

Preuves

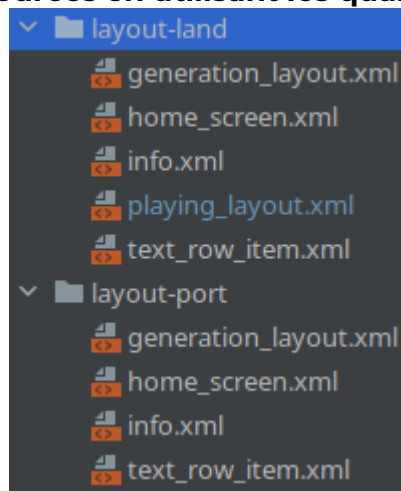
Je sais utiliser les Intent pour faire communiquer deux activités :

```
public void play(View view) {  
    Intent intent = new Intent( packageContext: this, Generation.class);  
    startActivity(intent);  
}
```

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

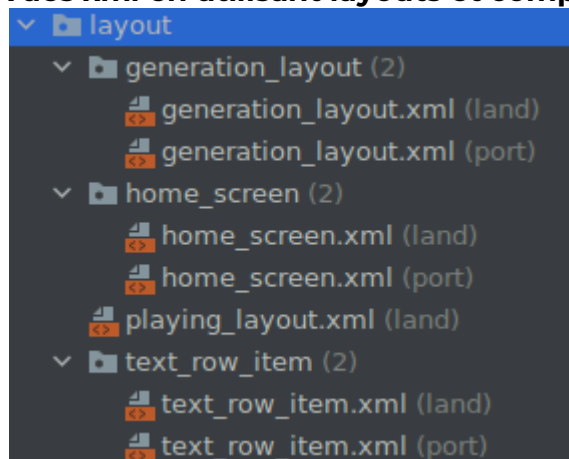
Vers la fin du projet nous nous sommes rendu compte que le SDK que nous utilisions était le 31, cela est dû à une erreur lors de la création du projet. Nous avons donc modifié ce que nous pouvions pour faire redescendre le SDK en 21 afin de prendre en compte un plus grand nombre d'appareils

Je sais distinguer les ressources en utilisant les qualifier :



Nous avons utilisé des qualifier pour différencier les ressources qui sont en vue portrait de celles qui sont en vue paysage.

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



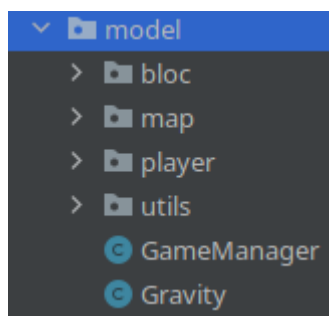
Je sais coder proprement mes activités, en m'assurant qu'elles ne font que relayer les événements :

```
//loading assets, map and player
drawableManager = DrawableManager.getInstance();
drawableManager.loadBitmaps( context: this, cellsize);
setContentView(R.layout.playing_layout);
LinearLayout = findViewById(R.id.LinearLayout);
canvas = new GameView( context: this);
gameManager = new GameManager(canvas, worldWidth, worldHeight, loadBitmaps(), cellsize);
gameManager.setGameView();

LinearLayout.addView(canvas);
ImageView.generateViewId();
```

(Voir PlayingActivity.java dans /java/fr.iut.minecraft2d/)
Utilisation de managers pour relayer les évènements.

Je sais coder une application en ayant un véritable métier :



Le dossier modèle contient nos classes métiers.

Je sais parfaitement séparer vue et modèle :

Pour gérer la vue nous passons par des View, telle que GameView qui gère la vue de jeu. Ainsi L'activité Playing Activity relaye les informations à GameManager qui lui même relaye les informations à la GameView.

