

TRAVAUX PRATIQUES – FILIÈRE DÉVELOPPEMENT DIGITAL

Option – Applications mobiles

M205 – Développer des interfaces utilisateurs sous Android

57 heures

SOMMAIRE



1. Créer des interfaces graphiques simples

- Activité n° 1 : Création de plusieurs interfaces graphiques et gestion des évènements
- Activité n° 2 : Création d'interfaces utilisant divers layouts
- Activité n° 3 : Création de plusieurs contrôles dont des RecyclerView

2. Créer des interfaces graphiques avancées

- Activité n° 1 : Création de plusieurs vues personnalisées
- Activité n° 2 : Dessiner des vues sur Canvas
- Activité n° 3 : Création de plusieurs animations entre plusieurs activités
- Activité n° 4 : Création de plusieurs animations (Zoom In/Out Animation, Rotate Animation…)

MODALITÉS PÉDAGOGIQUES



1

LE GUIDE DE SOUTIEN

Il contient le résumé théorique et le manuel des travaux pratiques



2

LA VERSION PDF

Une version PDF est mise en ligne sur l'espace apprenant et formateur de la plateforme WebForce Life



3

DES CONTENUS TÉLÉCHARGEABLES

Les fiches de résumés ou des exercices sont téléchargeables sur WebForce Life



4

DU CONTENU INTERACTIF

Vous disposez de contenus interactifs sous forme d'exercices et de cours à utiliser sur WebForce Life



5

DES RESSOURCES EN LIGNES

Les ressources sont consultables en synchrone et en asynchrone pour s'adapter au rythme de l'apprentissage

Copyright - Tout droit réservé - OFPPT



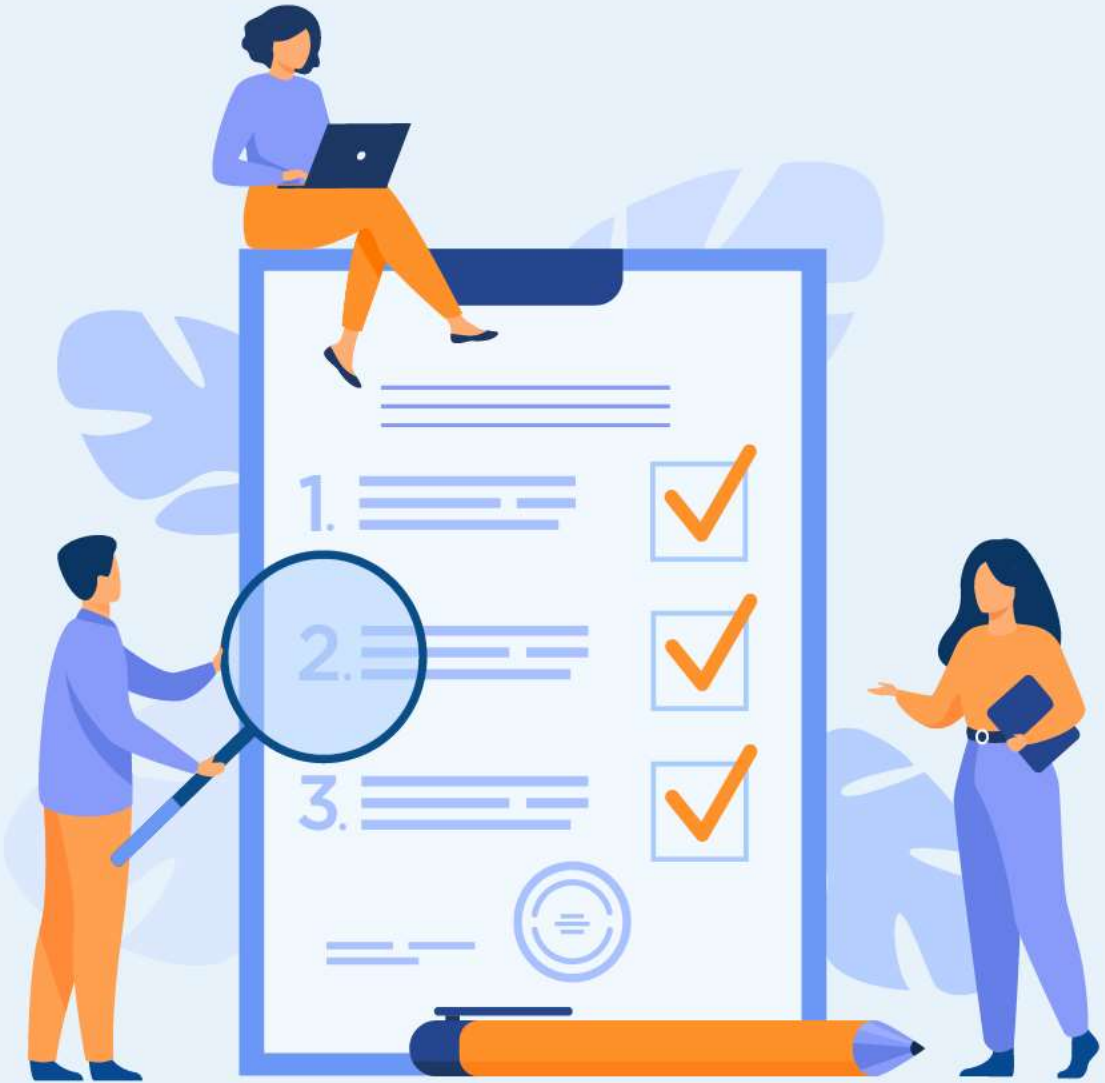
PARTIE 1

Créer des interfaces graphiques simples

Dans ce module, vous allez :

- Création facile de plusieurs interfaces graphiques ;
- Création correcte d'interfaces utilisant divers layouts.

29 heures



ACTIVITÉ n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

Compétences visées :

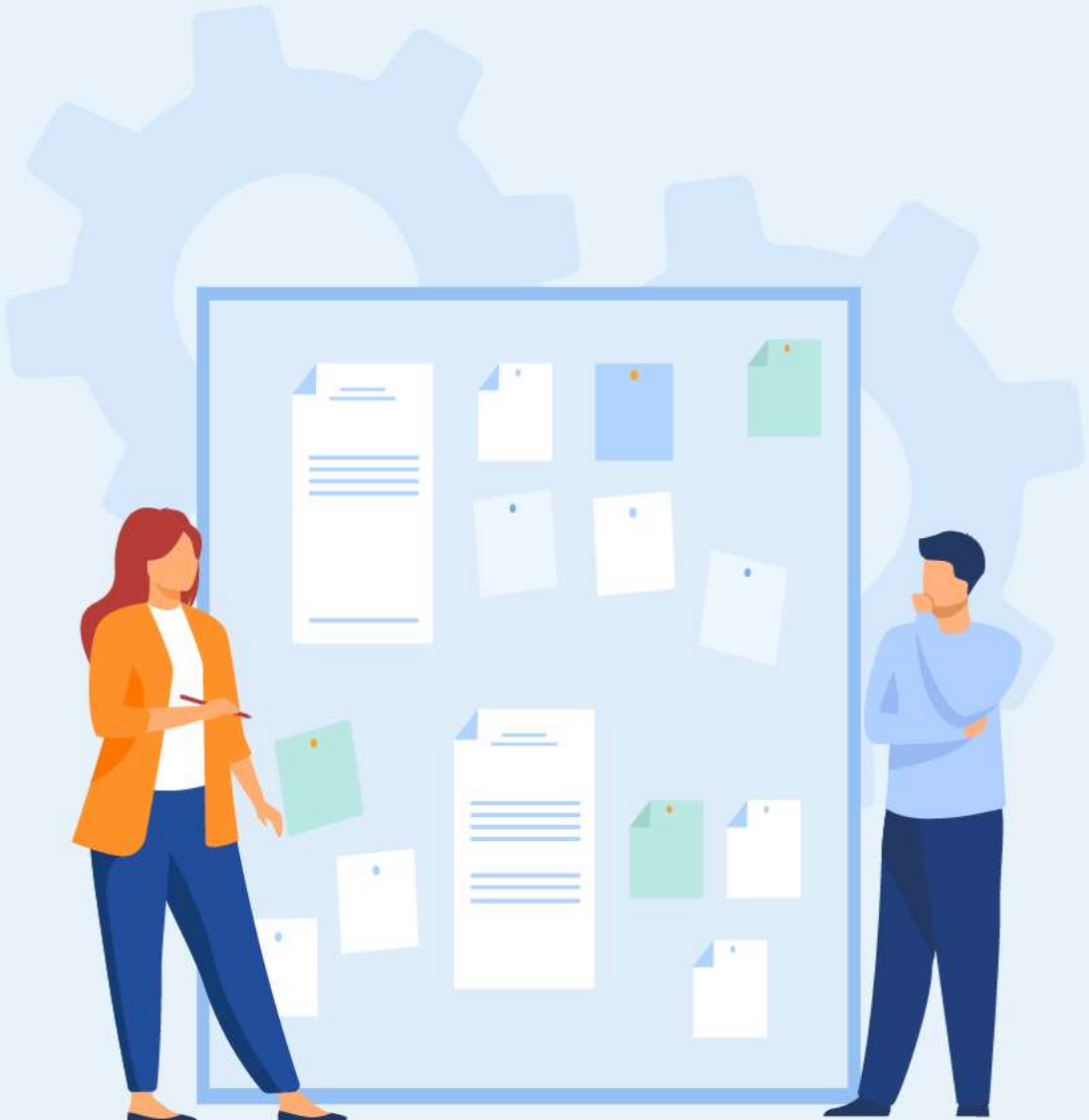
- Manipulation des évènements ;

Recommandations clés :

- Révision générale du résumé théorique ;
- Suivre les instructions du TP et organiser le dossier de travail.

5 heures

CONSIGNES



Pour le formateur :

- Faire un rappel de cours ;
- Demander aux stagiaires de suivre les étapes de l'activité.

Pour l'apprenant :

- Suivre les étapes indiquées dans l'activité.

Conditions de réalisation :

- Support de résumé théorique accompagnant ;
- Accès Internet pour installer les dépendances.

Critères de réussite :

- Le stagiaire doit être capable de :
  - manipuler les évènements ;
  - créer un menu ;
  - utiliser un calendrier.

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

PARTIE 1

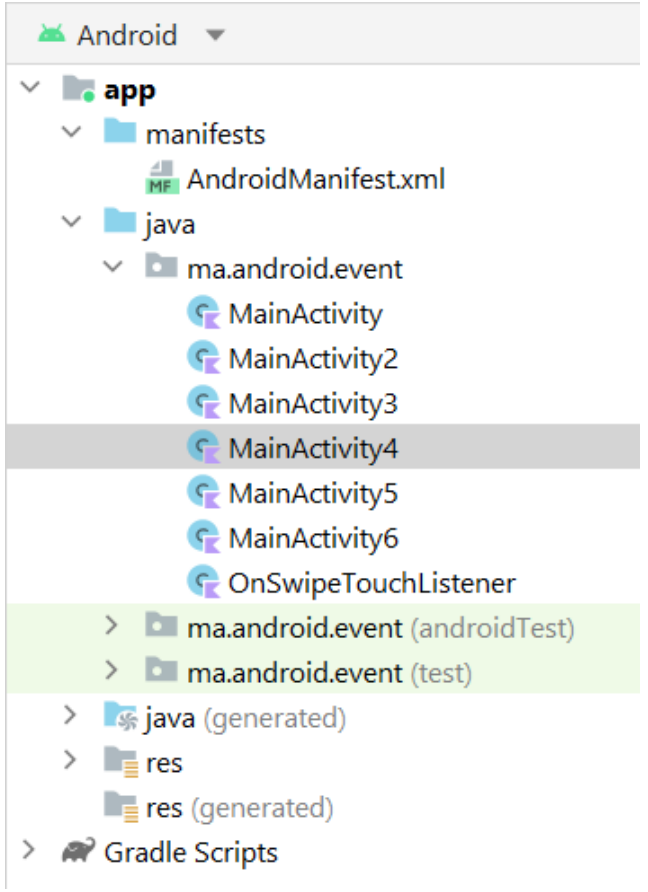
Enoncé

L'objectif de cette activité est de créer plusieurs interfaces graphiques et de gérer les évènements suivants :

- Click
- LongClick
- Swipe (gauche et droite)
- TextChange
- OnItemSelected
- OnTouch
- CreateContextMenu
- OndateSet

La navigation entre les activités sera effectuée en utilisant les évènements Swipe (gauche et droite).

Structure de projet et Démonstration





Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

Tâche 1 : Click et LongClick

Il est demandé de créer une activité (**MainActivity**) comprenant un bouton sur lequel seront appliqués les évènements de click suivants : **OnClickListener** et **OnLongClickListener**, comme illustré dans la figure ci-jointe.



PARTIE 1

Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

PARTIE 1

Etape 1 : Modifier le fichier layout

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity"
    android:id="@+id/activity1">

    <Button
        android:id="@+id/click"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="276dp"
        android:text="Click Button"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="136dp"
        android:text="Click and Long Click "
        android:textSize="26dp"
        android:textStyle="bold"
        style="@style/MaterialAlertDialog.MaterialComponents.Title.Icon"
        app:layout_constraintBottom_toTopOf="@+id/click"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.16" />

    <TextView
        android:id="@+id/longText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="120dp"
        android:text=""
        android:textColor="@color/purple_200"
        android:textSize="22dp"
        android:textStyle="italic"
        android:padding="5dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/click"
        app:layout_constraintVertical_bias="0.141" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

Etape 2 : Modifier le code de la classe activité

```
class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val avtivity1 = findViewById<ConstraintLayout>(R.id.activity1)
        var click = findViewById<Button>(R.id.click)
        val longText = findViewById<TextView>(R.id.longText)

        click.setOnClickListener {
            Toast.makeText(this, "Button click ", Toast.LENGTH_SHORT).show()
            longText.text = "Simple Click On Button "
        }

        click.setOnLongClickListener {
            Toast.makeText(this@MainActivity,"Button Long click",Toast.LENGTH_SHORT).show()
            longText.text = "Long Click On Button"
             true
        }
```

```
 avtivity1.setOnTouchListener(object :OnSwipeTouchListener(this@MainActivity)
{
            override fun onSwipeLeft(): Boolean {
                val i = Intent(this@MainActivity, MainActivity2::class.java )
                startActivity(i)
                return true;
            }
        })


    }
```

La classe **OnSwipeTouchListener** est présentée dans la diapositive suivante.

PARTIE 1

Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

PARTIE 1

Etape 3 : Créer la classe OnSwipeTouchListener

```
open class OnSwipeTouchListener(val context: Context?) : View.OnTouchListener {
    companion object {
        private const val SwipeThreshold = 100
        private const val SwipeVelocityThreshold = 100
    }

    private val gestureDetector = GestureDetector(context, GestureListener())

    override fun onTouch(v: View, event: MotionEvent): Boolean {
        return gestureDetector.onTouchEvent(event)
    }

    open fun onSwipeRight(): Boolean { return false }
    open fun onSwipeLeft(): Boolean { return false }
    open fun onSwipeUp(): Boolean { return false }
    open fun onSwipeDown(): Boolean { return false }

    private inner class GestureListener : GestureDetector.SimpleOnGestureListener() {
        override fun onDown(e: MotionEvent): Boolean {
            return true
        }

     override fun onFling(
            e1: MotionEvent,
            e2: MotionEvent,
            velocityX: Float,
            velocityY: Float,
        ): Boolean {
            try {
                val diffY = e2.y - e1.y
                val diffX = e2.x - e1.x

                if (abs(diffX) > abs(diffY)) {
                    if (abs(diffX) > SwipeThreshold && abs(velocityX) > SwipeVelocityThreshold)
{
                        return when {
                            diffX > 0 -> onSwipeRight()
                            else -> onSwipeLeft()
                        }
                    }
                } else if (abs(diffY) > SwipeThreshold && abs(velocityY) >
SwipeVelocityThreshold) {
                    return when {
                        diffY > 0 -> onSwipeDown()
                        else -> onSwipeUp()
                    }
                }
            } catch (exception: Exception) {
                exception.printStackTrace()
            }
            return false
        }
    }
```

Copyright - Tout droit réservé - OFPPT

Etape 3 : Créer la classe OnSwipeTouchListener

Ce code définit une classe appelée **OnSwipeTouchListener** qui implémente l'interface **View.OnTouchListener**. La classe contient une companion object qui définit deux constantes, **SwipeThreshold** et **SwipeVelocityThreshold**, ainsi qu'une instance de **GestureDetector** appelée **gestureDetector**. La classe contient également une série de méthodes vides, **onSwipeRight**, **onSwipeLeft**, **onSwipeUp**, et **onSwipeDown**, qui seront appelées lorsque l'utilisateur effectue un "swipe" (un geste de glissement) vers la droite, vers la gauche, vers le haut ou vers le bas respectivement.

La méthode **onTouch** de **OnSwipeTouchListener** est appelée lorsqu'un événement de toucher est détecté sur une vue. Elle appelle la méthode **onTouchEvent** de **gestureDetector** avec l'événement de toucher en tant qu'argument, et renvoie le résultat de cet appel.

La classe interne **GestureListener** est une sous-classe de **GestureDetector.SimpleOnGestureListener** qui surcharge la méthode **onFling**. Cette méthode est appelée lorsqu'un mouvement de glissement (un "swipe") est détecté. Elle calcule la différence entre la position de toucher initiale (**e1**) et la position de toucher finale (**e2**) en utilisant les coordonnées **x** et **y** de chacun. Si la différence en **x** est supérieure à la différence en **y**, cela signifie que l'utilisateur a effectué un swipe horizontal. Dans ce cas, la méthode vérifie si la différence en **x** est supérieure à **SwipeThreshold** et si la vitesse en **x** (**velocityX**) est supérieure à **SwipeVelocityThreshold**. Si ces conditions sont remplies, la méthode appelle **onSwipeRight** ou **onSwipeLeft** en fonction de la direction du swipe. Si la différence en **y** est supérieure à la différence en **x**, cela signifie que l'utilisateur a effectué un swipe vertical. Dans ce cas, la méthode vérifie si la différence en **y** est supérieure à **SwipeThreshold** et si la vitesse en **y** (**velocityY**) est supérieure à **SwipeVelocityThreshold**. Si ces conditions sont remplies, la méthode appelle **onSwipeUp** ou **onSwipeDown** en fonction de la direction du swipe. Si aucun de ces cas n'est vrai, la méthode renvoie **false**.

Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

Tâche 2 : TextChange

Il est demandé de créer une activité (**MainActivity2**) comprenant un champ de saisie de texte. Il est également demandé de créer un programme affichant chaque lettre saisie dans ce champ de texte dans un champ texte, comme illustré dans la figure ci-jointe.



PARTIE 1

Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

PARTIE 1

Etape 1 : Modifier le fichier layout

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity2"
    android:id="@+id/activity2">

    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="140dp"
        android:text="EditText Events"
        style="@style/MaterialAlertDialog.Material3"
        android:textSize="28dp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="80dp"
        android:text="TextView"
        android:textSize="26dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView4" />

    <EditText
        android:id="@+id/input"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="296dp"
        android:ems="10"
        android:hint="text value"
        android:inputType="textPersonName"
        android:text=""
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.532"
        app:layout_constraintStart_toStartOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

Etape 2 : Modifier le code de la classe activité

```
class MainActivity2 : AppCompatActivity() {
    @SuppressLint("ClickableViewAccessibility")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main2)
        val myInput = findViewById<EditText>(R.id.input)
        val textView = findViewById<TextView>(R.id.text)
        val avtivity2 = findViewById<ConstraintLayout>(R.id.activity2)

        myInput.addTextChangedListener(object :TextWatcher {
            override fun beforeTextChanged(p0: CharSequence?, p1: Int, p2: Int, p3: Int) {
                //
            }
            override fun onTextChanged(p0: CharSequence?, p1: Int, p2: Int, p3: Int) {
                textView.setText("Text : " + p0)
            }
            override fun afterTextChanged(p0: Editable?) {
                //
            }
        })
        myInput.setOnFocusChangeListener(object :OnFocusChangeListener{
            override fun onFocusChange(p0: View?, p1: Boolean) {
                if(p1) {
                    Toast.makeText(this@MainActivity2, "Edit Text As Focus",Toast.LENGTH_SHORT).show()
                }
                else{
                    Toast.makeText(this@MainActivity2, "EditText Lost Focus", Toast.LENGTH_SHORT).show()
                }
            }
        })
        myInput.setOnKeyListener(View.OnKeyListener{v, keyCode, event ->
            if (keyCode == KeyEvent.KEYCODE_ENTER && event.action ==
KeyEvent.ACTION_UP) {
                Toast.makeText(this@MainActivity2,"Key
Enter",Toast.LENGTH_SHORT).show()
                return@OnKeyListener true
            }
            false
        })

        avtivity2.setOnTouchListener(object : OnSwipeTouchListener(this@MainActivity2) {
            override fun onSwipeRight(): Boolean {
                val i = Intent(this@MainActivity2,MainActivity::class.java )
                startActivity(i)
                return true;
            }
            override fun onSwipeLeft(): Boolean {
                val i = Intent(this@MainActivity2,MainActivity3::class.java )
                startActivity(i)
                return true;
            }
        })
    }
}
```

PARTIE 1

Copyright - Tout droit réservé - OFPPT

Etape 2 : Modifier le code de la classe activité

Ce code définit une activité appelée **MainActivity2**. Dans la méthode **onCreate**, cette activité instancie un champ de saisie de texte (**myInput**), un champ de texte (**textView**) et un layout (**avtivity2**) en utilisant les ressources de layout définies dans **activity_main2.xml**.

Puis, elle ajoute un **TextWatcher** à **myInput**, qui est un objet qui peut être utilisé pour surveiller les changements de texte dans un champ de saisie de texte. Le **TextWatcher** surcharge les méthodes **beforeTextChanged**, **onTextChanged**, et **afterTextChanged**, qui sont appelées respectivement avant, pendant et après que le texte dans **myInput** change. Dans la méthode **onTextChanged**, le texte actuel de **myInput** est affiché dans **textView**.

Ensuite, **MainActivity2** définit un **OnFocusChangeListener** sur **myInput**, qui est un objet qui peut être utilisé pour surveiller les changements de focus dans un champ de saisie de texte. Lorsque **myInput** gagne ou perd le focus, la méthode **onFocusChange** est appelée et un toast (un message temporaire) est affiché pour indiquer si **myInput** a gagné ou perdu le focus.

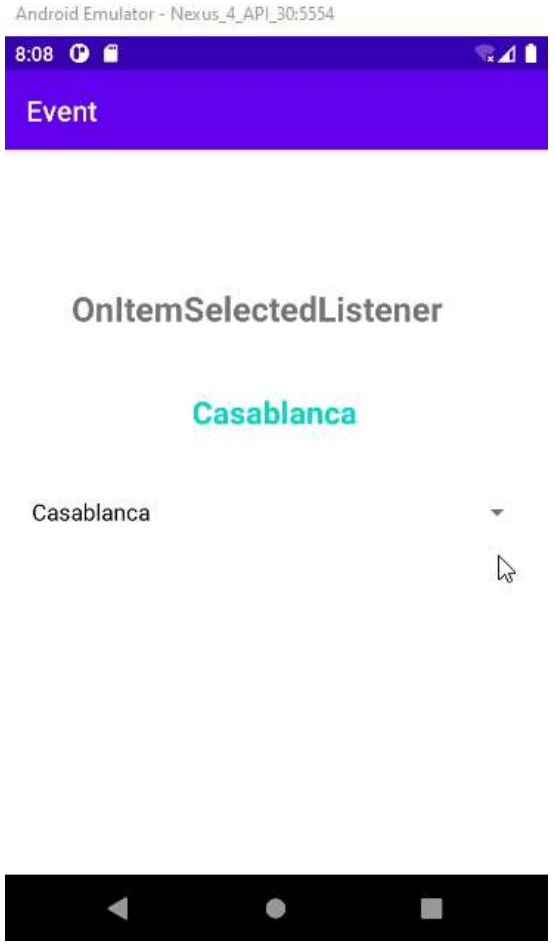
Ensuite, **MainActivity2** définit un **OnKeyListener** sur **myInput**, qui est un objet qui peut être utilisé pour surveiller les touches de clavier appuyées sur un champ de saisie de texte. Lorsque l'utilisateur appuie sur la touche "Entrée" dans **myInput**, la méthode **onKey** est appelée et un toast est affiché pour indiquer que la touche "Entrée" a été appuyée.

Enfin, **MainActivity2** définit un **OnSwipeTouchListener** sur **avtivity2**, qui est un objet qui peut être utilisé pour détecter les swipes (gestes de glissement) sur un layout. Si l'utilisateur effectue un swipe vers la droite sur **avtivity2**, une nouvelle activité **MainActivity** est lancée. Si l'utilisateur effectue un swipe vers la gauche sur **avtivity2**, une nouvelle activité **MainActivity3** est lancée.

Copyright - Tout droit réservé - OFPPT

Tâche 3 : OnItemSelected

On demande de créer une activité (**MainActivity3**) qui contient une liste des villes et un champ texte, dans lequel la ville sélectionnée sera affichée. Comme le montre la figure à côté.



Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

PARTIE 1

Etape 1 : Modifier le fichier layout

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity3"
    android:id="@+id/activity3">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="288dp"
        android:layout_height="48dp"
        android:layout_marginTop="96dp"
        android:text="OnItemSelectedListener"
        android:textSize="24sp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Spinner
        android:id="@+id/spinner"
        android:layout_width="360dp"
        android:layout_height="48dp"
        android:layout_marginTop="88dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView2" />

    <TextView
        android:id="@+id/spinnerText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="312dp"
        android:text="Text"
        android:textSize="22dp"
        android:textStyle="bold"
        android:textColor="@color/teal_200"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

PARTIE 1

Etape 2 : Modifier le code de la classe activité

```
class MainActivity3 : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main3)
        val villes = arrayOf("Rabat ", "Casablanca" , "Fes" , "El Jadida" , "Kenitra", "Agadir")
        val arrayAdapter = ArrayAdapter(this, android.R.layout.simple_spinner_dropdown_item,villes)
        val spinner = findViewById<Spinner>(R.id.spinner)
        val spinnerResult = findViewById<TextView>(R.id.spinnerText)
        val activity3 = findViewById<ConstraintLayout>(R.id.activity3)
        spinner.adapter = arrayAdapter
        spinner.onItemSelectedListener = object :
            AdapterView.OnItemSelectedListener {
            override fun onItemSelected(p0: AdapterView<*>?, p1: View?, p2: Int, p3: Long) {
                spinnerResult.text = villes[p2]
            }
            override fun onNothingSelected(p0: AdapterView<*>?) {
                //
            }
        }
        activity3.setOnTouchListener(object : OnSwipeTouchListener(this@MainActivity3) {
            override fun onSwipeRight(): Boolean {
                val i = Intent(this@MainActivity3,MainActivity2::class.java )
                startActivity(i)
                return true;
            }
            override fun onSwipeLeft(): Boolean {
                val i = Intent(this@MainActivity3,MainActivity4::class.java )
                startActivity(i)
                return true;
            }
        })
    }
}
```

Ce code permet de créer une activité Android qui affiche une liste déroulante (**Spinner**) de villes et qui détecte les gestes de glissement vers la droite ou vers la gauche sur la vue pour lancer des activités correspondantes.

La méthode **onCreate()** est la première méthode qui est appelée lorsque l'activité est créée. Elle permet de définir la vue à afficher et de configurer les différents éléments de l'interface utilisateur.

Le Spinner est configuré avec un **ArrayAdapter** qui adapte les éléments du tableau de chaînes de caractères à la vue du Spinner. Un écouteur d'événements est défini pour détecter les changements de sélection dans le Spinner et mettre à jour un **TextView** avec la ville sélectionnée.

Un écouteur de toucher est également défini pour la vue activity3. Cet écouteur détecte les gestes de glissement vers la droite ou vers la gauche sur la vue et déclenche l'intention de lancer une activité correspondante en utilisant des objets **Intent**.

Copyright - Tout droit réservé - OFPPT

Tâche 4 : OnTouch

Créer une activité Android (**MainActivity4**) qui affiche une image et qui permet à l'utilisateur de déplacer l'image sur l'écran en la faisant glisser du doigt, comme indiqué dans l'image jointe.



Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

Etape 1 : Modifier le fichier layout

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity4"
    android:id="@+id/activity4">
    <ImageView
        android:id="@+id/image"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:src="@drawable/android" />
    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Drag ImageView"
        android:textSize="26dp"
        android:textStyle="bold"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="20dp"/>
</RelativeLayout>
```

Mettre l'image dans le dossier drawable.

PARTIE 1

Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

Etape 2 : Modifier le code de la classe activité

```
class MainActivity4 : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        val actionBar = supportActionBar
        actionBar!!.hide()
        setContentView(R.layout.activity_main4)
        val image = findViewById<ImageView>(R.id.image)
        val activity4 = findViewById<RelativeLayout>(R.id.activity4)
        var linstener = View.OnTouchListener(function = {view, motionEvent ->
            if (motionEvent.action == MotionEvent.ACTION_MOVE) {
                view.y = motionEvent.rawY - view.height/2
                view.x = motionEvent.rawX - view.width/2
            }
            true
        })
        image.setOnTouchListener(linstener)
        activity4.setOnTouchListener(object : OnSwipeTouchListener(this@MainActivity4) {
            override fun onSwipeRight(): Boolean {
                val i = Intent(this@MainActivity4,MainActivity3::class.java )
                startActivity(i)
                return true;
            }
            override fun onSwipeLeft(): Boolean {
                val i = Intent(this@MainActivity4,MainActivity5::class.java )
                startActivity(i)
                return true;
            }
        })
    }
}
```

Ce code permet de créer une activité Android qui affiche une image et qui permet à l'utilisateur de déplacer l'image sur l'écran en la faisant glisser du doigt.

La méthode **onCreate()** est la première méthode qui est appelée lorsque l'activité est créée. Elle permet de définir la vue à afficher et de configurer les différents éléments de l'interface utilisateur.

Dans ce code, l'action bar (barre d'actions) est cachée et la vue **activity_main4** est définie comme vue de l'activité. L'image est affichée en utilisant un objet **ImageView** et un écouteur de toucher est défini pour l'objet **ImageView**. Cet écouteur détecte les mouvements de l'utilisateur sur l'écran et met à jour la position de l'image en conséquence.

Un écouteur de toucher est également défini pour la vue **activity4**. Cet écouteur détecte les gestes de glissement vers la droite ou vers la gauche sur la vue et déclenche l'intention de lancer une activité correspondante en utilisant des objets Intent.

En résumé, ce code définit une activité qui affiche une image que l'utilisateur peut déplacer sur l'écran en la faisant glisser du doigt et qui détecte les gestes de glissement vers la droite ou vers la gauche sur la vue pour lancer des activités correspondantes.

PARTIE 1

Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

Tâche 5 : OnDateSetListner

Créer une activité Android (**MainActivity5**) qui affiche un bouton pour ouvrir un calendrier et permettre à l'utilisateur de sélectionner une date. Affichez la date sélectionnée dans un champ de texte, comme indiqué dans l'image jointe.



PARTIE 1

Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

Etape 1 : Modifier le fichier layout

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity5"
    android:id="@+id/activity5">

    <TextView
        android:id="@+id/textView6"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="120dp"
        android:text="OnDateSetListener"
        android:textSize="28dp"
        style="@style/MaterialAlertDialog.Material3"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.489"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0" />

    <Button
        android:id="@+id/dateButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="112dp"
        android:backgroundTint="@color/teal_200"
        android:text="Selet Date"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView6" />
    <TextView
        android:id="@+id/dateSelect"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="96dp"
        android:text="TextView"
        android:textColor="@color/teal_700"
        android:textSize="24sp"
        android:textStyle="italic"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/dateButton" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

PARTIE 1

Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

PARTIE 1

Etape 2 : Modifier le code de la classe activité

```
class MainActivity5 : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main5)
        //Calendar
        val c = Calendar.getInstance()
        val year = c.get(Calendar.YEAR)
        val month = c.get(Calendar.MONTH)
        val day = c.get(Calendar.DAY_OF_MONTH)
        val bn = findViewById<Button>(R.id.dateButton)
        val date = findViewById<TextView>(R.id.dateSelect)
        val activity5 = findViewById<ConstraintLayout>(R.id.activity5)
        bn.setOnClickListener {
            val dpd = DatePickerDialog(this,DatePickerDialog.OnDateSetListener { view,
myear,              monthOfYear, dayOfMonth ->
                date.setText(""+dayOfMonth+"/"+ monthOfYear +"/" +myear)
            }, year, month, day)
            dpd.show()
        }
         activity5.setOnTouchListener(object : OnSwipeTouchListener(this@MainActivity5) {
            override fun onSwipeRight(): Boolean {
                val i = Intent(this@MainActivity5,MainActivity4::class.java )
                startActivity(i)
                return true;
            }
            override fun onSwipeLeft(): Boolean {
                val i = Intent(this@MainActivity5,MainActivity6::class.java )
                startActivity(i)
                return true;
            }
        })}}
```

Ce code permet de créer une activité Android qui affiche un bouton pour ouvrir un calendrier et permettre à l'utilisateur de sélectionner une date. La date sélectionnée est affichée dans un champ de texte. L'activité détecte également les gestes de glissement vers la droite ou vers la gauche sur la vue et lance des activités correspondantes.

La méthode **onCreate()** est la première méthode qui est appelée lorsque l'activité est créée. Elle permet de définir la vue à afficher et de configurer les différents éléments de l'interface utilisateur.

Le bouton est configuré avec un écouteur d'événements qui ouvre une boîte de dialogue de sélection de date lorsqu'il est cliqué. La boîte de dialogue de sélection de date est créée en utilisant l'objet **DatePickerDialog**. Un écouteur d'événements est défini pour détecter la sélection de date par l'utilisateur. Lorsqu'une date est sélectionnée, le champ de texte est mis à jour avec la date sélectionnée.

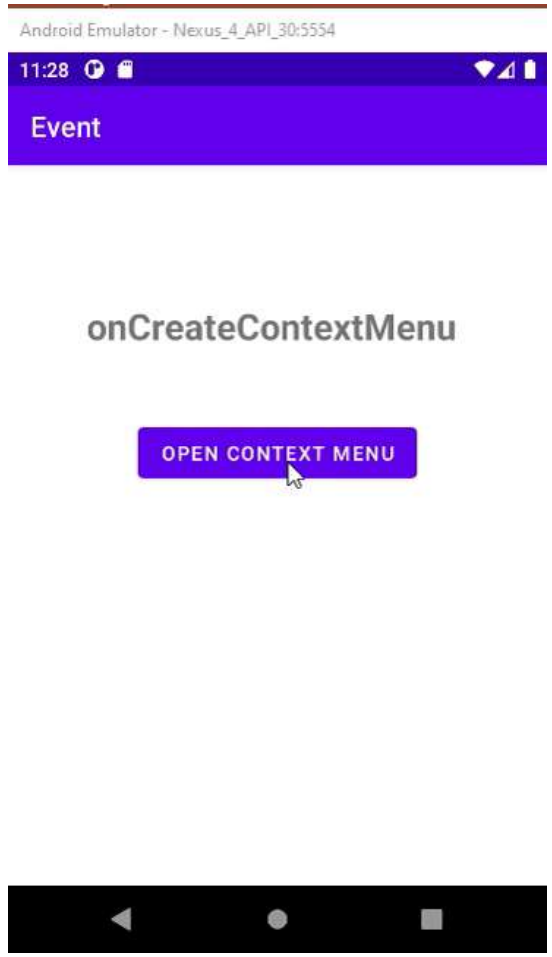
Un écouteur de toucher est également défini pour la vue **activity5**. Cet écouteur détecte les gestes de glissement vers la droite ou vers la gauche sur la vue et déclenche l'intention de lancer une activité correspondante en utilisant des objets Intent.

En résumé, ce code définit une activité qui affiche un bouton pour ouvrir un calendrier et permettre à l'utilisateur de sélectionner une date, qui affiche la date sélectionnée dans un champ de texte, et qui détecte les gestes de glissement vers la droite ou vers la gauche sur la vue pour lancer des activités correspondantes.

Copyright - Tout droit réservé - OFPPT

Tâche 6 : OnCreateContextMenu

Créer une activité Android (**MainActivity6**) qui affiche un bouton pour ouvrir un menu contextuel (ou pop-up menu), ensuite affiche un message toast en fonction de l'élément sélectionné dans le menu. Comme indiqué dans l'image jointe.



Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

Etape 1 : Modifier le fichier layout

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity6"
    android:id="@+id/activity6">

    <Button
        android:id="@+id/btnShowContextMenu"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="256dp"
        android:text="Open context menu"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView5"
        app:layout_constraintVertical_bias="0.61" />

    <TextView
        android:id="@+id/textView5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="92dp"
        android:text="onCreateContextMenu"
        android:textSize="26dp"
        android:padding="5dp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.462"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

PARTIE 1

Copyright - Tout droit réservé - OFPPT

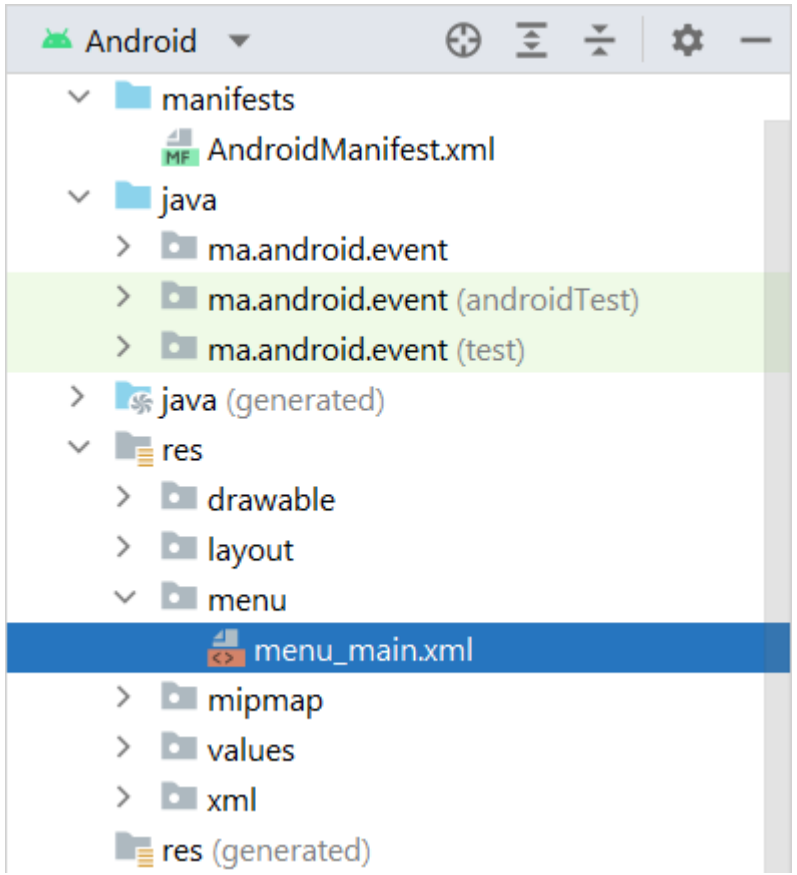
PARTIE 1

Etape 2 : Créer le fichier menu_main.xml dans le dossier menu

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/iItem1"
        android:title="Item 1"/>

    <item android:id="@+id/iItem2"
        android:title="Item 2"/>

    <item android:id="@+id/iItem3"
        android:title="Item 3"/>
</menu>
```



Copyright - Tout droit réservé - OFPPT

Activité n° 1

Création de plusieurs interfaces graphiques et gestion des évènements

Etape 3 : Modifier le code de la classe activité

```
class MainActivity6 : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main6)

        val btnShowContextMenu = findViewById<Button>(R.id.btnShowContextMenu)
        val activity6 = findViewById<ConstraintLayout>(R.id.activity6)
        registerForContextMenu(btnShowContextMenu)

        btnShowContextMenu.setOnClickListener {
            openContextMenu(it)
        }

        activity6.setOnTouchListener(object : OnSwipeTouchListener(this@MainActivity6) {
            override fun onSwipeRight(): Boolean {
                val i = Intent(this@MainActivity6,MainActivity5::class.java )
                startActivity(i)
                return true;
            }
        })
    }

    override fun onCreateContextMenu(menu: ContextMenu?, v: View?, menuInfo:
                    ContextMenu.ContextMenuInfo?) {
        super.onCreateContextMenu(menu, v, menuInfo)
        menuInflater.inflate(R.menu.menu_main, menu)
    }

    override fun onContextItemSelected(item: MenuItem): Boolean {
        when (item?.itemId) {
            R.id.iItem1 -> {
                showToast("Select item 1")
                return true
            }
            R.id.iItem2 -> {
                showToast("Selected item 2")
                return true
            }
            R.id.iItem3 -> {
                showToast("Selected item 3")
                return true
            }
            else -> return super.onContextItemSelected(item)
        }
    }

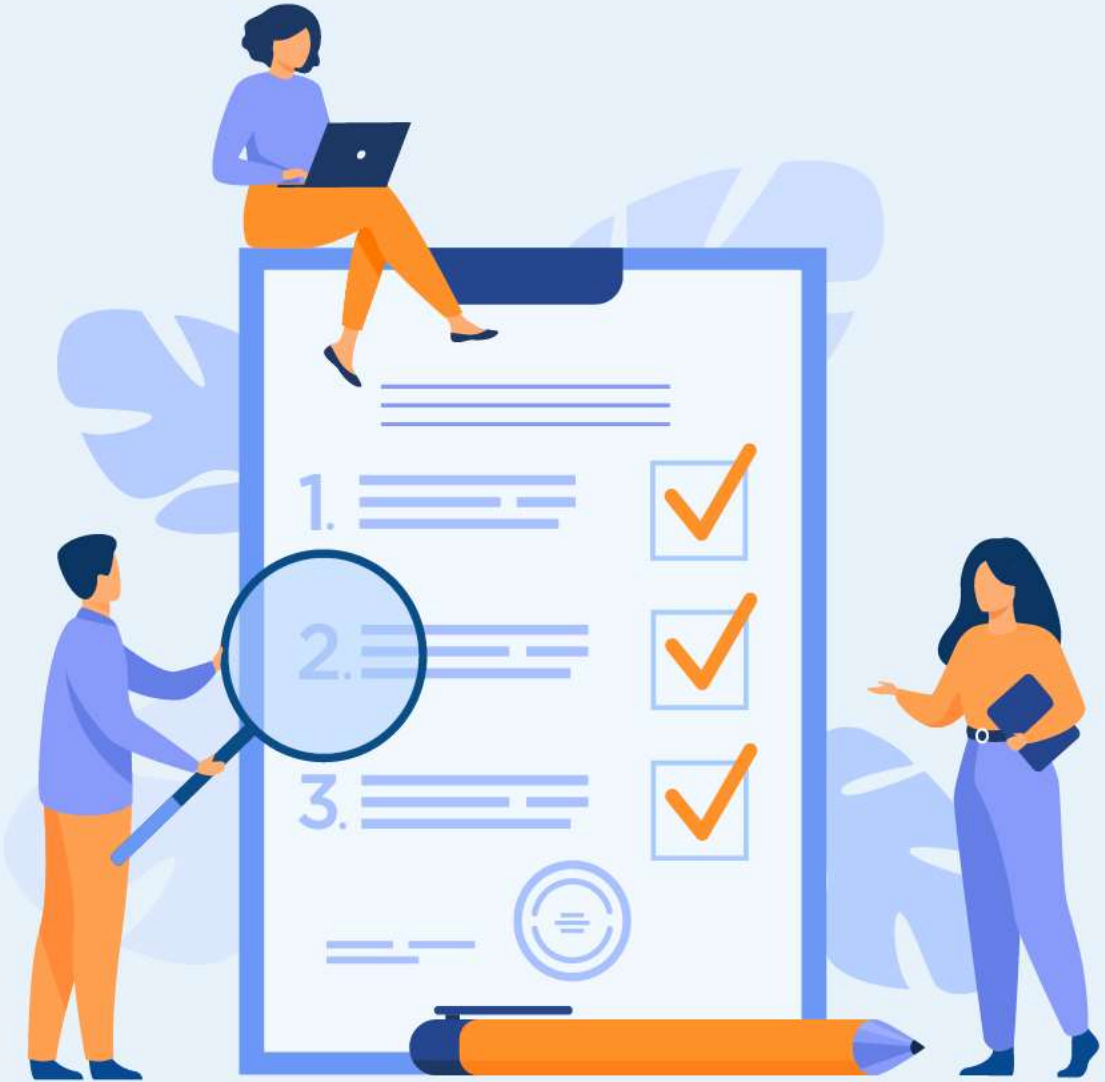
    private fun showToast(message: String) {
        Toast.makeText(this, message, Toast.LENGTH_SHORT).show()
    }
}
```

PARTIE 1

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 2

Création d'interfaces utilisant divers layouts



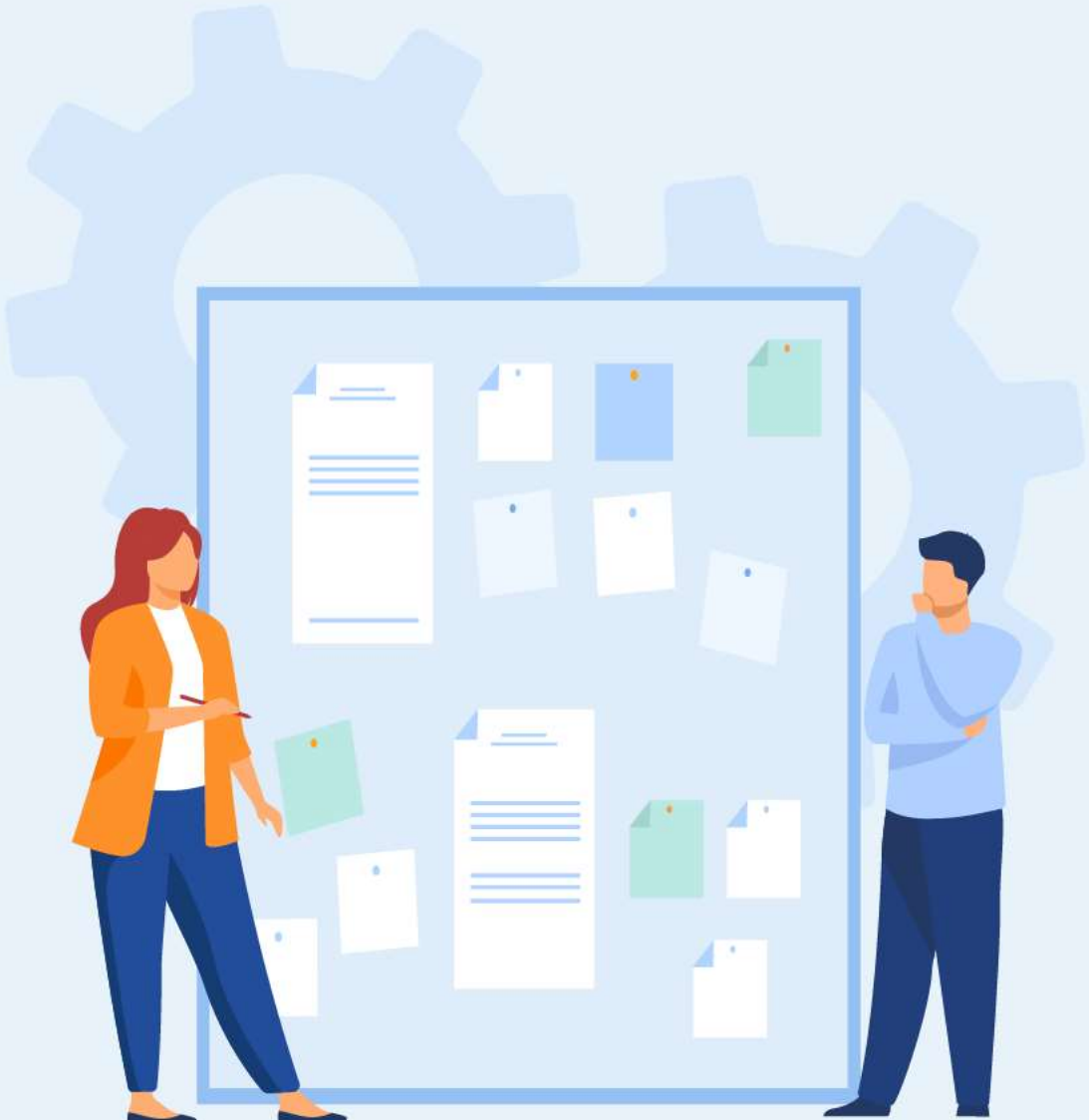
Compétences visées :

- Création d’une interfaces basée sur les fragments ;
- Manipulation d’une ListView.

Recommandations clés :

- Révision générale du résumé théorique ;
- Suivre les instructions du TP et organiser le dossier de travail.

12 heures



CONSIGNES

Pour le formateur :

- Rappeler les étapes à suivre pour mettre en place une interface avec ViewPager ;
- Rappeler les étapes à suivre pour réaliser une ListView ;
- Demander aux apprenants de suivre les étapes décrites dans l'activité.

Pour l'apprenant :

- Suivre les étapes indiquées dans l'activité.

Conditions de réalisation :

- Support de résumé théorique accompagnant ;
- Un ordinateur sur lequel est installée la dernière version stable d'Android Studio.

Critères de réussite :

- Le stagiaire doit être capable de :
  - réaliser des interfaces graphiques avec différents layout ;
  - manipuler une ListView ;
  - utiliser la patron de conception Singleton.

Activité n° 2

Création d'interfaces utilisant divers layouts

PARTIE 1

TP N° 1 : Converter

On souhaite développer une application mobile permettant d'offrir une interface pour la conversion Celsius ↔ Fahrenheit et Km ↔ Miles.

L'application contient 2 tabulations.

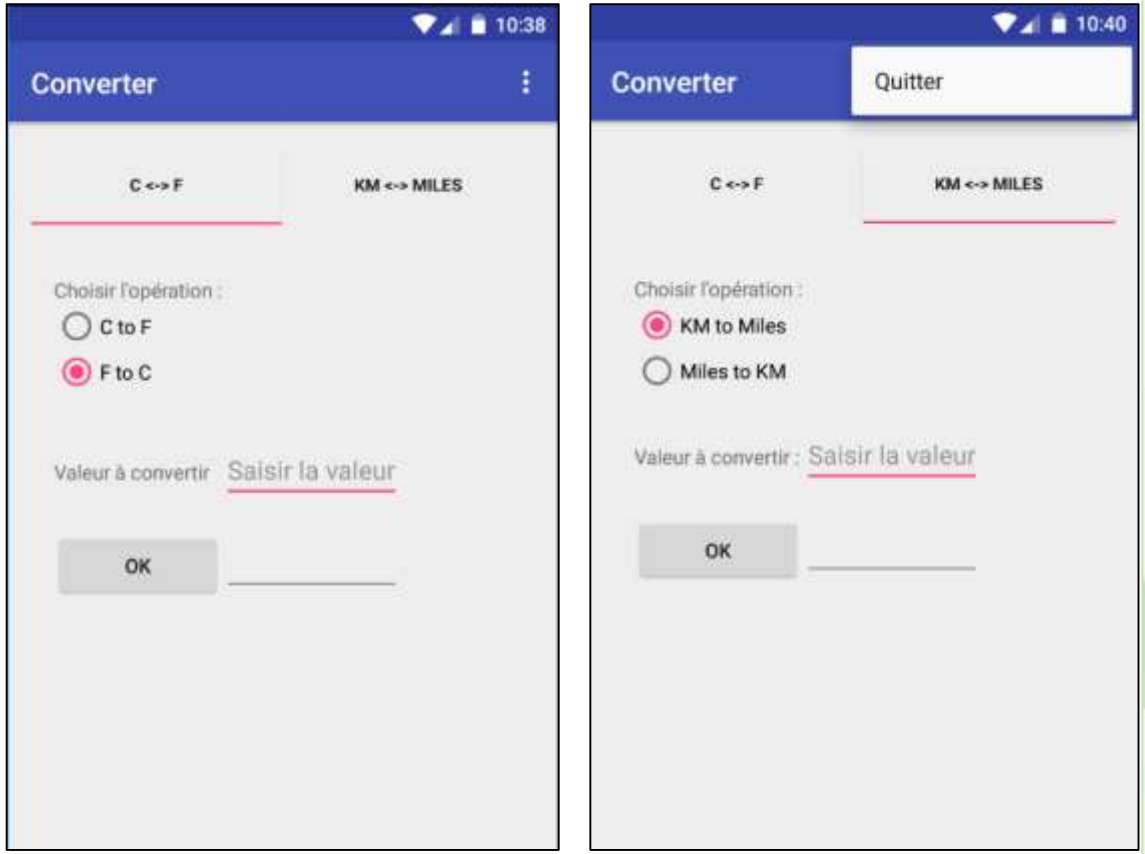
La première pour une conversion Celsius ↔ Fahrenheit et la deuxième pour une conversion Km ↔ Miles. Chaque tabulation contient les éléments suivants

- Un titre ;
- 2 boutons radio (C ↔ F) et (F ↔ C) pour la première tabulation et (Km ↔ Miles) et (Miles ↔ Km) pour la deuxième ;
- Une zone de texte pour saisir la valeur à convertir ;
- Un bouton pour calculer la valeur ;
- Une TextView pour afficher le résultat ;
- Ajouter un menu « Quitter » permettant de quitter l'application. Ajouter une AlertDialog pour confirmer le choix de l'utilisateur.

Indications :

T(°F) = 1,8 T(°C) + 32

1km équivaut à 0,6214 miles.



Copyright - Tout droit réservé - OFPPT

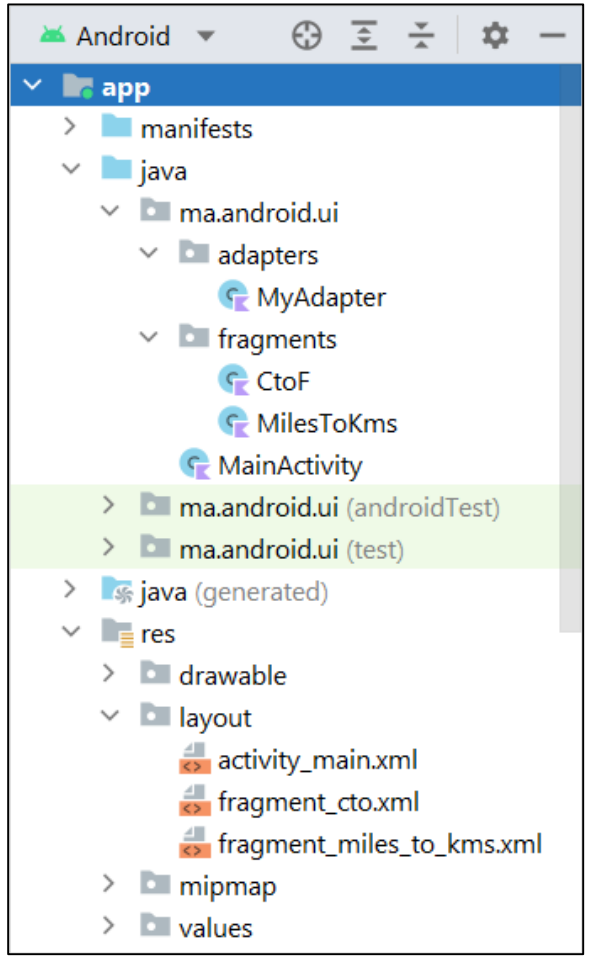
Activité n° 2
Création d'interfaces utilisant divers layouts

PARTIE 1

Solution

- Pour mettre en place les tabulations :
  - utiliser le composant ViewPager ;
  - créer deux fragments pour représenter le contenu de chaque tabulation.

Structure de projet



Copyright - Tout droit réservé - OFPPT

Activité n° 2

Création d'interfaces utilisant divers layouts

Etape 1 : Modifier le fichier activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <com.google.android.material.tabs.TabLayout
        android:id="@+id/tabLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/white"
        app:tabTextColor="@color/black">
    </com.google.android.material.tabs.TabLayout>

    <androidx.viewpager.widget.ViewPager
        android:id="@+id/viewPager"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/tabLayout"
        android:layout_centerInParent="true"
        android:layout_marginTop="100dp"
        tools:layout_editor_absoluteX="8dp" />

</RelativeLayout>
```

PARTIE 1

Copyright - Tout droit réservé - OFPPT

Etape 2 : Créer le fragement CtoF

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="ma.android.ui.fragments.CtoF">
    <!-- TODO: Update blank fragment layout -->
    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <RadioGroup
            android:id="@+id/radioGroup"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            app:layout_constraintTop_toBottomOf="@+id/textView"
            tools:ignore="MissingConstraints"
            tools:layout_editor_absoluteX="50dp">

            <RadioButton
                android:id="@+id/radioButton2"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:checked="true"
                android:text="C to F"
                tools:layout_editor_absoluteX="28dp"
                tools:layout_editor_absoluteY="230dp" />

            <RadioButton
                android:id="@+id/radioButton"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="F to C"
                tools:layout_editor_absoluteX="28dp"
                tools:layout_editor_absoluteY="230dp" />
        </RadioGroup>
        <TextView
            android:id="@+id/textView3"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginStart="16dp"
            android:layout_marginTop="68dp"
            android:text="Valeur à convertir :"
            app:layout_constraintEnd_toStartOf="@+id/txtValue"
            app:layout_constraintHorizontal_bias="0.641"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/radioGroup" />
```

Copyright - Tout droit réservé - OFPPT

PARTIE 1

Etape 2 : Créer le fragement CtoF (Suite)

```
<EditText
    android:id="@+id/txtValue"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="184dp"
    android:layout_marginTop="60dp"
    android:ems="10"
    android:inputType="textPersonName"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/radioGroup"
    tools:ignore="MissingConstraints" />

<EditText
    android:id="@+id/resultat"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="184dp"
    android:layout_marginTop="52dp"
    android:ems="10"
    android:inputType="textPersonName"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/txtValue"
    tools:ignore="MissingConstraints" />

<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="68dp"
    android:layout_marginEnd="36dp"
    android:text="OK"
    app:layout_constraintEnd_toStartOf="@+id/resultat"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView3" />

<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="36dp"
    android:layout_marginTop="20dp"
    android:text="Choisir l'opération :"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>

</FrameLayout>
```

Copyright - Tout droit réservé - OFPPT

PARTIE 1

Etape 3 : Créer le fragement MilestoKms

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    tools:context="ma.android.ui.fragments.MilesToKms">

    <!-- TODO: Update blank fragment layout -->
    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        >
        <RadioGroup
            android:id="@+id/radioGroup"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            app:layout_constraintTop_toBottomOf="@+id/textView"
            tools:ignore="MissingConstraints"
            tools:layout_editor_absoluteX="50dp">

            <RadioButton
                android:id="@+id/radioButton2"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="KM to MILES"
                tools:layout_editor_absoluteX="28dp"
                tools:layout_editor_absoluteY="230dp" />

            <RadioButton
                android:id="@+id/radioButton"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="MILES to KM"
                tools:layout_editor_absoluteX="28dp"
                tools:layout_editor_absoluteY="230dp"
                android:checked="true"/>
        </RadioGroup>

        <TextView
            android:id="@+id/textView3"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginStart="16dp"
            android:layout_marginTop="72dp"
            android:text="Valeur à convertir :"
            app:layout_constraintEnd_toStartOf="@+id/txtVal"
            app:layout_constraintHorizontal_bias="0.666"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/radioGroup" />
```

Copyright - Tout droit réservé - OFPPT

PARTIE 1

Etape 3 : Créer le fragement MilestoKms (Suite)

```
<EditText
        android:id="@+id/txtVal"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="184dp"
        android:layout_marginTop="60dp"
        android:ems="10"
        android:inputType="textPersonName"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/radioGroup"
        tools:ignore="MissingConstraints" />

    <EditText
        android:id="@+id/res"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="184dp"
        android:layout_marginTop="40dp"
        android:ems="10"
        android:inputType="textPersonName"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/txtVal"
        tools:ignore="MissingConstraints" />

<Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="52dp"
        android:layout_marginEnd="36dp"
        android:text="OK"
        app:layout_constraintEnd_toStartOf="@+id/res"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView3" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="60dp"
        android:layout_marginTop="20dp"
        android:text="Choisir l'opération :"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
  </androidx.constraintlayout.widget.ConstraintLayout>

</FrameLayout>
```

Copyright - Tout droit réservé - OFPPT

PARTIE 1

Etape 4 : Créer la classe CtoF

```
class CtoF : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
        val view: View = inflater!!.inflate(R.layout.fragment_cto, container, false)
        val Button: Button = view.findViewById(R.id.button)
        val valeur: EditText = view.findViewById(R.id.resultat)
        val ctf: RadioButton = view.findViewById(R.id.radioButton2)
        val ftc: RadioButton = view.findViewById(R.id.radioButton)
        val vall: EditText = view.findViewById(R.id.txtValue)

Button.setOnClickListener {
            Log.d("console", "Button pressed")
            if (vall.text.toString() == null) {
                Log.d("console", "vide")
            } else {
                if (ctf.isChecked) {
                    var valeur1: Float = vall.text.toString().toFloat()
                    var res: Float = (valeur1 * 9 / 5) + 32
                    valeur.setText(res.toString())
                }
                if (ftc.isChecked) {
                    var valeur1: Float = vall.text.toString().toFloat()
                    var res: Float = (valeur1 - 32) * 5 / 9
                    valeur.setText(res.toString())
                }
            }
        }
        return view
    }

}
```

Copyright - Tout droit réservé - OFPPT

Etape 4 : Créer la classe MilestoKms

```
class MilesToKms : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
        val view: View = inflater!!.inflate(R.layout.fragment_miles_to_kms, container, false)
        val Button: Button = view.findViewById(R.id.button)
        val valeur: EditText = view.findViewById(R.id.res)
        val mtk: RadioButton = view.findViewById(R.id.radioButton)
        val ktm: RadioButton = view.findViewById(R.id.radioButton2)
        val vall: EditText = view.findViewById(R.id.txtVal)

