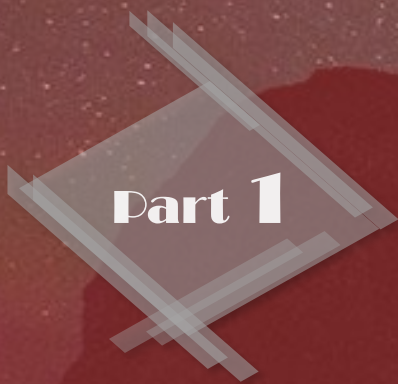


The background features a night sky with a starry field and a faint Milky Way galaxy. In the foreground, there are dark silhouettes of mountains. Overlaid on the scene are several large, semi-transparent geometric shapes: a red triangle pointing down, a green triangle pointing up, and several grey lines and rectangles forming a complex, abstract pattern. The text 'PROJECT Gravity Snake' is centered in white.

# PROJECT | Gravity Snake

By: SSR Team  
Qingshan Yao  
Shufan Huang  
Xiangyu Lin  
Feng Chang

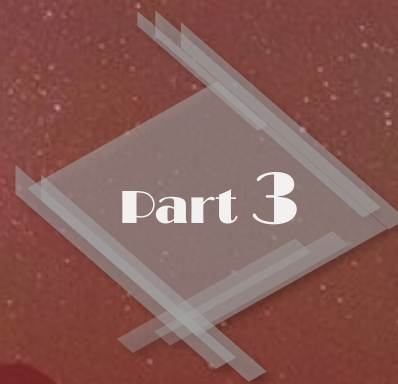
# Contents



Overview



Utilization of  
acceleration  
sensor



User interface



Key algorithms

A night sky with the Milky Way galaxy visible in the upper center. The sky is dark with numerous stars. In the foreground, there are dark silhouettes of rocks or hills. A blue light source is visible on the horizon, casting a glow. A semi-transparent rectangular frame is overlaid on the image, containing the text.

**Part 1**

# Overview



# The motivation

- New play of classic games.
- More interesting and challenging.
- Enlightenment of Flappy Bird.
- Broad development space.

# division of labor

Qingshan  
Yao

Xiangyu  
Lin

Shufan  
Huang

Feng  
Chang

Team leader

Team member

Team member

Team member

Utilization of acceleration  
sensor  
Connect some parts

User interface

Key algorithms

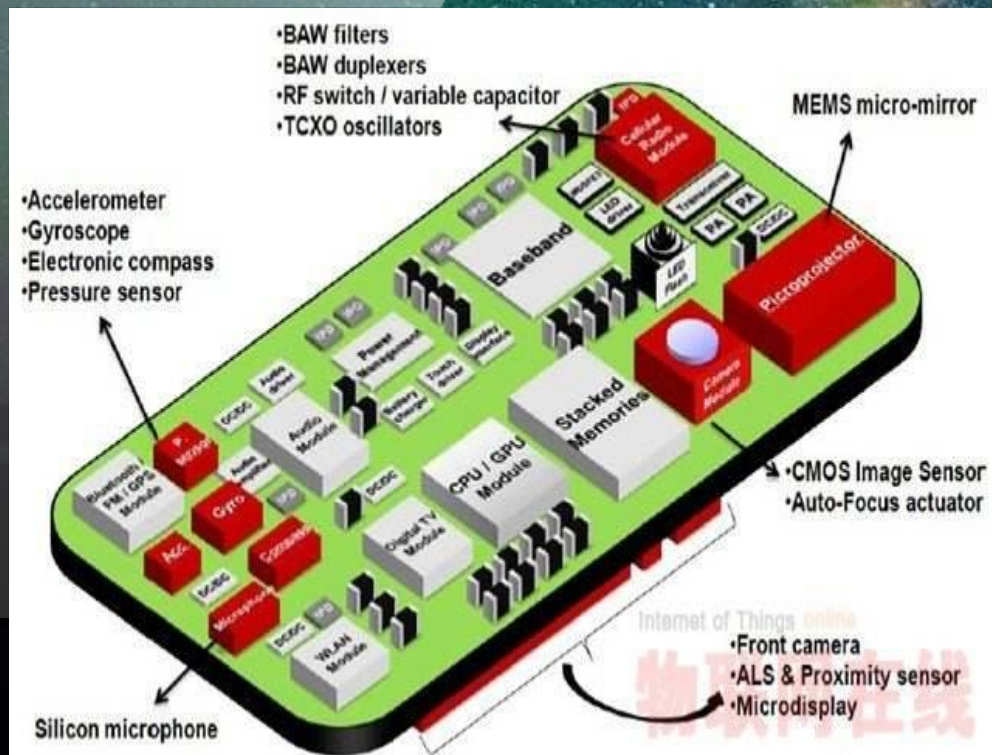
Art designer  
algorithms

The background of the slide is a night sky featuring the Milky Way galaxy, which appears as a bright, multi-colored band of stars stretching across the upper portion of the frame. The sky is filled with numerous smaller stars. In the foreground, the dark silhouettes of mountains or large rocks are visible against the starry background. A semi-transparent, multi-layered rectangular frame is centered on the slide, containing the text. A small, semi-transparent green circle is located at the bottom right corner of the frame.

