

- Project report -
**An application for mobile devices – Thief and
Policemen**

Université Cergy Pontoise

Authors : Bah Amadou Madjou
Phetramphand Antoine

Contents

I. Presentation of project	3
1. Description	3
2. Characters in the game	3
3. Software used.....	3
II. Team members.....	4
III. Activities	5
1. Acitivity "MainMenu"	5
2. Acitivity "GameService"	5
3. Acitivity "MySensor"	5
IV. Intents	6
V. Background Services / Threads	7
VI. Sensors Used	8
VII. User guide	9



I. Presentation of project

1. Description

The application that we created called “***Thief and Policemen***”. It’s a simple video game which involves guiding an avatar to jump in order to avoid enemies to advance the game. Here, we can control the ***Thief*** as the main character. The enemies are the ***Policemen***.

2. Characters in the game

The characters are:

-  The main character (the thief)
-  The IA (the policemen)

3. Software used

We used **Android Studio** to develop this game.
To emulate this application, we used **Nexus S API 10**.

II. Team members

Our team :

- **Bah Amadou Madjou**(e-mail : amsbert1@gmail.com)
- **Phetramphand Antoine**(e-mail : Antoine.phetramphand@gmail.com)

III. Activities

1. Acitivity “MainMenu”

It's the main Activity which contains the others activities and permits to execute them thanks to the method *launch()*.

It displays also a menu which allows to choose a mode.

2. Acitivity “GameService”

It's the Activity which allows to start the main game by using the class **GamePanel**.

```
public class GameService extends Activity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        /*setContentView(R.layout.second_layout);*/  
  
        //turn title off  
        requestWindowFeature(Window.FEATURE_NO_TITLE);  
  
        //set to full screen  
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN, WindowManager.LayoutParams.FLAG_FULLSCREEN);  
  
        setContentView(new GamePanel(this));  
    }  
}
```

3. Acitivity “MySensor”

It's the Activity which triggers the Sensor with type **Accelerometer**.

IV. Intents

The intents are used in the Activity **"MainMenu"**

```
public void launch(View view) {

    /*String message = "Welcome to my app";
    textView.setText(message);*/

    String button_text;
    button_text = ((Button)view).getText().toString();
    if(button_text.equals("Start the game")){

        Intent intent = new Intent(this, MyService.class);
        startActivity(intent);

    }
    else if(button_text.equals("Stop the game")){

        Intent intent = new Intent(this, MyService.class);
        startActivity(intent);

    }
    else if(button_text.equals("Use the Sensor")){

        Intent intent = new Intent(this, MySensor.class);
        startActivity(intent);

    }

    public void createMethod(View v) {

        Intent i = new Intent(this, MyService.class);
        startService(i);

    }

    public void startMethod(View v) {

        Intent intent = new Intent(this, GameService.class);
        startActivity(intent);

    }

    public void stopMethod(View v) {

        Intent i = new Intent(this, MyService.class);
        stopService(i);

    }

}
```

V. Background Services / Threads

```
public class MyService extends Service {  
  
    @Override  
    public IBinder onBind(Intent intent) { return null; }  
  
    @Override  
    public void onCreate() {  
        super.onCreate();  
        Toast.makeText(this, "The game is created", Toast.LENGTH_LONG).show();  
        Intent intent = new Intent(this, GameService.class);  
        /*startActivity(intent);*/  
    }  
  
    @Override  
    public int onStartCommand(Intent intent, int flags, int startId) {  
        Toast.makeText(this, "The game is started", Toast.LENGTH_LONG).show();  
        //stopSelf();  
  
        return super.onStartCommand(intent, flags, startId);  
    }  
  
    @Override  
    public void onDestroy() {  
        super.onDestroy();  
        Toast.makeText(this, "The game is stopped", Toast.LENGTH_LONG).show();  
        // stopSelf();  
    }  
}
```

Here, this class enables to create or stop the game and sends messages to user if it succeeds.