Button.setOnClickListener {
            Log.d("console", "Button pressed")
            if (vall.text.toString() == null) {
                Log.d("console", "vide")
            } else {
                if (mtk.isChecked) {
                    var valeur1: Float = vall.text.toString().toFloat()
                    var res: Double = valeur1 * 1.609344
                    valeur.setText(res.toString())
                }
                if (ktm.isChecked) {
                    var valeur1: Float = vall.text.toString().toFloat()
                    var res: Double = valeur1 / 1.609344
                    valeur.setText(res.toString())
                }
            }
        }
        return view
    }

}
```

Copyright - Tout droit réservé - OFPPT

Etape 5 : Créer la classe MyAdapter

```
class MyAdapter (
    var context: Context,
    fm: FragmentManager,
    var totalTabs: Int
) :
    FragmentPagerAdapter(fm) {
    override fun getItem(position: Int): Fragment {
        return when (position) {
            0 -> {
                CtoF()
            }
            1 -> {
                MilesToKms()
            }
            else -> getItem(position)
        }
    }
    override fun getCount(): Int {
        return totalTabs
    }
}
```

Ce code crée une classe d'adaptateur pour une vue avec onglets (**tabs**) dans une application Android. L'adaptateur implémente la classe FragmentPagerAdapter et prend en charge la création de fragments pour chaque onglet. L'adaptateur utilise une instruction "**when**" pour déterminer quel fragment doit être créé pour chaque onglet en fonction de sa position. La méthode **getCount()** renvoie le nombre total d'onglets.

Il y a deux fragments définis dans ce code : **CtoF()** et **MilesToKms().** Ces fragments seront créés respectivement pour les onglets en position 0 et 1. Si la position de l'onglet n'est pas 0 ou 1, la méthode **getItem()** est appelée de nouveau avec la même position en argument, ce qui peut entraîner une récursion infinie si aucun fragment n'est défini pour cette position.

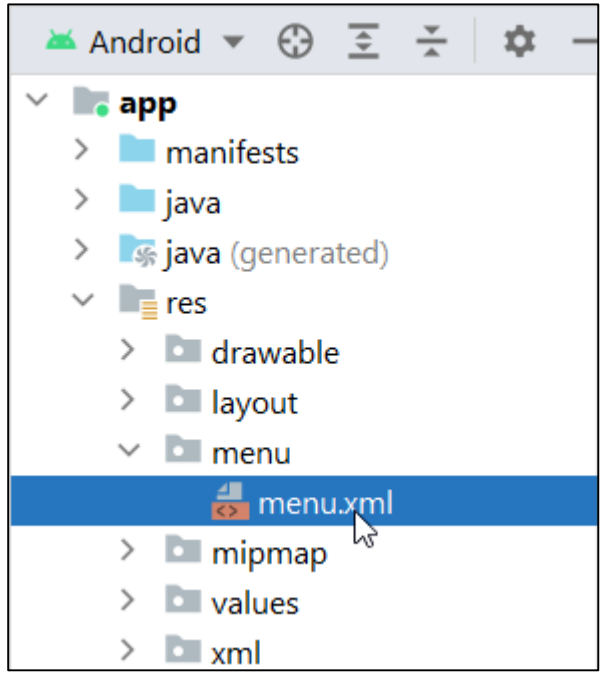
Copyright - Tout droit réservé - OFPPT

Etape 6 : Créer le fichier menu_main.xml dans le dossier menu

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    tools:context=".MainActivity">

    <item android:id="@+id/exit"
        app:showAsAction="never"
        android:title="Quitter"/>

</menu>
```



Copyright - Tout droit réservé - OFPPT

Etape 7 : Intégrer le menu dans l'activité

```
override fun onCreateContextMenu(menu: ContextMenu?, v: View?, menuInfo: ContextMenu.ContextMenuInfo?) {
    super.onCreateContextMenu(menu, v, menuInfo)
    menuInflater.inflate(R.menu.menu, menu)
}

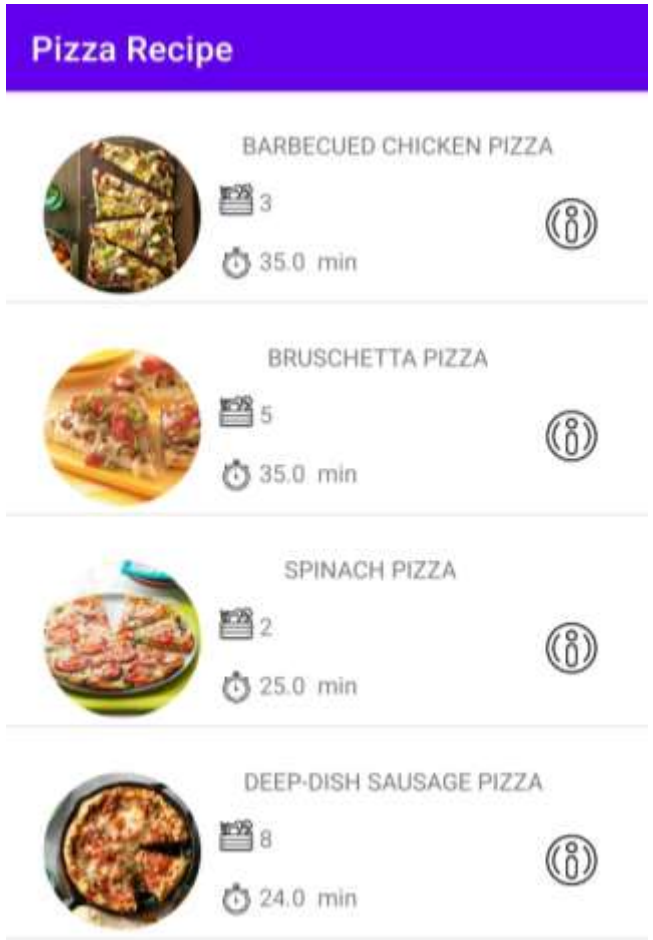
override fun onContextItemSelected(item: MenuItem): Boolean {
    when (item?.itemId) {
        R.id.exit -> {
            val alertDialogBuilder = AlertDialog.Builder (this)
            alertDialogBuilder.setMessage("Voulez vraiment quitter l'application ? ")
            alertDialogBuilder.setPositiveButton("Oui",
                DialogInterface.OnClickListener { arg0, arg1 ->
                    MainActivity@this.finish()
                })
            alertDialogBuilder.setNegativeButton("Non",
                DialogInterface.OnClickListener { dialog, which -> finish() })
            val alertDialog: AlertDialog = alertDialogBuilder.create()
            alertDialog.show()
            return true
        }
        else -> return super.onContextItemSelected(item)
    }
}
```

Copyright - Tout droit réservé - OFPPT

PARTIE 1

TP N° 2 : Pizza recipe

Créer une application Android qui utilise Kotlin et affiche une liste de recettes de pizzas en utilisant une ListView, comme illustré dans l'image jointe.



Copyright - Tout droit réservé - OFPPT

Activité n° 2

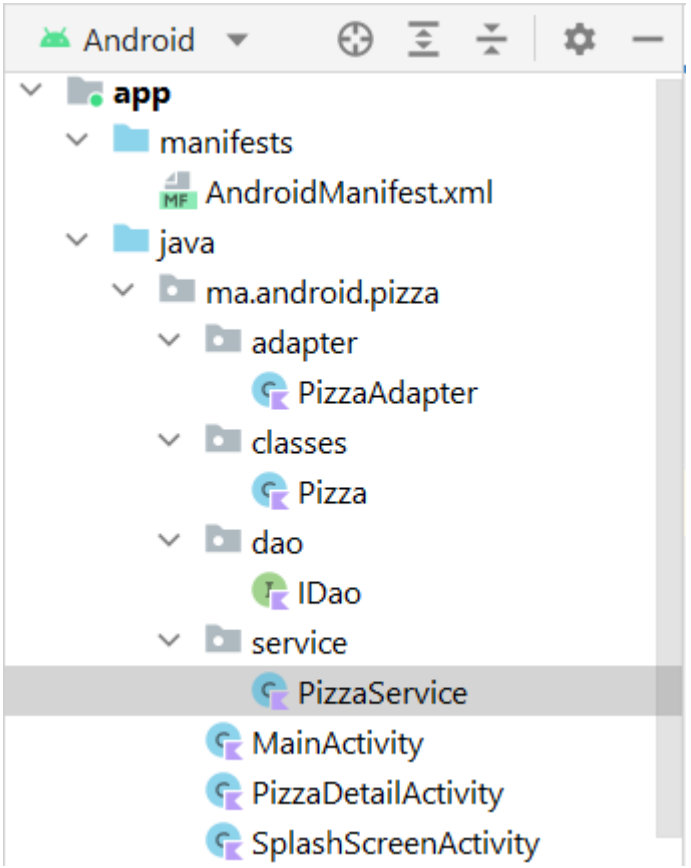
Création d'interfaces utilisant divers layouts

PARTIE 1

Etape 1 : Créer le projet (empty activity)

1. Créer une application mobile avec le langage Kotlin sous Android ;
2. Lancer l'émulateur.

Structurer le projet



Copyright - Tout droit réservé - OFPPT

PARTIE 1

Etape 2 : Créer la classe Pizza

Créer la classe **Pizza** dans le package « **classes** ».

```
package ma.android.pizza.classes

class Pizza {
    var id = 0
    var nom: String = ""
    var nbrIngredients = 0
    var photo = 0
    var duree = 0f
    var detailsIngred: String = ""
    var description: String = ""

    var preparation: String? = ""

    //default constructor
    constructor() {}

    constructor(
        nom: String,
        nbrIngredients: Int,
        photo: Int,
        duree: Int,
        detailsIngred: String,
        description: String,
        preparation: String
    ) {
        id = ++comp
        this.nom = nom
        this.nbrIngredients = nbrIngredients
        this.photo = photo
        this.duree = duree.toFloat()
        this.detailsIngred = detailsIngred
        this.description = description
        this.preparation = preparation
    }
    companion object {
        private var comp = 0
    }
}
```

Copyright - Tout droit réservé - OFPPT

Etape 3 : Créer l'interface IDao

Créer l'interface générique **IDao** dans le package « **dao** ».

```
package ma.android.pizza.dao

import ma.android.pizza.classes.Pizza

interface IDao <T>{
    fun create (o : T) : Boolean
    fun delete (o : T) : Boolean
    fun update (o : T) : Boolean
    fun findById (id : Int) : Pizza?
    fun findAll () : List<T>
}
```


Copyright - Tout droit réservé - OFPPT

Etape 4 : Créer la classe PizzaService

Créer la classe **PizzaService** dans le package « **service** ».

```
package ma.android.pizza.service

import ma.android.pizza.classes.Pizza
import ma.android.pizza.dao.IDao

class PizzaService private constructor() : IDao<Pizza> {
    private val pizzas: MutableList<Pizza>

    init {
        pizzas = ArrayList()
    }

    override fun create(o: Pizza): Boolean {
        return pizzas.add(o)
    }

    override fun delete(o: Pizza): Boolean {
        return pizzas.remove(o)
    }

    override fun update(o: Pizza): Boolean {
        for (p in pizzas) {
            if (p.id == o.id) {
                p.nom = o.nom
                p.description = o.description
                p.detailsIngred = o.detailsIngred
                p.duree = o.duree
                p.photo = o.photo
                p.nbrIngredients = o.nbrIngredients
                p.preparation = o.preparation
                return true
            }
        }
        return false
    }
```

Copyright - Tout droit réservé - OFPPT

Etape 4 : Créer la classe PizzaService (Suite)

```
override fun findById(id: Int): Pizza? {
        for (p in pizzas) {
            if (p.id == id) {
                return p
            }
        }
        return null
    }

    override fun findAll(): List<Pizza> {
        return pizzas
    }

    companion object {
        var instance: PizzaService? = null
            get() {
                if (field == null) field = PizzaService()
                return field
            }
            private set
    }
}
```


Copyright - Tout droit réservé - OFPPT

Etape 5 : Modifier la classe MainActivity

1. Déclarer la variable globale :

```
private lateinit var fs: PizzaService
```

2. Dans la méthode onCreate(), instalcier la l'objet fs :

```
fs = PizzaService.instance!!
```

3. Dans la méthode onCreate (), créer 10 objets Pizza (Un exemple ci-dessous, la suite dans le dossier ressources):

```
fs.create(
    Pizza(
        "BARBECUED CHICKEN PIZZA",
        3,
        R.mipmap.pizza1,
        35,
        "- 2 boneless skinless chicken breast halves (6 ounces each)\n- 1/4 teaspoon pepper\n- 1 cup barbecue sauce, divided\n- 1 tube (13.8 ounces) refrigerated pizza crust\n- 2 teaspoons olive oil\n-2 cups shredded Gouda cheese\n-1 small red onion, halved and thinly sliced\n-1/4 cup minced fresh cilantro",
        "So fast and so easy with refrigerated pizza crust, these saucy, smoky pizzas make quick fans with their hot-off-the-grill, rustic flavor. They're perfect for spur-of-the- moment cookouts and summer dinners on the patio. —Alicia Trevithick, Temecula, California",
        "STEP 1:\n\n Sprinkle chicken with pepper; place on an oiled grill rack over medium heat. Grill, covered, until a thermometer reads 165°, 5-7 minutes per side, basting frequently with 1/2 cup barbecue sauce during the last 4 minutes. Cool slightly. Cut into cubes.\n\nSTEP 2:\n\n Divide dough in half. On a well-greased large sheet of heavy-duty foil, press each portion of dough into a 10x8-in. rectangle; brush lightly with oil. Invert dough onto grill rack; peel off foil. Grill, covered, over medium heat until bottom is lightly browned, 1-2 minutes.\n\nSTEP 3:\n\n Remove from grill. Spread grilled sides with remaining barbecue sauce. Top with cheese, chicken and onion. Grill, covered, until bottom is lightly browned and cheese is melted, 2-3 minutes. Sprinkle with cilantro. Yield: 2 pizzas (4 pieces each)."
    )
)
```


Copyright - Tout droit réservé - OFPPT

Etape 5 : Modifier la classe MainActivity

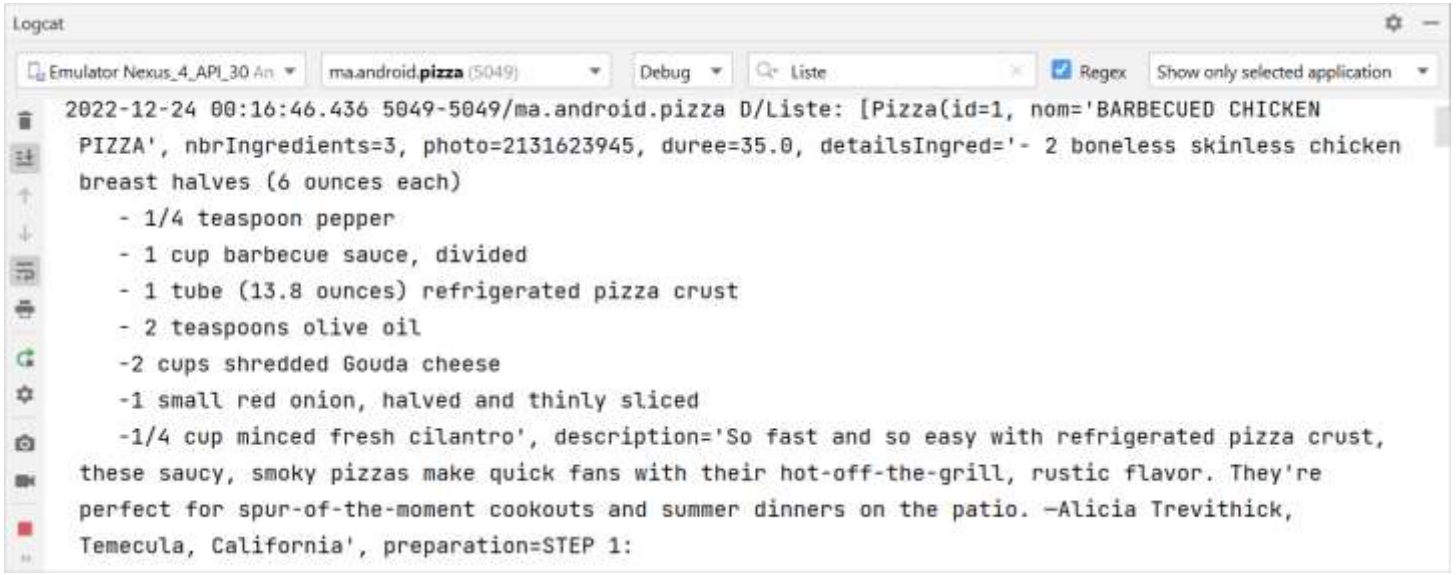
4. Afficher la liste des Pizzas dans le log. Ajouter dans la méthode onCreate () :

```
Log.d("Liste", fs.findAll().toString());
```

5. Dans la classe Pizza redéfinir la méthode toString () :

```
override fun toString(): String {
        return "Pizza(id=$id, nom='$nom', nbrIngredients=$nbrIngredients, photo=$photo, duree=$duree, detailsIngred='$detailsIngred',
description='$description',
                preparation=$preparation)"
    }
```

3. Afficher le résultat d'exécution dans logcat :



PARTIE 1

Copyright - Tout droit réservé - OFPPT

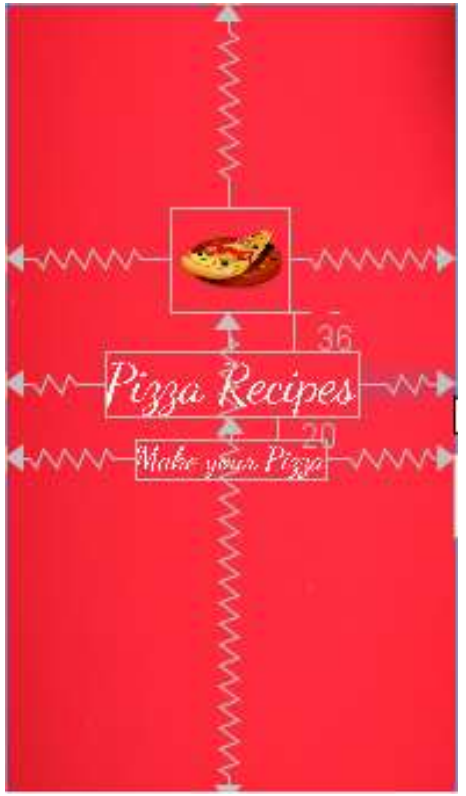
PARTIE 1

Etape 6 : Développer la couche présentation

Une fois que la couche de données (ici une liste) et la couche accès aux données sont mises en place. On demande de mettre en place la couche de présentation. Dans cette couche les données seront listées dans une liste.

Tâche 1 : Interface de démarrage de l'application

On demande de créer un écran de démarrage de l'application (**SplashScreenActivity**). L'écran restera lisible pendant 3 seconds, ensuite l'utilisateur sera redirigé vers la liste des pizzas. Après, l'écran de démarrage sera détruit.



Copyright - Tout droit réservé - OFPPT

PARTIE 1

Tâche 1 : Interface de démarrage de l'application

Le layout : activity_splashscreen.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:background="@mipmap/back"
    android:layout_height="match_parent">

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="109dp"
        android:layout_height="96dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.292"
        app:srcCompat="@mipmap/icon" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="36dp"
        android:fontFamily="cursive"
        android:text="Pizza Recipes"
        android:textColor="@color/white"
        android:textSize="50dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/imageView2" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:fontFamily="cursive"
        android:text="Make your Pizza"
        android:textColor="@color/white"
        android:textSize="30dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Copyright - Tout droit réservé - OFPPT

Activité n° 2
Création d'interfaces utilisant divers layouts

PARTIE 1

Tâche 1 : Interface de démarrage de l'application

La classe SplashScreenActivity :

```
package ma.android.pizza

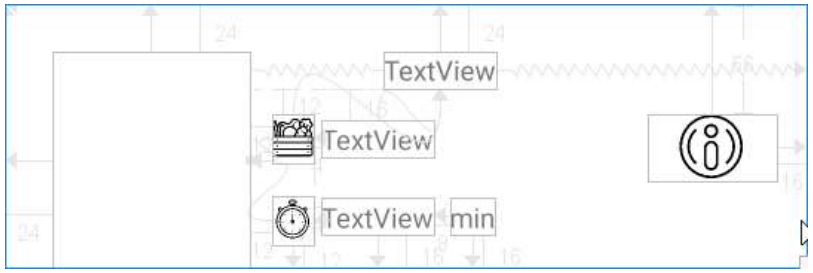
import android.content.Intent
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.os.Handler

class SplashScreenActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_splashscreen)

        supportActionBar?.hide()
        Handler().postDelayed({
            val intent = Intent(this,MainActivity::class.java)
            startActivity(intent)
            finish()
        },3000)
    }
}
```

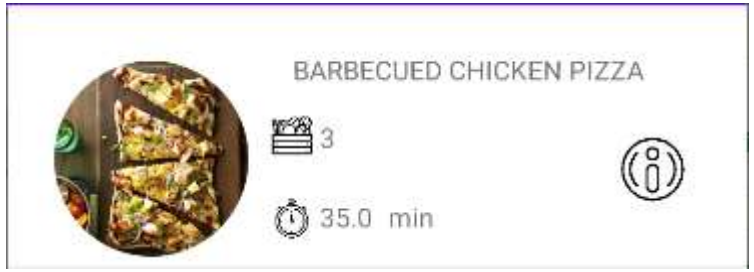
Tâche 2 : Création de l'item

On demande de créer élément qui présentera chaque Pizza dans la liste.



Pour l'image de chaque Pizza utiliser le composant **CircleImageView.** Pour ce faire, ajouter la dépendance :

```
implementation 'de.hdodenhof:circleimageview:3.1.0'
```



Copyright - Tout droit réservé - OFPPT

Tâche 2 : Création de l'item

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <de.hdodenhof.circleimageview.CircleImageView
        android:id="@+id/image"
        android:layout_width="101dp"
        android:layout_height="111dp"
        android:layout_marginStart="24dp"
        android:layout_marginTop="24dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@mipmap/pizza3" />

    <TextView
        android:id="@+id/nom"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="24dp"
        android:rotationX="-4"
        android:text="TextView"
        android:textAppearance="@style/TextAppearance.AppCompat.Small"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.303"
        app:layout_constraintStart_toEndOf="@+id/image"
        app:layout_constraintTop_toTopOf="parent" />

    <ImageView
        android:id="@+id/image1"
        android:layout_width="66dp"
        android:layout_height="34dp"
        android:layout_marginTop="56dp"
        android:layout_marginEnd="16dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:srcCompat="@mipmap/info2" />

    <TextView
        android:id="@+id/id"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:visibility="gone"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/duration"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="4dp"
        android:layout_marginBottom="16dp"
        android:text="TextView"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toEndOf="@+id/imageView2" />
```

PARTIE 1

Copyright - Tout droit réservé - OFPPT

Tâche 2 : Création de l'item

```
<ImageView
    android:id="@+id/imageView"
    android:layout_width="21dp"
    android:layout_height="25dp"
    android:layout_marginStart="12dp"
    android:layout_marginTop="12dp"
    app:layout_constraintStart_toEndOf="@+id/image"
    app:layout_constraintTop_toBottomOf="@+id/nom"
    app:srcCompat="@mipmap/harvest" />

<TextView
    android:id="@+id/ingredients"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="4dp"
    android:layout_marginTop="16dp"
    android:text="TextView"
    app:layout_constraintStart_toEndOf="@+id/imageView"
    app:layout_constraintTop_toBottomOf="@+id/nom" />

<ImageView
    android:id="@+id/imageView2"
    android:layout_width="21dp"
    android:layout_height="25dp"
    android:layout_marginStart="12dp"
    android:layout_marginBottom="12dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toEndOf="@+id/image"
    app:srcCompat="@mipmap/time" />

<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:layout_marginBottom="16dp"
    android:text="min"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toEndOf="@+id/duration" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

PARTIE 1

Copyright - Tout droit réservé - OFPPT

Activité n° 2
Création d'interfaces utilisant divers layouts

Tâche 3 : Modifier le fichier activity_main.xml

Insérer le composant **ListView** dans le layout :

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <ListView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/listview"
        tools:listitem="@layout/list_item"/>

</androidx.constraintlayout.widget.ConstraintLayout>
```

Tâche 4 : Créer la classe PizzaAdapter

```
class PizzaAdapter(activity: Activity, pizzas: List<Pizza>) :
    BaseAdapter() {
    private val pizzas: List<Pizza>
    private val inflater: LayoutInflater

    init {
        this.pizzas = pizzas
        inflater =
activity.getSystemService(Context.LAYOUT_INFLATER_SERVICE) as
                LayoutInflater
    }

    override fun getCount(): Int {
        return pizzas.size
    }

    override fun getItem(i: Int): Any {
        return pizzas[i]
    }

    override fun getItemId(p0: Int): Long {
        return pizzas[p0].id.toLong()
    }
```

PARTIE 1

Copyright - Tout droit réservé - OFPPT

PARTIE 1

Tâche 4 : Créer la classe PizzaAdapter

```
override fun getView(i: Int, convertView: View?, parent: ViewGroup): View
{

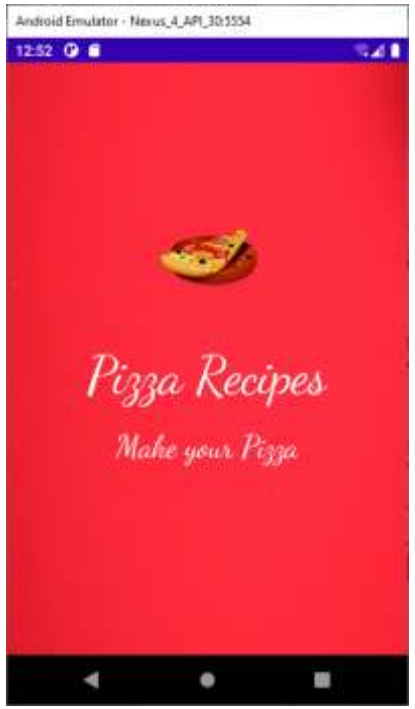
    // val inflater : LayoutInflater = LayoutInflater.from(context)
     val view : View = inflater.inflate(R.layout.list_item,null);
     val nom = view.findViewById<TextView>(R.id.nom)
     val ingredients = view.findViewById<TextView>(R.id.ingredients)
     val duration = view.findViewById<TextView>(R.id.duration)
     val id = view.findViewById<TextView>(R.id.id)
     val image = view.findViewById<ImageView>(R.id.image)
     nom.text = pizzas[i].nom
     ingredients.text = pizzas[i].nbrIngredients.toString()
     duration.text= pizzas[i].duree.toString()
     id.text = pizzas[i].id.toString()
     image.setImageResource(pizzas[i].photo)
     return view
   }
}
```

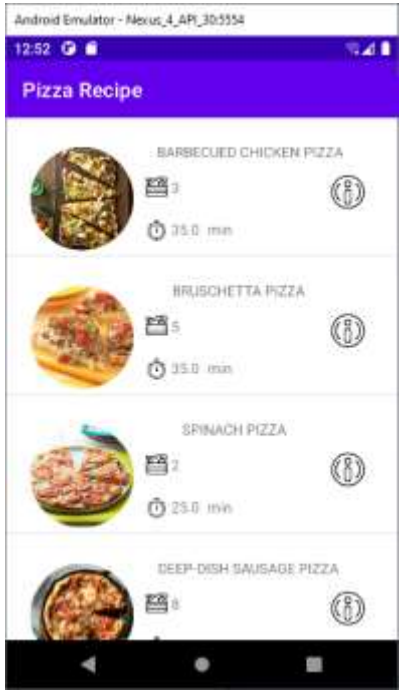
Tâche 5 : Associer l'adapter à la liste

Dans la classe **MainActivity** associer l'adapter à la liste :

```
Log.d("Liste", fs.findAll().toString());
liste.adapter = PizzaAdapter(this, fs.findAll())
```

Ensuite, lancer l'application.





Copyright - Tout droit réservé - OFPPT

Etape 7 : Intégrer le menu

On demande d'intégrer un menu permettant de partager l'application via les applications de messageries (WhatsApp, Messenger, …).

Tâche 1 : Créer le menu

Dans le dossier menu, créer le fichier XML « menu.xml » :

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">

    <item
        android:id="@+id/share"
        android:title="share"
        android:icon="@drawable/share"
        app:showAsAction="always"/>

</menu>
```



Copyright - Tout droit réservé - OFPPT

PARTIE 1

Tâche 2 : Associer le menu à l'activité

Dans la classe **MainActivity**. Ajouter les deux méthodes :

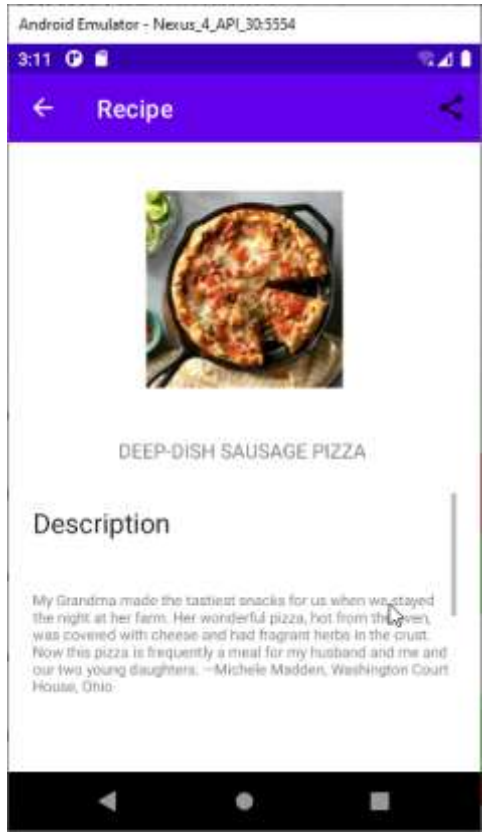
```
override fun onCreateOptionsMenu(menu: Menu?): Boolean {
    menuInflater.inflate(R.menu.menu, menu)
    return super.onCreateOptionsMenu(menu)
}

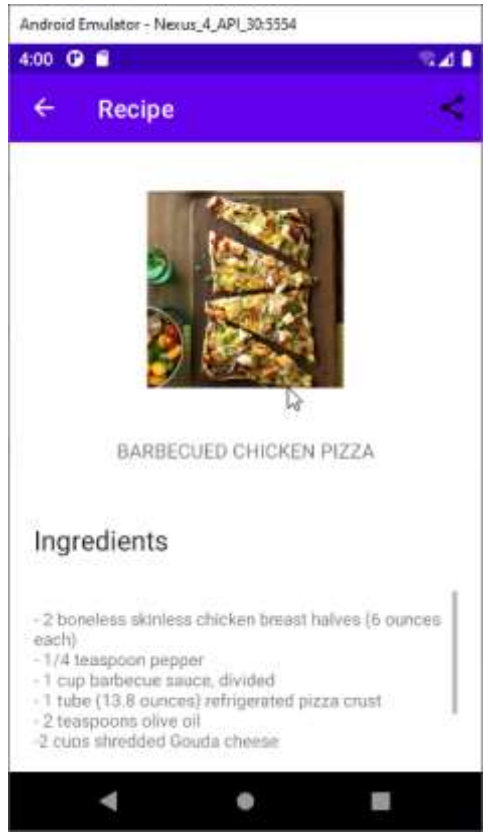
override fun onOptionsItemSelected(item: MenuItem): Boolean {
    val intent = Intent(Intent.ACTION_SEND)
    intent.type = "text/plain"
    intent.putExtra(Intent.EXTRA_SUBJECT, "Check out these cool
recipes!!!")
    intent.putExtra(Intent.EXTRA_TEXT, "The apps link's here....")
    startActivity(Intent.createChooser(intent, "Share via"))
    return super.onOptionsItemSelected(item)
}
```

Lancer l'application et tester le menu.