Je maîtrise le cycle de vie de mon application :

```

@Override
protected void onStart() {
    Log.d( tag: "PlayingActivity.java", msg: "onStart");
    super.onStart();
}

@Override
protected void onStop() {
    //Save the data
    try {
        sauveur.save(openFileOutput(FILE_NAME,MODE_PRIVATE),counter);
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
    Log.d( tag: "PlayingActivity.java", msg: "onStop");
    super.onStop();
}

@Override
protected void onDestroy() {
    // fileSaver.save(String.valueOf(counter),this);
    Log.d( tag: "PlayingActivity.java", msg: "onDestroy");
    Toast.makeText(getApplicationContext(), text: "Partie supprimée", Toast.LENGTH_LONG).show();
    super.onDestroy();
}

@Override
protected void onPause() {
    //fileSaver.save(String.valueOf(counter),this);
    Toast.makeText(getApplicationContext(), text: "Partie en pause", Toast.LENGTH_LONG).show();
    super.onPause();
}

@Override
protected void onResume() {
    Log.d( tag: "PlayingActivity.java", msg: "OnResume");
    super.onResume();
}

```

Je sais utiliser le findViewById à bon escient :

```
linearLayout = findViewById(R.id.linearLayout);
```

Je sais gérer les permissions dynamiques de mon application :

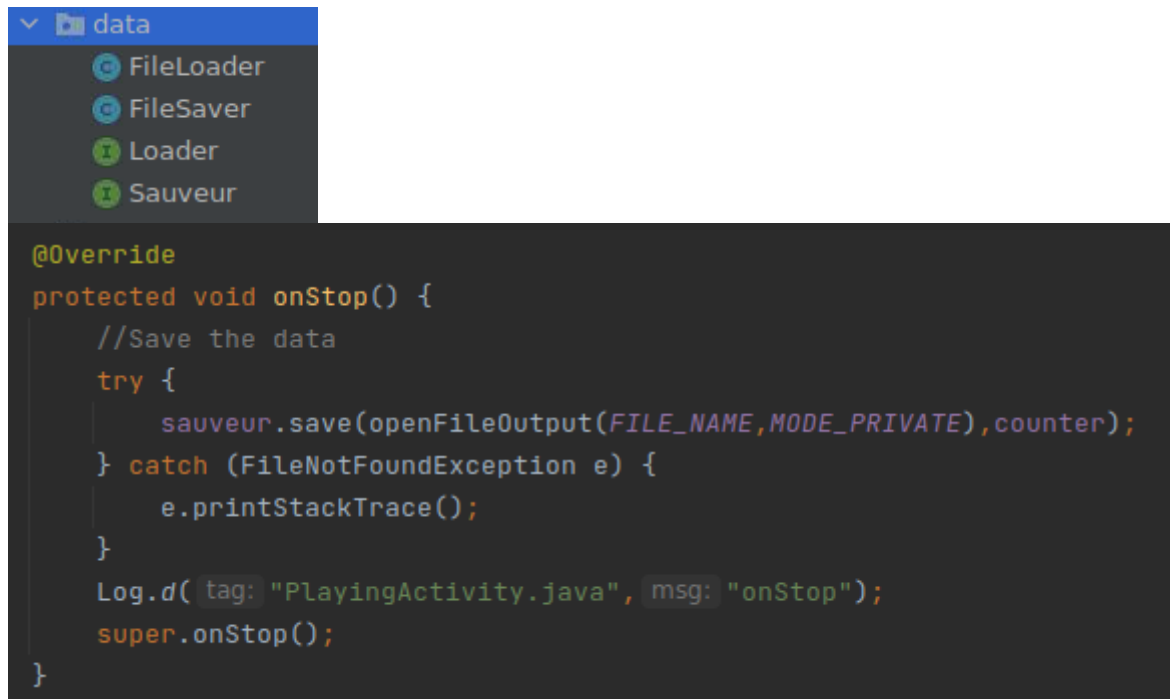
Je sais gérer la persistance légère de mon application :

```

<Button
    android:id="@+id/buttonMondePetit"
    android:id="@+id/timer_textView"

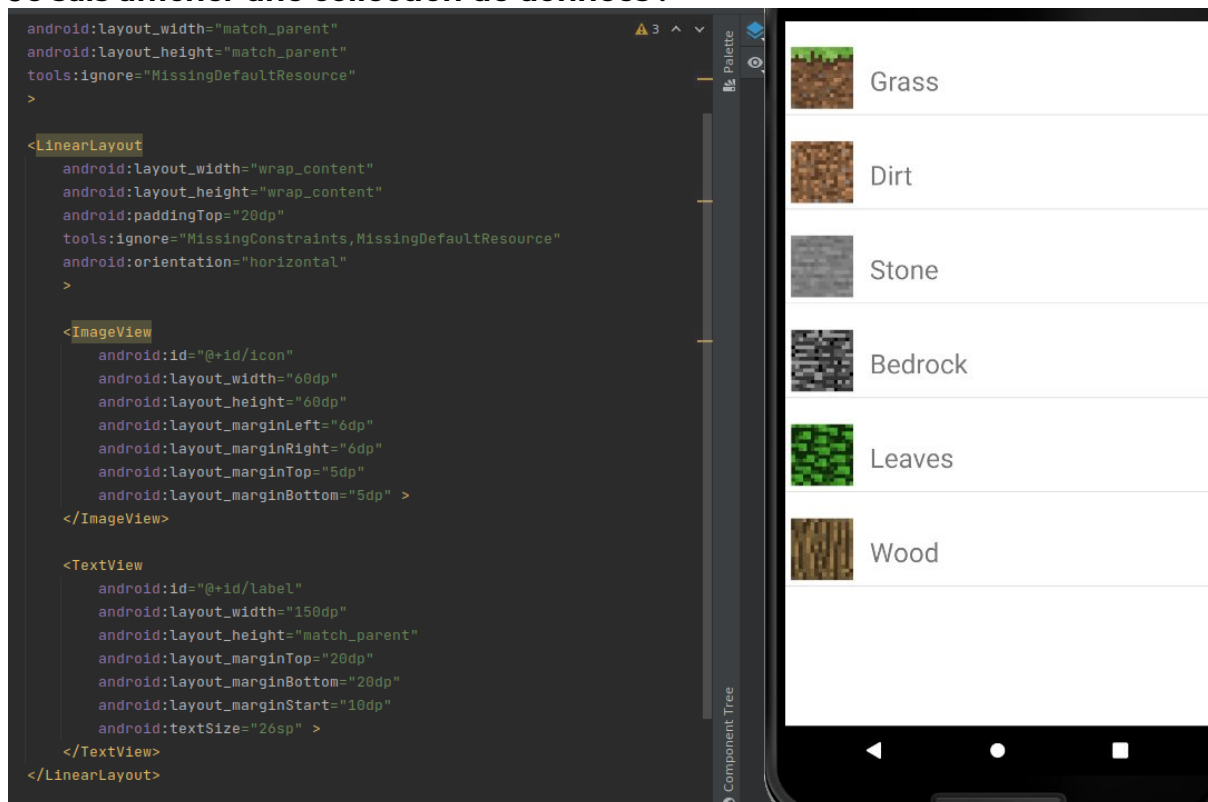
```

