Développement mobile

JAFFUER Pierre - LE LIDEC Tristan

Décembre 2021

Introduction

Contexte :

- UE développement mobile
- Application Android
- Java

Notions utilisées :

- Capteurs (accéléromètre)
- Gestes courants (double tap et balayage vers le haut)
- Appareil photo

Sommaire

Première idée

Un jeu de bataille navale :

- en 3D
- utilisation de AR core
- réalité virtuelle

Première idée

Les + :

- utilisation de beaucoup de capteurs
- gestuelle très utilisée

Les - :

- demande beaucoup de travail
- du temps

Idée réalisée

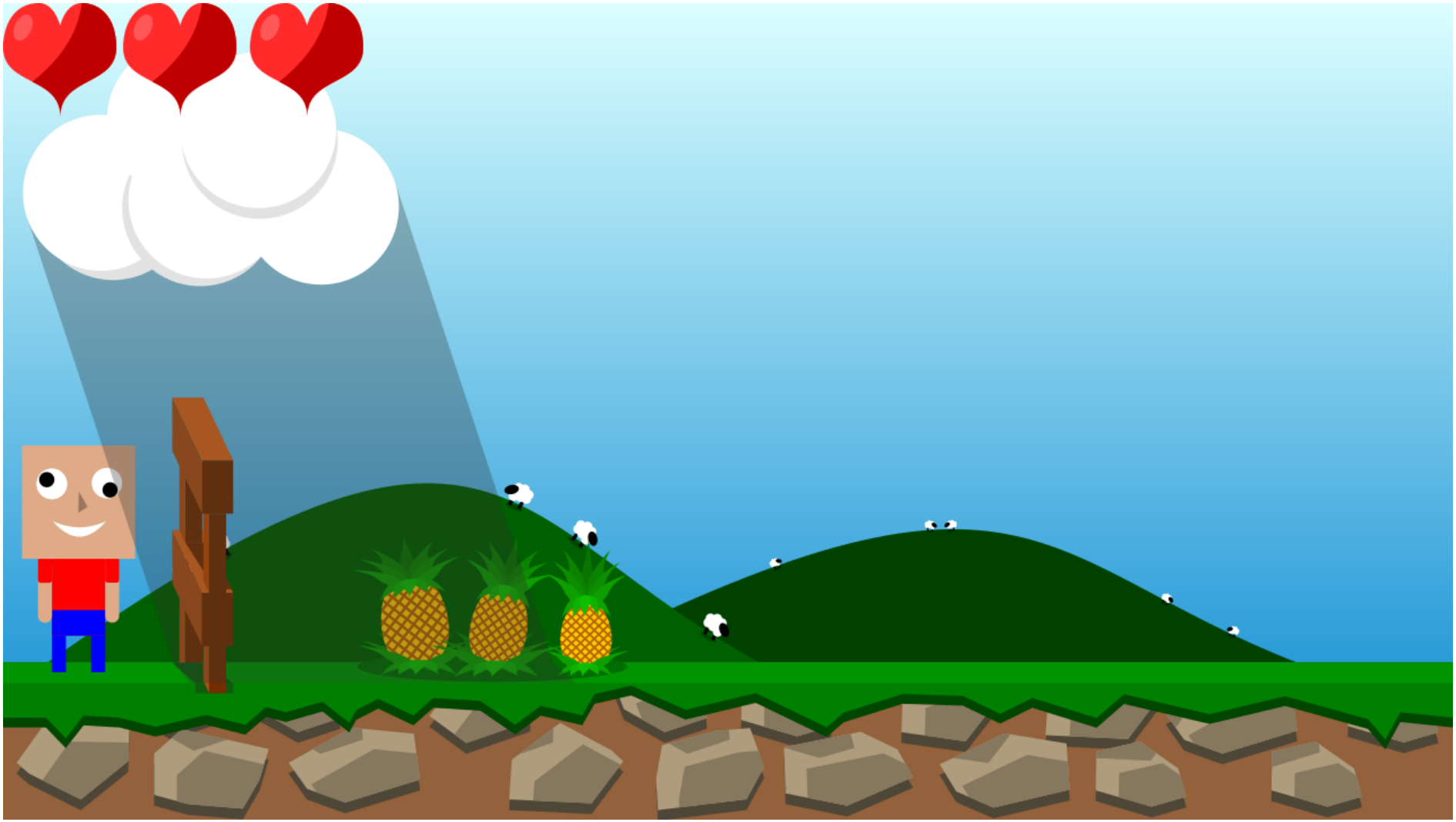


Figure – Jumpy

Idée réalisée

- android vanilla
- MediaPlayer et SoundPool

Logiciels utilisés :

- graphismes : Inkscape
- audio : Audacity, BoscaCeoil, sfxr

Le joueur

Le joueur est divisé en 2 parties :