Etape 8 : Afficher les détails d'un Pizza

Au clic sur un élément de la liste, afficher les détails d'une Pizza.





Copyright - Tout droit réservé - OFPPT

Tâche 1 : Associer un événement de click à l'item

1. Dans la classe **MainActivity** implémenter l'interface **OnItemClickListener**.

```
class MainActivity : AppCompatActivity(), OnItemClickListener
```

2. Dans la méthode onCreate(), associer l'événement :

```
liste.adapter = PizzaAdapter(this, fs.findAll())
liste.onItemClickListener = this    //ligne à intégrer
```

3. Ensuite, redéfinir la méthode **onItemClick** :

```
override fun onItemClick(adapterView: AdapterView<*>?, view: View, i: Int, l: Long) {
    val id = view.findViewById<TextView>(R.id.id)
    val intent = Intent(this, PizzaDetailActivity::class.java)
    //intent.putExtra("Description",id.getText());
    intent.putExtra("id", id.text)
    startActivity(intent)
}
```

Ici, juste l'id qui sera transmet vers l'activité **PizzaDetailActivity**.

Copyright - Tout droit réservé - OFPPT

PARTIE 1

Tâche 2 : Créer l'activité PizzaDetailActivity

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="20dp"
    tools:context=".PizzaDetailActivity">


    <ImageView
        android:id="@+id/imageView5"
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_gravity="center"
        android:padding="20dp"
        app:srcCompat="@drawable/ic_launcher_background" />

    <TextView
        android:id="@+id/pizza_name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center_horizontal"
        android:paddingTop="20dp"
        android:paddingBottom="20dp"

android:textAppearance="@style/TextAppearance.AppCompat.Display2"
        android:textSize="16sp" />
```

```
<ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="694dp"
            android:orientation="vertical">

            <TextView
                android:id="@+id/textView2"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:paddingTop="20dp"
                android:paddingBottom="20dp"
                android:text="Description"
                android:textAppearance="@style/TextAppearance.AppCompat.Large" />

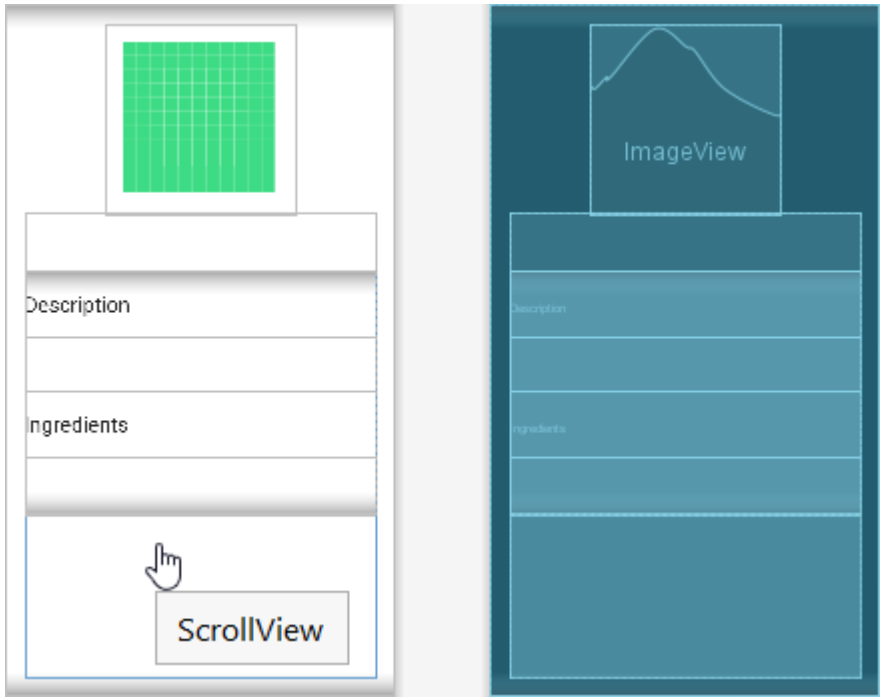
            <TextView
                android:id="@+id/description"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:paddingTop="20dp"
                android:paddingBottom="20dp"
                android:textSize="12sp" />
```

Copyright - Tout droit réservé - OFPPT

Tâche 2 : Créer l'activité PizzaDetailActivity

```
<TextView
            android:id="@+id/textView4"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:paddingTop="20dp"
            android:paddingBottom="20dp"
            android:text="Ingredients"
            android:textAppearance="@style/TextAppearance.AppCompat.Large" />

        <TextView
            android:id="@+id/ingredient"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:paddingTop="20dp"
            android:paddingBottom="20dp" />
    </LinearLayout>
  </ScrollView>
</LinearLayout>
```



PARTIE 1

Copyright - Tout droit réservé - OFPPT

PARTIE 1

Tâche 3 : Modifier la classe PizzaDetailActivity

```
class PizzaDetailActivity : AppCompatActivity() {

    private lateinit var binding: ActivityPizzaBinding
    lateinit var pizza_name: TextView
    lateinit var description: TextView
    lateinit var ingredient: TextView
    lateinit var image: ImageView
    lateinit var ps: PizzaService

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityPizzaBinding.inflate(layoutInflater)
        setContentView(binding.root)

        image = findViewById(R.id.imageView5)
        pizza_name = findViewById(R.id.pizza_name)
        description = findViewById(R.id.description)
        ingredient = findViewById(R.id.ingredient)
        val intent = intent
        ps = PizzaService.instance!!
        val p: Pizza? = ps.findById(Integer.parseInt(intent.getStringExtra("id")))

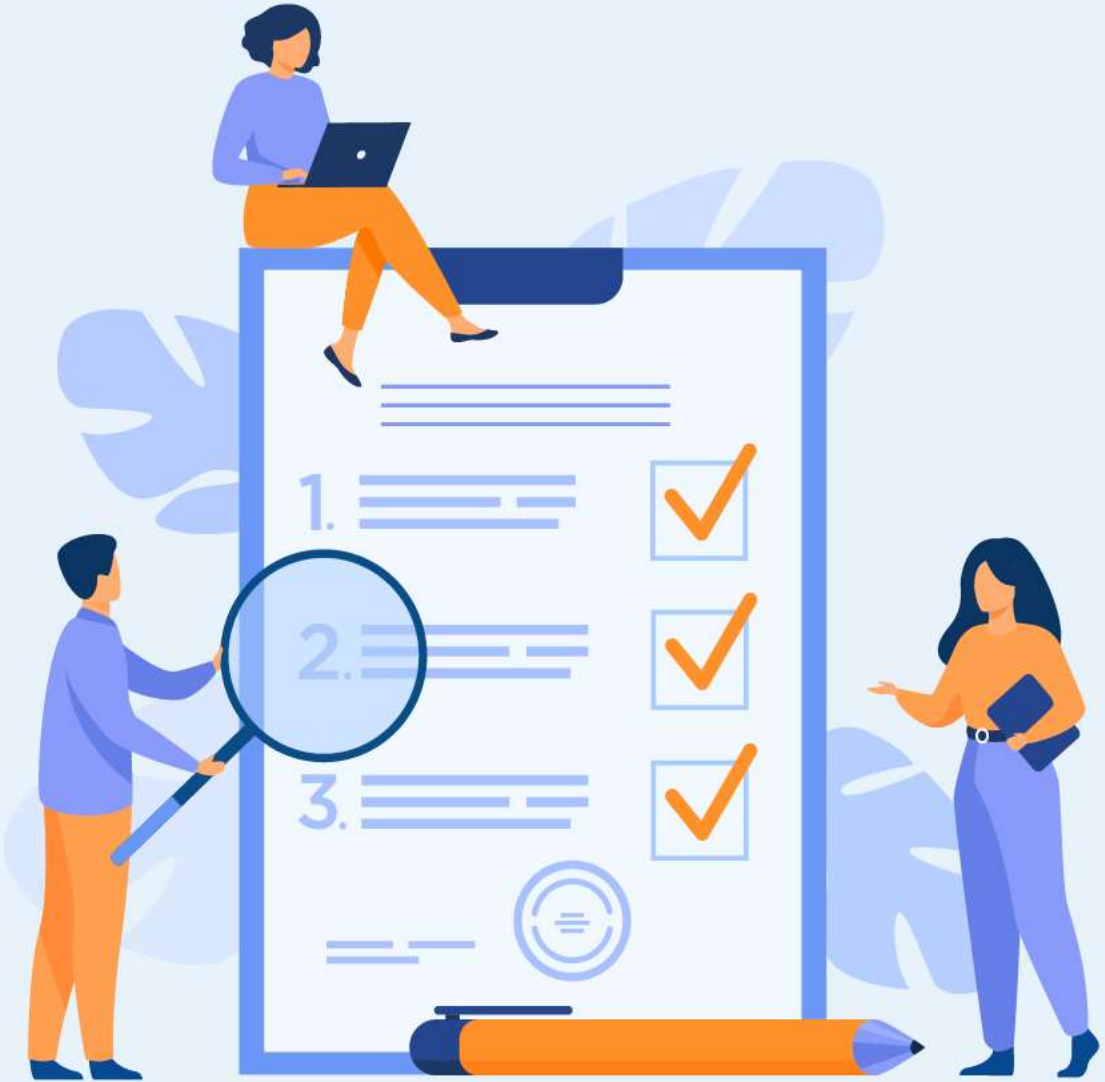
        if (p != null) {
            pizza_name.text = p.nom
            image.setImageResource(p.photo)
            description.text = p.description
            ingredient.text = p.detailsIngred
        }
    }

    //share button
    override fun onCreateOptionsMenu(menu: Menu?): Boolean {
        menuInflater.inflate(R.menu.menu, menu)
        return super.onCreateOptionsMenu(menu)
    }
    override fun onOptionsItemSelected(item: MenuItem): Boolean {
        when (item.itemId) {
            R.id.share -> {
                shareApp()
            }
        }
        return super.onOptionsItemSelected(item)
    }
    private fun shareApp() {
        val msg: String = "hi there, checkout these cool recipes"
        val intent = Intent()
        intent.action = Intent.ACTION_SEND
        intent.putExtra(Intent.EXTRA_TEXT, msg)
        intent.type = "text/plain"
        startActivity(intent)
    }
}
```

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 3

Création de plusieurs contrôles dont des RecyclerView



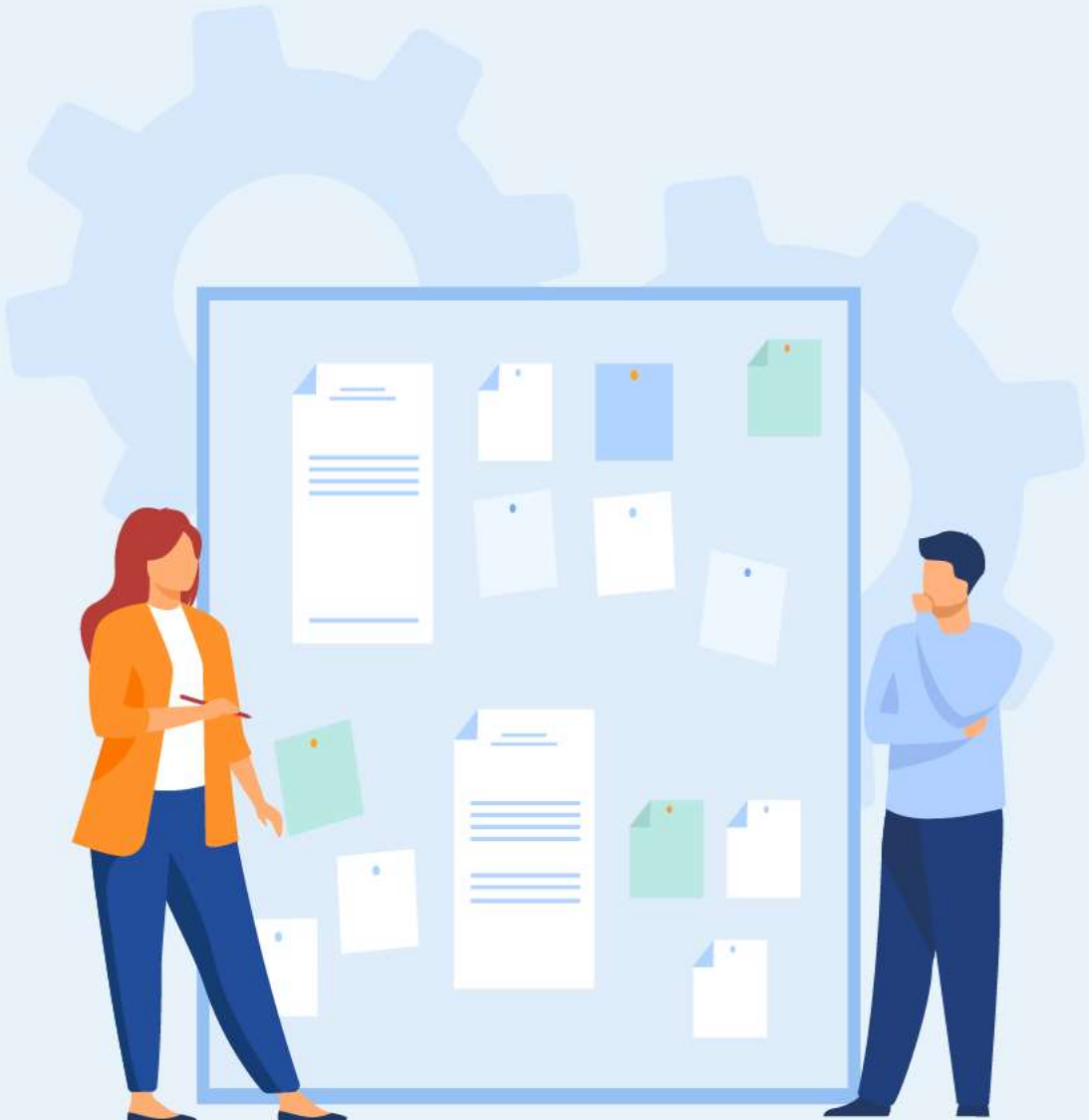
Compétences visées :

- Réaliser un écran de démarrage d'une application mobile
- Comprendre l'utilisation d'un RecyclerView
- Créer les menus
- Réaliser une RecyclerView avec filtrage des données
- Utiliser les images cerclées (CircularView, CircleImageView)
- Utiliser la bibliothèque Glide

Recommandations clés :

- Révision générale du résumé théorique ;
- Suivre les instructions du TP et organiser le dossier de travail.

12 heures



CONSIGNES

Pour le formateur :

- Rappeler les étapes à suivre pour mettre en place un RecyclerView ;
- Demander aux apprenants de suivre les étapes décrites dans l'activité.

Pour l'apprenant :

- Suivre les étapes indiquées dans l'activité.

Conditions de réalisation :

- Support de résumé théorique accompagnant ;
- Un ordinateur sur lequel est installée la dernière version stable d'Android Studio ;
- Connexion internet.

Critères de réussite :

- Le stagiaire doit être capable de :
  - manipuler un RecyclerView ;
  - utiliser la bibliothèque Glide ;
  - développer une application selon une architecture en couche.

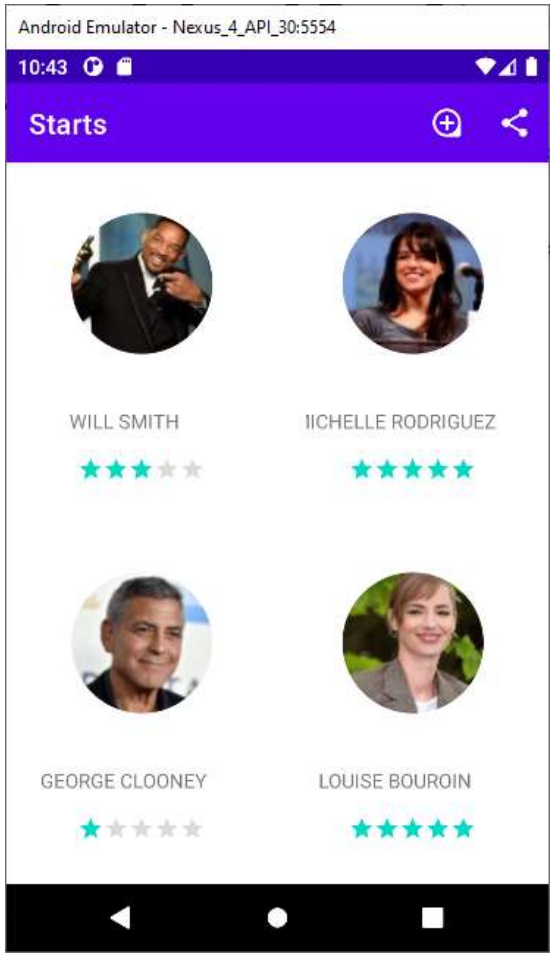
Activité n° 3

Création de plusieurs contrôles dont des RecyclerView

Enoncé

Créer une application Android qui affiche une liste d'objets représentant des stars de cinéma, en utilisant la bibliothèque Glide pour télécharger les images depuis un site web. L'application doit permettre à l'utilisateur de filtrer la liste en fonction d'un critère de recherche et de donner une note (de 1 à 5) à chaque star en cliquant sur un élément de la liste. La note doit être mise à jour dans la source de données et l'élément de la liste doit être mis à jour dans l'interface utilisateur.

En plus de la possibilité de filtrer et de noter les stars de cinéma, l'application doit également permettre à l'utilisateur de partager les informations sur l'application via des systèmes de messagerie tels que WhatsApp et Messenger.



PARTIE 1

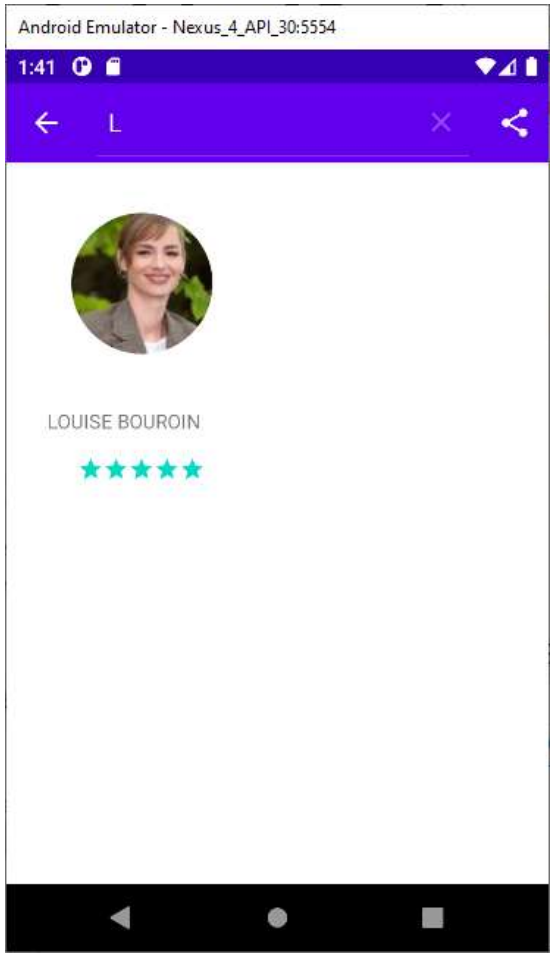
Copyright - Tout droit réservé - OFPPT

Activité n° 3

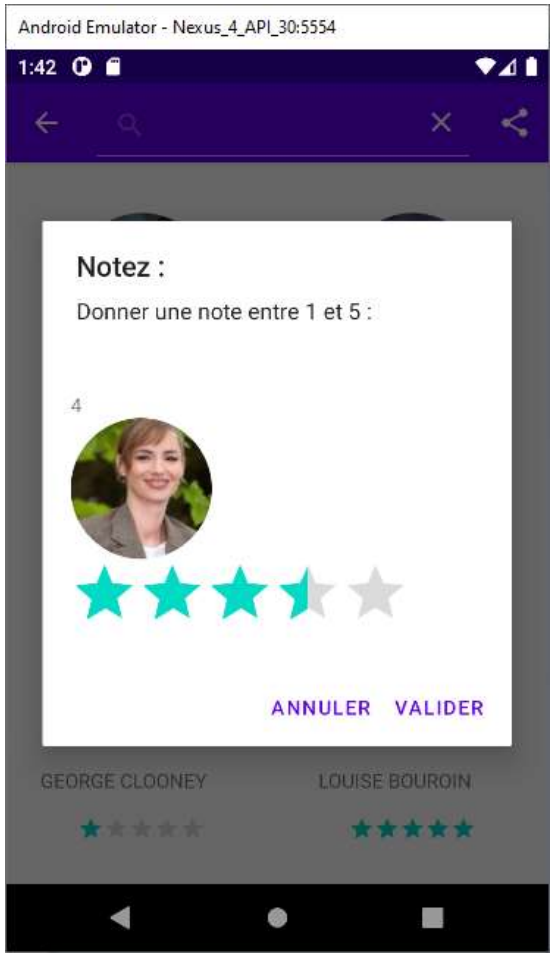
Création de plusieurs contrôles dont des RecyclerView

PARTIE 1

Filtrer la liste



Noter un Star



Copyright - Tout droit réservé - OFPPT

Activité n° 3

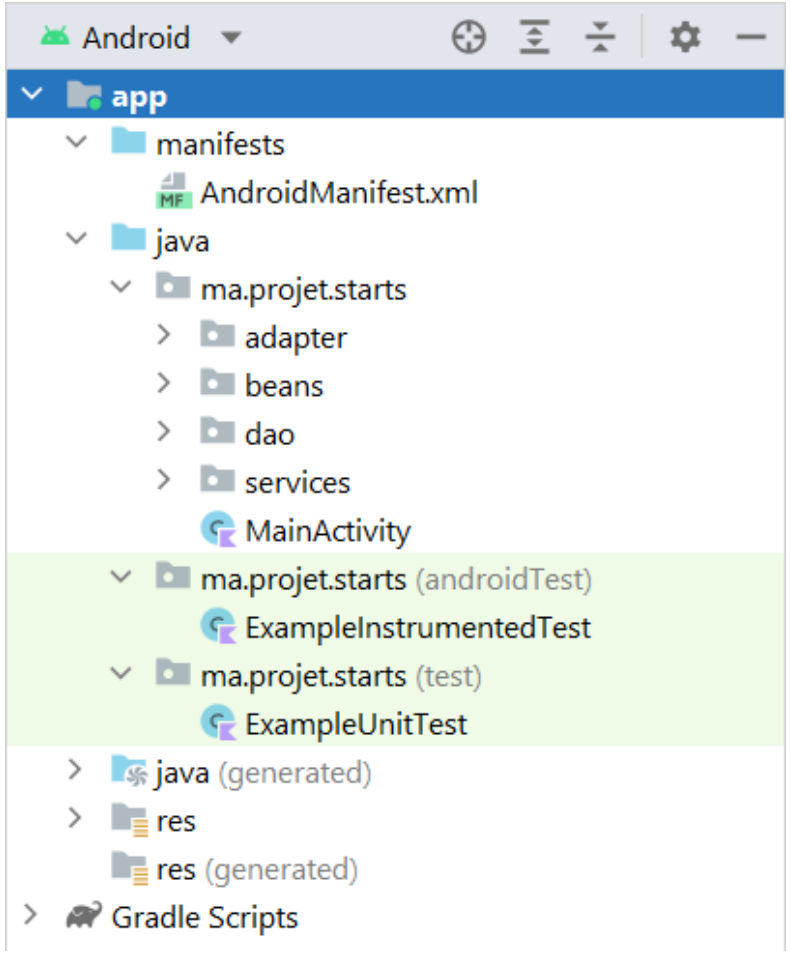
Création de plusieurs contrôles dont des RecyclerView

Etape 1 : Architecture de projet

On demande de créer une application mobile Android avec le langage Kotlin.

Cette application sera implémenter selon une architecture en couche, ainsi, on demande de créer les sous-packages suivants :

- adapter ;
- beans ;
- dao ;
- Services.



PARTIE 1

Copyright - Tout droit réservé - OFPPT

Activité n° 3
Création de plusieurs contrôles dont des RecyclerView

PARTIE 1

Etape 2 : Classe Star

Dans le package **beans**, créer la classe Star suivante :

```
class Star(name: String, img: String, star: Float) {
    var id: Int
    var name: String
    var img: String
    var star: Float

    companion object {
        private var comp = 0
    }

    init {
        id = ++comp
        this.name = name
        this.img = img
        this.star = star
    }
}
```

Etape 3 : Interface IDao

Dans le package **dao**, créer l'interface **IDao** suivante :

```
package ma.projet.acteurs.dao

interface IDao<T> {
    fun create(o: T): Boolean
    fun update(o: T): Boolean
    fun delete(o: T): Boolean
    fun findById(id: Int): T
    fun findAll(): List<T>?
}
```

Copyright - Tout droit réservé - OFPPT

Activité n° 3
Création de plusieurs contrôles dont des RecyclerView

PARTIE 1

Etape 4 : Classe StarService

Dans le package **services**, créer la classe **StarService** suivante :

```
class StarService private constructor() : IDao<Star?> {
    private val stars: MutableList<Star>

    init {
        stars = ArrayList()
    }

    companion object {
        @JvmStatic
        var instance: StarService? = null
            get() {
                if (field == null) field = StarService()
                return field
            }
            private set
    }

    override fun create(o: Star?): Boolean {
        return stars.add(o!!)
    }

    override fun update(o: Star?): Boolean {
        for (s in stars) {
            if (o != null) {
                if (s.star == o.id.toFloat()) {
                    s.img = o.img
                    s.name = o.name
                    s.star = o.star
                }
            }
        }
        return true
    }
    override fun delete(o: Star?): Boolean {
        return stars.remove(o)
    }

    override fun findById(id: Int): Star? {
        for (s in stars) {
            if (s.id == id) return s
        }
        return null
    }
    override fun findAll(): List<Star> {
        return stars
    }
```

Copyright - Tout droit réservé - OFPPT

Activité n° 3

Création de plusieurs contrôles dont des RecyclerView

Etape 5 : Dépendances de projet

Ajouter les dépendances suivantes dans gradle (module) :

```
implementation "androidx.recyclerview:recyclerview:1.2.1"
implementation "androidx.recyclerview:recyclerview-selection:1.1.0"
implementation 'de.hdodenhof:circleimageview:3.0.1'
implementation 'com.github.bumptech.glide:glide:4.8.0'
annotationProcessor 'com.github.bumptech.glide:compiler:4.8.0'
```

Etape 6 : activity_main.xml

Dans le fichier **activity_main.xml**, ajouter le composant **RecyclerView** :

```
<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="ma.projet.starts.MainActivity">


    <androidx.recyclerview.widget.RecyclerView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/recycle_view"/>

</LinearLayout>
```

PARTIE 1

Copyright - Tout droit réservé - OFPPT

PARTIE 1

Etape 7 : Layout d'un élément Star

Créer le fichier **star_item.xml** dans le dossier layout, qui représentera chaque star dans la liste.




