

Preuve Projet Android

Projet par : Adrien BLAY A3, Hector PITEAU B2

Code :

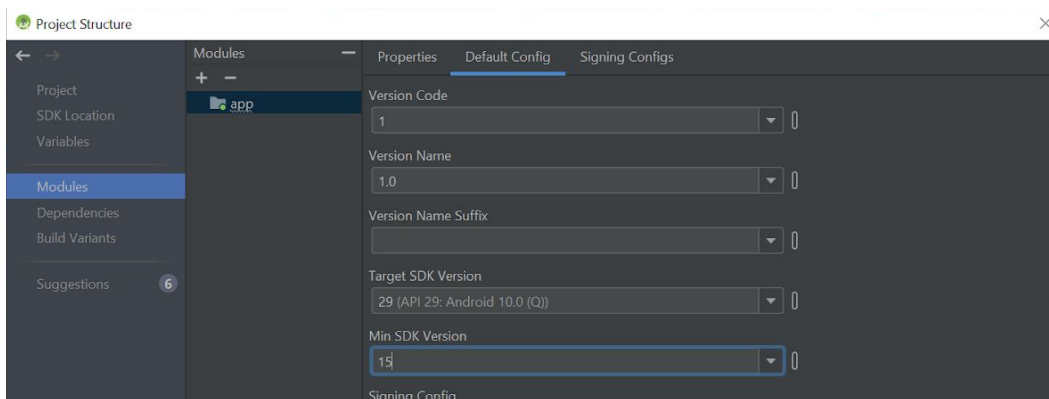
- Je sais utiliser les Intent pour faire communiquer deux activités

```
findViewById(R.id.Settings).setOnClickListener((view) -> {  
    startActivity(new Intent( packageContext: MainGame.this, SettingsActivity.class));  
});
```

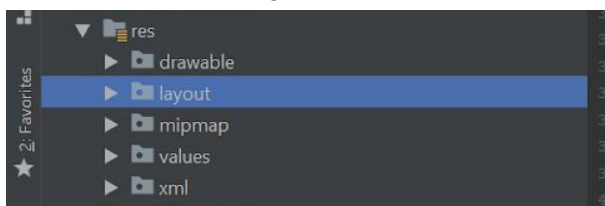
Oui, ici par exemple avec l'id Settings qui est dans MainActivity.xml, on va utiliser un intent pour passer de l'activité MainGame à l'activité SettingsActivity.

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

La version minimale de notre projet est la version 15.



- Je sais distinguer mes ressources en utilisant les qualifier



Oui dans le dossier res on a 5 packages et c'est là que l'on sépare nos ressources. Dans notre package drawable se trouve toutes nos images. Le package layout contient toutes nos activity. Dans notre package values on peut retrouver les colors.xml, arrays.xml, strings.xml et les styles.xml.

- Je sais modifier le manifeste de l'application en fonction de mes besoins

Oui voir AndroidManifest.xml dans le package manifests. Par exemple, à la ligne 20, on définit que le MainActivity soit en Orientation uniquement portrait. A la ligne 24, avec la balise `<intent-filter></intent-filter>`, on spécifie sur qu'elle activity notre application va se lancer. Ici, cela correspond à l'activity LoadActivity.

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3    package="com.example.blais_piteau_android"
4    android:versionCode="1"
5    android:versionName="1.0">
6
7    <application
8      android:allowBackup="true"
9      android:icon="@mipmap/ic_launcher"
10     android:label="BLAY_PITEAU_android"
11     android:roundIcon="@mipmap/ic_launcher_round"
12     android:screenOrientation="portrait"
13     android:supportRtl="true"
14     android:theme="@style/AppTheme">
15     <activity android:name=".GameOverView"></activity>
16     <activity
17       android:name=".SettingsActivity"
18       android:label="Settings" />
19     <activity android:name=".StatsActivity" />
20     <activity
21       android:name=".MainActivity"
22       android:screenOrientation="portrait" />
23     <activity android:name=".MainGame"></activity>
24     <activity android:name=".LoadActivity">
25       <intent-filter>
26         <action android:name="android.intent.action.MAIN" />
27
28         <category android:name="android.intent.category.LAUNCHER" />
29       </intent-filter>
30     </activity>
31   </application>

```

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

Oui voir dans stats_activity.xml (ConstraintLayout) ou encore dans activity_game_over.xml (LinearLayout). On utilise aussi des EditText a la ligne 14 de main_activity.xml ou encore imageView à la ligne 25. Ici on utilise un LinearLayout. Dans ce linearLayout, on place notre orientation en verticale ainsi que notre gravity en center ce qui va nous permettre d'avoir un texte aligné au centre de la page. Pour TextView, on affiche un texte Game Over avec une taille de 30sp. On change la couleur du texte en blanc (text2 etant defini dans color.xml)

