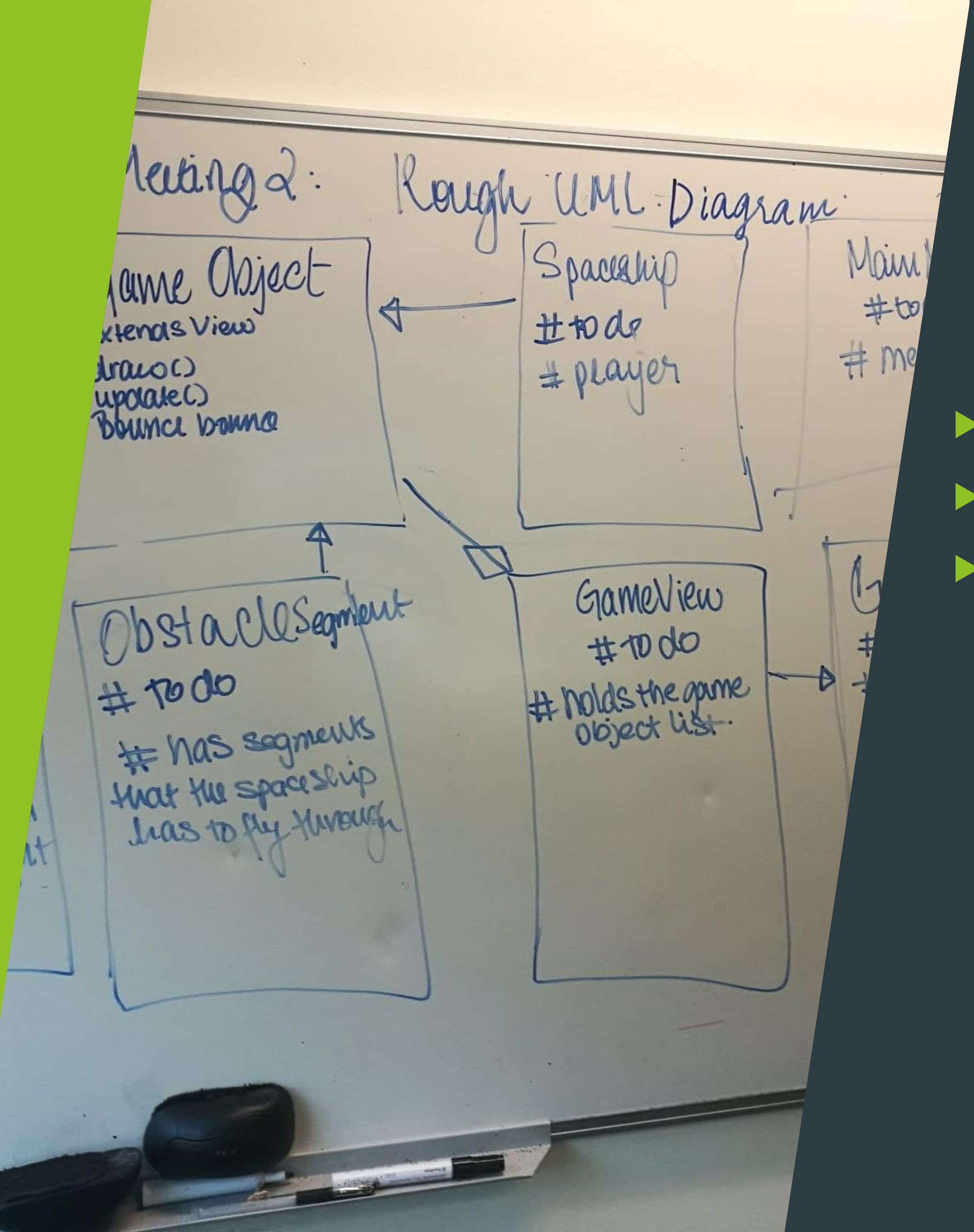
Retro Game Demonstration

By Jasper McNiven, Kriti Tripathi, Septian Razi and Yatindra Tripathi

Introduction

- Space Crusaders
- Inspired by Flappy Birds
- Theme – space
- Simple implementation allowed for additional features
- Designed our own obstacles and character
- Unique features



