Rapport Sausage Shower

Ǵoomanee Izhaar, L3 Informatique

14 avril 2019

Résumé

1 Introduction

Sausage Shower est une application unique dans son genre. Il s'inspire des paysages uniques et variés de l'île de la Réunion. Donc, on a voulu élaborer un jeu qui répresente cette île intense au multiple de facette. Cette implication incite à découvrir ce paysage et les différentes saveurs que cultivent les Réunionnais. De ce fait, le fond d'écran représente les forêts multiples qu'il y a sur l'ile, par exemple la forêt de Bébour. L'avatar est un oiseau tropical, une espèce endémique de la Réunion, appelé communément à la réunion « Zoizo la Vierge » ou également « Zoizo malheur"Terpsiphone bourbonnensis bourbonnensis" ». Et en dernier lieu, quand on pense à l'île de la Réunion, on pense aussi aux fameux rougaille saucisse. C'est un plat typique de l'île et très apprécié par les réunionnais qui est représentée dans ce jeu. Sausage Shower permet de découvrir un lieu fabuleux de rêve et d'aventure.

2 Le but et le fonctionnement de l'application

Voici une belle capture d'écran d'une partie en cours :

2.1 Le but du Jeu

Le jeu est simple : le but est de faire manger à l'oiseau le maximum de nourriture possible, dans ce jeu l'oiseau mange des saucisses pour obtenir le plus de points. On note qu'il y a deux types de saucisses. Les saucisses de taille importante rapportent 20 points et les plus petites saucisses rapportent 10 points. Pour faire bouger l'oiseau, il suffit d'appuyer sur l'écran, mais il ne faut pas omettre qu'il y a également des braconniers qui sont placés à des endroits bien spécifiques, et ils attendent patiemment le passage de l'oiseau, pour lui lancer des flèches. Pour finir, vous avez 3 vies et à la fin des 3 vies utilisées, le jeu se termine.

2.2 Le description générale de l'implementation du jeu

Le jeu comprend plusieurs fenêtres différentes. Tout d'abord, vous avez la fenêtre principale sur laquelle vous pouvez lancer le jeu si vous le souhaitez. Une fois lancer, une fenêtre apparaît et le jeu se met en marche, l'oiseau débute sur sa position initiale, l'oiseau ne peut dépasser l'écran du jeu, il a une position minimale et maximale à respecter. Lorsque le joueur a perdu, une autre fenêtre surgit, « game over ». Sur cette fenêtre est affiché le score réalisé de cette partie par le joueur et également le meilleur score obtenu auparavant. Le meilleur score du joueur est enregistré. De plus, il y a un bouton « Play Again », si le joueur souhaite rejouer une partie, dans ce cas le jeu recommence depuis le début.

Voici les différentes classes et activités Java utilisés en**Android** :

— MainActivity > Page principale
— GameActivty > Lance le jeu
— GameOverActivity > Page GameOver et resultats
— GameView > Contient tout les codes et le méthodes du jeu

Voici les différentes classes Swift sur **IOS** :

— GameScene > on écrit tout le code dedans
— GameViewController > Ce fichier on le modifie pas, ca sert a lancer le jeu
Une liste numérotée :

1. premier item
2. second item

3 Architecture du code

Contrairement à la section 2, on va détailler les codes en plus *approfondis* On va commencer tout d'abord par **Android**, en décrivant dans l'ordre de l'execution. Puis par la suite, on aura la section iOS

3.1 Android

Voici le **MainActivity**
verbatim :

```
public class MainActivity extends AppCompatActivity implements View.OnClickListener{

    private ImageButton buttonPlay;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);


        //recuperer le ImageButton

        buttonPlay=(ImageButton) findViewById(R.id.buttonplay);

        //quand on appuie sur le button
        buttonPlay.setOnClickListener(this);

    }

    @Override
    public void onClick(View v) {

        startActivity(new Intent(this, GameActivity.class));

    }
}
```

Comme dériver plus haut, ce code représente la page principale du jeu, il y a une Image Button(play Now») qu'on récupère grâce à findviewbyid. Puis un OnclickListener pour l'action sur le bouton, et on a utilisé un Intent, pour passer à la classe GameActiivty pour commencer la jouer.