```

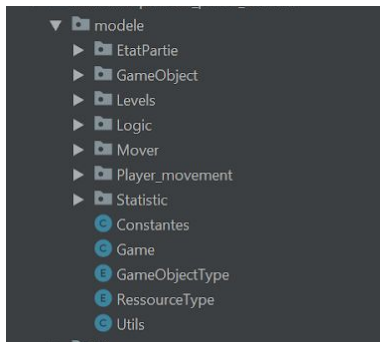
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".GameOverView"
  android:orientation="vertical"
  android:gravity="center"
  android:background="@drawable/bg2"
  >

  <TextView
    android:id="@+id/game_over"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textSize="30sp"
    android:text="GAME OVER"
    android:textColor="@color/text2"
    />

  <TextView
    android:id="@+id/score"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textSize="30sp"
    android:text="0"
    android:layout_marginTop="80dp"
    android:textColor="@color/text2"
    />

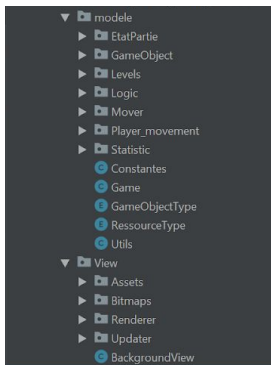
```

- Je sais coder une application en ayant un véritable métier



Oui, voir package modèle. Dans notre package modèle se trouve nos classes métiers. Nous avons aussi plusieurs packages avec d'autres classes à l'intérieur.

- Je sais parfaitement séparer vue et modèle



Oui voir dans com.example.blais_piteau_android, il y a un package view et un modele où l'on sépare la vue du modèle.

- Je maîtrise le cycle de vie de mon application

Oui, voir dans MainActivity, à partir de la ligne 40. Ici, nous créons des méthodes onStart, onResume, onPause, onDestroy, onStop pour gérer le cycle de vie de notre application.

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

@Override
protected void onResume() {
    super.onResume();
    background.resume();
    game.resume();
}

@Override
protected void onPause() {
    super.onPause();
    background.pause();
    game.pause();
}

@Override
protected void onDestroy() { super.onDestroy(); }

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

- Je sais utiliser le findViewById à bon escient

Oui par exemple, dans GameOverView.java à la ligne 22. Ici, on va chercher id bestScore qui se trouve dans activity_game_over.xml et qui est dans un TextView. Le findViewById ici, va rechercher la vue qui est identifiée par l'id best_score.

```
15
16
17 @Override
18 protected void onCreate(Bundle savedInstanceState) {
19     super.onCreate(savedInstanceState);
20     setContentView(R.layout.activity_game_over);
21
22     TextView score = (TextView) findViewById(R.id.score);
23     TextView bestScore = (TextView) findViewById(R.id.best_score);
24
25     int scoreFin = getIntent().getIntExtra(Constants.SCORE_MESSAGE, 0);
26     score.setText(Integer.toString(scoreFin));
27
28     SharedPreferences settings = getSharedPreferences("GAME", Context.MODE_PRIVATE);
29
30     int meilleurScore = settings.getInt(Constants.MEILLEUR_SCORE, 0);
31
32     SharedPreferences.Editor e = settings.edit();
33     if(scoreFin > meilleurScore){
34         bestScore.setText("Meilleur Score : " + scoreFin);
35         e.putInt(Constants.MEILLEUR_SCORE, scoreFin);
36     }
37     else{
38         bestScore.setText("Meilleure Score : " + meilleurScore);
39     }
}
```

- Je sais gérer les permissions dynamiques de mon application

L'application va demander l'autorisation. L'application avec cette ligne de code va demander la permission pour avoir accès à internet.