Our Group

- Weekly meetings
- 5 accounted minuted meetings
- Initially divided into 3 parts – UI, assets, logic (collision) Video recorded progress

Overview

- The aim - jump through/avoid obstacles
- Collect coins to gain additional points
- Speed increases as game progresses
- Indefinite game
- End of game – when player crashes/collide

Demonstration

Design Approach

Assignment separated into four parts:

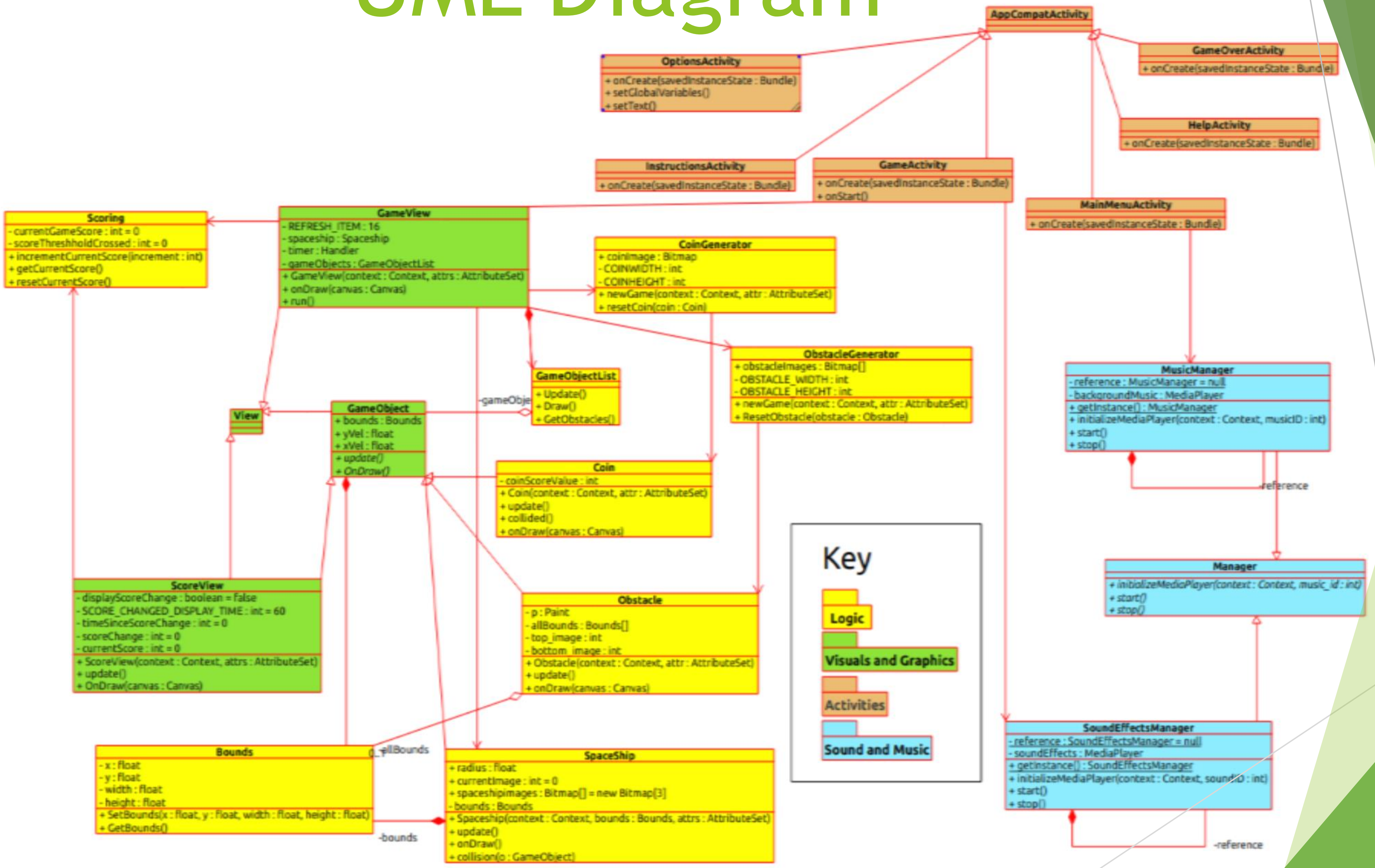
- Activities
- Logic
- Sound and Music
- Visual and Graphics