**Part 2**  
Utilization of  
acceleration sensor

```
getService(SENSOR_SERVICE);
getService(SENSOR_SERVICE);
getService(SENSOR_SERVICE);
getService(SENSOR_SERVICE);
getDefaultSensor(Sensor.TYPE_GRAVITY);
```

# sensors



```
getService(SENSOR_SERVICE);
```

```
getDefaultSensor(Sensor.TYPE_GRAVITY);
```

```
SensorEventListener
```

```
onSensorChanged, onAccuracyChanged
```

# Utilization

obtain

manage

debug

obtain

Use tools above to get the measurable statics

manage

Manage the threshold values and other conditions

debug

According to the main function, adjust the values





Part 3  
User Interface

How to unite all the things beautifully and logically?

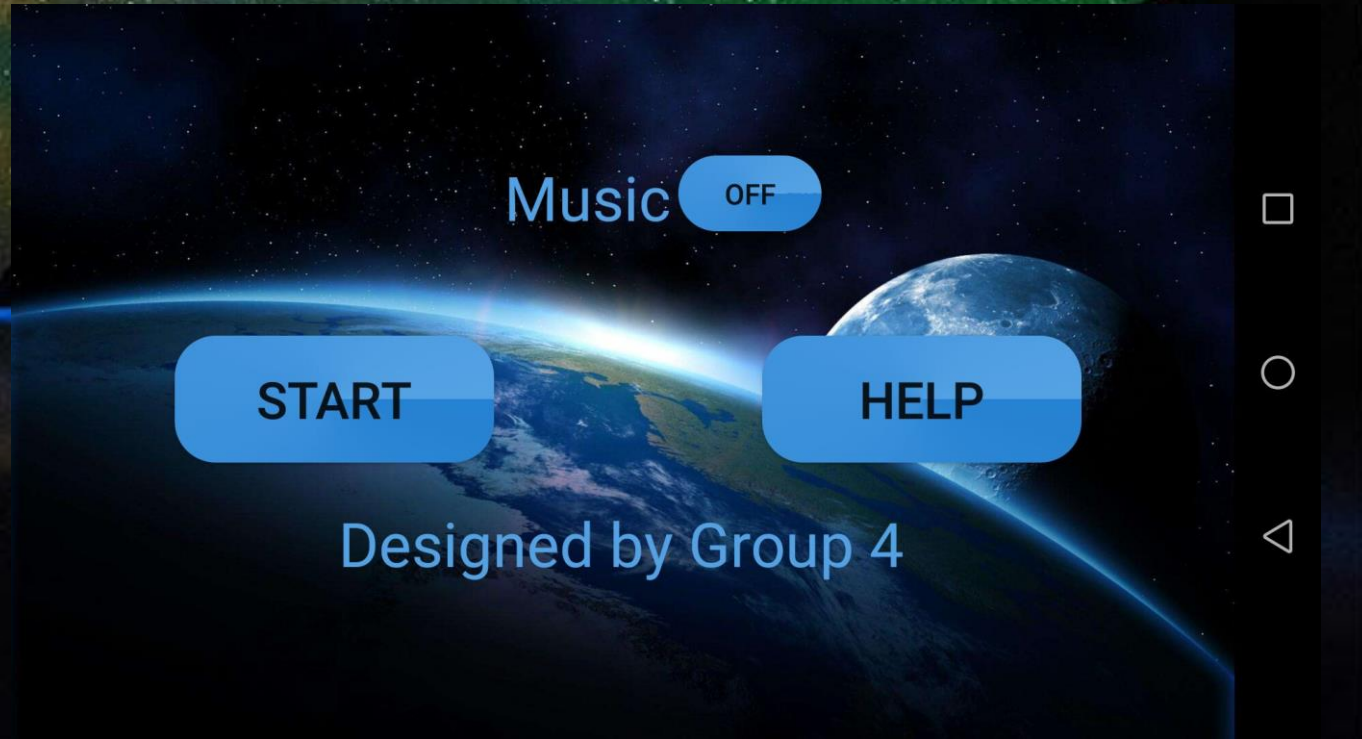
Music

Picture

Background

Snake

Button



How to use codes to create more beautiful things?

Artist?



How to arrange the beautiful pictures and buttons with the most rational codes?