```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:padding="20dp"
    android:id="@+id/parent"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">

        <de.hdodenhof.circleimageview.CircleImageView
            android:id="@+id/img"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:layout_marginTop="16dp"
            android:src="@mipmap/star"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.498"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />
```

Copyright - Tout droit réservé - OFPPT

Etape 7 : Layout d'un élément Star (Suite)

```
<RatingBar
    android:id="@+id/stars"
    style="@android:style/Widget.Material.RatingBar.Indicator"
    android:layout_width="183dp"
    android:layout_height="35dp"
    android:accessibilityHeading="false"
    android:clickable="false"
    android:isIndicator="true"
    android:numStars="5"
    android:scaleX="0.5"
    android:scaleY="0.5"
    android:stepSize="0.1"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/name" />

<TextView
    android:id="@+id/name"
    android:layout_width="179dp"
    android:layout_height="48dp"
    android:layout_marginTop="16dp"
    android:paddingTop="15dp"
    android:text="TextView"
    android:gravity="center"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/img" />

<TextView
    android:id="@+id/ids"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="TextView"
    android:visibility="gone" />
</androidx.constraintlayout.widget.ConstraintLayout>

</RelativeLayout>
```

PARTIE 1

Copyright - Tout droit réservé - OFPPT

Etape 8 : Menu principal

Dans le dossier **menu**, créer le menu suivante :

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/app_bar_search"
        android:icon="@drawable/ic_baseline_loupe_24"
        android:title="Search"
        app:actionViewClass="android.widget.SearchView"
        app:showAsAction="always|collapseActionView"

        />
    <item
        android:id="@+id/share"
        android:icon="@drawable/ic_baseline_share_24"
        android:title="Search"
        app:actionViewClass="android.widget.SearchView"
        app:showAsAction="always|collapseActionView"

        />
</menu>
```





PARTIE 1

Copyright - Tout droit réservé - OFPPT

Etape 9 : Classe Adapter

Dans le package **adapter**, créer la classe **StarAdapter** suivante :

```
class StarAdapter(private val context: Context, private val stars: List<Star>) : RecyclerView.Adapter<StarViewHolder>(), Filterable {
    private var starsFilter: MutableList<Star>
    private val mfilter: NewFilter
    override fun onCreateViewHolder(viewGroup: ViewGroup, i: Int): StarViewHolder {
        val v = LayoutInflater.from(context).inflate(
            R.layout.star_item,
                viewGroup, false)
        val holder = StarViewHolder(v)
        holder.itemView.setOnClickListener { v ->
            val popup = LayoutInflater.from(context).inflate(R.layout.star_edit_item, null,
                    false)
            val img = popup.findViewById<ImageView>(R.id.img)
            val bar = popup.findViewById<RatingBar>(R.id.ratingBar)
            val idss = popup.findViewById<TextView>(R.id.idss)
            val bitmap = ((v.findViewById<View>(R.id.img) as ImageView).drawable as BitmapDrawable).bitmap
            img.setImageBitmap(bitmap)
            bar.rating = (v.findViewById<View>(R.id.stars) as RatingBar).rating
            idss.text = (v.findViewById<View>(R.id.ids) as TextView).text.toString()
```

Copyright - Tout droit réservé - OFPPT

Activité n° 3
Création de plusieurs contrôles dont des RecyclerView

Etape 9 : Classe Adapter (Suite)

```
val dialog = AlertDialog.Builder(context)
            .setTitle("Notez : ")
            .setMessage("Donner une note entre 1 et 5 :")
            .setView(popup)
            .setPositiveButton("Valider") { dialog, which ->
                val s = bar.rating
                val ids = idss.text.toString().toInt()
                val star = instance!!.findById(ids)
                if (star != null) {
                    star.star = s
                }
                instance!!.update(star)
                notifyItemChanged(holder.adapterPosition)
            }
            .setNegativeButton("Annuler", null)
            .create()
        dialog.show()
    }
    return holder
}
```

```
override fun onBindViewHolder(starViewHolder: StarViewHolder, i: Int) {
    Log.d(TAG, "onBindView call ! $i")
    Glide.with(context)
        .asBitmap()
        .load(starsFilter[i].img)
        .apply(RequestOptions().override(100, 100))
        .into(starViewHolder.img)
    starViewHolder.name.text =
            starsFilter[i].name.uppercase(Locale.getDefault())
    starViewHolder.stars.rating = starsFilter[i].star
    starViewHolder.idss.text = starsFilter[i].id.toString() + ""
}

override fun getItemCount(): Int {
    return starsFilter.size
}

override fun getFilter(): Filter {
    return mfilter
}
```

PARTIE 1

Copyright - Tout droit réservé - OFPPT

PARTIE 1

Etape 9 : Classe Adapter (Suite)

```
 inner class StarViewHolder(itemView: View) :
RecyclerView.ViewHolder(itemView) {
        var idss: TextView
        var img: ImageView
        var name: TextView
        var stars: RatingBar
        var parent: RelativeLayout

        init {
            idss = itemView.findViewById(R.id.ids)
            img = itemView.findViewById(R.id.img)
            name = itemView.findViewById(R.id.name)
            stars = itemView.findViewById(R.id.stars)
            parent = itemView.findViewById(R.id.parent)
        }
    }
```

```
 inner class NewFilter(var mAdapter: RecyclerView.Adapter<*>) : Filter() {
        override fun performFiltering(charSequence: CharSequence):
FilterResults {
            starsFilter.clear()
            val results = FilterResults()
            if (charSequence.length == 0) {
                starsFilter.addAll(stars)
            } else {
                val filterPattern =
charSequence.toString().lowercase(Locale.getDefault()).trim { it <= ' ' }
                for (p in stars) {
                    if
(p.name.lowercase(Locale.getDefault()).startsWith(filterPattern)) {
                        starsFilter.add(p)
                    }
                }
            }
            results.values = starsFilter
            results.count = starsFilter.size
            return results
        }
```

Copyright - Tout droit réservé - OFPPT

Etape 9 : Classe Adapter (Suite)

```
 override fun publishResults(charSequence: CharSequence, filterResults: FilterResults) {
            starsFilter = filterResults.values as MutableList<Star>
            mAdapter.notifyDataSetChanged()
        }
    }

    companion object {
        private const val TAG = "StarAdapter"
    }

    init {
        starsFilter = ArrayList()
        starsFilter.addAll(stars)
        mfilter = NewFilter(this)
    }
}
```


Copyright - Tout droit réservé - OFPPT

Activité n° 3
Création de plusieurs contrôles dont des RecyclerView

Etape 10 : Modifier le layout activity_main.xml

Dans le fichier **activity_main.xml**, déclarer le composant **RecyclerView**.

```
<?xml version="1.0" encoding="utf-8" ?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="ma.projet.starts.MainActivity">

    <androidx.recyclerview.widget.RecyclerView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/recycle_view"/>

</LinearLayout>
```

Etape 11 : Modifier la classe MainActivity

```
class MainActivity : AppCompatActivity() {
    private var stars: List<Star>? = null
    private var recyclerView: RecyclerView? = null
    private var starAdapter: StarAdapter? = null
    private var service: StarService? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        stars = ArrayList()
        service = StarService.instance
        init()
        recyclerView = findViewById(R.id.recycle_view)
        starAdapter = StarAdapter(this, service!!.findAll())
        if(recyclerView != null){
            recyclerView!!.adapter = starAdapter
            recyclerView!!.layoutManager = GridLayoutManager(this, 2)
        }
    }
    fun init() {
        service!!.create(
            Star("Blake Lively", "http://www.stars-photos.com/image.php?id=3278", 3F)
        )
        service!!.create(
            Star("Jennifer Lawrence","http://www.stars-photos.com/image.php?id=4593",5F)
        )
        //Ajouter d'autres starts
    }
```

PARTIE 1

Copyright - Tout droit réservé - OFPPT

Activité n° 3
Création de plusieurs contrôles dont des RecyclerView

Etape 12 : Associer le menu à l'activité

Dans la classe **MainActivity**, ajouter les deux méthodes :

```
override fun onCreateOptionsMenu(menu: Menu): Boolean {
    menuInflater.inflate(R.menu.menu, menu)
    val menuItem = menu.findItem(R.id.app_bar_search)
    val searchView = MenuItemCompat.getActionView(menuItem) as SearchView
    searchView.setOnQueryTextListener(object : SearchView.OnQueryTextListener {
        override fun onQueryTextSubmit(query: String): Boolean {
            return true
        }
        override fun onQueryTextChange(newText: String): Boolean {
            if (starAdapter != null) {
                starAdapter!!.filter.filter(newText)
            }
            return true
        }
    })
    return true
}
```

```
override fun onOptionsItemSelected(item: MenuItem): Boolean {
    if (item.itemId == R.id.share) {
        val txt = "Stars"
        val mimeType = "text/plain"
        ShareCompat.IntentBuilder
            .from(this)
            .setType(mimeType)
            .setChooserTitle("Stars")
            .setText(txt)
            .startChooser()
    }
    return super.onOptionsItemSelected(item)
  }
}
```

PARTIE 1

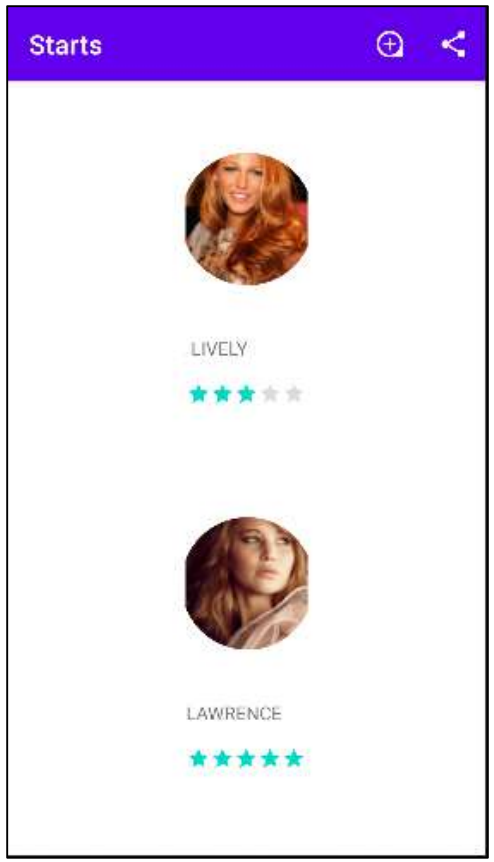
Copyright - Tout droit réservé - OFPPT

Activité n° 3

Création de plusieurs contrôles dont des RecyclerView

Etape 13 : Affichage linéaire

Modifier le code afin d'afficher les stars dans **LinearLayout**.



Solution

Modifier la ligne :

```
recyclerView!!.layoutManager = GridLayoutManager(this, 2)
```

Par :

```
recyclerView!!.layoutManager = LinearLayoutManager(this)
```

PARTIE 1

Copyright - Tout droit réservé - OFPPT

Activité n° 3

Création de plusieurs contrôles dont des RecyclerView

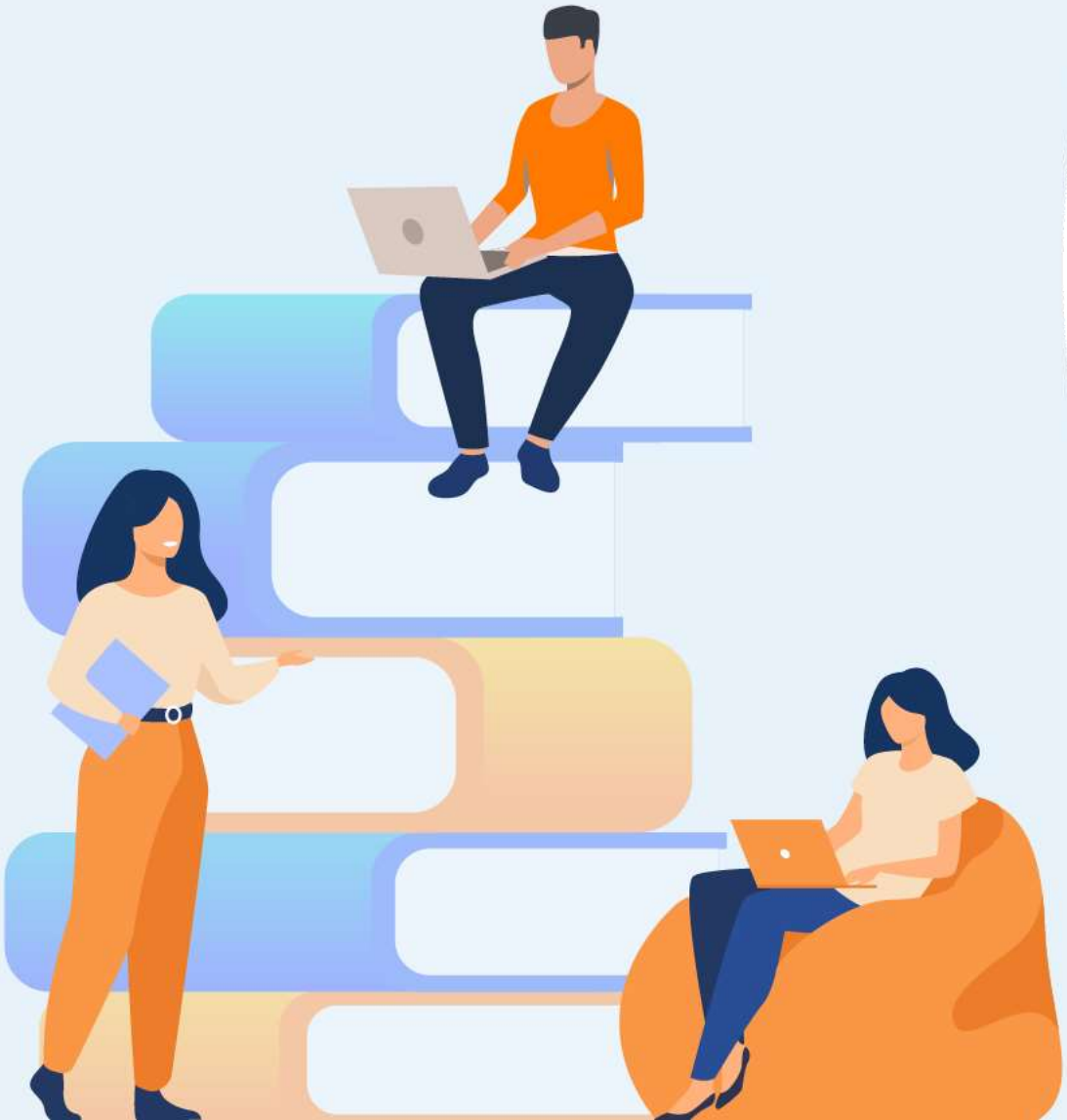
Challenge

Ajouter les fonctionnalités suivantes à l'application :

- Supprimer un star de la liste en effectuant un balayage (swipe) vers la droite sur cet élément ;
- Afficher un popup pour ajouter un star à la liste en effectuant un balayage (swipe) vers la gauche."

PARTIE 1

Copyright - Tout droit réservé - OFPPT



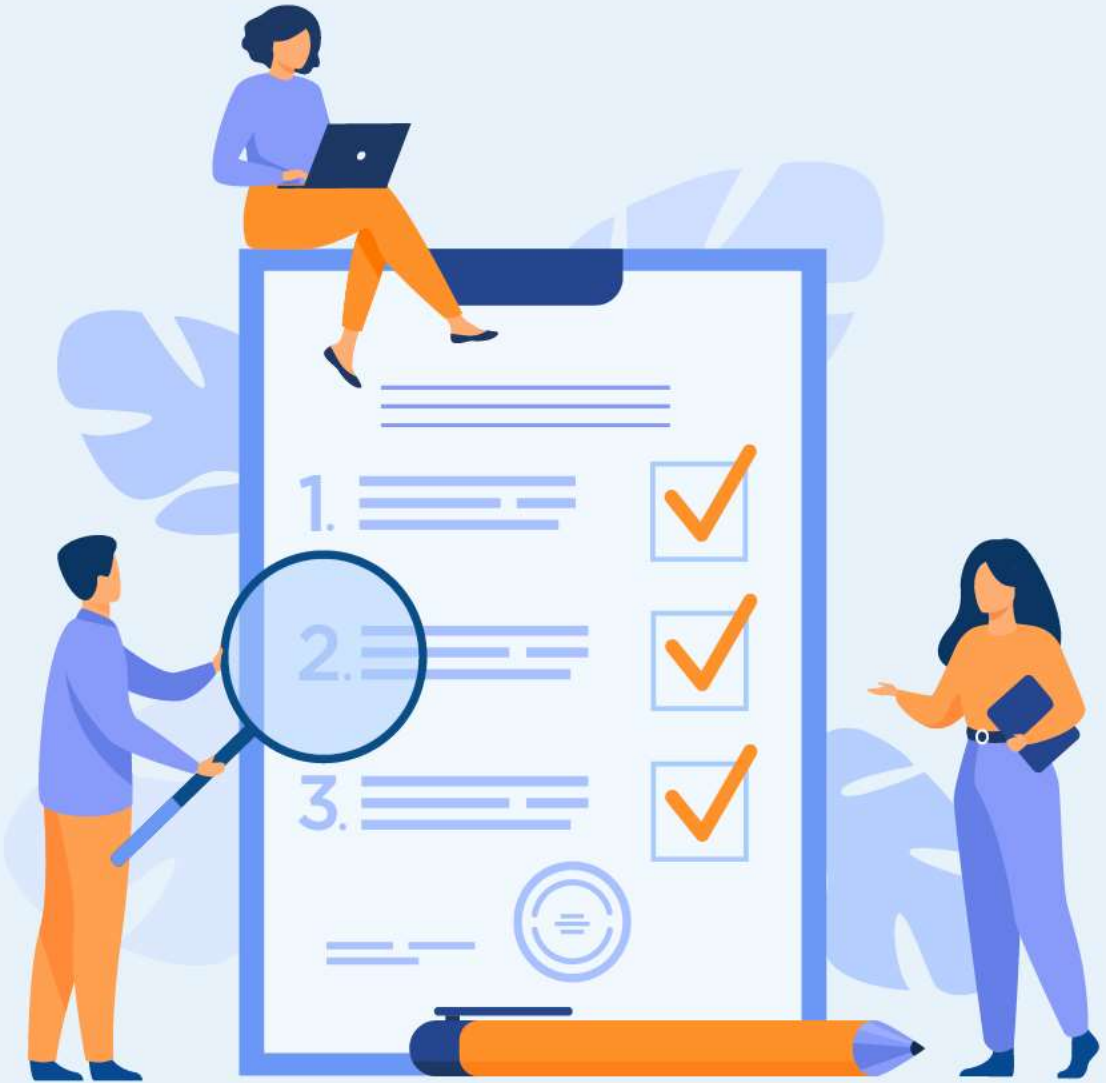
PARTIE 2

Créer des interfaces graphiques avancées

Dans ce module, vous allez :

- Intégration correcte des styles et des ressources ;
- Utilisation des Canvas ;
- Implémenter des vues en utilisant le Material design ;
- Gestion correcte des animations entre les activités.

28 heures



ACTIVITÉ n° 1

Création de plusieurs vues personnalisées

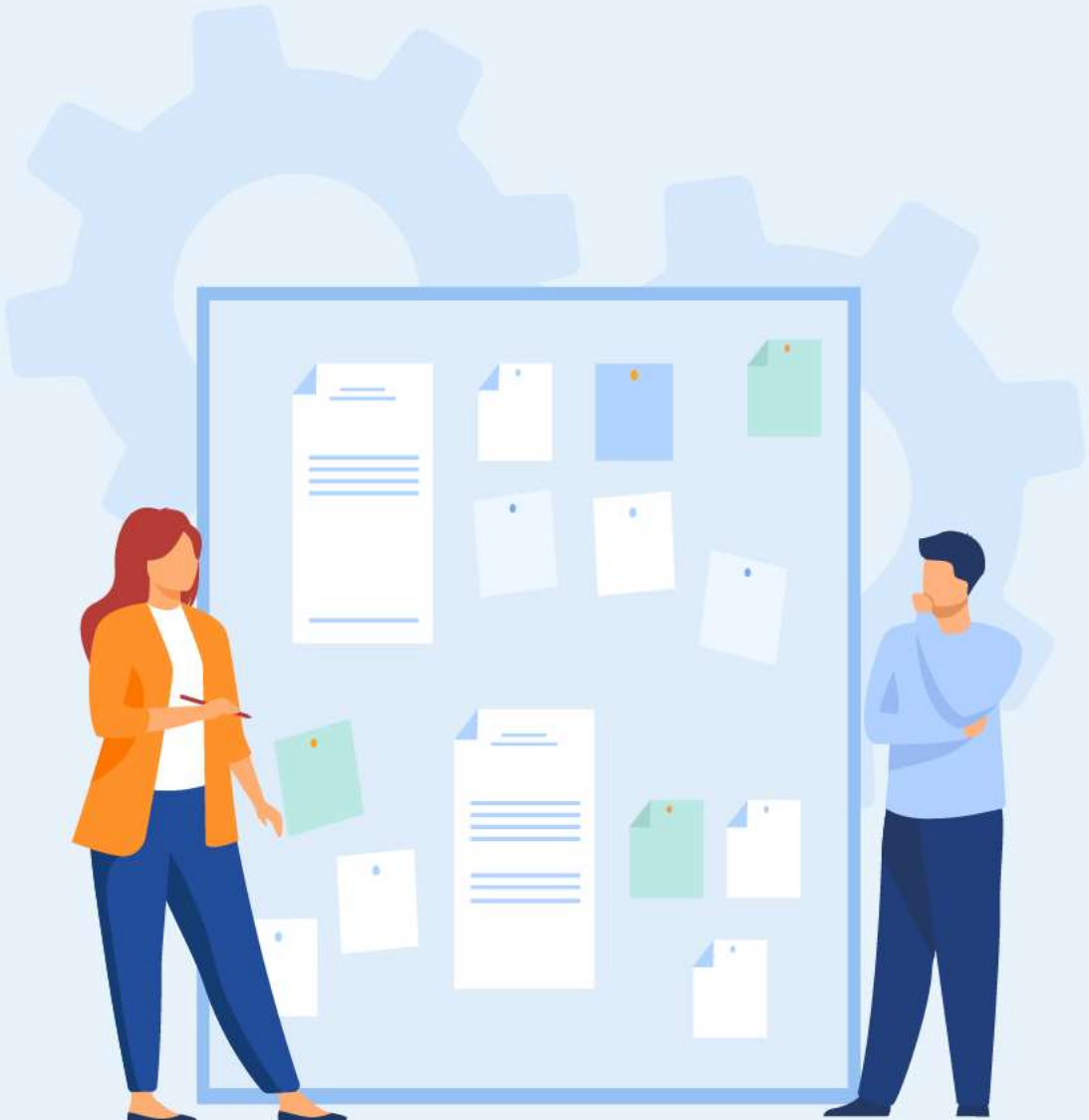
Compétences visées :

- Faciliter la navigation dans l'application en offrant un accès rapide aux différentes sections et fonctionnalités de l'application ;
- Améliorer l'expérience utilisateur en fournissant une interface de navigation intuitive et cohérente dans toutes les sections de l'application ;
- Adapter l'interface de navigation en fonction des besoins de l'application;
- Faciliter le développement et la maintenance de l'application en fournissant une interface de navigation modulaire et réutilisable.

Recommandations clés :

- Révision générale du résumé théorique ;
- Suivre les instructions du TP et organiser le dossier de travail.

10 heures



CONSIGNES

1. Pour le formateur :

- Demander aux apprenants de suivre les étapes décrites dans l’activité.

2. Pour l’apprenant :

- Suivre les étapes indiquées dans l’activité.

3. Conditions de réalisation :

- Support de résumé théorique accompagnant ;
- Un ordinateur sur lequel est installée la dernière version stable d'Android Studio.

4. Critères de réussite :

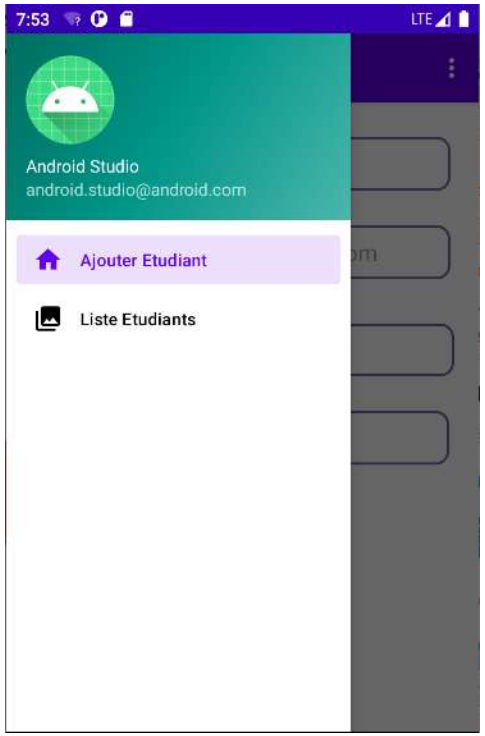
- Le stagiaire doit être capable de :
  - manipuler une navigation darwer ;
  - utiliser la caméra pour prendre et stocker des photos.

ACTIVITÉ n° 1

Création de plusieurs vues personnalisées

Enoncé

Créer une application Android qui utilise une activité de type **NavigationDrawer** et deux fragments pour gérer une liste d'étudiants. Le premier fragment doit permettre à l'utilisateur de remplir un formulaire pour ajouter de nouveaux étudiants à la liste, tandis que le second doit afficher la liste des étudiants existants. Les étudiants sont caractérisés par leur nom, prénom, date de naissance et ville.





PARTIE 2

Copyright - Tout droit réservé - OFPPT

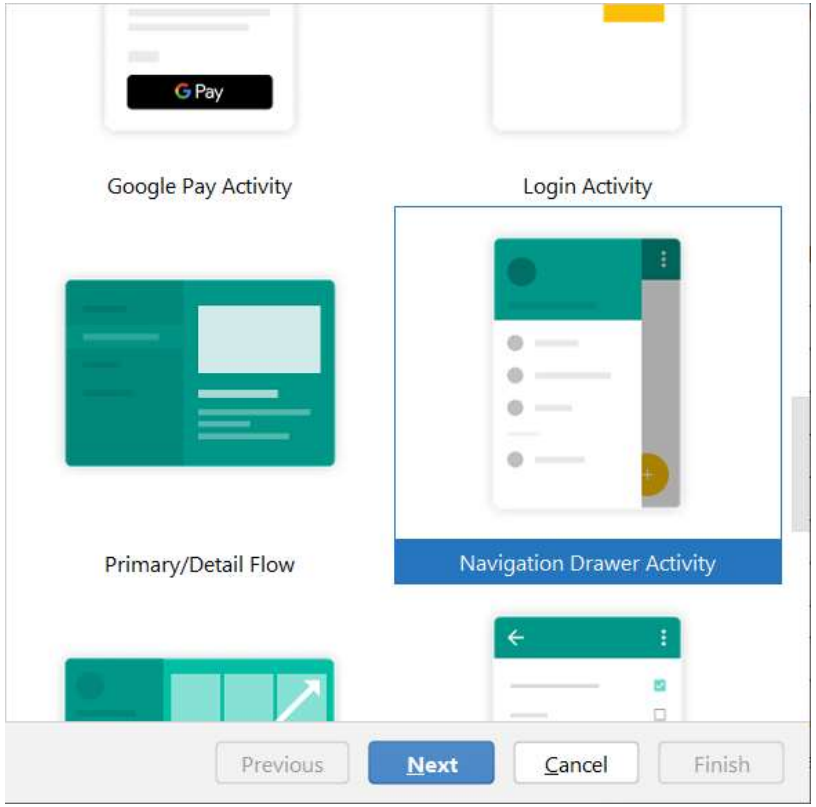
ACTIVITÉ n° 1

Création de plusieurs vues personnalisées

PARTIE 2

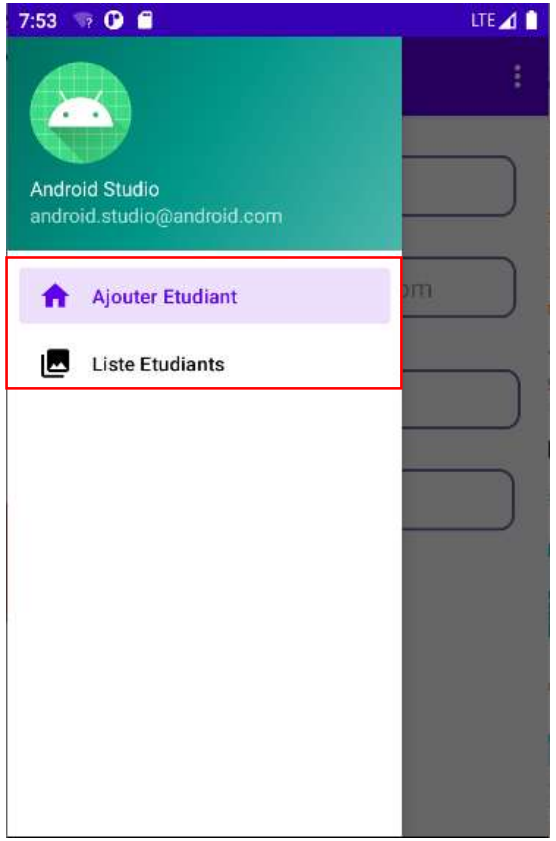
Etape 1 : Créer un projet sur Android Studio

On demande de créer un projet sur Android Studio avec le langage **Kotlin**, et sélectionner **NavigationDrawer**



Etape 2 : Menu de l'application

- Lancer l'application dans l'émulateur ou dans un smartphone ;
- Modifier le code afin de créer le menu ci-dessous :



Copyright - Tout droit réservé - OFPPT

Etape 2 : Menu de l'application (Suite)

Modifier le fichier **activity_main_drawer.xml** qui figure dans le dossier menu comme suit :

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    tools:showIn="navigation_view">

    <group android:checkableBehavior="single">
        <item
            android:id="@+id/nav_add"
            android:icon="@drawable/ic_baseline_home_24"
            android:title="Ajouter Etudiant" />
        <item
            android:id="@+id/nav_list"
            android:icon="@drawable/ic_menu_gallery"
            android:title="Liste Etudiants" />

    </group>
</menu>
```

Changer les icônes des items.

Copyright - Tout droit réservé - OFPPT

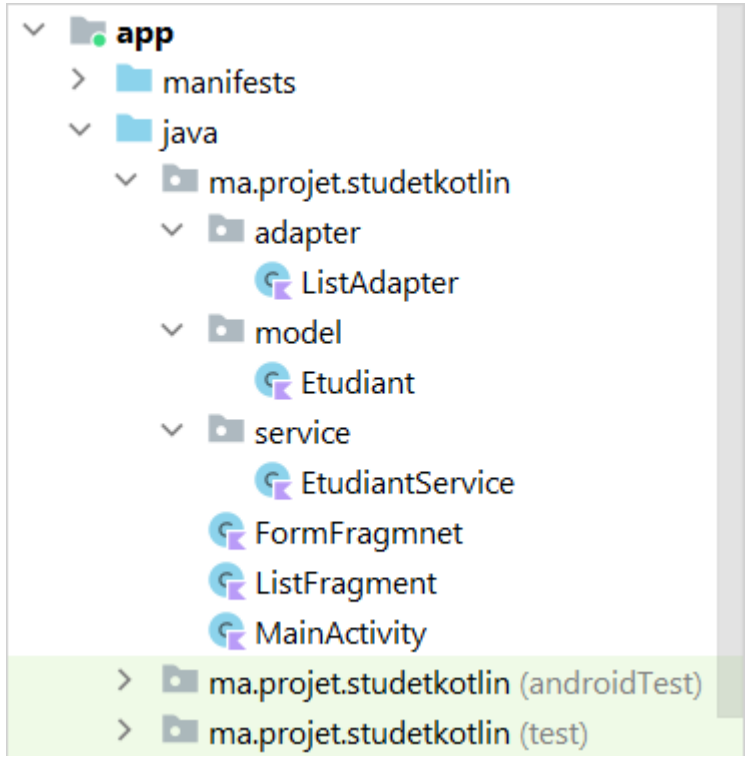
ACTIVITÉ n° 1

Création de plusieurs vues personnalisées

Etape 3 : Modifier la structure de projet

- Modifier la structure de projet ;
- Créer les sous packages :
    - adapter ;
    - model ;
    - service ;

Structure de projet



PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 1

Création de plusieurs vues personnalisées

PARTIE 2

Etape 4 : Créer la classe Etudiant

Créer la classe **Etudiant** dans le package « **model** » :

```
class Etudiant {
    var nom: String? = null
    var prenom: String? = null
    var ville: String? = null
    var dateNaissance: String? = null

    constructor(nom: String?, prenom: String?, ville: String?,
dateNaissance: String?) {
        this.nom = nom
        this.prenom = prenom
        this.ville = ville
        this.dateNaissance = dateNaissance
    }
}
```

Etape 5 : Créer la classe EtudiantService

Créer la classe **EtudiantService** dans le package « **service** » :

```
class EtudiantService private constructor() {
    var students: ArrayList<Etudiant>
    fun addStudent(e: Etudiant) {
        students.add(e)
    }
    companion object {
        var instance: EtudiantService? = null
            get() {
                if (field == null) {
                    synchronized(EtudiantService::class.java) {
                        if (field == null) {
                            field = EtudiantService()
                        }
                    }
                }
                return field
            }
            private set
    }
    init {
        students = ArrayList()
    }
}
```

Copyright - Tout droit réservé - OFPPT

Etape 6 : Créer le fragment FormFragment

Créer le fragment FormFragment permettant d'ajouter un nouveau étudiant :




```xml
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".FormFragmnet">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <TextView
            android:id="@+id/textViewville"
            android:layout_width="102dp"
            android:layout_height="26dp"
            android:layout_marginTop="264dp"
            android:text="Ville :"
            android:textSize="20sp"
            app:layout_constraintBottom_toBottomOf="@+id/send"
            app:layout_constraintEnd_toStartOf="@+id/ville"
            app:layout_constraintHorizontal_bias="0.366"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintVertical_bias="0.0" />
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT

Etape 6 : Créer le fragment FormFragment (Suite)

```
<EditText
    android:id="@+id/nom"
    android:layout_width="225dp"
    android:layout_height="44dp"
    android:layout_marginTop="28dp"
    android:background="@drawable/custuminput"
    android:ems="10"
    android:hint="  Entrez votre nom"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.854"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<EditText
    android:id="@+id/prenom"
    android:layout_width="225dp"
    android:layout_height="44dp"
    android:layout_marginTop="28dp"
    android:background="@drawable/custuminput"
    android:ems="10"
    android:hint=" Entrez votre prenom"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.854"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/nom" />
```

```
<TextView
    android:id="@+id/textViewnom"
    android:layout_width="117dp"
    android:layout_height="32dp"
    android:text="Nom :"
    android:textSize="20sp"
    app:drawableTint="#000000"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toStartOf="@+id/ville"
    app:layout_constraintHorizontal_bias="0.484"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.057" />

<EditText
    android:id="@+id/date"
    android:layout_width="184dp"
    android:layout_height="43dp"
    android:layout_marginTop="36dp"
    android:layout_marginEnd="20dp"
    android:background="@drawable/custuminput"
    android:ems="10"
    android:hint="  DD/MM/YY"
    android:inputType="date"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/prenom" />
```


