Compétences devant être maîtrisées le 27 mars 2022

DOCUMENTATION
Je sais décrire le contexte de mon application, pour que n'importe qui soit capable de comprendre à quoi elle sert.
Voir contexte

Je sais concevoir et décrire un diagramme de cas d'utilisation pour mettre en avant les différentes fonctionnalités de mon application.
Voir le fichier starUML diagramme

Je sais concevoir un diagramme UML de qualité représentant mon application.
Voir le fichier starUML diagramme

Je sais décrire mon diagramme UML en mettant en valeur et en justifiant les éléments essentiels.
Voir le fichier starUML diagramme

CODE
Je sais utiliser les Intent pour faire communiquer deux activités.

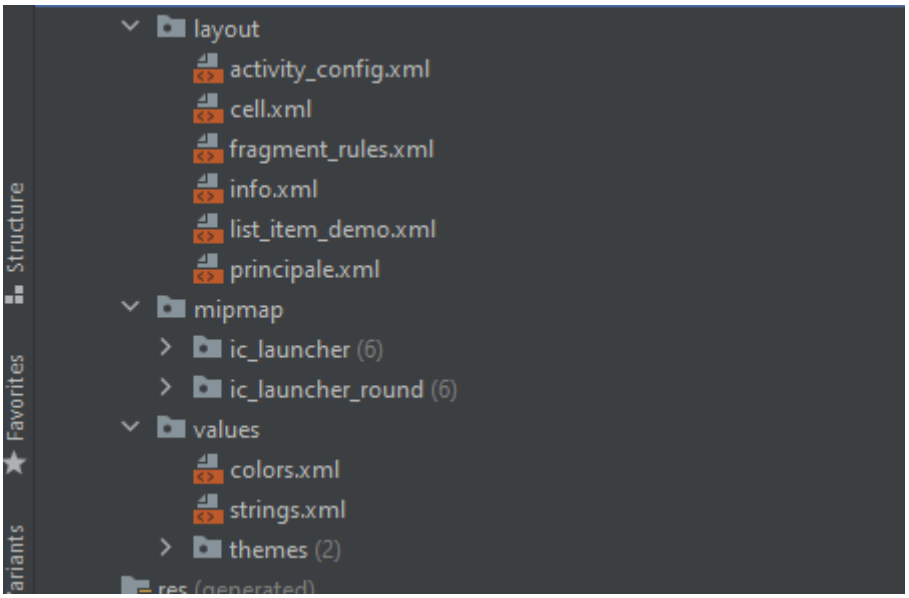
```
<intent-filter>
    <action android:name="android.intent.action.MAIN"/>
    <category android:name="android.intent.category.LAUNCHER"/>
</intent-filter>


public void clicSurBoutonConfig(){
    Intent monIntent = new Intent( packageContext: this, ConfigActivity.class);
    startActivity(monIntent);
}

public void clicSurBoutonInfo(){
    Intent monIntent = new Intent( packageContext: this, Info.class);
    startActivity(monIntent);
}


public void clicSurBoutonPlay(){
```

Je sais développer en utilisant le SDK le plus bas possible.
Voir sur le projet
Je sais distinguer mes ressources en utilisant les qualifier



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

```
    <Button
        android:id="@+id/clear"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_row="0"
        android:layout_column="1"
        android:layout_marginEnd="12dp"
        android:layout_marginRight="12dp"
        android:text="@string/clear"
        app:layout_constraintBottom_toBottomOf="@+id/Avancer"
        app:layout_constraintEnd_toStartOf="@+id/Avancer" />

    <GridLayout
        android:id="@+id/Table"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="160dp"
        app:layout_constraintTop_toBottomOf="@+id/config"
        tools:layout_editor_absoluteX="56dp">


    </GridLayout>


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

Je sais coder proprement mes activités, en m’assurant qu’elles ne font que relayer les évènements

```
clear.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        dieu.clearGrid();
        actualiser();
    }
});
avancer.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) { start(); }
});
```

Je sais coder une application en ayant un véritable métier
Voir application
Je sais parfaitement séparer vue et modèle

```
public class BaseApplication extends Application {

    private static BaseApplication instance;

    private Dieu dieu;

    public static BaseApplication getInstance() { return instance; }

    public Dieu getDieu() {
        if(dieu == null){
            dieu=new Stub().Loader(getBaseContext()).get("recursif");
        }
        return dieu;
    }

