

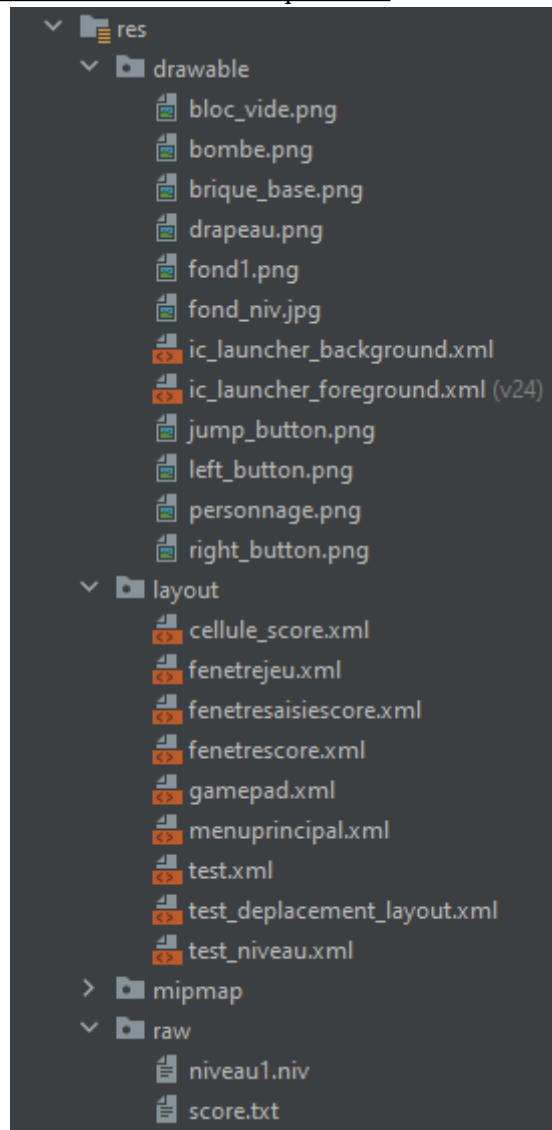
## Codes

Je sais utiliser les Intent pour faire communiquer deux activités.

```
public void cliqueJouer(View view) {  
    Intent monIntent = new Intent( packageContext: this, Jeu.class);  
    startActivity(monIntent);  
}
```

Je sais développer en utilisant le SDK le plus bas possible.

Je sais distinguer mes ressources en utilisant les qualifier.



Je sais faire des vues xml en utilisant layouts et composants adéquats.

Ici le code de la fenêtre du menu principal :

```

<ScrollView
    android:id="@+id/scrollView2"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toStartOf="parent">

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

        <androidx.constraintlayout.widget.ConstraintLayout
            android:id="@+id/constraintLayout"
            android:layout_width="match_parent"
            android:layout_height="458dp">

            <Button...>

            <Button...>

            <Button...>

            <Button...>

        </androidx.constraintlayout.widget.ConstraintLayout>
    </LinearLayout>
</ScrollView>

```

Je sais coder proprement mes activités, en m'assurant qu'elles ne font que relayer les évènements.  
Ici le code de l'une de nos activités :

```

public class SaisieScore extends AppCompatActivity {
    private TextView textViewTemps;
    private TextView textViewNiveau;
    private EditText editText;
    private LesScores lesScores;
    private int temps;
    private int niveau;

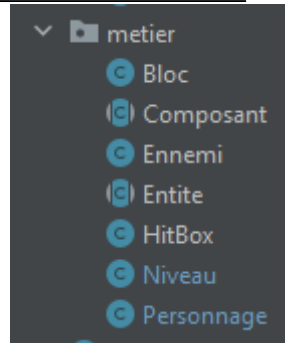
    @SuppressWarnings("SetTextI18n")
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {...}

    @Override
    protected void onSaveInstanceState(@NonNull Bundle outState) {...}

    public void cliqueValiderScore(View view) {...}
}

```

Je sais coder une application en ayant un véritable métier.