VI. Sensors Used

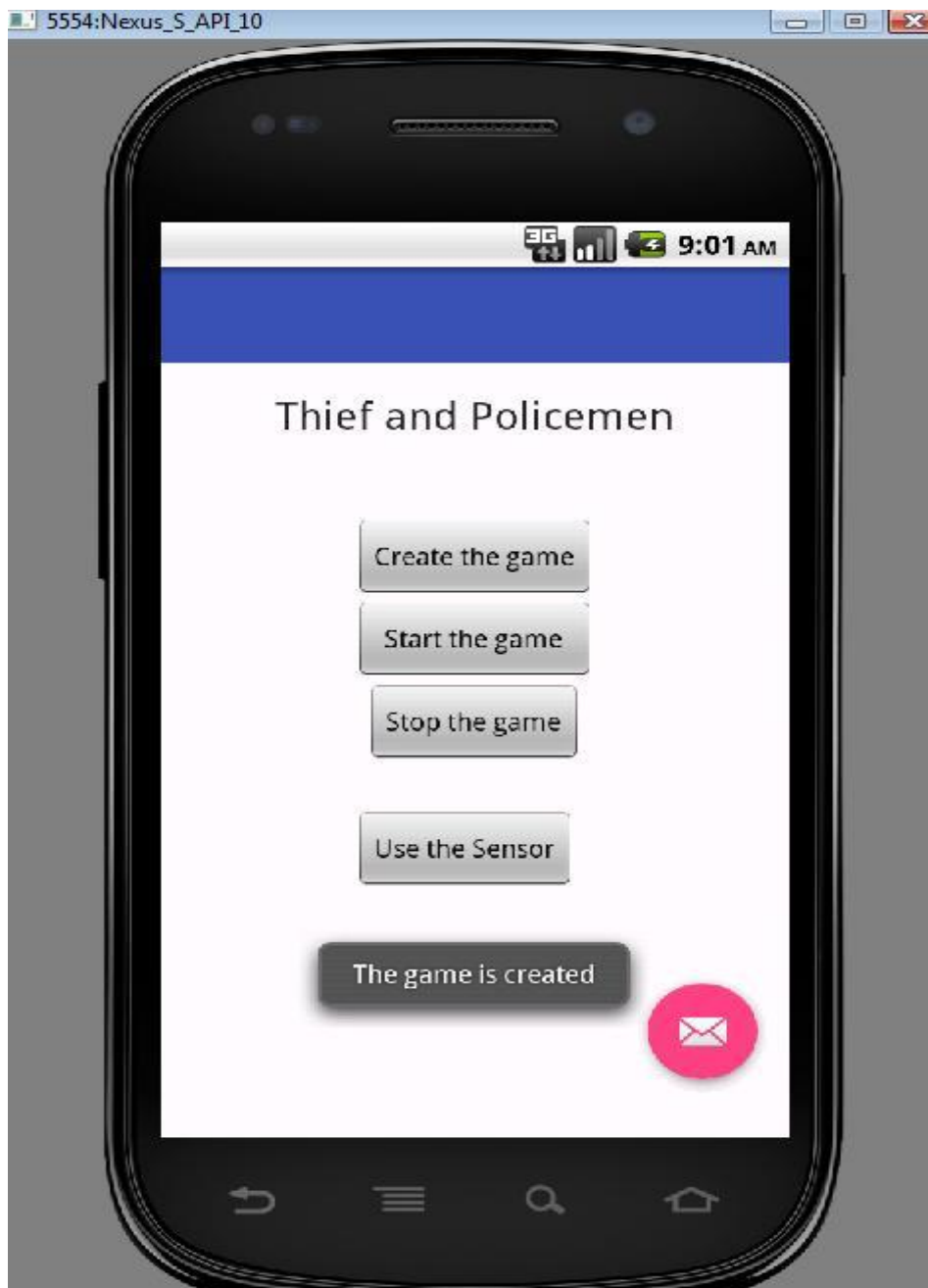
```
public class MySensor extends Activity implements SensorEventListener {  
  
    private TextView displayReading;  
    private Sensor sensor;  
    private SensorManager sm;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.third_layout);  
  
        sm=(SensorManager) getSystemService(SENSOR_SERVICE);  
        sensor = sm.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);  
        sm.registerListener(this, sensor, SensorManager.SENSOR_DELAY_NORMAL);  
        displayReading=(TextView) findViewById(R.id.display_reading);  
    }  
}
```

It uses the type **Accelerometer**.