- La tête (Image personnalisable)
- Le corps

La tête

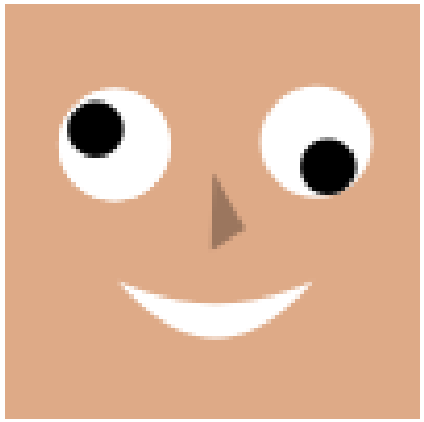


Figure – Tête par défaut du joueur

- Bitmap purement cosmétique
- Remplaçable par un selfie du joueur

Le corps

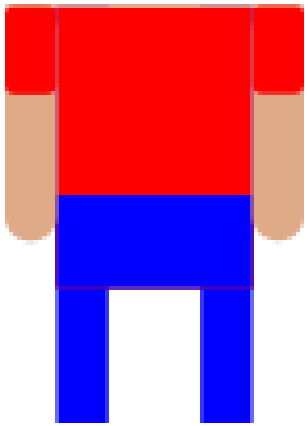


Figure – Corps du joueur

- collisions
- déplacements

Gestion des collisions

Pool d'objets

```
private final ArrayList<Sprite> pool = new ArrayList<>();
```

Figure – Instanciation pool objets

Gestion des collisions

```
public void actualiser(float delta, float vitesse){
    obstacleDistance -= vitesse*delta;

    // Ajout d'un obstacle
    if(obstacleDistance <= 0){
        obstacleDistance = 680 + RANDOM.nextFloat()*600; // distance min = 680n max = 1280
        // Barrière.
        if(RANDOM.nextInt( bound: 2) == 0)
            pool.add(new Sprite( x: 1280+bmpManager.BARRIERE.getWidth()/2.0f,  y: 348+bmpManager.BARRIERE.getHeight()/2.0f, bmpManager.BARRIERE));
        // Ananas
        else
            pool.add(new Sprite( x: 1280+bmpManager.ANANAS.getWidth()/2.0f,  y: 470+bmpManager.ANANAS.getHeight()/2.0f, bmpManager.ANANAS));
    }

    // Déplacement des obstacles
    pool.forEach(o->o.setX(o.getX()-vitesse*delta));

    // On retire les obstalces hors camera
    pool.removeIf(o->o.getX()+o.getWidth() /2.0f <= 0);
}
```

Figure – Actualisation du pool d'objets

Gestion des collisions

```
public boolean collision(Rectangle autre){
    return !(autre.getX()-autre.getWidth()/2 > x+width/2 || autre.getX()+autre.getWidth()/2 < x-width/2 ||
            autre.getY()-autre.getHeight()/2 > y+height/2 || autre.getY()+ autre.getHeight()/2 < y-height/2);
}
```

Figure – Détection de collision

Gestion des collisions

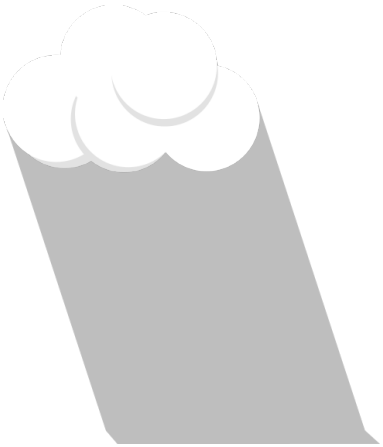
```
public boolean collision(Rectangle autre){
    for(Sprite s : pool) {
        if (s.collision(autre))
            return true;
    }
    return false;
}
```