Object oriented design approach

Base class – GameObject extended by all other classes

Attempt at an observer design pattern

UML Diagram



AppCompatActivity

OptionsActivity
+ onCreate(savedInstanceState : Bundle)
+ setGlobalVariables()
+ setText()

GameOverActivity
+ onCreate(savedInstanceState : Bundle)

HelpActivity
+ onCreate(savedInstanceState : Bundle)

InstructionsActivity
+ onCreate(savedInstanceState : Bundle)

GameActivity
+ onCreate(savedInstanceState : Bundle)
+ onStart()

MainMenuActivity
+ onCreate(savedInstanceState : Bundle)

Scoring
- currentGameScore : int = 0
- scoreThreshholdCrossed : int = 0
+ incrementCurrentScore(increment : int)
+ getCurrentScore()
+ resetCurrentScore()

GameView
- REFRESH_ITEM : 16
- spaceship : Spaceship
- timer : Handler
- gameObjects : GameObjectList
+ GameView(context : Context, attrs : AttributeSet)
+ onDraw(canvas : Canvas)
+ run()

CoinGenerator
+ coinImage : Bitmap
- COINWIDTH : int
- COINHEIGHT : int
+ newGame(context : Context, attr : AttributeSet)
+ resetCoin(coin : Coin)

GameObjectList
+ Update()
+ Draw()
+ GetObstacles()

ObstacleGenerator
+ obstacleImages : Bitmap[]
- OBSTACLE_WIDTH : int
- OBSTACLE_HEIGHT : int
+ newGame(context : Context, attr : AttributeSet)
+ ResetObstacle(obstacle : Obstacle)

MusicManager
- reference : MusicManager = null
- backgroundMusic : MediaPlayer
+ getInstance() : MusicManager
+ initializeMediaPlayer(context : Context, musicID : int)
+ start()
+ stop()

View

GameObject
+ bounds : Bounds
+ yVel : float
+ xVel : float
+ *update()*
+ *OnDraw()*

Coin
- coinScoreValue : int
+ Coin(context : Context, attr : AttributeSet)
+ update()
+ collided()
+ onDraw(canvas : Canvas)

Manager
+ *initializeMedioPlayer(context : Context, music_id : int)*
+ *start()*
+ *stop()*

ScoreView
- displayScoreChange : boolean = false
- SCORE_CHANGED_DISPLAY_TIME : int = 60
- timeSinceScoreChange : int = 0
- scoreChange : int = 0
- currentScore : int = 0
+ ScoreView(context : Context, attrs : AttributeSet)
+ update()
+ OnDraw(canvas : Canvas)

Obstacle
- p : Paint
- allBounds : Bounds[]
- top_image : int
- bottom_image : int
+ Obstacle(context : Context, attr : AttributeSet)
+ update()
+ onDraw(canvas : Canvas)

Bounds
- x : float
- y : float
- width : float
- height : float
+ SetBounds(x : float, y : float, width : float, height : float)
+ GetBounds()

SpaceShip
+ radius : float
+ currentImage : int = 0
+ spaceshipimages : Bitmap[] = new Bitmap[3]
- bounds : Bounds
+ Spaceship(context : Context, bounds : Bounds, attrs : AttributeSet)
+ update()
+ onDraw()
+ collision(o : GameObject)

SoundEffectsManager
- reference : SoundEffectsManager = null
- soundEffects : MediaPlayer
+ getInstance() : SoundEffectsManager
+ initializeMediaPlayer(context : Context, soundID : int)
+ start()
+ stop()