Je sais parfaitement séparer vue et modèle.

Voir diagrammes de classes véritables séparation entre vue et modèle, en vert on a la couche graphique, en bleu clair on retrouve les classes du modèle logique et du métier.

Je maîtrise le cycle de vie de mon application.

```
@Override
protected void onCreate(@Nullable Bundle savedInstanceState) {...}

@Override
protected void onResume() {
    super.onResume();
}
```

Je sais utiliser le findViewById à bon escient.

```
public class VoirScore extends AppCompatActivity {

    private LesScores lesScores;
    private RecyclerView recyclerView;

    @SuppressWarnings("ResourceType")
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {...}

    @Override
    protected void onResume() {
        super.onResume();
        recyclerView = findViewById(R.id.recyclerview);
        recyclerView.setLayoutManager(new LinearLayoutManager(context, LinearLayoutManager.VERTICAL, false));
        recyclerView.setAdapter(new MonAdaptateur(lesScores, activiteParente: this));
    }
}
```

Je sais gérer les permissions dynamiques de mon application.

Nous n'en avons pas eu besoin pour créer un plateformer.

Je sais gérer la persistance légère de mon application.

Dans la classe MenuPrincipal :

```
@Override
protected void onSaveInstanceState(@NonNull Bundle outState) {
    outState.putParcelable("lesScores", lesScores);
    super.onSaveInstanceState(outState);
}
```

Je sais gérer la persistance profonde de mon application.

Appelle dans la classe SaisieScore de la méthode sauver de SauveurScore qui sauvegarde le fichier sur le téléphone dans une zone où nous avons accès à la modification de fichiers.

```
public void cliqueValiderScore(View view) {
    File file = getDir( name: "score", mode: 0);
    SauveurDeScores sauveurDeScores = new SauveurDeScores();
    lesScores.ajouterScore(new Score(String.valueOf(editText.getText()), temps, niveau));
    try {
        sauveurDeScores.sauver(lesScores, new FileOutputStream( name: file.getAbsolutePath() + "/score.txt"));
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
    Intent monIntent = new Intent( packageContext: this, MenuPrincipal.class);
    monIntent.putExtra( name: "lesScores", (Parcelable) lesScores);
    startActivity(monIntent);
}
```

Je sais afficher une collection de données.

J'utilise une RecyclerView dans la classe VoirScore.

```
@Override
protected void onResume() {
    super.onResume();
    recyclerView = findViewById(R.id.recyclerview);
    recyclerView.setLayoutManager(new LinearLayoutManager( context: this, LinearLayoutManager.VERTICAL,
        reverseLayout: false));
    recyclerView.setAdapter(new MonAdaptateur(lesScores, activiteParente: this));
}
```

Je sais coder mon propre adaptateur.

La classe MonAdpatateur :

```

public class MonAdaptateur extends RecyclerView.Adapter {

    private LesScores lesScores;
    private AppCompatActivity activiteParente;

    public MonAdaptateur(LesScores lesScores, AppCompatActivity activiteParente) {
        super();
        this.lesScores = lesScores;
        this.activiteParente = activiteParente;
    }

    @NonNull
    @Override
    public RecyclerView.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = activiteParente.getLayoutInflater().inflate(R.layout.cellule_score, parent, attachToRoot: false);
        return new ViewHolderScore(view);
    }

    @Override
    public void onBindViewHolder(@NonNull RecyclerView.ViewHolder holder, int position) {
        Score score = lesScores.getListeScores().get(position);
        ((ViewHolderScore) holder).getTextViewNom().setText(score.getNom());
        ((ViewHolderScore) holder).getTextViewNiveau().setText(String.valueOf(score.getNiveau()));
        ((ViewHolderScore) holder).getTextViewTemps().setText(String.valueOf(score.getTemps()));
    }












    @Override
    public int getItemCount() { return lesScores.getListeScores().size(); }
}

```

Je maîtrise l'usage des fragments.