Copyright - Tout droit réservé - OFPPT

Etape 6 : Créer le fragment FormFragment (Suite)

```
<TextView
    android:id="@+id/textViewprenom"
    android:layout_width="102dp"
    android:layout_height="26dp"
    android:layout_marginStart="16dp"
    android:layout_marginEnd="33dp"
    android:text="Prénom :"
    android:textSize="20sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toStartOf="@+id/prenom"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.168" />

<Spinner
    android:id="@+id/ville"
    android:layout_width="225dp"
    android:layout_height="44dp"
    android:layout_marginTop="28dp"
    android:layout_marginEnd="24dp"
    android:background="@drawable/custuminput"
    android:entries="@array/villes"
    android:spinnerMode="dropdown"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/date" />
```

```
<Button
    android:id="@+id/send"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="32dp"
    android:background="@drawable/custombutton"
    android:text="Enregistrer"
    app:backgroundTint="#7F669D"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/ville"
    app:layout_constraintVertical_bias="0.091" />
<TextView
        android:id="@+id/textView2"
        android:layout_width="143dp"
        android:layout_height="29dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="196dp"
        android:layout_marginEnd="20dp"
        android:text="Date Naissance :"
        android:textSize="19dp"
        app:layout_constraintBottom_toTopOf="@+id/textViewville"
        app:layout_constraintEnd_toStartOf="@+id/date"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0" />

    </androidx.constraintlayout.widget.ConstraintLayout>

</FrameLayout>
```


Copyright - Tout droit réservé - OFPPT

Etape 7 : Modifier la classe FormFragment

Modifier la classe FormFragment

```
class FormFragment : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
         val view : View = inflater.inflate(R.layout.fragment_form_fragmnet, container, false)
        val nom : EditText = view.findViewById(R.id.nom);
        val prenom : EditText = view.findViewById(R.id.prenom);
        val date : EditText = view.findViewById(R.id.date);
        val ville : Spinner = view.findViewById(R.id.ville);
        val send : Button = view.findViewById(R.id.send);
        send.setOnClickListener{
            val e : Etudiant = Etudiant(nom.text.toString(), prenom.text.toString(), date.text.toString(), ville.selectedItem.toString())
            val es : EtudiantService = EtudiantService.instance!!
            es.addStudent(e)
            Log.d("Test", "onCreateView: " + es.students[0].nom)
        }

        return view
    }

}
```

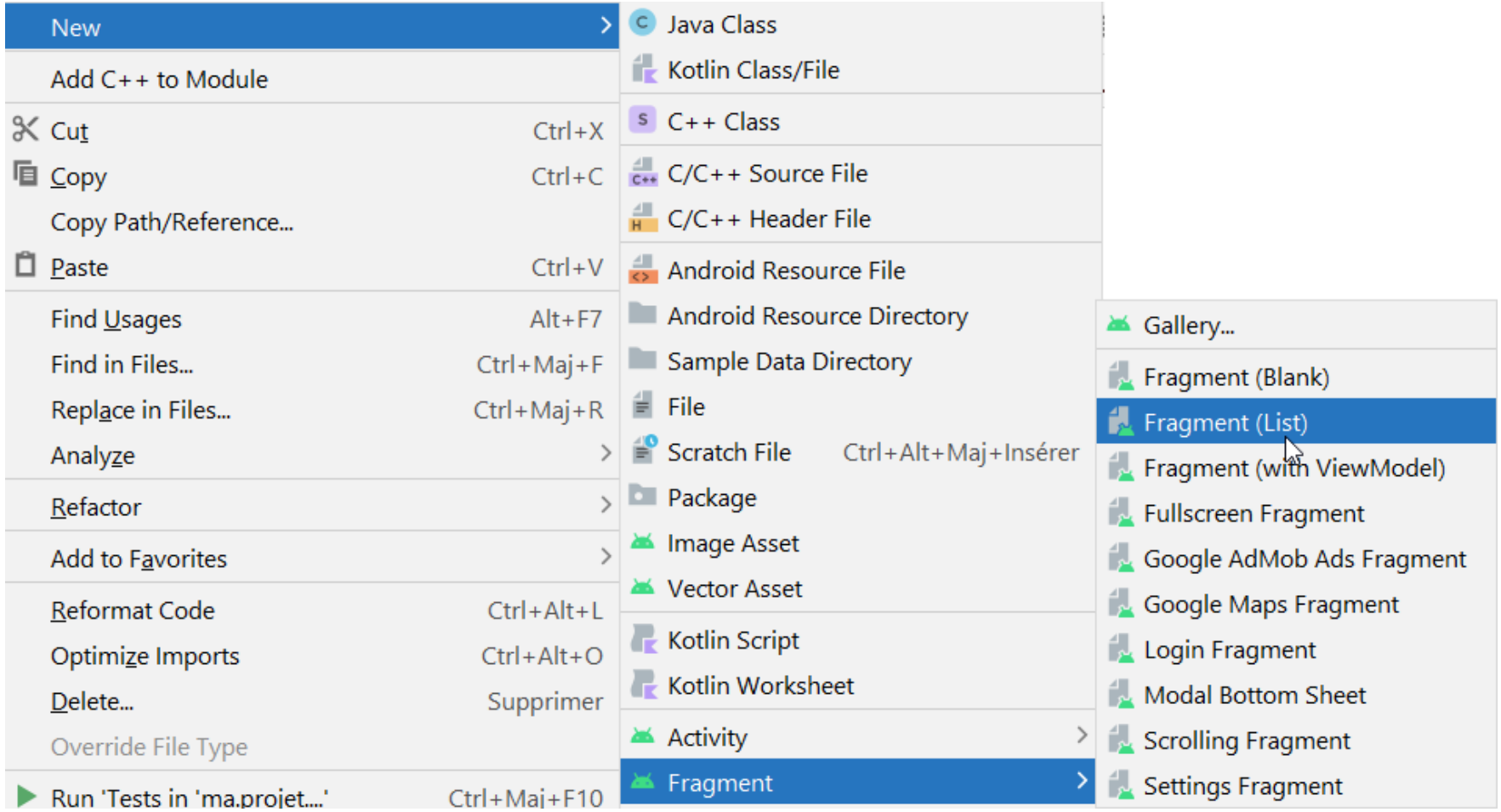

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 1

Création de plusieurs vues personnalisées

PARTIE 2

Etape 8 : Créer le fragment ListFragment



Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 1

Création de plusieurs vues personnalisées

Etape 9 : Modifier l'élément list_item.xml




```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="15dp"
    android:orientation="horizontal">
    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center" >
        <ImageView
            android:id="@+id/icon"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            app:srcCompat="@mipmap/ic_launcher" />

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:orientation="vertical">

            <TextView
                android:id="@+id/nomLs"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="title"
                android:textStyle="bold"
                android:layout_marginLeft="15dp"
                android:layout_marginStart="15dp"
                android:textAppearance="@style/Base.TextAppearance.AppCompat.Medium"
            />


            <TextView
                android:id="@+id/villeLs"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="description"
                android:layout_marginLeft="15dp"
                android:layout_marginStart="15dp"
                android:layout_marginTop="5dp"
                android:textSize="16sp"/>
        </LinearLayout>
    </LinearLayout>
</LinearLayout>
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 1

Création de plusieurs vues personnalisées

Etape 11 : Modifier fragment_list.xml

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ListFragment">


    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <ListView
            android:id="@+id/listView"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.5"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />
    </androidx.constraintlayout.widget.ConstraintLayout>
</FrameLayout>
```

Etape 12 : Modifier la classe ListAdapter

```
class ListAdapter(private val context: Activity, private val etds: ArrayList<Etudiant>)
    : ArrayAdapter<Etudiant>(context, R.layout.list_item, etds) {

    override fun getView(position: Int, view:
    View?, parent: ViewGroup
    ): View {
        val inflater = context.layoutInflater
        val rowView = inflater.inflate(R.layout.list_item, null, true)

        val titleText = rowView.findViewById(R.id.nomLs) as TextView
        val subtitleText = rowView.findViewById(R.id.villeLs) as TextView

        titleText.text = etds[position].nom + " " + etds[position].prenom
        subtitleText.text = etds[position].ville +" "+etds[position].dateNaissance

        return rowView
    }
}
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT

Etape 13 : Modifier la classe ListFragment

```
class ListFragment : Fragment() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

    }

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
        val view : View = inflater.inflate(R.layout.fragment_list, container, false)

        val listAdapter = ListAdapter(requireActivity(), EtudiantService.instance!!.students)
        val listView : ListView = view.findViewById(R.id.listView)
        listView.adapter = listAdapter

        return view
    }
}
```


Copyright - Tout droit réservé - OFPPT

Etape 15 : Modifier la navigation

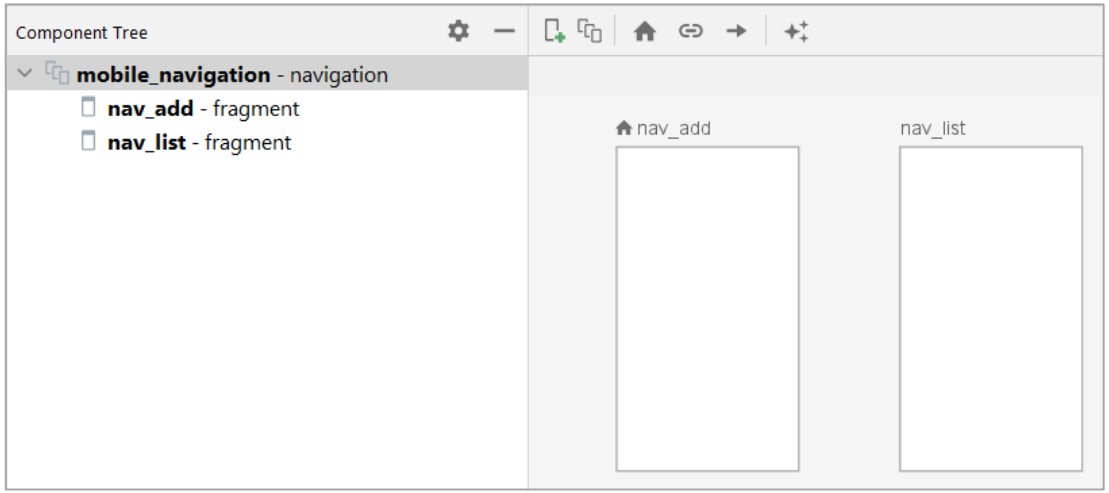
Dans le dossier navigation, modifier le fichier **mobile_navigation.xml** comme suit :

```
<?xml version="1.0" encoding="utf-8"?>
<navigation xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/mobile_navigation"
    app:startDestination="@+id/nav_add">

    <fragment
        android:id="@+id/nav_add"
        android:name="ma.projet.studetkotlin.FormFragment"
        android:label="Ajouter Etudiant"
        tools:layout="@layout/fragment_home" />

    <fragment
        android:id="@+id/nav_list"
        android:name="ma.projet.studetkotlin.ListFragment"
        android:label="Liste Etudiants"
        tools:layout="@layout/fragment_gallery" />


</navigation>
```



PARTIE 2

Copyright - Tout droit réservé - OFPPT

Etape 16 : Modifier la classe MainActivity

```
class MainActivity : AppCompatActivity() {

    private lateinit var appBarConfiguration: AppBarConfiguration
    private lateinit var binding: ActivityMainBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)

        setSupportActionBar(binding.appBarMain.toolbar)

        binding.appBarMain.fab.setOnClickListener { view ->
            Snackbar.make(view, "Replace with your own action", Snackbar.LENGTH_LONG)
                .setAction("Action", null).show()
        }
        val drawerLayout: DrawerLayout = binding.drawerLayout
        val navView: NavigationView = binding.navView
        val navController = findNavController(R.id.nav_host_fragment_content_main)
        // Passing each menu ID as a set of Ids because each
        // menu should be considered as top level destinations.
        appBarConfiguration = AppBarConfiguration(
            setOf(
                R.id.nav_add, R.id.nav_list
            ), drawerLayout
        )
        setupActionBarWithNavController(navController, appBarConfiguration)
        navView.setupWithNavController(navController)
    }

    override fun onCreateOptionsMenu(menu: Menu): Boolean {
        // Inflate the menu; this adds items to the action bar if it is present.
        menuInflater.inflate(R.menu.main, menu)
        return true
    }

    override fun onSupportNavigateUp(): Boolean {
        val navController = findNavController(R.id.nav_host_fragment_content_main)
        return navController.navigateUp(appBarConfiguration) ||
super.onSupportNavigateUp()
    }
}
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 1
Création de plusieurs vues personnalisées

Challenge

On demande de modifier l'exercice précèdent comme suit :

- Ajouter le champ image dans le formulaire d'ajout ;
- Sélectionner une image à partir de la galerie ou prendre une photo par la caméra ;
- Ajouter un champ pour sélectionner la date dans un calendrier ;
- Voir l'image gif à côté pour plus de détails.



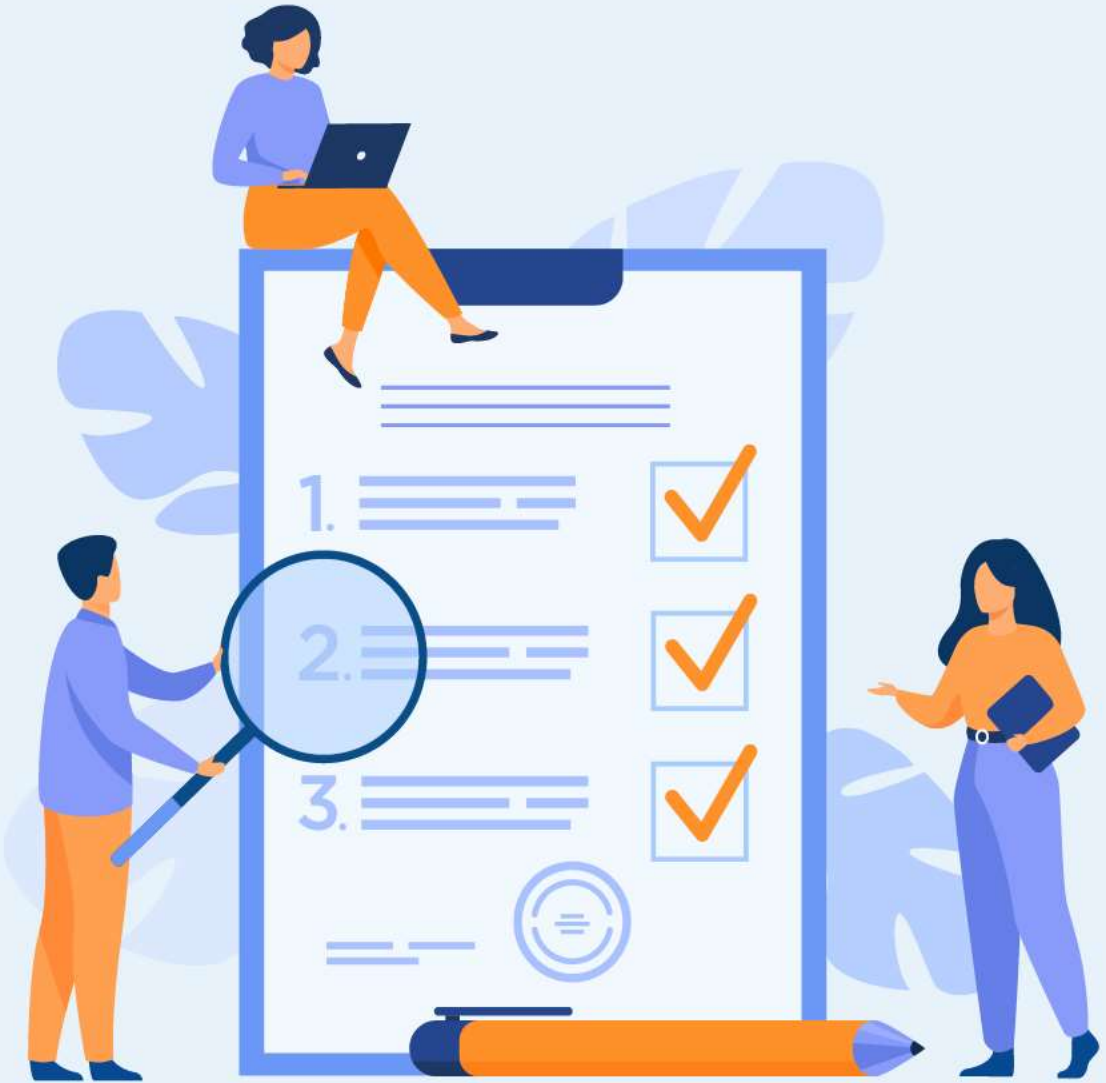
Indications

Déclarer les permissions suivantes :

```
<uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.CAMERA"/>
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT



ACTIVITÉ n° 2

Dessiner des vues sur Canvas

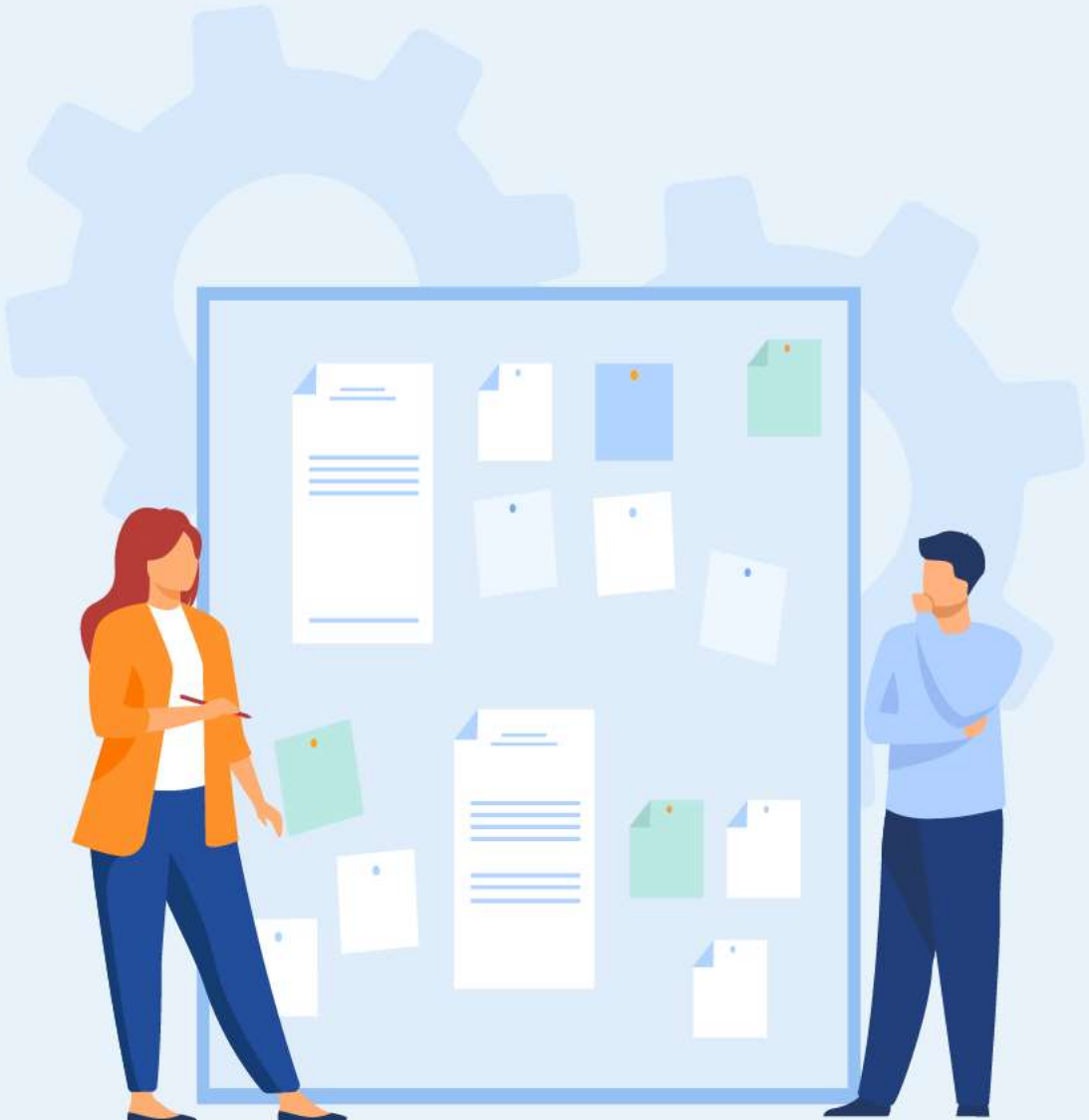
Compétences visées :

- Apprendre à utiliser les différents outils de dessin disponibles sur les Canvas, tels que les pinceaux, les rectangles, les cercles et les ligne ;
- Savoir comment dessiner des éléments animés sur les Canvas ;
- Maitriser les concepts de clip et de région de dessin pour limiter le dessin aux parties souhaitées de l'écran.

Recommandations clés :

- Révision générale du résumé théorique ;
- Suivre les instructions du TP et organiser le dossier de travail.

8 heures



CONSIGNES

1. Pour le formateur

- Rappeler les différents outils de dessin disponibles sur les Canvas ;
- Demander aux apprenants de suivre les étapes décrites dans l'activité.

2. Pour l'apprenant

- Suivre les étapes indiquées dans l'activité.

3. Conditions de réalisation :

- Support de résumé théorique accompagnant ;
- Un ordinateur sur lequel est installée la dernière version stable d'Android Studio.

4. Critères de réussite :

- Le stagiaire doit être capable de :
  - ➢ utiliser les différents outils de dessin disponibles sur les Canvas, tels que les pinceaux, les rectangles, les cercles et les ligne ;
  - ➢ dessiner des éléments animés sur les Canvas ;

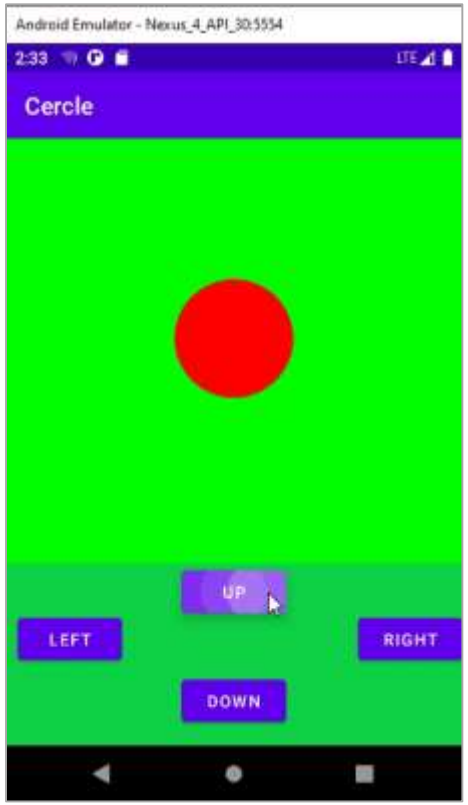
ACTIVITÉ n° 2

Dessiner des vues sur Canvas

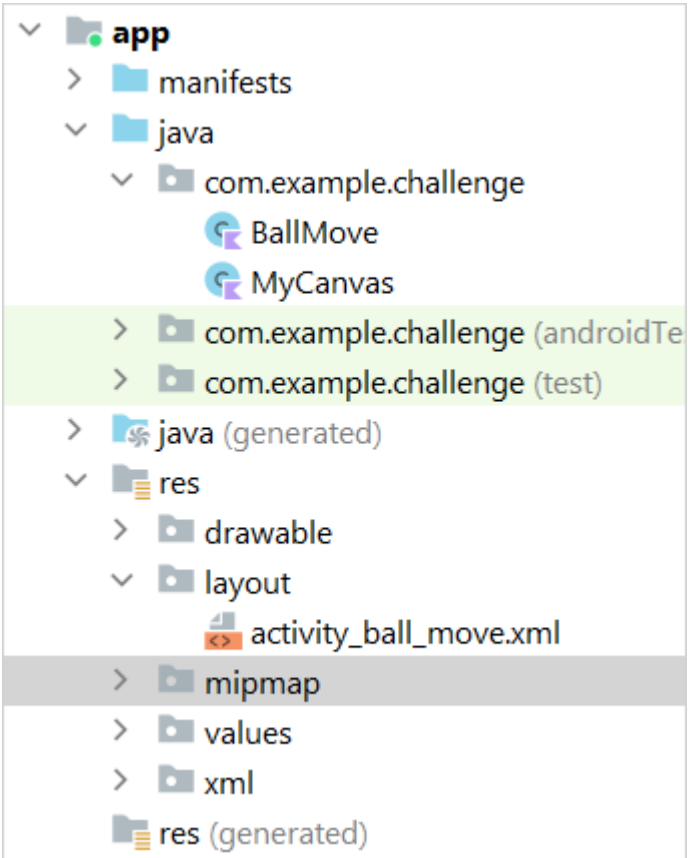
PARTIE 2

TP n° 1 : Enoncé

Dans cette activité, on demande de dessiner un cercle dans un "canvas view" et le déplacer dans toutes les directions (haut, bas, gauche, droite) en utilisant le langage kotlin. Comme illustré dans la figure suivante :



Structure de projet



Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 2

Dessiner des vues sur Canvas

PARTIE 2

Etape 1 : Créer la classe MyCanvas

1. Créer une classe nommée **MyCanvas** qui vas hérite de la classe view et qui contient deux constructeurs.

```
class MyCanvas : View {
    constructor(context: Context?) : super(context) {

    }
    constructor(context: Context?, attrs: AttributeSet?) : super(context, attrs)
{

    }
}
```

2. Déclarer un objet Paint a l' intérieure de la classe « MyCanvas », puis définir la méthode **init ()** qui va déterminer les attributs de l'élément a dessiner comme la couleur , la largeur. Puis, appeler cette méthode à l'intérieure des constructeurs comme suit ;

```
class MyCanvas : View {
    var paint: Paint? = null

    constructor(context: Context?) : super(context) {
        init()
    }
    constructor(context: Context?, attrs: AttributeSet?) : super(context, attrs)
{
        init()
    }
    fun init() {
        paint = Paint()
        paint!!.color = Color.RED
        paint!!.strokeWidth = 8f
        paint!!.style = Paint.Style.FILL
    }
}
```

3. Redéfinir la méthode **onDraw()** qui prend en paramètre un objet de type Canvas qui va servir à dessiner le cercle comme suit :

```
override fun onDraw(canvas: Canvas) {
    super.onDraw(canvas)
    canvas.drawColor(Color.GREEN)
    canvas.drawCircle((this.width / 2).toFloat(),
                (this.height / 2).toFloat(), 100f,paint!!)
}
```

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 2

Dessiner des vues sur Canvas

Etape 1 : Créer la classe MyCanvas (Suite)

4. Déclarer des variables booléens pour vérifier quelle option a été choisi (left, right, down, up) et deux variables de type float qui vont déterminer le mouvement du cercle selon le x et le y .

```
class MyCanvas : View {
    var paint: Paint? = null
    var check = false
    var left = false
    var right = false
    var up = false
    var down = false
    var x = 0.0
    var y = 0.0

    constructor(context: Context?) : super(context) {
        init()
    }

    constructor(context: Context?, attrs: AttributeSet?) : super(context, attrs) {
        init()
    }

    fun init() {
        paint = Paint()
        paint!!.color = Color.RED
        paint!!.strokeWidth = 8f
        paint!!.style = Paint.Style.FILL
    }

    override fun onDraw(canvas: Canvas) {
        super.onDraw(canvas)
        canvas.drawColor(Color.GREEN)
        if (check == false) {
            canvas.drawCircle((this.width / 2).toFloat(),(this.height / 2).toFloat(),
100f,paint!!)
            x = (this.width / 2).toFloat().toDouble()// current x and y
            y = (this.height/2).toFloat().toDouble()
        } else if (left) {
            x=x-20
            canvas.drawCircle(x.toFloat(), y.toFloat(), 100f, paint!!)
        } else if (right) {
            x=x+20
            canvas.drawCircle(x.toFloat(), y.toFloat(), 100f, paint!!)
        } else if (down) {
            y=y+20
            canvas.drawCircle(x.toFloat(), y.toFloat(), 100f, paint!!)
        } else if (up) {
            y=y-20
            canvas.drawCircle(x.toFloat(), y.toFloat(), 100f, paint!!)
        }
    }
```

Remarques

Cette classe permet de dessiner un cercle de couleur rouge sur un canvas de couleur verte. La position du cercle est initialement au centre du canvas, mais peut être déplacée vers la gauche, la droite, le haut ou le bas en modifiant les variables booléennes **left**, **right**, **up** ou **down**. La position du cercle est mise à jour à chaque appel de la méthode **onDraw** en utilisant les variables **x** et **y** qui représentent les coordonnées du centre du cercle.

PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 2

Dessiner des vues sur Canvas

Etape 2 : Modifier le layout activity_ball_move.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#0ed145"
    android:orientation="vertical">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
        android:weightSum="1">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_weight="0.3"
            android:orientation="vertical">
            <com.example.challenge.MyCanvas
                android:id="@+id/myview"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                tools:ignore="ExtraText" />
            <Button
                android:id="@+id/button"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:text="Button" />
        </LinearLayout>
```

```
        <androidx.constraintlayout.widget.ConstraintLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent">
            <Button
                android:id="@+id/up"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="up"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                tools:ignore="NotSibling"
                tools:layout_editor_absoluteY="2dp" />
            <Button
                android:id="@+id/down"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginTop="92dp"
                android:text="down"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent" />
            <Button
                android:id="@+id/left"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginStart="10dp"
                android:layout_marginTop="40dp"
                android:text="left"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent" />
```

PARTIE 2

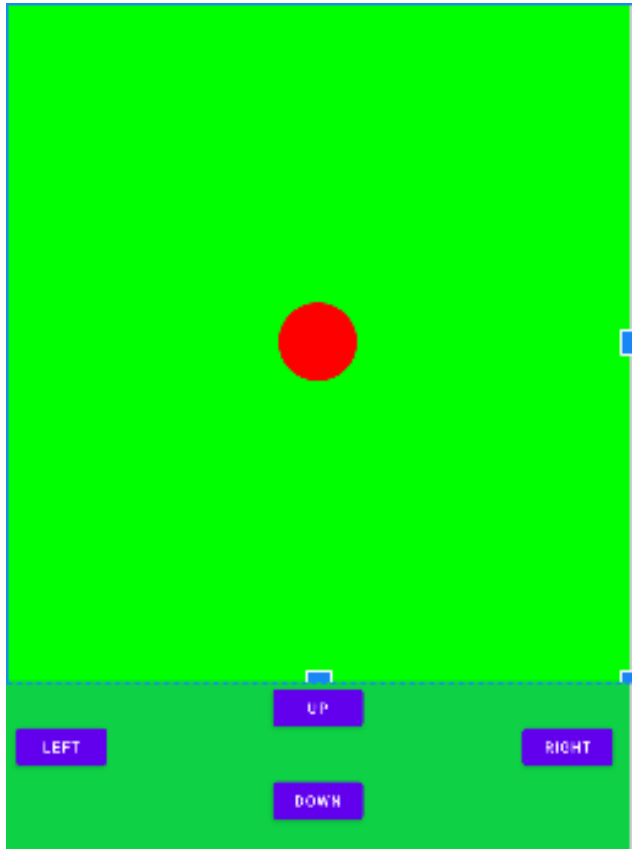
Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 2

Dessiner des vues sur Canvas

Etape 2 : Modifier le layout activity_ball_move.xml

```
    <Button
            android:id="@+id/right"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginStart="200dp"
            android:layout_marginTop="40dp"
            android:text="right"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintHorizontal_bias="0.927"
            app:layout_constraintStart_toEndOf="@+id/left"
            app:layout_constraintTop_toTopOf="parent" />
      </androidx.constraintlayout.widget.ConstraintLayout>
   </LinearLayout>
</LinearLayout>
```



PARTIE 2

Copyright - Tout droit réservé - OFPPT

Etape 3 : Modifier la classe BallMove.kt

```
class BallMove : AppCompatActivity() {
    @SuppressLint("MissingInflatedId")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_ball_move)
        var canvas = findViewById<View>(R.id.myview) as MyCanvas
        var left = findViewById<Button>(R.id.left)
        var right = findViewById<Button>(R.id.right)
        var up = findViewById<Button>(R.id.up)
        var down = findViewById<Button>(R.id.down)
        left.setOnClickListener(object : View.OnClickListener {
            override fun onClick(view: View?) {
                canvas!!.check = true
                canvas!!.left = true
                canvas!!.right = false
                canvas!!.up = false
                canvas!!.down = false
                canvas!!.invalidate()
            }
        })
        right.setOnClickListener(object : View.OnClickListener {
            override fun onClick(view: View?) {
                canvas!!.check = true
                canvas!!.left = false
                canvas!!.right = true
                canvas!!.up = false
                canvas!!.down = false
                canvas!!.invalidate()
            }
        })
        up.setOnClickListener(object : View.OnClickListener {
            override fun onClick(view: View?) {
                canvas!!.check = true
                canvas!!.left = false
                canvas!!.right = false
                canvas!!.up = true
                canvas!!.down = false
                canvas!!.invalidate()
            }
        })
        down.setOnClickListener(object : View.OnClickListener {
            override fun onClick(view: View?) {
                canvas!!.check = true
                canvas!!.left = false
                canvas!!.right = false
                canvas!!.up = false
                canvas!!.down = true
                canvas!!.invalidate()
            }
        })
    }
}
```

Remarques

Ce code représente une activité Android qui affiche un canvas et 4 boutons. Lorsque l'un des boutons est cliqué, le canvas est mis à jour en déplaçant le cercle vers la gauche, la droite, le haut ou le bas en fonction du bouton qui a été cliqué. Les variables **`left`**, **`right`**, **`up`** ou **`down`** de la classe **`MyCanvas`** sont mises à jour pour indiquer la direction dans laquelle le cercle doit être déplacé, et la méthode **`invalidate`** est appelée pour forcer la mise à jour du canvas.