```

1 android:versionName="1.0">
2 <uses-permission android:name="android.permission.INTERNET"/>
3 <application

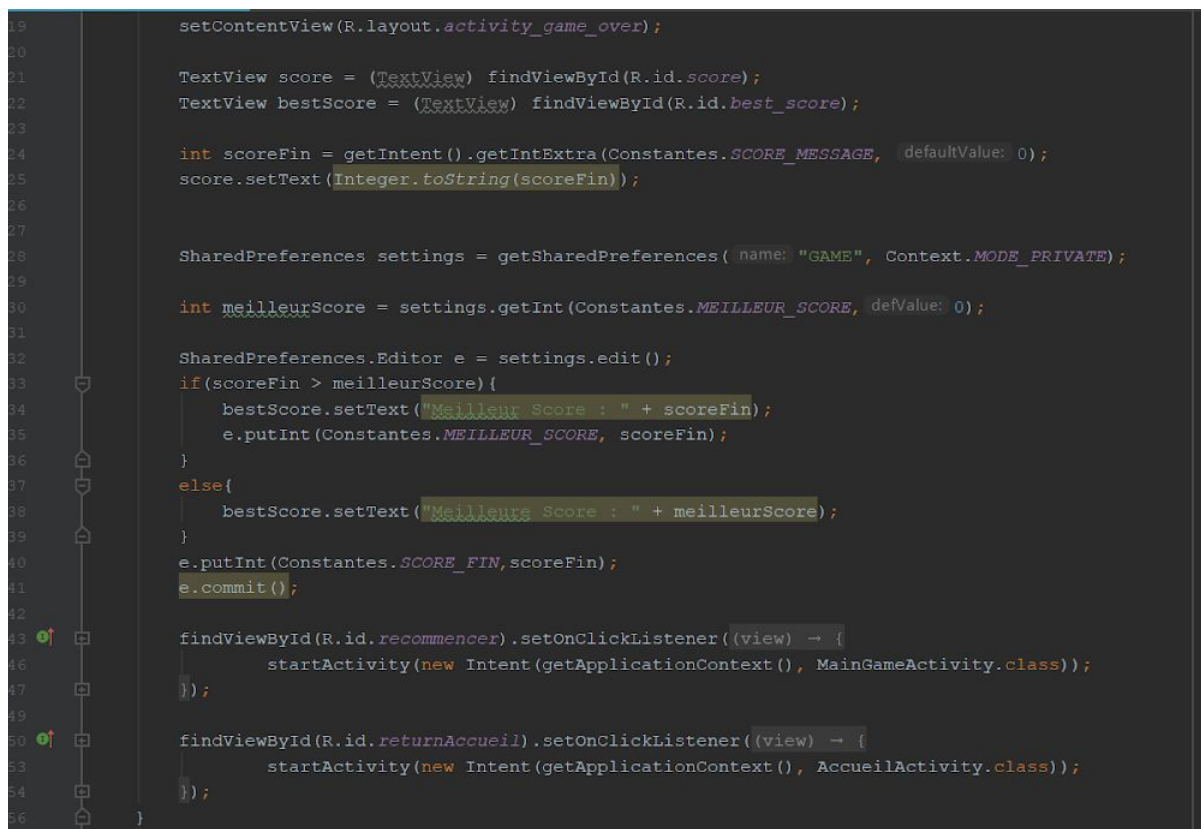
```

- Je sais gérer la persistance légère de mon application

Non, nous n'avons pas utilisé.

- Je sais gérer la persistance profonde de mon application

Oui dans GameOverActivity.java à la ligne 28 par exemple. Avec le SharedPreferences, l'application pourra garder en mémoire le meilleur score par exemple après la fermeture de l'application et ainsi quand on relancera l'application, notre meilleur score sera sauvegardé. Le putInt à la ligne 35 va sauvegarder un int. Ici, scoreFin qui correspondra au score de la fin de la partie. Ce score va donc remplacer le meilleur score si il était plus élevé.