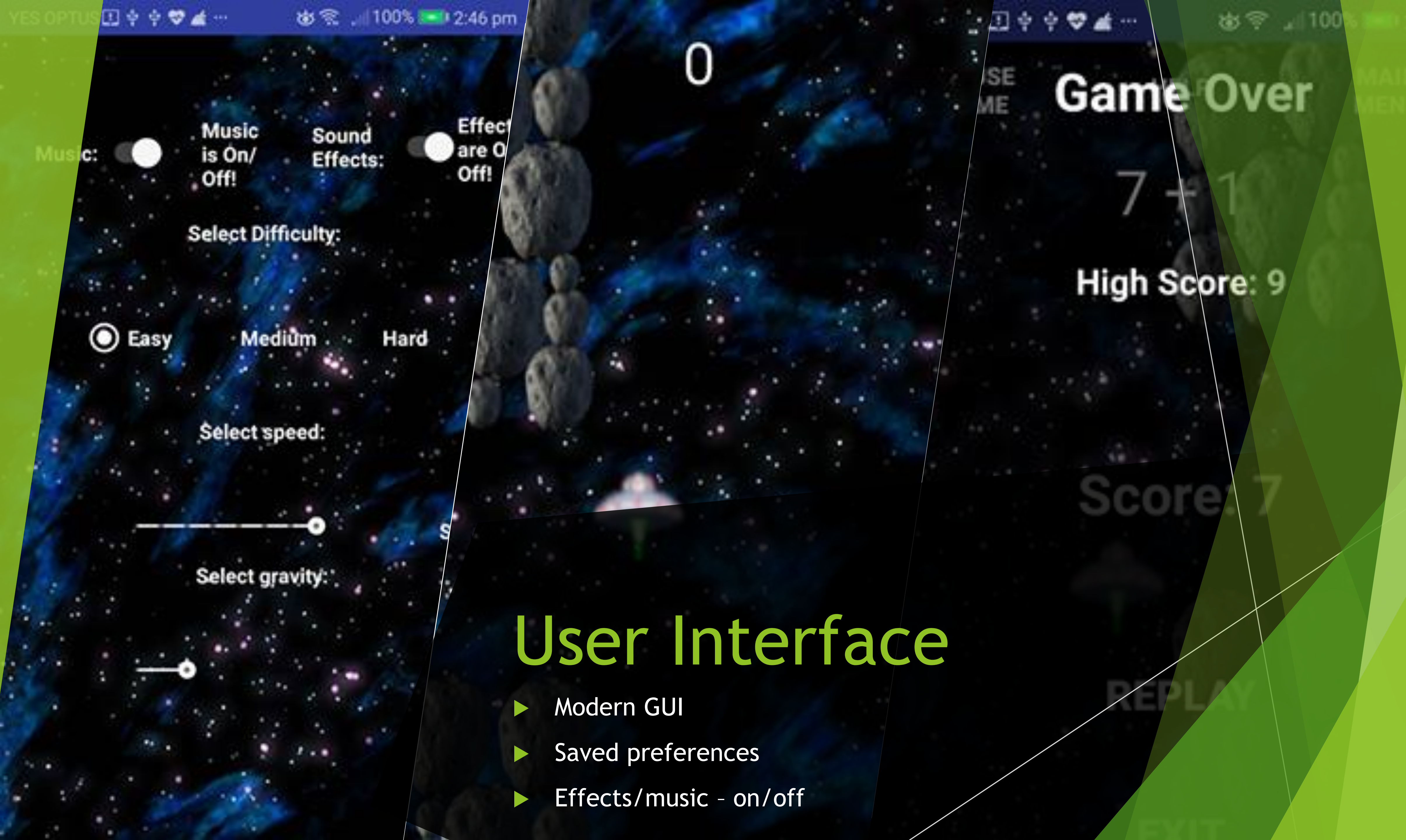
Key

🟨 Logic

🟩 Visuals and Graphics

🟧 Activities

🟦 Sound and Music



Music: Music is On/ Off!

Sound Effects: Effect are O Off!

Select Difficulty:

◉ Easy Medium Hard

Select speed:

Select gravity:

Game Over

7 + 1

High Score: 9

Score: 7

User Interface

- Modern GUI
- Saved preferences
- Effects/music – on/off

Logic

Modules implemented individually

Brought together in the end

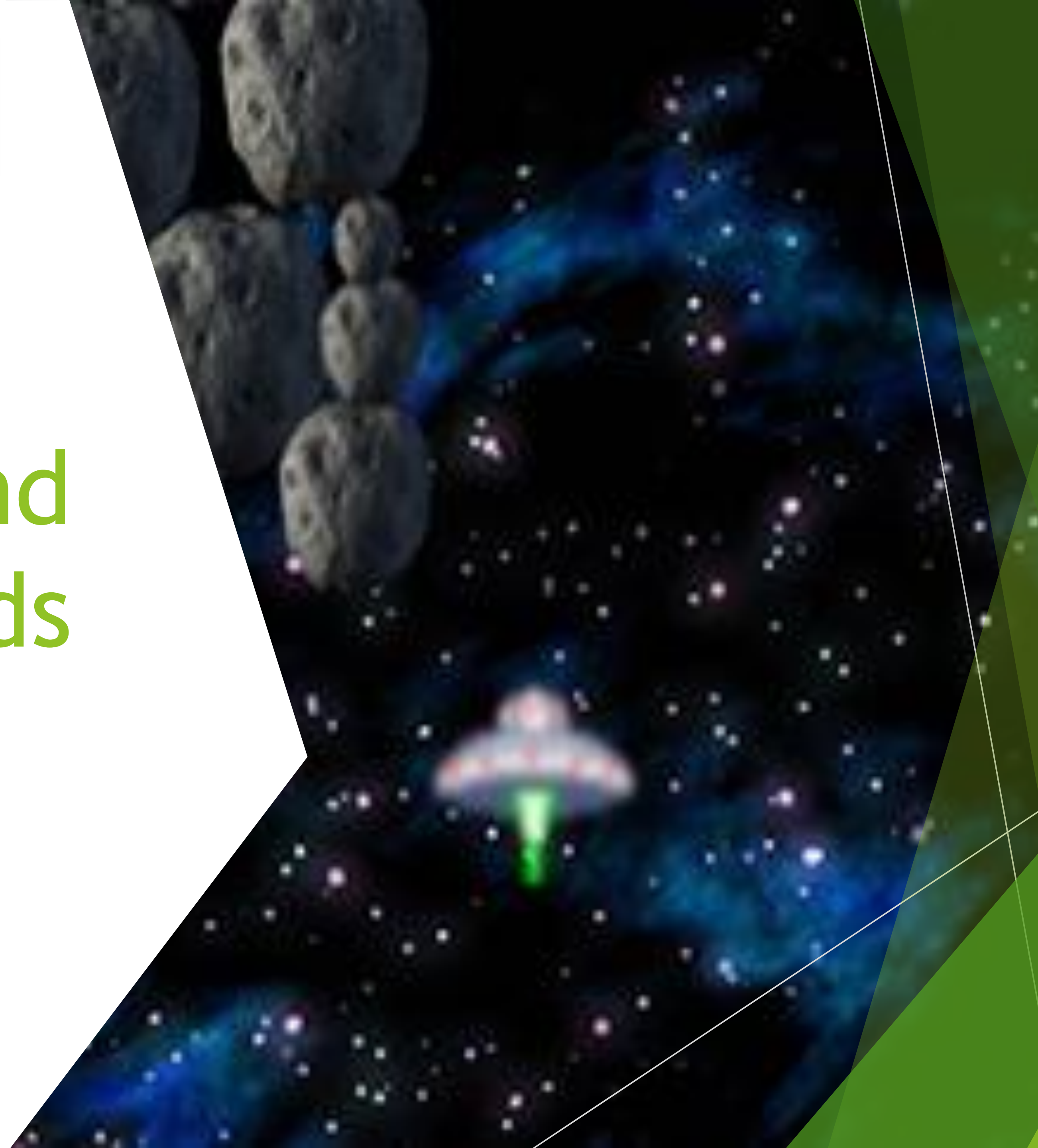
Ex – collision, obstacle generation. etc

Appropriated into the game

GameObject – key class

Graphics and Sounds

- Scrolling backgrounds
- Game images/sounds
- Use of Gimp



Testing

- Junit tests
- Many types of testing carried:
    - Blackbox testing - obstacle generator working as expected
    - Whitebox testing – tests to check collision of the space ship with obstacles (with good coverage)
    - Integration testing – sample simulation of the entire game
    - Manual testing – frequently playing the game, crashing the player intentionally to ensure game over features work, etc.

Improvements, Extensions and Optimisation

- Improvements to:
  - Design approach: adopted façade/decorator pattern, better plan and modularised it better
  - Implementation: Use SoundPool instead of MediaPlayer, bitmaps for saved preferences using a separate cache class, etc.
  - Time management/milestones: implementation of collision which is the heart of the game and hence delayed everything else.
- Extensions easily added as a result of our chosen design approach

Questions ???

Thank you