Nous n'en avons pas utiliser.

Je maîtrise l'utilisation de Git.

|   |  |
|---|--|
|    | <b>contexte.odt</b><br>figaugirar authored 22 hours ago  |
|    | <b>Ajout centrage niveau en fonction du téléphone début de la mise en place des...</b> <span>...</span><br>Antoine VITON authored 23 hours ago |
| 08 Mar, 2022 1 commit   |  |
|    | <b>Modification chargement du niveau, mise à jour diagrammes de classes,...</b> <span>...</span><br>Antoine VITON authored 1 week ago          |
| 02 Mar, 2022 5 commits  |  |
|    | <b>Résolution conflit</b><br>Antoine VITON authored 1 week ago   |
|    | <b>Mise en place du chargement et de l'affichage du niveau, modification de la...</b> <span>...</span><br>Antoine VITON authored 1 week ago    |
|    | <b>debut de la gestion des collisions (problème dans la détection de la taille de l'écran)</b><br>figaugirar authored 1 week ago               |
|    | <b>deplacement fonctionelles, le personnage peut se déplacer sur l'axe horizontal...</b> <span>...</span><br>figaugirar authored 1 week ago    |
|    | <b>debut des déplacements</b><br>figaugirar authored 1 week ago  |
| 01 Mar, 2022 5 commits  |  |
|    | <b>Revert "Merge remote-tracking branch 'origin/main' into main"</b> <span>...</span><br>figaugirar authored 2 weeks ago                       |
|    | <b>Merge remote-tracking branch 'origin/main' into main</b> <span>...</span><br>figaugirar authored 2 weeks ago                                |
|   | <b>gamepad.xml + images associées + création fonctions associées</b><br>figaugirar authored 2 weeks ago  |
|  | <b>Test sur l'affichage du niveau, affichage d'une tuile réusssi, nettoyage ManagerJeu et Afficheur</b><br>Antoine VITON authored 2 weeks ago  |

## Application

Je sais développer une application sans utiliser de librairies externes.

Nous n'avons pas utilisé de librairie externe.

Je sais développer une application publiable sur le store.

Nous en avons fait la demande à monsieur Bouhours certainement trop tard.

Je sais développer un jeu intégrant une boucle de jeu threadée observable.

La classe Boucle :

```
public class Boucle extends BoucleAbstraite{

    private Thread threadInterne;
    public static final double TPSRAFF = 1000.0/30;

    @Override
    public void run() {
        while (jeuEnCours) {
            try {
                threadInterne.sleep((long)TPSRAFF);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
            notifier();
        }
    }
}
```

Je sais développer un jeu graphique sans utiliser de SurfaceView.

Création de notre propre VueJeu :

```
public class VueJeu extends View {

    private List<BlocGraphique> listeBlocsGraphiques;
    private List<EntiteGraphique> listeEntitesGraphiques;
    private Bitmap fond;
    private AfficheurAndroid afficheurAndroid;
    private int temps;

    public VueJeu(Context context, List<BlocGraphique> listeBlocsGraphiques,
        List<EntiteGraphique> listeEntitesGraphiques, AfficheurAndroid afficheur) {...}

    @Override
    protected void onDraw(Canvas canvas) {
        @SuppressWarnings("DrawAllocation") Paint paint = new Paint(Paint.ANTI_ALIAS_FLAG);
        paint.setTextSize(250);
        canvas.drawBitmap(fond, left: 0, top: 0, paint);
        for (BlocGraphique blocGraphique : listeBlocsGraphiques) {...}
        for (EntiteGraphique entiteGraphique : listeEntitesGraphiques){
            Bitmap bitmap = entiteGraphique.getImage();
            canvas.drawBitmap(bitmap, left: afficheurAndroid.getDecalage() + entiteGraphique.getEntite().
                getEntite().getPositionX(), top: afficheurAndroid.getDecalage() + entiteGraphique.getEntite().
                getPositionY(), paint);
        }
    }
}
```