Je sais gérer la persistance profonde de mon application :




Sauvegarde des données dans un fichier .txt
(Voir PlayingActivity.java dans /java/fr.iut.minecraft2d/)

Je sais afficher une collection de données :



Je sais coder mon propre adaptateur :

 AdaptateurDeListe

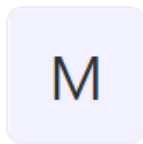
```
public class AdaptateurDeListe extends ArrayAdapter<String>
```

(voir AdaptateurDeListe dans /java/fr.iut.minecraft2d/)

Je maîtrise l'usage des fragments :






```
public class InfoFragment extends Fragment {  
    public InfoFragment() { super(R.layout.info); }  
}
```

Je maîtrise l'utilisation de Git :



Minecraft2D Android-studio 

Project ID: 2305

 **57** Commits  **1** Branch  **0** Tags  **57.3 MB** Files  **57.3 MB** Storage

24 Mar, 2022 1 commit



ajout d'un fragment Info + ajout d'un bouton pour aller vers le bas... 

Ben Gregory authored 20 hours ago

18 Mar, 2022 1 commit



essai infructueux de faire fonctionner les collisions

begregory authored 1 week ago

17 Mar, 2022 4 commits



Ajout dans la doc de ce qui a été fait précédemment

arpeyronon authored 1 week ago



Modification de l'icone de l'app

arpeyronon authored 1 week ago



cleanning code + javadoc

arpeyronon authored 1 week ago



Thread ok + loader et saver de la data ok

arpeyronon authored 1 week ago