PARTIE 2

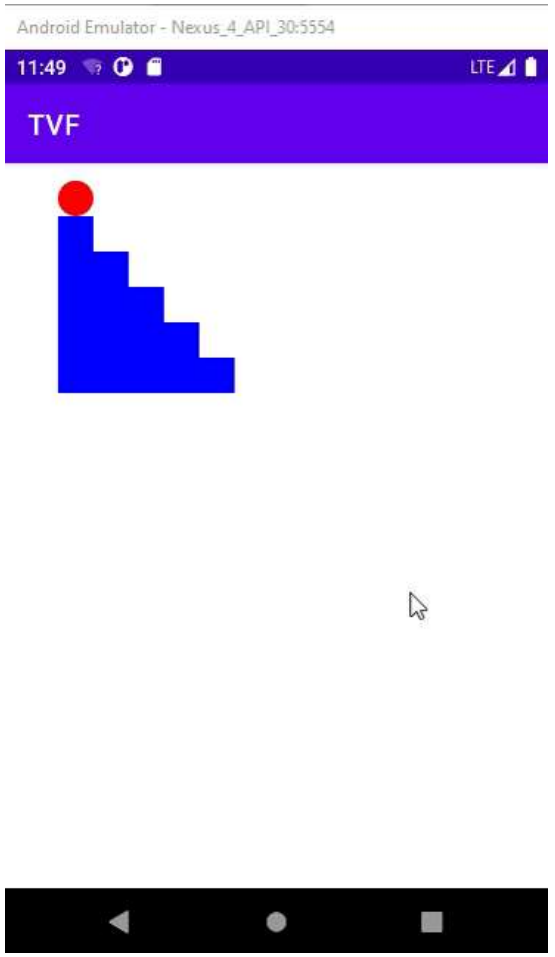
Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 2

Dessiner des vues sur Canvas

TP n° 2 : Enoncé

On souhaite développer une application mobile permettant de simuler le mouvement d'une boule rouge descendant les marches d'un escalier, comme illustré dans l'image ci-jointe.



PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 2

Dessiner des vues sur Canvas

Etape 1 : Créer un projet sous Android Studio

On demande de créer un projet sous Android Studio avec le langage **Kotlin**, et de choisir **Empty Activity** par défaut.

Enuiste, créer la classe **DrawingView** permettant de déssiner la boucle et l'escalier.

Il s'agit d'une classe de vue personnalisée qui affiche un cercle et un chemin à l'écran. Le cercle est dessiné à l'aide de la méthode drawCircle de la classe Canvas, qui prend comme arguments les coordonnées x et y du centre du cercle, et le rayon du cercle. Le chemin est dessiné à l'aide de la méthode drawPath de la classe Canvas, qui prend comme argument un objet Path qui définit la forme à dessiner.

Etape 2 : Déssiner la boule et l'escalier

```
class DrawingView internal constructor(context: Context?) : View(context) {
    val paint = Paint()
    val path = Path()
    var posX = 100
    var posY = 50

    override fun onDraw(canvas: Canvas) {
        super.onDraw(canvas)
        paint.style = Paint.Style.FILL
        paint.color = Color.RED
        canvas.drawCircle(posX.toFloat(), posY.toFloat(), 25f, paint)
        paint.isAntiAlias = true
        paint.color = Color.BLUE
        paint.style = Paint.Style.FILL
        path.moveTo(75f, 75f)
        path.lineTo(125f, 75f)
        path.lineTo(125f, 125f)
        path.lineTo(175f, 125f)
        path.lineTo(175f, 175f)
        path.lineTo(225f, 175f)
        path.lineTo(225f, 225f)
        path.lineTo(275f, 225f)
        path.lineTo(275f, 275f)
        path.lineTo(325f, 275f)
        path.lineTo(325f, 325f)
        path.lineTo(75f, 325f)
        path.close()
        canvas.drawPath(path, paint)
    }
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 2

Dessiner des vues sur Canvas

Etape 3 : Simuler le mouvement

La vue surcharge également la méthode onTouchEvent, qui est appelée lorsque l'utilisateur touche l'écran. Lorsque cette méthode est appelée, elle crée un nouveau Thread et le démarre. Le thread déplace le cercle vers la droite et vers le bas de 50 pixels toutes les 300 et 500 millisecondes, respectivement, pour un total de 5 fois. Une fois la boucle terminée, le cercle est réinitialisé à sa position d'origine (100, 50).

Enfin, la méthode invalidate est appelée pour redessiner la vue et mettre à jour la position du cercle à l'écran.

```
override fun onTouchEvent(event: MotionEvent): Boolean {
    val t: Thread = object : Thread() {
        override fun run() {
            for (i in 0..3) {
                posX += 50
                invalidate()
                try {
                    sleep(300)
                } catch (e: InterruptedException) {
                    e.printStackTrace()
                }
                posY += 50
                invalidate()
                try {
                    sleep(500)
                } catch (e: InterruptedException) {
                    e.printStackTrace()
                }
            }
        }
    }
    t.start()
    return super.onTouchEvent(event)
}
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 2

Dessiner des vues sur Canvas

Challenge

On demande de modifier le projet existant afin de rendre la boule à son état initial une fois en bas de l'escalier.

Indication :

Ce code permet d'initialiser la boule à son état initial :

```
posX = 100
posY = 50
invalidate()
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 2

Dessiner des vues sur Canvas

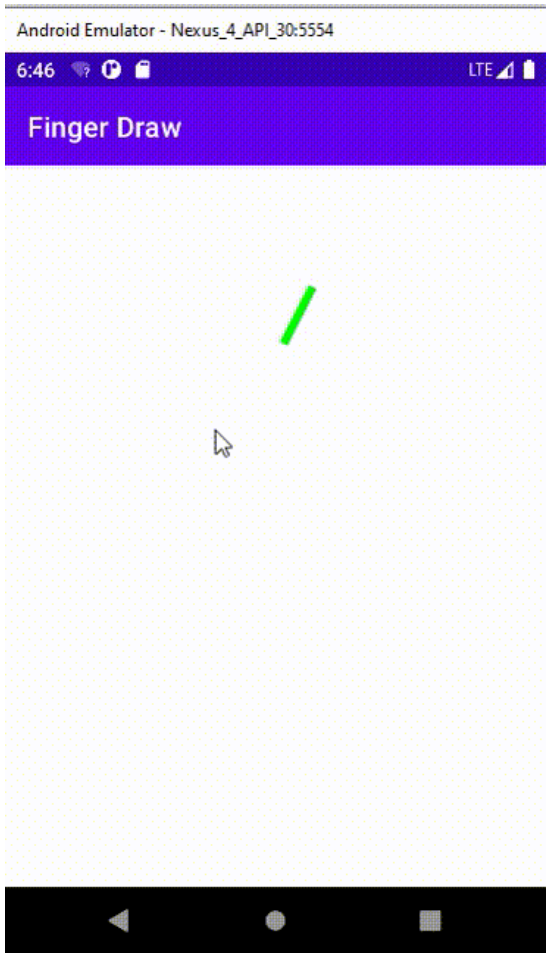
TP n° 3 : Enoncé

Créer une application de dessin simple : Commencer par créer une surface blanche et permettre à l'utilisateur de dessiner dessus avec son doigt. Ensuite, ajouter des fonctions de base comme la modification de la taille et de la couleur du pinceau.

On demande de créer la classe **DrawLineCanvas** permet de dessiner des lignes sur un canvas en utilisant les événements de toucher de l'utilisateur. Lorsqu'un utilisateur touche l'écran et déplace son doigt, une ligne est tracée sur le canvas en suivant le mouvement de l'utilisateur. Lorsque l'utilisateur relâche son doigt, la ligne est ajoutée au canvas de manière permanente.

La classe **DrawLineCanvas** étend la classe **View** de Android, ce qui lui permet d'afficher du contenu graphique sur l'écran. Elle définit plusieurs constructeurs pour créer une instance de la classe **DrawLineCanvas**, ainsi que des méthodes pour gérer les événements de toucher de l'utilisateur et pour dessiner sur le canvas.

La classe utilise également plusieurs objets de la classe **Paint** pour définir les couleurs et les styles de dessin, ainsi que la classe **Path** pour tracer les lignes sur le canvas. Elle utilise également la classe **Canvas** pour dessiner sur le canvas de manière permanente et la classe **Bitmap** pour stocker le canvas dessiné afin de pouvoir le réafficher ultérieurement.



PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 2

Dessiner des vues sur Canvas

Etape 1 : Créer la classe DrawLineCanvas

```
class DrawLineCanvas : View {
    private lateinit var c: Canvas
    private lateinit var pLine: Paint
    private lateinit var pBg: Paint
    private lateinit var touchPath: Path
    private lateinit var b: Bitmap

    constructor(context: Context?) : super(context)
    constructor(context: Context?, attrs: AttributeSet?) : super(context, attrs) {
        pBg = Paint()
        pBg!!.setColor(Color.WHITE)

        pLine = Paint()
        pLine!!.setColor(Color.GREEN)
        pLine!!.isAntiAlias = true
        pLine!!.style = Paint.Style.STROKE
        pLine!!.strokeWidth = 12F
        touchPath = Path()
    }

    constructor(context: Context?, attrs: AttributeSet?, defStyleAttr: Int) :
            super(context, attrs, defStyleAttr)

    override fun onSizeChanged(w: Int, h: Int, oldw: Int, oldh: Int) {
        super.onSizeChanged(w, h, oldw, oldh)
        b = Bitmap.createBitmap(w, h, Bitmap.Config.ARGB_8888)
        c = Canvas(b)
    }

    override fun onTouchEvent(event: MotionEvent?): Boolean {
        val touchX: Float = event!!.x
        val touchY: Float = event!!.y
        when (event!!.action) {
            MotionEvent.ACTION_DOWN -> {
                touchPath!!.moveTo(touchX, touchY)
            }
            MotionEvent.ACTION_MOVE -> {
                touchPath!!.lineTo(touchX, touchY)
            }
            MotionEvent.ACTION_UP -> {
                touchPath!!.lineTo(touchX, touchY)
                c!!.drawPath(touchPath, pLine)
                touchPath = Path()
            }
            else -> {
                return false
            }
        }
        invalidate()
        return true
    }
    override fun onDraw(canvas: Canvas?) {
        super.onDraw(canvas)
        canvas!!.drawBitmap(b, 0F, 0F, pBg)
        canvas!!.drawPath(touchPath, pLine)
    }
}
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 2
Dessiner des vues sur Canvas

Etape 1 : Créer la classe DrawLineCanvas

La méthode **onTouchEvent** est appelée chaque fois que l'utilisateur interagit avec l'écran, et elle enregistre la position du toucher dans un objet Path. Lorsque l'utilisateur retire son doigt de l'écran (**MotionEvent.ACTION_UP**), le chemin est dessiné sur un objet Canvas, qui est ensuite dessiné sur l'écran par la méthode **onDraw**. Les objets **Paint** sont utilisés pour personnaliser l'apparence des lignes et de l'arrière-plan de la vue.

Etape 2 : Création de layout

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <ma.projet.finger.DrawLineCanvas
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>

</androidx.constraintlayout.widget.ConstraintLayout>
```

Le code XML qui représente la mise en page de l'interface utilisateur de l'application Android. Il utilise la vue **DrawLineCanvas** décrite précédemment pour afficher une zone de dessin sur l'écran. La vue **DrawLineCanvas** occupe toute la largeur et la hauteur de l'écran grâce aux attributs **layout_width** et **layout_height** qui ont la valeur **match_parent**.
La vue **DrawLineCanvas** est contenue dans un layout de type **ConstraintLayout**, qui est un type de layout flexible et performant qui permet de définir des contraintes entre les vues pour créer des mises en page complexes. Le layout **ConstraintLayout** est défini par l'élément **androidx.constraintlayout.widget.ConstraintLayout** et a les attributs **layout_width** et **layout_height** qui indiquent qu'il doit occuper toute la largeur et la hauteur de l'écran.

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 2

Dessiner des vues sur Canvas

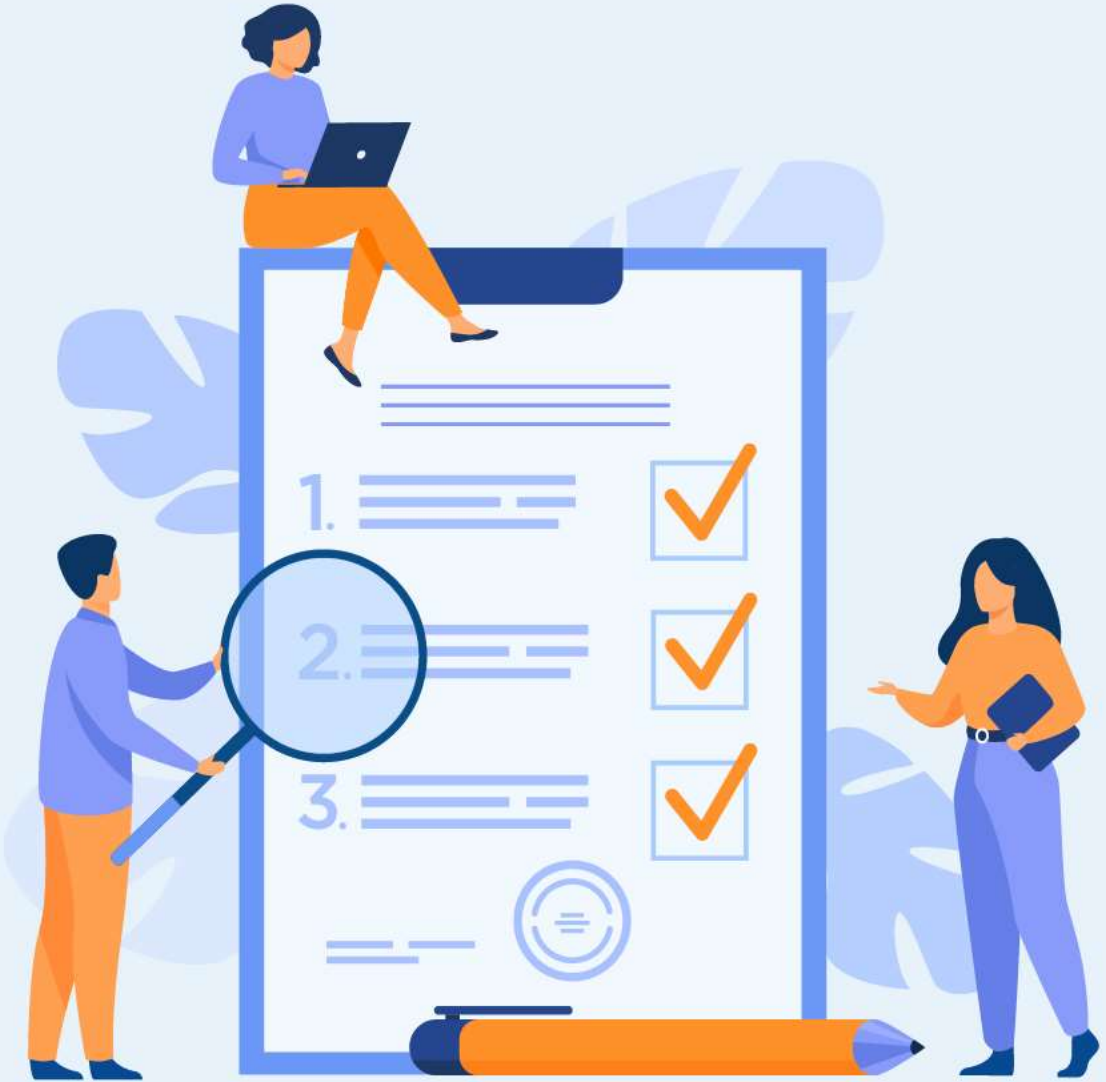
Challenge

On demande d'ajouter les fonctionnalités suivantes :

- Ajouter un bouton pour afficher un popup pour configurer la taille et de la couleur du pinceau ;
- Ajouter un bouton pour effacer le contenu de la surface blanche ;

PARTIE 2

Copyright - Tout droit réservé - OFPPT



ACTIVITÉ n° 3

Création de plusieurs animations entre plusieurs activités

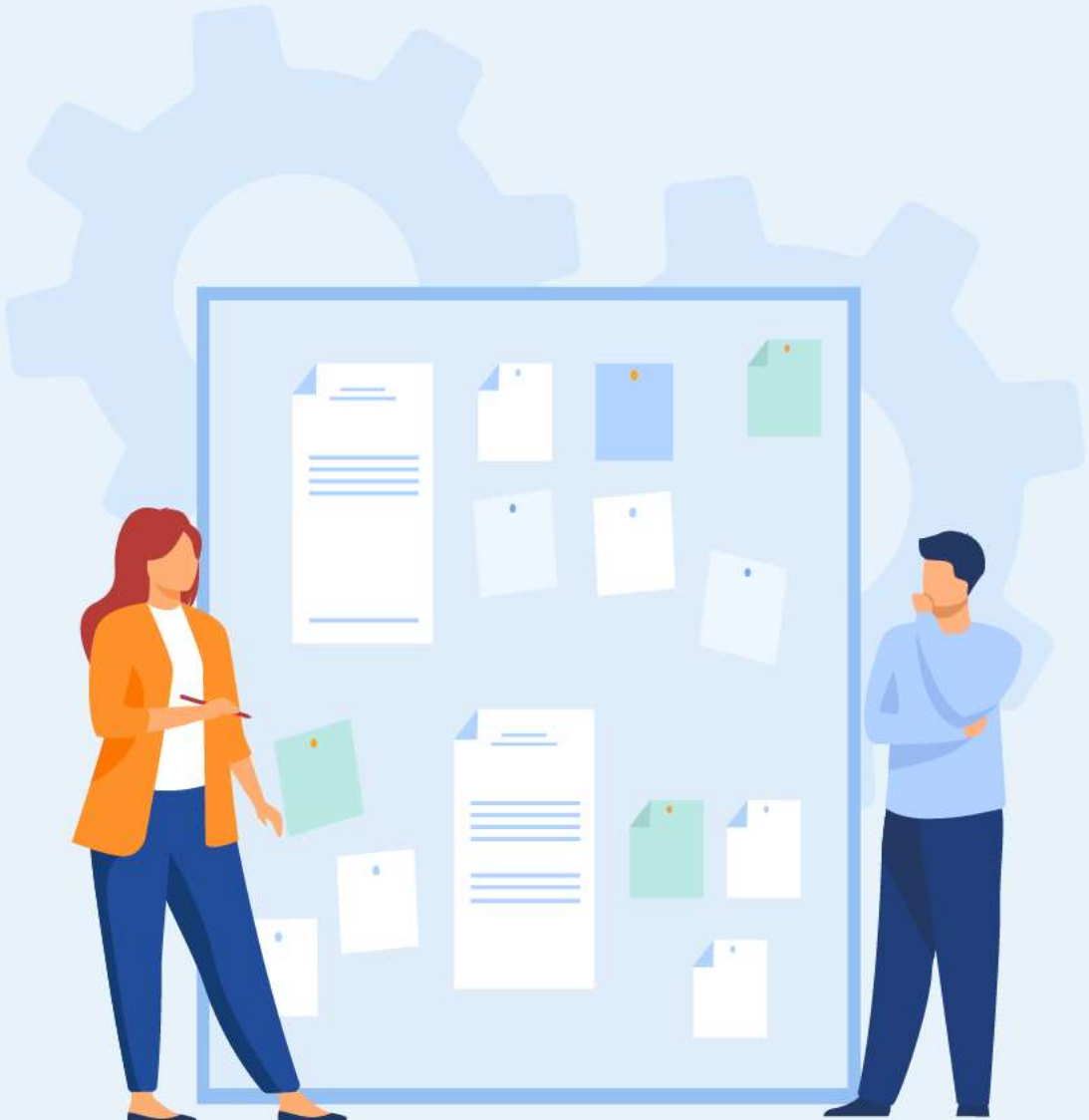
Compétences visées :

- Appliquer des transitions animées entre les activités ;

Recommandations clés :

- Révision générale du résumé théorique ;
- Suivre les instructions du TP et organiser le dossier de travail.

5 heures



CONSIGNES

1. Pour le formateur

- Rappeler les étapes à suivre pour appliquer des transitions animées entres les activités ;
- Demander aux apprenants de suivre les étapes décrites dans l'activité.

2. Pour l'apprenant

- Suivre les étapes indiquées dans l'activité.

3. Conditions de réalisation :

- Support de résumé théorique accompagnant ;
- Un ordinateur sur lequel est installée la dernière version stable d'Android Studio.

4. Critères de réussite :

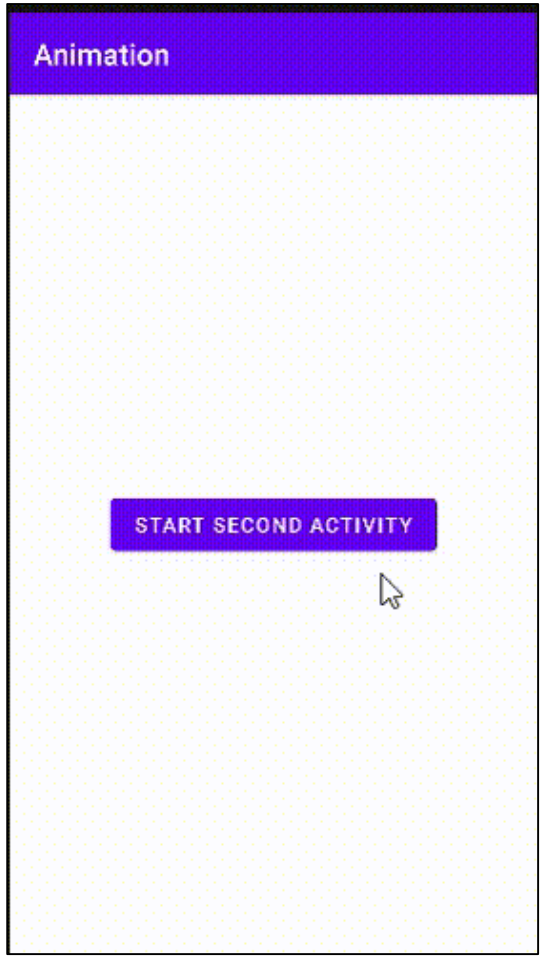
- Le stagiaire doit être capable de :
  - ➢ appliquer des transitions animées entre les activités ;

ACTIVITÉ n° 3

Création de plusieurs animations entre plusieurs activités

TP n° 1 : Enoncé

Dans cette activité, on demande de créer une transition animée entre deux activités dans une application Android. Lorsque l'utilisateur clique sur le bouton défini dans la première activité (MainActivity), il est redirigé vers la seconde activité (SecondActivity) avec une animation de "slide up". Lorsque l'utilisateur clique sur le bouton défini dans la seconde activité, il est ramené à la première activité avec une animation de "slide down".



PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 3

Création de plusieurs animations entre plusieurs activités

Etape 1

Définir les ressources d'animation à utiliser pour l'animation de glissement. Pour cela, créer deux fichiers de ressources d'animation XML dans le répertoire **res/anim** de projet. Le premier fichier définira l'animation de glissement vers le haut (par exemple, **slide_up.xml**) et le second définira l'animation de glissement vers le bas (par exemple, **slide_down.xml**). Voici un exemple de code pour l'animation de glissement vers le haut :

```
<!-- slide_up.xml -->
<set xmlns:android="http://schemas.android.com/apk/res/android">
   <translate
      android:fromYDelta="100%p"
      android:toYDelta="0"
      android:duration="@android:integer/config_shortAnimTime" />
</set>
```

Et voici un exemple de code pour l'animation de glissement vers le bas :

```
<!-- slide_down.xml -->
<set xmlns:android="http://schemas.android.com/apk/res/android">
   <translate
      android:fromYDelta="0"
      android:toYDelta="100%p"
      android:duration="@android:integer/config_shortAnimTime" />
</set>
```

Etape 2

Lancer une nouvelle activité avec une animation de glissement, utiliser la méthode **startActivity** de l'objet **Context** pour lancer l'activité, puis appeler la méthode **overridePendingTransition** sur l'objet **Activity** courant pour définir l'animation à utiliser. Voici un exemple de code qui lance une nouvelle activité avec une animation de glissement vers le haut :

```
// Lancer l'activité avec l'animation de glissement vers le haut
val intent = Intent(this, SecondActivity::class.java)
startActivity(intent)
overridePendingTransition(R.anim.slide_up, R.anim.slide_up)
```

Et voici un exemple de code qui lance une nouvelle activité avec une animation de glissement vers le bas :

```
// Lancer l'activité avec l'animation de glissement vers le bas
val intent = Intent(this, SecondActivity::class.java)
startActivity(intent)
overridePendingTransition(R.anim.slide_down, R.anim.slide_down)
```

La méthode **overridePendingTransition** prend en paramètre deux entiers qui représentent les ressources d'animation à utiliser pour l'entrée et la sortie de l'activité. Dans les exemples ci-dessus, les mêmes ressources d'animation sont utilisées pour l'entrée et la sortie de l'activité, mais il est possible également d'utiliser des ressources d'animation différentes.

Il est important de noter que cette méthode ne fonctionnera que si l'animation a été définie avant l'appel à **startActivity**. Si **startActivity est appelée** avant d'appeler **overridePendingTransition**, l'animation ne sera pas prise en compte et l'activité sera simplement lancée sans animation.

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 3

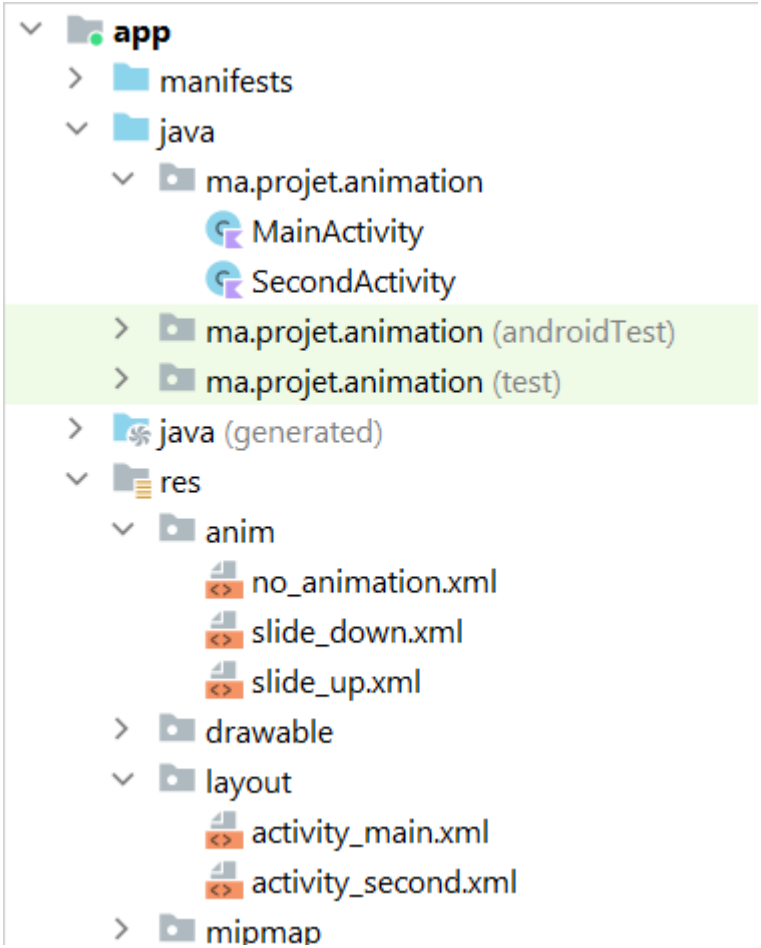
Création de plusieurs animations entre plusieurs activités

PARTIE 2

Solution

À base des étapes précédentes, une simple implémentation de l'animation est fournie dans les diapositives suivantes :

Structure de projet



Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 3

Création de plusieurs animations entre plusieurs activités

PARTIE 2

Animations

slide_up.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <translate
        android:duration="1000"
        android:fromYDelta="-100%"
        android:toYDelta="0" />
</set>
```

slide_down.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <translate
        android:duration="1000"
        android:fromYDelta="0"
        android:toYDelta="-100%" />
</set>
```

no_animation.xml