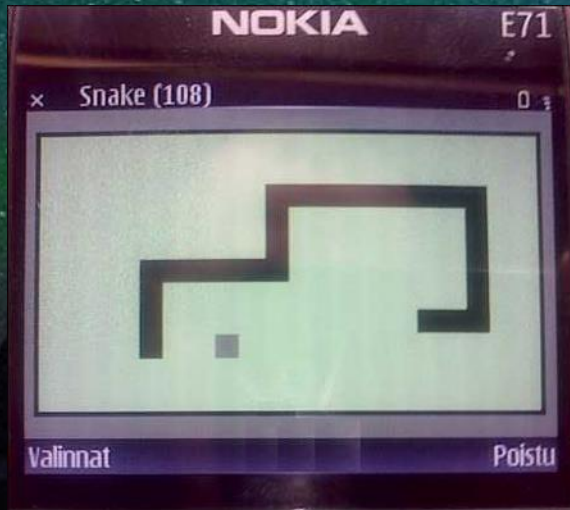
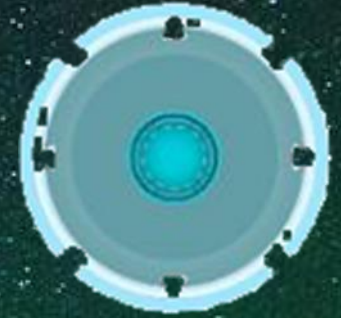
Programmer?



A night sky with the Milky Way galaxy visible in the upper center. The sky is dark with numerous stars. In the foreground, there are dark silhouettes of rocks or hills. A semi-transparent rectangular frame with a double border is centered on the slide. The text "Part 4" is written in a white, sans-serif font inside the frame.

Part 4

# Key algorithms



IDEA

More advanced sensor → Evolution of games



# SnakeView

- m 🔒 SnakeView(Context, AttributeSet)
- m 🔒 SnakeView(Context, AttributeSet, int)
- m 🔒 initSnakeView(): void
- m 🔒 initNewGame(): void
- m 🔒 coordArrayListToArray(ArrayList<Coordinate>): int[]
- m 🔒 saveState(): Bundle
- m 🔒 coordArrayToArrayList(int[]): ArrayList<Coordinate>
- m 🔒 restoreState(Bundle): void
- m 🔒 setTextView(TextView): void
- m 🔒 setStartButton(Button): void
- m 🔒 setMode(int): void
- m 🔒 addRandomApple(): void
- m 🔒 update(): void
- m 🔒 updateWalls(): void
- m 🔒 updateApples(): void
- m 🔒 updateSnake(): void
- m 🔒 onClick(View): void
- m 🔒 onAccuracyChanged(Sensor, int): void ↑SensorEventListener
- m 🔒 onSensorChanged(SensorEvent): void ↑SensorEventListener

# Important Variables

f 🔒 mMoveDelay: long = 600

f 🔒 mDirection: int = NORTH

f 🔒 mNextDirection: int = NORTH

f 🔒 mScore: long = 0

```
mDirection = mNextDirection;

switch (mDirection) {
    case EAST: {
        newHead = new Coordinate(head.x + 1, head.y);
        break;
    }
    case WEST: {
        newHead = new Coordinate(head.x - 1, head.y);
        break;
    }
    case NORTH: {
        newHead = new Coordinate(head.x, head.y - 1);
        break;
    }
    case SOUTH: {
        newHead = new Coordinate(head.x, head.y + 1);
        break;
    }
}
```

```
int applecount = mAppleList.size();
for (int appleindex = 0; appleindex < applecount; appleindex++) {
    Coordinate c = mAppleList.get(appleindex);
    if (c.equals(newHead)) {
        mAppleList.remove(c);
        addRandomApple();

        mScore++;

        growSnake = true;
    }
}
```

```
getService(SENSOR_SERVICE);
getService(SENSOR_SERVICE);
getService(SENSOR_SERVICE);
getService(SENSOR_SERVICE);
```

# tileview

```
public class TileView extends View {
    protected static int mTileSize;

    protected static int mXTileCount;
    protected static int mYTileCount;

    private static int mXOffset;
    private static int mYOffset;
```

```
private Bitmap[] mTileArray;
```

```
private int[][] mTileGrid;
```

```
public TileView(Context context, AttributeSet attrs, int defStyle)
{
    super(context, attrs, defStyle);
    TypedArray a = context.obtainStyledAttributes(attrs,
        R.styleable.TileView);

    mTileSize = a.getInt(R.styleable.TileView_tileSize, 50);

    a.recycle();
}
```

```
@Override
public void onDraw(Canvas canvas) {
    super.onDraw(canvas);
    for (int x = 0; x < mXTileCount; x += 1) {
        for (int y = 0; y < mYTileCount; y += 1) {
            if (mTileGrid[x][y] > 0) {
                canvas.drawBitmap(mTileArray[mTileGrid[x]
                [y]],

                mXOffset + x * mTileSize,
                mYOffset + y * mTileSize,
                mPaint);
            }
        }
    }
}
```

A two-dimensional array of integers

obtain the new attribute value

Draw the canvas onto the mobile phone





Thanks