

TP - Le Cycle de Vie d'une Activité Android

Étape 1 : Création du projet

Commencez par créer un nouveau projet Android dans Android Studio avec une "Empty Activity". Donnez-lui le nom `CycleVieActivite`, et utilisez le langage de votre choix (Java ou Kotlin).

Étape 2 : Comprendre le Cycle de Vie d'une Activité

Les méthodes clés du cycle de vie d'une activité sont :

- **onCreate()** : Lors de la création de l'activité, vous initialisez les vues.
- **onStart()** : Juste avant que l'activité devienne visible.
- **onResume()** : Quand l'activité commence à interagir avec l'utilisateur.
- **onPause()** : Lorsque l'activité perd le focus.
- **onStop()** : Lorsque l'activité devient invisible.
- **onRestart()** : Quand l'activité revient après être arrêtée.
- **onDestroy()** : Avant que l'activité ne soit détruite.

Étape 3 : Implémentation du Cycle de Vie

Dans `MainActivity.java`, redéfinissez les méthodes pour observer l'appel du cycle de vie.

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Log.d("CycleVie", "onCreate appelée");
    }

    @Override
    protected void onStart() {
        super.onStart();
        Log.d("CycleVie", "onStart appelée");
    }
}
```

```
}
```

```
@Override
```

```
protected void onResume() {
```

```
    super.onResume();
```

```
    Log.d("CycleVie", "onResume appelée");
```

```
}
```

```
@Override
```

```
protected void onPause() {
```

```
    super.onPause();
```

```
    Log.d("CycleVie", "onPause appelée");
```

```
}
```

```
@Override
```

```
protected void onStop() {
```

```
    super.onStop();
```

```
    Log.d("CycleVie", "onStop appelée");
```

```
}
```

```
@Override
```

```
protected void onRestart() {
```

```
    super.onRestart();
```

```
    Log.d("CycleVie", "onRestart appelée");
```

```
}
```

```
@Override
```

```
protected void onDestroy() {
```

```
        super.onDestroy();  
        Log.d("CycleVie", "onDestroy appelée");  
    }  
}
```

Étape 4 : Gérer les changements d'orientation

Testez la rotation de l'écran et observez les appels des méthodes du cycle de vie. Utilisez `onSaveInstanceState()` pour sauvegarder l'état de l'activité et `onRestoreInstanceState()` pour le restaurer après une rotation.

Étape 5 : Créer une seconde activité

Créez une nouvelle activité `SecondActivity` et lancez-la depuis `MainActivity`. Utilisez un `Intent` pour passer d'une activité à l'autre.

```
Intent intent = new Intent(MainActivity.this, SecondActivity.class);  
startActivity(intent);
```

Étape 6 : Retour à MainActivity

Ajoutez un bouton dans `SecondActivity` pour revenir à `MainActivity` et observer les transitions dans le log.

Conclusion

Vous avez maintenant une compréhension du cycle de vie d'une activité dans Android et comment gérer les transitions entre les activités et leurs états. Cela vous permet d'optimiser la gestion des ressources et d'améliorer l'expérience utilisateur.