ALKendi
CENTRE DE PRÉPARATION BTS

Développement Mobile

TP7

Nom complet: Mohamed Amine BARAKAT

16 Janvier 2026

2025/2026

1. Contexte de l'application:

Le code fourni est celui de LoginActivity.java, une activité Android. Cette activité présente une interface de connexion simple à l'utilisateur. Elle contient deux champs de saisie (EditText) pour le nom d'utilisateur et le mot de passe, ainsi qu'un bouton de connexion (Button).L'objectif de cette activité est de permettre à l'utilisateur de s'authentifier. Après une validation basique (vérification que les champs ne sont pas vides), l'application redirige l'utilisateur vers l'écran principal, représenté par MainActivity.

1. Interface de Connexion:

(activity_login.xml)Ce fichier correspond à LoginActivity.java. Il doit contenir deux champs de texte pour le nom d'utilisateur et le mot de passe, ainsi qu'un bouton pour se connecter.Chemin du fichier : app/src/main/res/layout/activity_login.xml

2. Interface Principale:

(activity_main.xml)Ce fichier correspond à MainActivity.java (qui est appelée après une connexion réussie). Pour l'instant, nous pouvons y mettre un simple message de bienvenue.Chemin du fichier : app/src/main/res/layout/activity_main.xml

Rôle du cycle de vie dans l’application:

Le cycle de vie d’une *Activity* Android représente l’ensemble des états par lesquels passe une application au cours de son utilisation. Il est entièrement géré par le système Android, qui appelle automatiquement des méthodes spécifiques en fonction des actions de l’utilisateur et des contraintes du système.

Dans cette application, le cycle de vie joue un rôle essentiel pour comprendre et maîtriser le comportement de l’application lors des différentes situations telles que le lancement, la mise en arrière-plan, le retour à l’application, la rotation de l’écran ou la fermeture définitive.

Chaque méthode du cycle de vie est utilisée de manière cohérente avec l’état de l’application :

- **onCreate()** permet d’initialiser l’interface graphique, de lier les composants et de récupérer les données transmises entre les activités.
- **onStart()** indique que l’application devient visible à l’écran.
- **onResume()** confirme que l’application est active et prête à recevoir les interactions de l’utilisateur.
- **onPause()** signale que l’application perd le focus, ce qui permet d’effectuer des actions de sauvegarde ou de journalisation.
- **onStop()** correspond au passage de l’application en arrière-plan, informant l’utilisateur que l’application n’est plus visible.
- **onRestart()** marque le retour de l’utilisateur vers l’application après une interruption.
- **onDestroy()** est appelée lors de la fermeture définitive de l’application et permet de libérer les ressources utilisées.

Ainsi, le cycle de vie garantit une gestion correcte des ressources, améliore la stabilité de l’application et assure une expérience utilisateur fluide. Dans ce TP, son implémentation permet d’observer concrètement l’enchaînement des états et de mieux comprendre le fonctionnement interne d’une application Android.