Voici maintenant une partie la classe **GameView**, dedans on voit comment définir une variable de type bitmap, qui sert à dessiner un image,

```
private Bitmap bird[] = new Bitmap[2];
private Bitmap background;

 bird[0] = BitmapFactory.decodeResource( getResources(), R.drawable.cutebird1 );
    bird[1] = BitmapFactory.decodeResource( getResources(), R.drawable.cutebird111);

    background = BitmapFactory.decodeResource( getResources(), R.drawable.bgforest );
```

Là le programme dessine l'image Bird[1] à chaque fois que l'utilisateur appuie sur l'écran, sinon le programme dessine l'image Bird[0]. C'est simplement pour avoir un effet volant ou une animation de **l'oiseau**

```
if (touch) {
            canvas.drawBitmap( bird[1], birdX, birdY, null );
            touch = false;
        } else canvas.drawBitmap( bird[0], birdX, birdY, null );
```

Dans ce morceau ci-dessous, c'est la partie du code, ou la nourriture se déplace. Si l'oiseau obtient la saucisse, le programme redessine là l'image au début. Toutefois si la saucisse sort de l'écran, le programme redessine l'image.

```
// déplacement des saucisses
        foodX= foodX-foodSpeed;

        // Si l'oiseau obtient la saucisse
        if(hitSaucisse( foodX, foodY )){
            scorebird = scorebird+10;
            foodX =-100;
        }


        // Si le saucisse sort de l'écran, alors on lui afffecte une nouvelle valeur de x et y(aléea
        if(foodX<0){
            foodX= birdwidth + 20;
            foodY= (int) Math.floor( Math.random()*(maxpos-initialpos) )+initialpos;

        }
        //puis le programme redessine la saucisse
        canvas.drawBitmap( birdfood, foodX, foodY, null );
```

Le GameOverActivity, on utilise Sharedpreferences pour sauvegarder les **HighScores**

```
 public class GameOver extends AppCompatActivity implements View.OnClickListener {

 private Button playagain;

 @Override
 protected void onCreate(Bundle savedInstanceState) {
     super.onCreate( savedInstanceState );
     setContentView( R.layout.activity_game_over );

     TextView scoreLabel = (TextView) findViewById( R.id.scorelabel );
     TextView HighscoreLabel = (TextView) findViewById( R.id.highscorelabel );


     playagain= (Button) findViewById( R.id.playagainbutton);
     playagain.setOnClickListener( this );


     int Score= getIntent().getIntExtra( "SCORE",0 );
     scoreLabel.setText( Score+ "" );

     SharedPreferences settings = getSharedPreferences( "GAME_DATA", Context.MODE_PRIVATE );
     int Highscore =  settings.getInt( "HIGH_SCORE",0 );

     if(Score>Highscore){
         HighscoreLabel.setText( "High Score : " + Score );

         SharedPreferences.Editor editor = settings.edit();
         editor.putInt( "HIGH_SCORE",Score );
         editor.commit();
     }else{
         HighscoreLabel.setText( "High Score:"+ Highscore );

     }

 }

@Override
 public void onClick(View v) {
     startActivity( new Intent( GameOver.this, GameActivity.class));

 }
}
```

3.2 iOS

Voici la fonction `didMove`, qu'on utilise pour lancer le jeu.

```
override func didMove(to view: SKView) {

        self.anchorPoint=CGPoint(x:0,y:0)
```

```
EcranX = self.size.width

EcranY = self.size.height

physicsWorld.contactDelegate = self as! SKPhysicsContactDelegate


physicsWorld.gravity = grav

self.physicsBody = SKPhysicsBody(edgeLoopFrom: CGRect(x: frame.minX, width: frame.width*2, h

self.physicsBody?.collisionBitMask = sceneCat


let background = SKSpriteNode(imageNamed: "background2")

background.name="background"

background.zPosition = -1

background.anchorPoint = CGPoint(x:0, y:0)

background.position=CGPoint(x:0,y:0)


self.addChild(background)

addBird()

addSaucisse()
```

4 Quelques points délicats/intéressants

Au cours du projet, on ne peut s'empêcher de constater que l'interface Android n'est pas " User-Friendly" et il n'y a pas les mêmes fonctions spécifiques comme sur IOS pour la physique et tout. Toutefois IOS n'étant pas accessible sur l'ordinateur Windows, cela a été compliquer pour développer le projet sur IOS.

5 Conclusion

Ce projet m'a été instructif. Une multitude de recherche devait se faire pour pouvoir réaliser ce projet, car j'ai dû faire beaucoup de recherches, pour pouvoir comprendre et connaître les différentes étapes pour réaliser ce jeu. En tout cas, mon expérience a évolué dans ce domaine, un véritable atout majeur dans mon savoir-faire. Je vois les choses différemment désormais quand je joue à un jeu sur un téléphone ou une tablette.

Finalement, pour moi cela a été un réel plaisir de réaliser ce jeu qui est fondé sur le paysage

de l'île de La Réunion et de pouvoir représenter l'harmonie et la splendeur de cette île par un jeu.