, comment Android gère votre application et comment utiliser le SDK.



Chapter III

Objectifs

Vous allez devoir réaliser diverses tâches qui vous feront comprendre le fonctionnement d'une application Android en JAVA ou Kotlin. Le but est d'avoir une application permettant de créer un contact (avec au minum 5 informations), de l'éditer et de le supprimer. Une fois un contact enregistré il devra être possible de dialoguer avec lui en utilisant les SMS.

Les contacts sont enregistrés de manière persistante (base de donnée SQLite, n'utilisez pas la table de contact partagée, mais créez bien la votre). Un résumé de chaque contact sera présent sur l'accueil de l'application sous forme de liste, un clic sur un résumé permettra de voir toutes les informations concernant le contact correspondant.

Votre application devra également fonctionner sous deux langues dont une par défaut (changez la langue du système pour tester). Lorsque vous êtes sur l'accueil et que vous passez l'application en arrière-plan, la date sera sauvegardée et affichée dans un *toast* lors de votre retour au premier plan. Un menu permettra de changer la couleur du header de l'application. Et pour terminer, l'icone de l'application sera le logo de 42.

Chapter IV

Consignes générales

- Ce projet ne sera évalué que par des humains.
- Le projet devra être en JAVA ou en Kotlin
- Aucune librairie externe (même pour le design) n'est autorisée.

Il est fortement conseillé d'utiliser Android Studio comme IDE. Faites attention, le plugin ADT pour Eclipse n'est plus supporté par Google.

Chapter V

Partie obligatoire

Voici ce que vous devrez réaliser:

- Création d'un contact.
- Edition d'un contact.
- Suppression d'un contact.
- Page d'accueil avec un résumé de chaque contact.
- Pouvoir recevoir les SMS de vos contacts enregistrés.
- Pouvoir envoyer des SMS à vos contacts.
- Un menu doit être disponible pour changer la couleur du header.
- L'application devra supporter deux langues.
- Afficher l'heure de la mise en arrière plan lors du retour sur l'application.
- L'application fonctionne en mode portrait et paysage.
- Le logo de l'application sera celui de 42.

Chapter VI

Partie bonus

- Avoir une photo pour les contacts.
- A la réception d'un SMS, un contact avec le numéro en nom est directement créé
- C'est beau ! Le Material Design c'est cool.
- On peut appeller le contact.

Faites-vous plaisir, plein de choses peuvent améliorer l'application.

Chapter VII

Émulation

Cette partie vous explique comment utiliser un émulateur Android.

L'émulateur fourni avec Android Studio ne fonctionne pas pour le moment sur les dumps. Je vous invite à utilisez Android Studio sur une VM ou un vrai device si vous voulez tester les SMS.

Plusieurs émulateurs Android existent, certains plus ou moins biens : GenyMotion est très performant et facile à mettre en place (VirtualBox est requis) mais ne permet pas l'envoi de SMS dans sa version gratuite.
L'AVD (Android Virtual Device) qui est fourni avec Android Studio n'est pas le plus performant, mais vous avez le contrôle. Il se met en place tout seul avec Android Studio, c'est intuitif.

Pour envoyer un SMS depuis l'AVD :

```
srudent@e1r1p1$ telnet localhost {port, (titre de la fenetre)}
#Trying 127.0.0.1...
#Connected to localhost.
#Escape character is '^]'.
#Android Console: type 'help' for a list of commands
#OK
sms send {numero} {message}
```

Pour un appel :

```
srudent@e1r1p1$ telnet localhost {port, (titre de la fenetre)}
#[...]
gsm call {numero}
```

Lorsqu'une image est en cours d'éxecution, l'ordinateur la reconnait comme un téléphone. Choississez simplement l'image lorsque vous lancez l'application depuis Android Studio.

Chapter VIII

Rendu et peer-évaluation

Attention à votre dépot, plein de fichiers plus ou moins utiles sont générés dans votre projet. Pensez à bien configurer votre .gitignore hint.

Pour la correction le projet sera compilé et installé avec:

```
$./gradlew installDebug
```