```
<?xml version="1.0" encoding="utf-8"?>
<translate xmlns:android="http://schemas.android.com/apk/res/android"
    android:fromYDelta="0"
    android:toYDelta="0"/>
```

Activités

```
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        var button=findViewById<Button>(R.id.bn_up)
        button.setOnClickListener{
            val intent = Intent(this, SecondActivity::class.java)
            startActivity(intent)
            overridePendingTransition(R.anim.slide_up,
R.anim.no_animation);
        }
    }
}

class SecondActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_second)
        var button=findViewById<Button>(R.id.bn_down)
        button.setOnClickListener{
            finish();
            overridePendingTransition(R.anim.no_animation,
R.anim.slide_down);
        }
    }
}
```

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 3

Création de plusieurs animations entre plusieurs activités

PARTIE 2

layouts

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <Button
        android:id="@+id/bn_up"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Start Second Activity" />
</LinearLayout>
```

activity_second.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical"
    tools:context=".SecondActivity">
    <Button
        android:id="@+id/bn_down"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Start First Activity" />
</LinearLayout>
```

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 3

Création de plusieurs animations entre plusieurs activités

TP n° 2 : Enoncé

Développer une application mobile Android en utilisant le langage Kotlin. L'application comprendra trois activités reliées entre elles par des transitions (Slide Right, Slide Left). Lorsqu'un utilisateur cliquera sur un bouton, il sera redirigé vers l'une des activités suivantes en utilisant l'une de ces transitions. La figure suivante illustre le fonctionnement de l'application.



PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 3

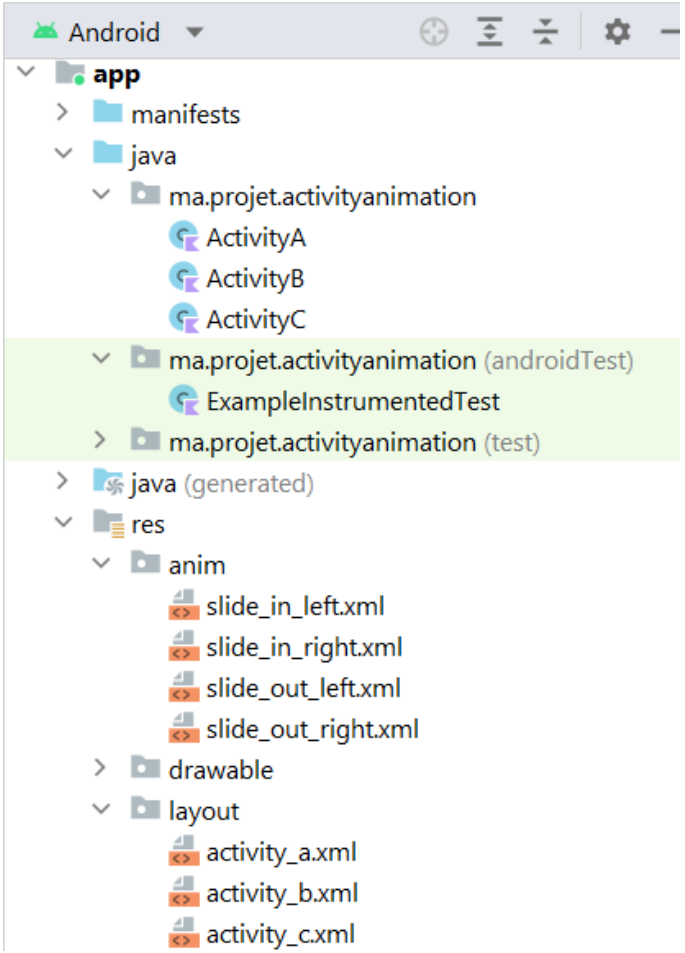
Création de plusieurs animations entre plusieurs activités

PARTIE 2

Etapes

1. Créer un nouveau projet Android en utilisant le modèle d'application vide ;
2. Créer les trois activités en utilisant le langage Kotlin. Chaque activité devra avoir sa propre vue (layout) avec des éléments de l'interface utilisateur tels que des boutons et des textes ;
3. Définir les transitions entre les activités en utilisant la méthode **overridePendingTransition**. Ajouter cette méthode à la fin de chaque méthode **startActivity** ou **finish** pour définir les transitions. Utiliser les constantes **R.anim.slide_in_right** et **R.anim.slide_out_left** pour définir la transition Slide Right, et utilisez les constantes **R.anim.slide_in_left** et **R.anim.slide_out_right** pour définir la transition Slide Left ;
4. Ajouter les boutons à chaque activité pour permettre à l'utilisateur de naviguer entre les activités. Utilisez les méthodes **startActivity** et **finish** pour lancer une nouvelle activité et terminer l'activité en cours ;
5. Tester l'application en utilisant l'émulateur ou un appareil Android réel pour vérifier que les transitions fonctionnent correctement.

Structure de projet



Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 3

Création de plusieurs animations entre plusieurs activités

PARTIE 2

activity_a.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#00BCD4"
    android:gravity="center"
    android:orientation="vertical"
    tools:context=".ActivityA">
    <Button
        android:layout_width="360dp"
        android:layout_height="wrap_content"
        android:onClick="intentB"
        android:text="Activity B"
        android:textColor="@color/black"
        app:backgroundTint="#FFFFFF" />
    <Button
        android:layout_width="360dp"
        android:layout_height="wrap_content"
        android:onClick="intentC"
        android:text="Activity C"
        android:textColor="@color/black"
        app:backgroundTint="#FFFFFF" />
</LinearLayout>
```

activity_b.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#7BC57E"
    android:gravity="center"
    android:orientation="vertical"
    tools:context=".ActivityB">
    <Button
        android:layout_width="360dp"
        android:layout_height="wrap_content"
        android:onClick="intentA"
        android:text="Activity A"
        android:textColor="@color/black"
        app:backgroundTint="#D3D3D3" />
    <Button
        android:layout_width="360dp"
        android:layout_height="wrap_content"
        android:onClick="intentC"
        android:text="Activity C"
        android:textColor="@color/black"
        app:backgroundTint="#D3D3D3" />
</LinearLayout>
```

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 3

Création de plusieurs animations entre plusieurs activités

activity_c.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#FF9800"
    android:gravity="center"
    android:orientation="vertical"
    tools:context=".ActivityC">
    <Button
        android:layout_width="360dp"
        android:layout_height="wrap_content"
        android:onClick="intentA"
        android:text="Activity A"
        android:textColor="@color/black"
        app:backgroundTint="#D3D3D3" />
    <Button
        android:layout_width="360dp"
        android:layout_height="wrap_content"
        android:onClick="intentB"
        android:text="Activity B"
        android:textColor="@color/black"
        app:backgroundTint="#D3D3D3" />
</LinearLayout>
```

Animations

slide_in_left

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <translate android:fromXDelta="-100%p" android:toXDelta="0"
        android:duration="@android:integer/config_mediumAnimTime"/>
</set>
```

slide_in_right

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <translate android:fromXDelta="100%p" android:toXDelta="0"
        android:duration="@android:integer/config_mediumAnimTime"/>
</set>
```

slide_out_left

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <translate android:fromXDelta="0" android:toXDelta="-100%p"
        android:duration="@android:integer/config_mediumAnimTime"/>
</set>
```

slide_out_right

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <translate android:fromXDelta="0" android:toXDelta="100%p"
        android:duration="@android:integer/config_mediumAnimTime"/>
</set>
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 3

Création de plusieurs animations entre plusieurs activités

PARTIE 2

ActivityA.java

```
class ActivityA : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_a)
        val actionBar = supportActionBar
        actionBar!!.title ="Activity A"
    }

    fun intentB(view: View?) {
        startActivity(Intent(this, ActivityB::class.java))
        overridePendingTransition(R.anim.slide_in_right, R.anim.slide_out_left);
    }

    fun intentC(view: View?) {
        startActivity(Intent(this, ActivityC::class.java))
        overridePendingTransition(R.anim.slide_in_left, R.anim.slide_out_right);
    }

}
```

ActivityB.java

```
class ActivityB : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_b)
        val actionBar = supportActionBar
        actionBar!!.title ="Activity B"
    }

    fun intentA(view: View?) {
        startActivity(Intent(this, ActivityA::class.java))
        overridePendingTransition(R.anim.slide_in_left, R.anim.slide_out_right);
    }

    fun intentC(view: View?) {
        startActivity(Intent(this, ActivityC::class.java))
        overridePendingTransition(R.anim.slide_out_left, R.anim.slide_in_right);
    }

}
```

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 3

Création de plusieurs animations entre plusieurs activités

ActivityC.java

```
class ActivityC : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_c)
        val actionBar = supportActionBar
        actionBar!!.title ="Activity C"
    }

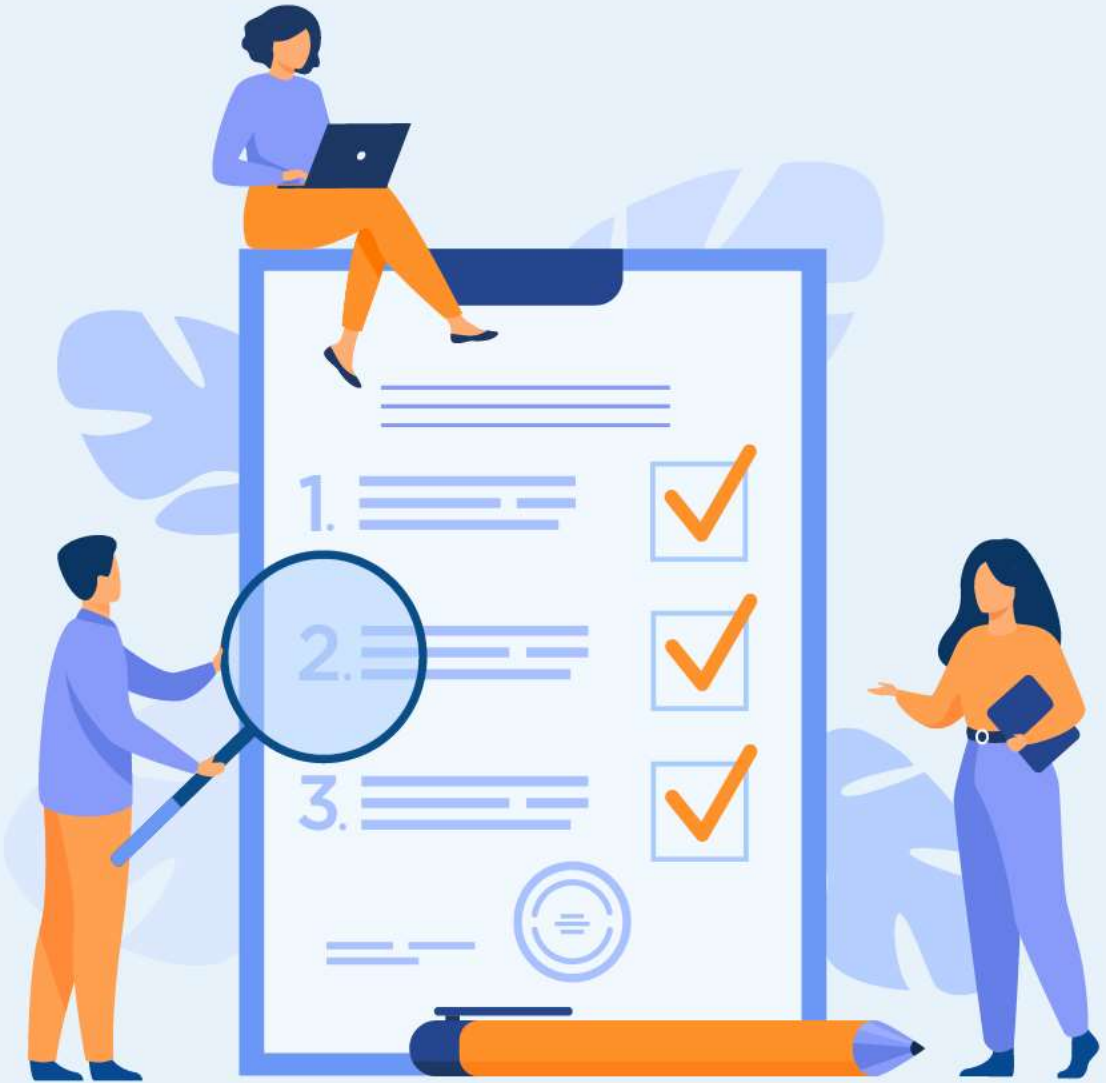
    fun intentA(view: View?) {
        startActivity(Intent(this, ActivityA::class.java))
        overridePendingTransition(R.anim.slide_out_left, R.anim.slide_in_right);
    }

    fun intentB(view: View?) {
        startActivity(Intent(this, ActivityB::class.java))
        overridePendingTransition(R.anim.slide_in_left, R.anim.slide_out_right);
    }

}
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT



ACTIVITÉ n° 4

Création de plusieurs animations (Zoom In/Out Animation, Rotate Animation…)

Compétences visées :

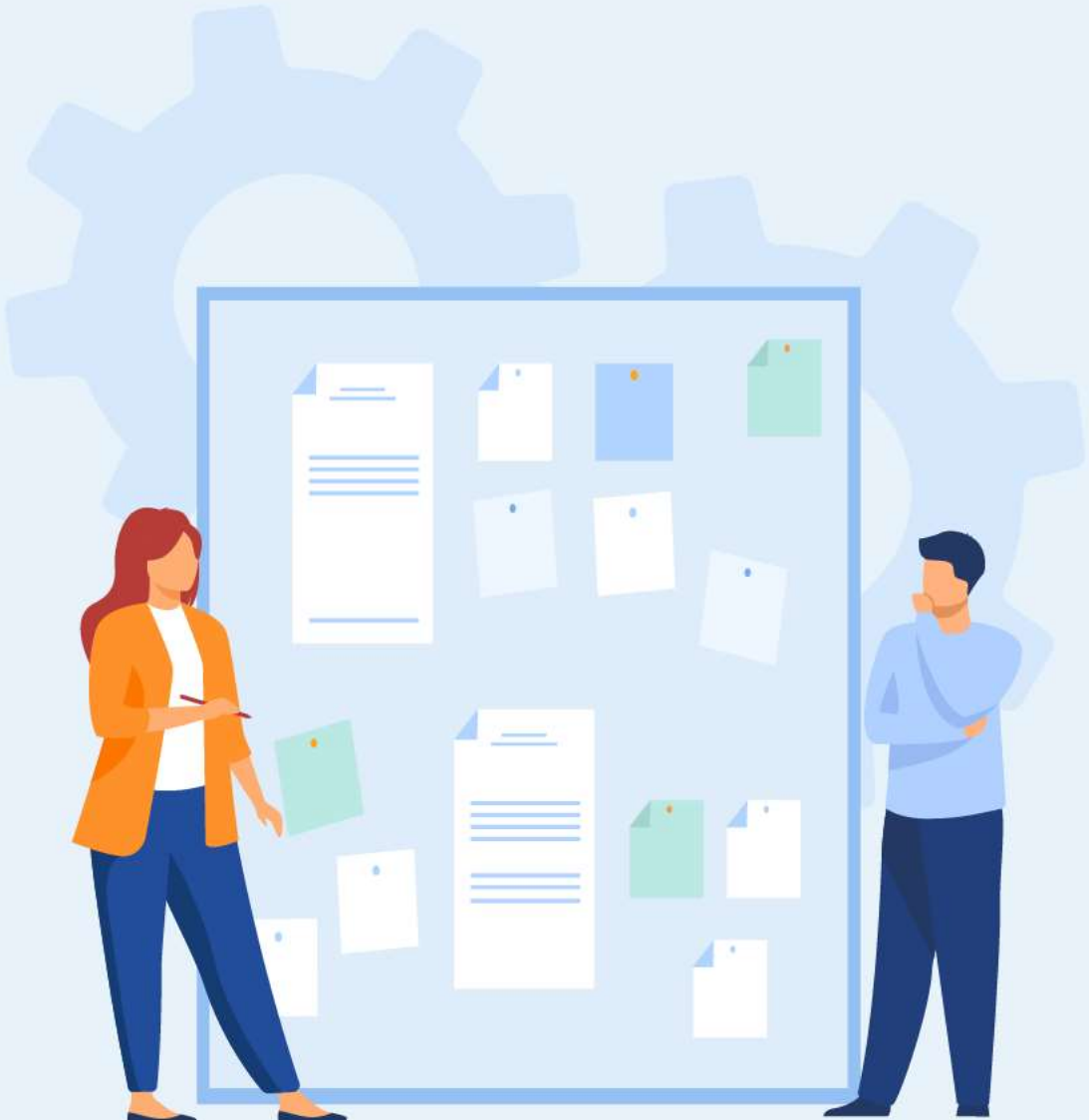
- Création de plusieurs animations (Zoom In/Out Animation, Rotate Animation…).

Recommandations clés :

- Révision générale du résumé théorique ;
- Suivre les instructions du TP et organiser le dossier de travail.

5 heures

CONSIGNES



1. Pour le formateur

- Rappeler les différents types d'animations ;
- Rappeler les étapes à suivre pour appliquer une animation sur un objet ;
- Demander aux apprenants de suivre les étapes décrites dans l'activité.

2. Pour l'apprenant

- Suivre les étapes indiquées dans l'activité.

3. Conditions de réalisation :

- Support de résumé théorique accompagnant ;
- Un ordinateur sur lequel est installée la dernière version stable d'Android Studio.

4. Critères de réussite :

- Le stagiaire doit être capable de :
  - créer plusieurs animations (Zoom In/Out Animation, Rotate Animation…);
  - appliquer les animations sur des objet .

ACTIVITÉ n° 4

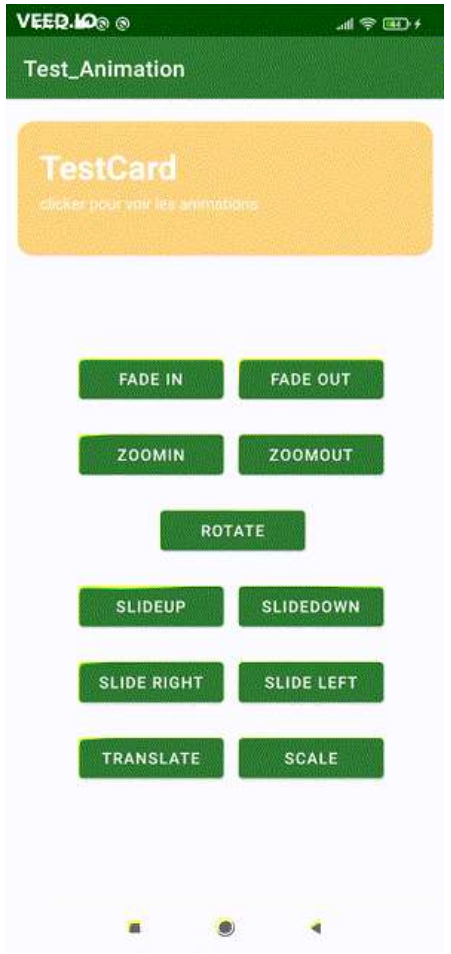
Création de plusieurs animations (Zoom In/Out Animation, Rotate Animation…)

TP n° 1 : Enoncé

Dans cette activité, on demande de créer une application Android avec le langage Kotlin. Ensuite, appliquer une série d'animations sur un objet Card. Comme le montre la figure suivante.

Les animations demandées sont :

- Fade In (code XML) ;
- Fade Out (code XML) ;
- Zoom In (code XML) ;
- Zoom Out (code XML) ;
- Slide UP (code XML) ;
- Slide Down (code XML) ;
- Slide Right (code XML) ;
- Slide Left (code XML) ;
- Rotate (code kotlin) ;
- Translate (code kotlin) ;
- Scale (code kotlin).



PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 4

Création de plusieurs animations (Zoom In/Out Animation, Rotate Animation…)

Etape 1 : Créer les animations dans le dossier anim

fade_in

```
<?xml version="1.0" encoding="utf-8"?>
<set
xmlns:android="http://schemas.android.com/apk/res/android"
   android:interpolator="@android:anim/linear_interpolator">
   <alpha
      android:duration="2000"
      android:fromAlpha="0.1"
      android:toAlpha="1.0">
   </alpha>
</set>
```

fade_out

```
<?xml version="1.0" encoding="utf-8"?>
<set
xmlns:android="http://schemas.android.com/apk/res/android"
   android:interpolator="@android:anim/linear_interpolator">
   <alpha
      android:duration="2000"
      android:fromAlpha="1.0"
      android:toAlpha="0.1" >
   </alpha>
</set>
```

slide_down

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
   android:interpolator="@android:anim/linear_interpolator">
   <scale
      android:duration="500"
      android:fromXScale="1.0"
      android:fromYScale="0.0"
      android:toXScale="1.0"
      android:toYScale="1.0" />
</set>
```

slide_up

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
   android:interpolator="@android:anim/linear_interpolator">
   <scale
      android:duration="500"
      android:fromXScale="1.0"
      android:fromYScale="1.0"
      android:toXScale="1.0"
      android:toYScale="0.0" />
</set>
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 4

Création de plusieurs animations (Zoom In/Out Animation, Rotate Animation…)

Etape 1 : Créer les animations dans le dossier anim

slide_left

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    android:interpolator="@android:anim/linear_interpolator">
    <scale
        android:duration="500"
        android:fromXScale="1.0"
        android:fromYScale="1.0"
        android:toXScale="0.0"
        android:toYScale="1.0" />
</set>
```

slide_right

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    android:interpolator="@android:anim/linear_interpolator">
    <scale
        android:duration="500"
        android:fromXScale="0.0"
        android:fromYScale="1.0"
        android:toXScale="1.0"
        android:toYScale="1.0" />
</set>
```

zoom_in

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:fillAfter = "true">
    <scale
        xmlns:android = "http://schemas.android.com/apk/res/android"
        android:duration = "1000"
        android:fromYScale = "1"
        android:pivotX = "50%"
        android:pivotY = "50%"
        android:toXScale = "2"
        android:toYScale = "2"/>
</set>
```

zoom_out

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:fillAfter = "true">
    <scale
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:duration = "2500"
        android:fromXScale = "1"
        android:fromYScale = "1"
        android:pivotX = "50%"
        android:pivotY = "50%"
        android:toXScale = ".2"
        android:toYScale = ".2" />
</set>
```

PARTIE 2

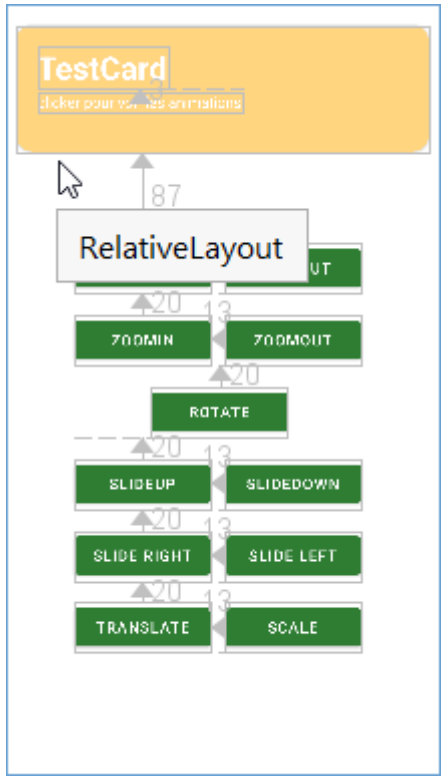
Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 4

Création de plusieurs animations (Zoom In/Out Animation, Rotate Animation…)

PARTIE 2

Etape 2 : Modifier le layout




```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:paddingLeft="10dp"
    android:paddingRight="10dp">
    <androidx.cardview.widget.CardView
        android:id="@+id/imgvw"
        android:layout_width="match_parent"
        android:layout_height="120dp"
        app:cardCornerRadius="15dp"
        android:layout_marginTop="20dp">
        <RelativeLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:background="#FFD580"
            android:padding="10dp">
            <TextView
                android:id="@+id/t1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="TestCard"
                android:textSize="30dp"
                android:textColor="#ffff"
                android:textStyle="bold"
                android:layout_marginTop="10dp"
                android:layout_marginLeft="10dp"/>
```

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 4

Création de plusieurs animations (Zoom In/Out Animation, Rotate Animation…)

Etape 2 : Modifier le layout

```
 <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="clicker pour voir les animations"
            android:layout_below="@+id/t1"
            android:layout_marginLeft="10dp"
            android:layout_marginTop="3dp"
            android:textColor="#ffff"/>
      </RelativeLayout>
   </androidx.cardview.widget.CardView>

<Button
   android:id="@+id/btnFadeIn"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_below="@+id/imgvw"
   android:layout_marginLeft="55dp"
   android:layout_marginTop="87dp"
   android:width="130dp"
   android:text="Fade In" />

<Button
   android:id="@+id/btnFadeOut"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_alignBottom="@+id/btnFadeIn"
   android:layout_marginLeft="13dp"
   android:layout_toRightOf="@+id/btnFadeIn"
   android:width="130dp"
   android:text="Fade Out" />
```

```
<Button
   android:id="@+id/zoomIn"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_below="@+id/btnFadeIn"
   android:layout_marginLeft="55dp"
   android:layout_marginTop="20dp"
   android:width="130dp"
   android:text="ZoomIn" />

<Button
   android:id="@+id/zoomOut"
   android:width="130dp"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_alignBottom="@+id/zoomIn"
   android:layout_marginLeft="13dp"
   android:layout_toRightOf="@+id/zoomIn"
   android:text="zoomOut" />


<Button
   android:id="@+id/Rotate"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_below="@+id/zoomIn"
   android:layout_marginLeft="128dp"
   android:layout_marginTop="20dp"
   android:width="130dp"
   android:text="rotate" />
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 4

Création de plusieurs animations (Zoom In/Out Animation, Rotate Animation…)

PARTIE 2

Etape 2 : Modifier le layout

```
<Button
    android:id="@+id/SlideUp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/Rotate"
    android:layout_marginLeft="55dp"
    android:layout_marginTop="20dp"
    android:width="130dp"
    android:text="SlideUp" />

<Button
    android:id="@+id/SlideDown"
    android:width="130dp"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBottom="@+id/SlideUp"
    android:layout_marginLeft="13dp"
    android:layout_toRightOf="@+id/SlideUp"
    android:text="SlideDown" />

<Button
    android:id="@+id/SlideRight"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/SlideUp"
    android:layout_marginLeft="55dp"
    android:layout_marginTop="20dp"
    android:width="130dp"
    android:text="Slide Right" />

<Button
        android:id="@+id/SlideLeft"
        android:width="130dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/SlideRight"
        android:layout_marginLeft="13dp"
        android:layout_toRightOf="@+id/SlideRight"
        android:text="Slide Left" />

    <Button
        android:id="@+id/translate"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/SlideRight"
        android:layout_marginLeft="55dp"
        android:layout_marginTop="20dp"
        android:width="130dp"
        android:text="Translate" />

    <Button
        android:id="@+id/Scale"
        android:width="130dp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBottom="@+id/translate"
        android:layout_marginLeft="13dp"
        android:layout_toRightOf="@+id/translate"
        android:text="Scale" />
</RelativeLayout>
```

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 4

Création de plusieurs animations (Zoom In/Out Animation, Rotate Animation…)

Etape 3 : Modifier la classe MainActivity

```
class MainActivity : AppCompatActivity() {
    lateinit var card: CardView
    lateinit var rotateButton: Button
    lateinit var scale: Button
    lateinit var translateButton: Button
    lateinit var SlideUp: Button
    lateinit var SlideDown: Button
    lateinit var SlideLeft: Button
    lateinit var SlideRight: Button
    lateinit var fadeIn: Button
    lateinit var fadeOut: Button
    lateinit var zoomIn: Button
    lateinit var zoomOut: Button

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        card = findViewById(R.id.imgvw)
        rotateButton = findViewById<Button>(R.id.Rotate)
        translateButton = findViewById<Button>(R.id.translate)
        SlideUp = findViewById<Button>(R.id.SlideUp)
        SlideDown = findViewById<Button>(R.id.SlideDown)
        SlideLeft = findViewById<Button>(R.id.SlideLeft)
        SlideRight = findViewById<Button>(R.id.SlideRight)
        fadeIn = findViewById<Button>(R.id.btnFadeIn)
        fadeOut = findViewById<Button>(R.id.btnFadeOut)
        zoomIn = findViewById<Button>(R.id.zoomIn)
        zoomOut = findViewById<Button>(R.id.zoomOut)
        scale = findViewById<Button>(R.id.Scale)

zoomOut.setOnClickListener {
    val animZoomOut = AnimationUtils.loadAnimation(this,
        R.anim.zoom_out)
    card.startAnimation(animZoomOut)
}

SlideUp.setOnClickListener {
    val slideUp = AnimationUtils.loadAnimation(this,
        R.anim.slide_up)
    card.startAnimation(slideUp)
}

SlideDown.setOnClickListener {
    val slide_down = AnimationUtils.loadAnimation(this,
        R.anim.slide_down)
    card.startAnimation(slide_down)
}

SlideLeft.setOnClickListener {
    val slide_left = AnimationUtils.loadAnimation(this,
        R.anim.slide_left)
    card.startAnimation(slide_left)
}

SlideRight.setOnClickListener {
    val slide_right = AnimationUtils.loadAnimation(this,
        R.anim.slide_right)
    card.startAnimation(slide_right)
}
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT

ACTIVITÉ n° 4

Création de plusieurs animations (Zoom In/Out Animation, Rotate Animation…)

Etape 3 : Modifier la classe MainActivity

```
fadeIn.setOnClickListener {
        val animZoomIn = AnimationUtils.loadAnimation(this,
            R.anim.fade_in)
        card.startAnimation(animZoomIn)
    }

    fadeOut.setOnClickListener {
        val animZoomIn = AnimationUtils.loadAnimation(this,
            R.anim.fade_out)
        card.startAnimation(animZoomIn)
    }

    zoomIn.setOnClickListener {
        val animZoomIn = AnimationUtils.loadAnimation(this,
            R.anim.zoom_in)
        card.startAnimation(animZoomIn)
    }

    rotateButton.setOnClickListener {
        rotater()
    }

    translateButton.setOnClickListener {
        translater()
    }

    scale.setOnClickListener {
        scaler()
    }
}
```

```
 private fun rotater() {
        val animator = ObjectAnimator.ofFloat(card, View.ROTATION, -360f, 0f)
        animator.duration = 1000
        animator.start()
    }

    private fun translater() {
        val animator = ObjectAnimator.ofFloat(card, View.TRANSLATION_X, 200f)
        animator.repeatCount = 1
        animator.duration = 1500
        animator.repeatMode = ObjectAnimator.REVERSE
        animator.start()
    }

    private fun scaler() {
        val scaleX = PropertyValuesHolder.ofFloat(View.SCALE_X, 4f)
        val scaleY = PropertyValuesHolder.ofFloat(View.SCALE_Y, 4f)
        val animator = ObjectAnimator.ofPropertyValuesHolder(
            card, scaleX, scaleY)
        animator.repeatCount = 1
        animator.repeatMode = ObjectAnimator.REVERSE
        animator.start()
    }
}
```

PARTIE 2

Copyright - Tout droit réservé - OFPPT