

Guide : Ajouter une nouvelle interface dans MVVM Android

1. Créer le package UI

- Package : ui/driver (exemple pour une interface conducteur)
- Clic droit sur ui -> New > Package -> Nom : driver

2. Créer l'Activity

- Clic droit sur ui/driver -> New > Java Class -> Nom : DriverActivity

```
package com.example.drive_2_go.ui.driver;
```

```
import android.os.Bundle;
```

```
import androidx.appcompat.app.AppCompatActivity;
```

```
public class DriverActivity extends AppCompatActivity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_driver);  
    }  
}
```

3. Créer le layout XML

- Chemin : res/layout/activity_driver.xml

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:orientation="vertical"  
    android:padding="16dp">  
  
    <TextView  
        android:id="@+id/tvDriverTitle"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:text="Liste des Conducteurs"  
        android:textSize="24sp"/>  
</LinearLayout>
```

4. Créer un ViewModel (optionnel)

- Package : viewmodel
- Java Class : DriverViewModel

```

package com.example.drive_2_go.viewmodel;

import android.app.Application;
import androidx.annotation.NonNull;
import androidx.lifecycle.AndroidViewModel;
import androidx.lifecycle.LiveData;
import androidx.lifecycle.MutableLiveData;
import java.util.List;

public class DriverViewModel extends AndroidViewModel {
    private MutableLiveData<List<String>> drivers;

    public DriverViewModel(@NonNull Application application) {
        super(application);
        drivers = new MutableLiveData<>();
    }

    public LiveData<List<String>> getDrivers() {
        return drivers;
    }

    public void loadDrivers() {
        // Charger les données depuis Repository
    }
}

```

5. Créer Model et Repository (si nécessaire)

- data/model/Driver.java -> objet métier
- data/repository/DriverRepository.java -> gestion des données

6. Créer Adapter (si affichage d'une liste)

- Package : ui/adapter -> DriverAdapter.java
- Même logique que CarAdapter.java pour RecyclerView

7. Ajouter la navigation

```

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

```

Résumé des étapes:

1. Créer package UI (ui/driver)
2. Créer Activity Java (DriverActivity.java)
3. Créer layout XML (activity_driver.xml)
4. Créer ViewModel (DriverViewModel.java)

5. Créer Model et Repository (Driver.java et DriverRepository.java)
6. Créer Adapter (DriverAdapter.java)
7. Ajouter navigation avec Intent ou NavController