Figure – Collision entre des sprites

Le dessin

Utilisation d'un canvas pour tout le jeu

- surfaceView
- canvas GPU (Android 8)
- Thread d'actualisation



(a) Nuage

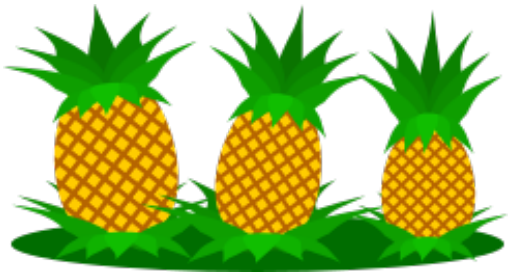
(b) ciel

(c) collines

(d) sol

Figure – Nos assets

Les obstacles



(a) Ananas

(b)
Barriere

Figure – Obstacles

La gestuelle

- double tap
- glissement vers le haut

Double tap

```java
// classe interne pour capturer l'instance de la la variable "game"
class MyGestureListener extends GestureDetector.SimpleOnGestureListener {

    @Override
    public boolean onDoubleTap(MotionEvent e) {
        game.rejouer();
        return true;
    }

    @Override
    public boolean onFling(MotionEvent event1, MotionEvent event2, float velocityX, float velocityY) {
        // detection d'un mouvement vert le haut

        // Vecteur directeur du mouvement
        double dx = event2.getX()-event1.getX();
        double dy = event2.getY()-event1.getY();
        double d = Math.sqrt(dx*dx+dy*dy);
        // Vecteur directeur normalisé
        double nx = dx / d;
        double ny = dy / d;

        // pointe vers le haut (repère de l'écran) et dans un conne de 45°
        // ie |nx| < sqrt(2)/2
        if(ny < 0 && Math.abs(nx) <= 0.7071067812){
            // On indique au jeu que l'on souhaite sauter
            game.sauter();
        }
        return true;
    }
}
```

Figure – Classe fille SimpleOnGestureListener

Glissement vers le haut

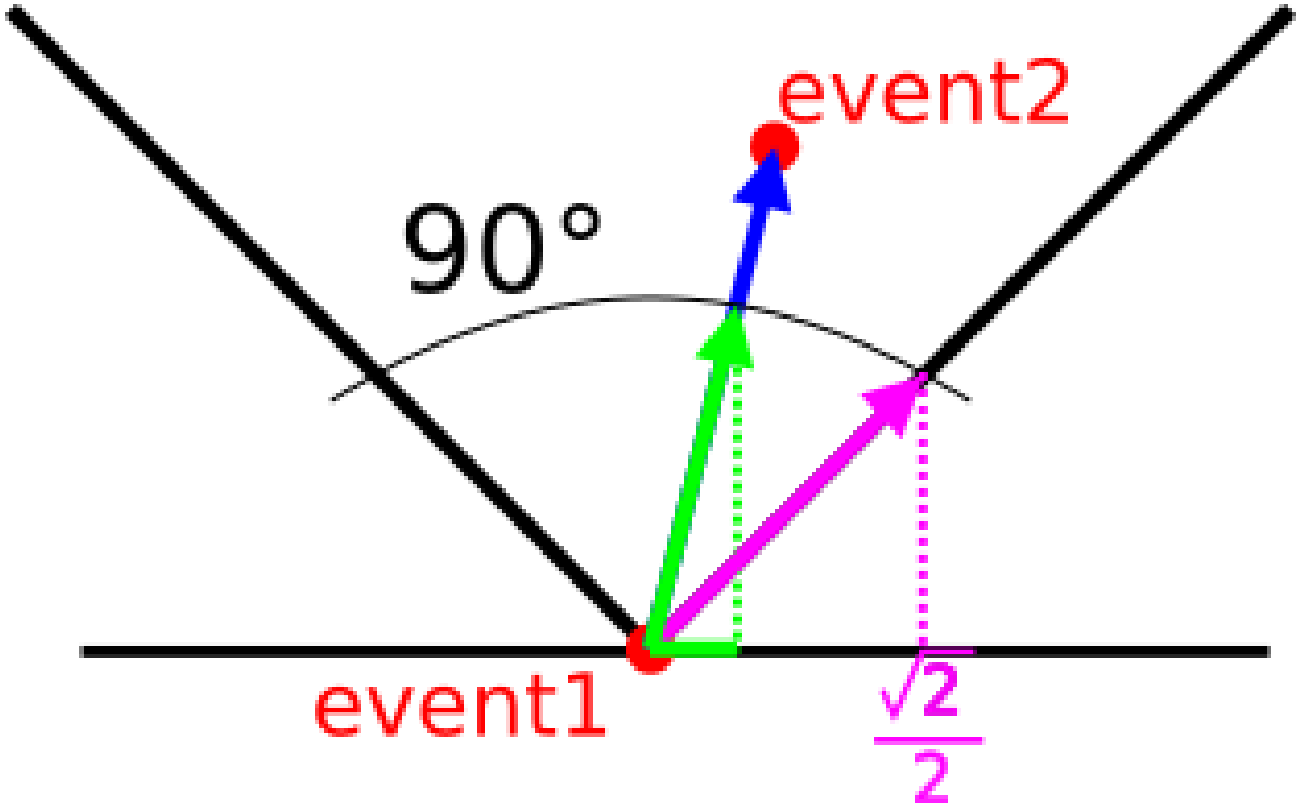


Figure – Détection d'un balayage vers le haut

Les capteurs

- Accéléromètre
- appareil photo

Accéléromètre

```
protected void onResume() {
    super.onResume();
    sensorManager.registerListener(game, sensor, SensorManager.SENSOR_DELAY_GAME);
}
```