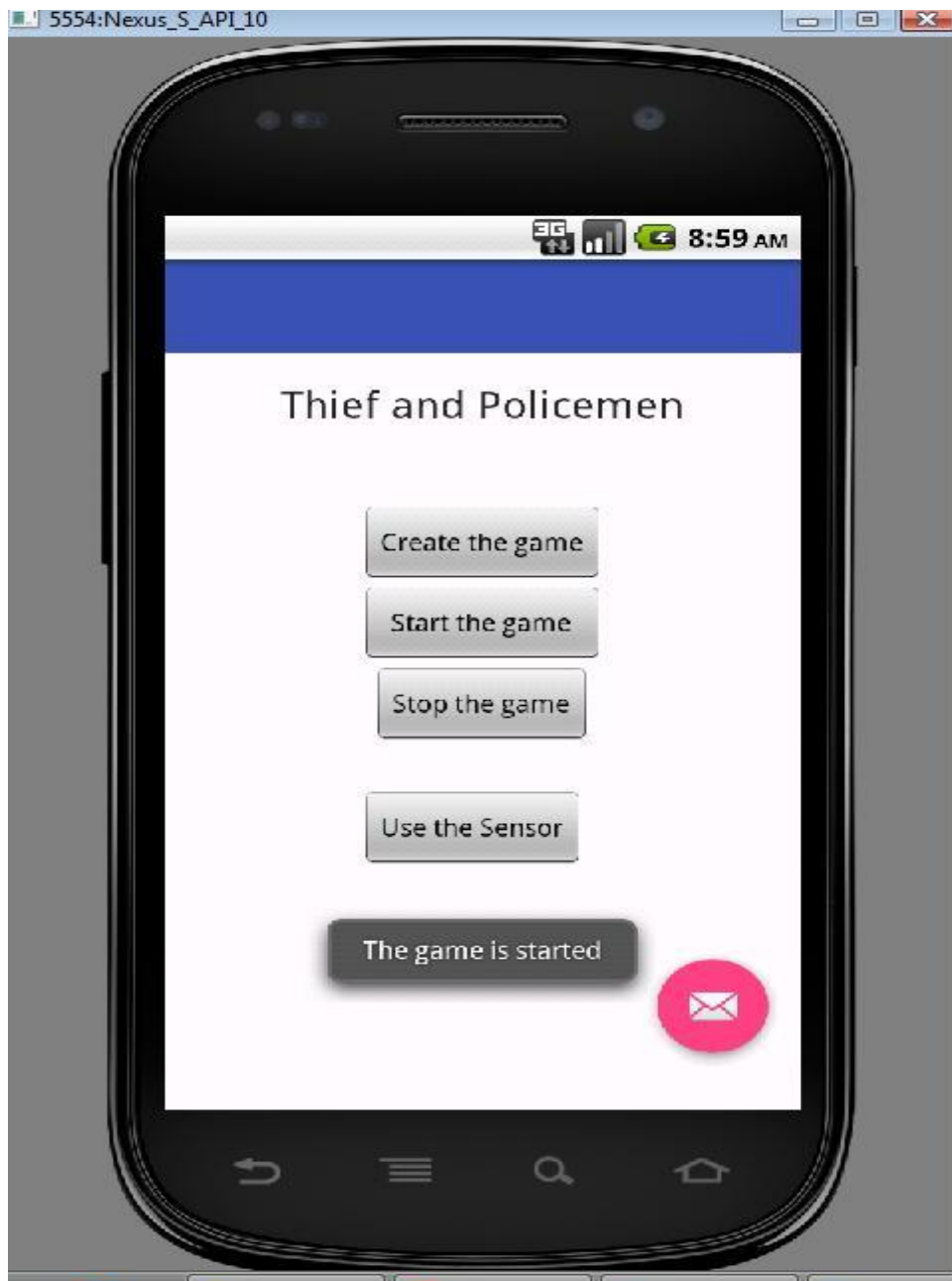
VII. User guide



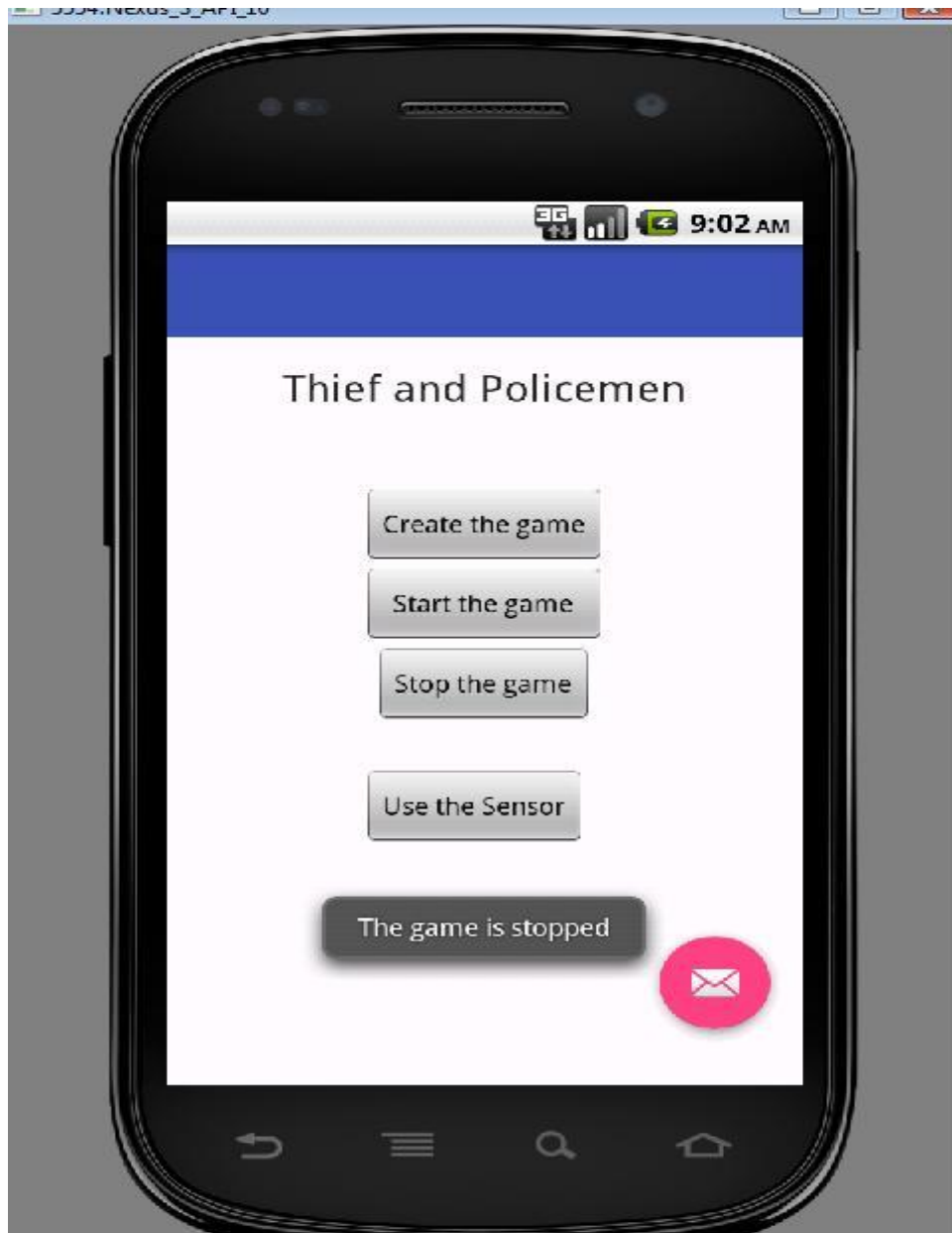
Here, this is the main menu when we execute the program.



To call the service, you press the button « **Create the game** » and show this message "**The game is created**".



It starts the game and click on "***Start the game***" to begin.



If you want to stop the service, click on « ***Stop the game*** ».



When you click on « *Use the sensor* », it displays an other window.



This is main game. Before to start, it's much pleasant to see the application by using **Ctrl+F11**. It enables the mode *Landscape*.

This window displays the commands of the game. For going up with the character, you need to press once or twice left button of the mouse and then hold the button. If you want to go down, you just release the button. The word “**distance**” means the distance you covered during the game.



When the game starts, enemies (policemen) appear on the screen and they go towards the left. You have to avoid them. If one of them hit you, the game is over.