一張含有 文字, 飛機, 行動電話, 螢幕擷取畫面 的圖片

自動產生的描述

ITP4721 Mobile Applications and Game Technology

Leung Tung Lam(220280643)

Contents

[Game Title 1](#_Toc165224278)

[Game Type 1](#_Toc165224279)

[Game story 1](#_Toc165224280)

[Game Play 1](#_Toc165224281)

[Mobile Game Multimedia Design 2](#_Toc165224282)

[Technical report - Mobile Game Program Design 2](#_Toc165224283)

## Demo

<https://youtu.be/HaajW0Z9Z3Y>

## Game Title

Falcon Strike

## Game Type

Flying Shooting

## Game story

In 20XX, an evil organization called the Black Order, almost occupied the whole world and planned to build a huge air battleship to help them conquer the world. The Coalition Government sent pilots to stop their plan. You play as pilots to fly fighter jet and defeat that airship.

## Game Play

The player's spaceship can be moved by dragging it on the screen. The spaceship automatically fires bullets when it's being dragged.

There are three types of enemy spaceships (blue, green, and red), each with different speeds and behaviors. The enemy spaceships appear from the top of the screen and move downwards.

The game has multiple levels. The level increases when the player reaches a certain score. As the level increases, the enemy spaceships become faster and more numerous.

The player's score increases by 10 points for each enemy spaceship shot down.

The player has 3 lives. A life is lost when the player's spaceship collides with an enemy spaceship. The game is over when all lives are lost.

The game is won when the player reaches a score of 300.

## Mobile Game Multimedia Design

The game uses bitmap graphics for the player's spaceship, the enemy spaceships, and the bullets. The graphics are drawn on a `Canvas` object using the `drawBitmap` method.

The game has a background image that scrolls vertically to give the illusion of movement.

The game uses sound effects for shooting and explosions. The sound effects are played using a `SoundPool` object.

## Mobile Game Program Design

Game Initialization: The game is initialized in the `start` method of the `GamePanel` class. This method sets up the player's spaceship, starts the game loop, and starts the enemy spawning task.

Game Update and Render Game Cycle: The game update and render cycle is handled in the `update` and `render` methods of the `GamePanel` class. The `update` method updates the game state (e.g., moving the spaceships and bullets, checking for collisions), and the `render` method draws the game state on the screen.

2D Graphics on Canvas and Draw: The game uses 2D bitmap graphics drawn on a `Canvas` object. The `Canvas` object is obtained from a `SurfaceView` object, which provides the game's drawing surface.

Managing Game Data: The game data (e.g., the player's spaceship, the enemy spaceships, the bullets) is managed in the `GamePanel` class. The game objects are stored in `CopyOnWriteArrayList` collections, which allow for safe iteration even when the collections are modified concurrently.

Handling sizes of Android devices, and Portrait and Landscape Modes: The game is designed to be played in portrait mode. The game scales the graphics based on the display density of the device, so the game looks consistent on devices with different screen sizes and resolutions.

Boundary Checking and Virtual World: The game checks for boundary collisions in the `update` method of the `GamePanel` class. When a spaceship or bullet moves outside the screen, it's removed from the game.

Drawing and Game Messages: The game draws text messages (e.g., the score, the "Game Over" message) on the screen using the `drawMultilineText` method of the `GamePanel` class. The text is drawn on the `Canvas` object using a `Paint` object.