    @Override
    public void onCreate() {
        super.onCreate();
        instance = this;
    }
}
```

La classe Dieu joue le rôle de manager.

Je maîtrise le cycle de vie de mon application

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

@Override
public void onCreate(@Nullable Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);
    setContentView(R.layout.principale);
    boutonLancement=findViewById(R.id.lancement);
```

Je sais utiliser le findViewById à bon escient

```
public void onCreate(@Nullable Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);
    setContentView(R.layout.principale);
    boutonLancement=findViewById(R.id.lancement);
    boutonConfig=findViewById(R.id.config);
```

Je sais gérer les permissions dynamiques de mon application

Je sais gérer la persistance légère de mon application

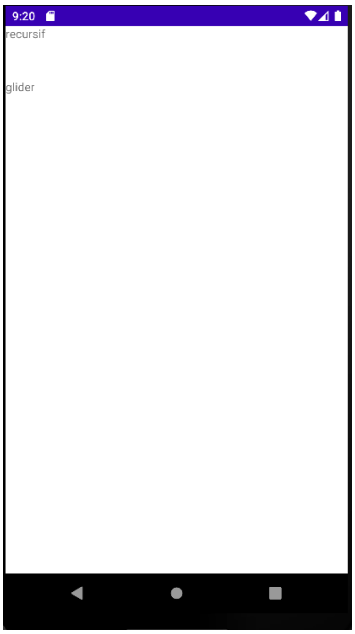
```
<?xml version="1.0" encoding="utf-8"?>
<CheckBox xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/checkbox"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

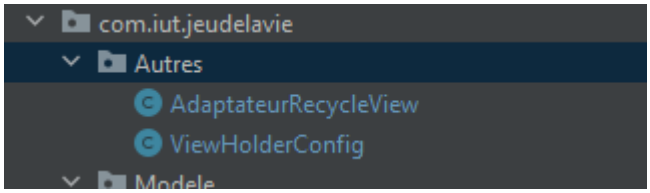
Voir code

Je sais gérer la persistance profonde de mon application

Voir code

Je sais afficher une collection de données





Je sais coder mon propre adaptateur

```java
public class AdaptateurRecycleView extends  RecyclerView.Adapter  {

    private HashMap<String, Monde> lesConfig;
    private OnItemClick onItemClick;

    public AdaptateurRecycleView(HashMap<String, Monde> lesConfig,AdaptateurRecycleView.OnItemClick adp) {
        this.lesConfig = lesConfig;
        this.onItemClick=adp;
    }

    @NonNull
    @Override
    public RecyclerView.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        LinearLayout ln= (LinearLayout) LayoutInflater.from(parent.getContext()).inflate(R.layout.list_item_demo, parent,  attachToRoot: false);
        return new com.iut.jeudelavie.autres.ViewHolderConfig(ln, onItemClick);
    }

    @Override
    public void onBindViewHolder(@NonNull RecyclerView.ViewHolder holder, int position) {
        //recuperer mon model item grave a la position
        //binder mon model item sur mon view holder
        ArrayList<String> laList=new ArrayList<>();
        laList.addAll(lesConfig.keySet());
        String elementCourant =laList.get(position);
        ((com.iut.jeudelavie.autres.ViewHolderConfig)holder).getTextView().setText(elementCourant);

    }

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

    public interface OnItemClick {
        void onMondeClicked(Monde monde);
    }
}
```

Je maîtrise l’usage des fragments
Voir code
Je maîtrise l’utilisation de Git
Voir sur le repository

APPLICATION
Je sais développer une application sans utiliser de librairies externes.
Voir code
Je sais développer une application publiable sur le store.

Voir l’application.
Je sais développer un jeu intégrant une boucle de jeu threadée observable.

```
    @Override
    public void run() {
        while(true){
            if(BoucleDeJeu.played){
                notifyObservers();
            }
            try {
                Thread.sleep(time);
            } catch(Exception e){
                e.printStackTrace();
            }
        }

    }

    public void notifyObservers(){
        for (Observer o : listObserver){
            o.update();
        }
    }

    public void addListener(Observer observer) { listObserver.add(observer); }
}
```

Voir code

Je sais développer un jeu graphique sans utiliser de SurfaceView
Voir code