Figure – Accéléromètre

Appareil photo

Vérification des permissions impérative.

```
photo.setOnClickListener(view -> {
    if (checkSelfPermission(Manifest.permission.CAMERA) != PackageManager.PERMISSION_GRANTED) {
        requestPermissions(new String[]{Manifest.permission.CAMERA}, MY_CAMERA_REQUEST_CODE);
    } else {
        prendrePhoto();
    }
});
```

Figure – Bouton pour prendre une photo

Appareil photo

Invocation à une application de caméra déjà installée.

```
public void prendrePhoto(){
    Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    // Il y a une application pour prendre une photo
    if (intent.resolveActivity(getPackageManager()) != null) {
        someActivityResultLauncher.launch(intent);
    } else {
        Toast.makeText( context: SkinActivity.this,  text: "Aucunne application pour effectuer la photo.", Toast.LENGTH_LONG).show();
    }
}
```

Figure – Caption

Interface

Traduction anglaise et française de l'UI.

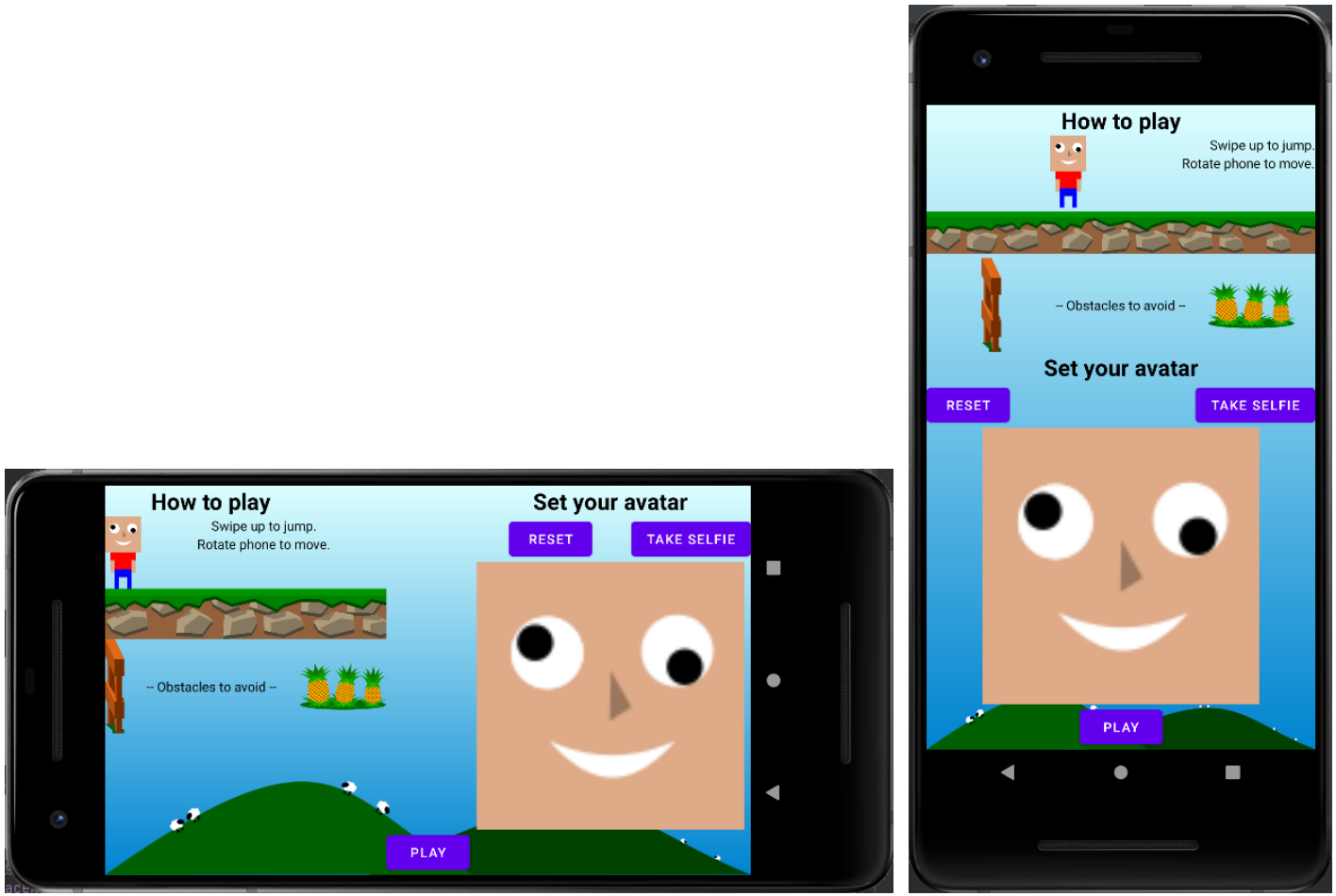


Figure – Adaptation à l'orientation

Conclusion

- Utilisation de capteurs
- simple mais fonctionnel
- Gestion des compromis

démonstration