```

19 setContentView(R.layout.activity_game_over);
20
21 TextView score = (TextView) findViewById(R.id.score);
22 TextView bestScore = (TextView) findViewById(R.id.best_score);
23
24 int scoreFin = getIntent().getIntExtra(Constants.SCORE_MESSAGE, defaultValue: 0);
25 score.setText(Integer.toString(scoreFin));
26
27
28 SharedPreferences settings = getSharedPreferences( name: "GAME", Context.MODE_PRIVATE);
29
30 int meilleurScore = settings.getInt(Constants.MEILLEUR_SCORE, defValue: 0);
31
32 SharedPreferences.Editor e = settings.edit();
33 if(scoreFin > meilleurScore){
34     bestScore.setText("Meilleur Score : " + scoreFin);
35     e.putInt(Constants.MEILLEUR_SCORE, scoreFin);
36 }
37 else{
38     bestScore.setText("Meilleure Score : " + meilleurScore);
39 }
40 e.putInt(Constants.SCORE_FIN, scoreFin);
41 e.commit();
42
43 findViewById(R.id.recommencer).setOnClickListener((view) -> {
44     startActivity(new Intent(getApplicationContext(), MainGameActivity.class));
45 });
46
47
48
49
50 findViewById(R.id.returnAccueil).setOnClickListener((view) -> {
51     startActivity(new Intent(getApplicationContext(), AccueilActivity.class));
52 });
53
54
55 }

```

- Je sais afficher une collection de données

Nous n'avons pas implémenté de collection de données.

- Je sais coder mon propre adaptateur

Non

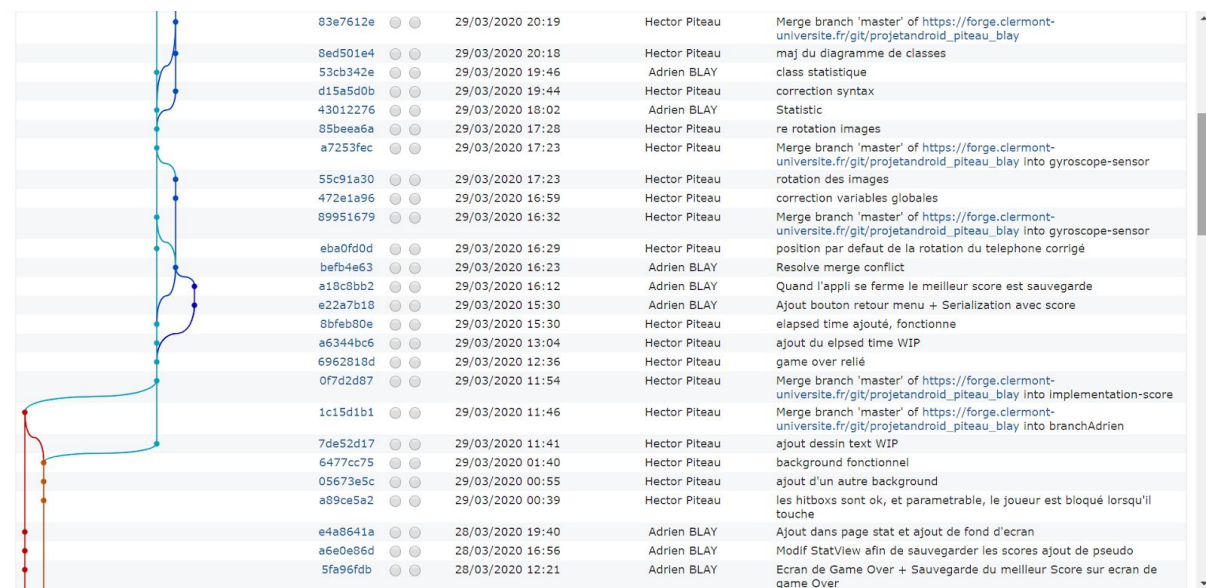
- Je maîtrise l'usage des fragments

Non

- Je maîtrise l'utilisation de Git

Oui voir le GIT

(https://forge.clermont-universite.fr/projects/projetandroid_piteau_blay/repository)



L'utilisation de git, la création de branche, merge des branches, résoudre des conflits.

Application :

- Je sais développer une application publiable sur le store

Elle n'est pas publiable sur le store.

- Je sais utiliser l'accéléromètre et boussole.

Oui dans GyroscopeController, à la ligne 88.

```
public void onSensorChanged(SensorEvent sensorEvent) {
    if(sensorEvent.sensor.getType() == Sensor.TYPE_ACCELEROMETER)
        accelOutput = sensorEvent.values;
    else if(sensorEvent.sensor.getType() == Sensor.TYPE_MAGNETIC_FIELD)
        magOutput = sensorEvent.values;
    if(accelOutput != null && magOutput != null) {
        //matrix 3x3 :
        float[] R = new float[9];
        float[] I = new float[9];

        boolean success = SensorManager.getRotationMatrix(R, I, accelOutput, magOutput);
        if(success) {
            SensorManager.getOrientation(R, orientation);
            Log.println(Log.DEBUG, tag: "HEC557788", msg: "accelOutput : " + Arrays.toString(orientation));
            if(startOrientation == null) {
                startOrientation = new float[orientation.length];
                System.arraycopy(orientation, srcPos: 0, startOrientation, destPos: 0, orientation.length);
                startOrientation[1] = 0;
                startOrientation[2] = 0;
            }
        }
    }
}
```

Nous avons décidé d'utiliser le gyroscope et la boussole plutôt que le gyroscope car certains téléphones ne sont pas équipés de gyroscope. Comme nous pouvons le voir nous l'utilisons afin de déterminer la position du joueur sur l'axe X principalement.