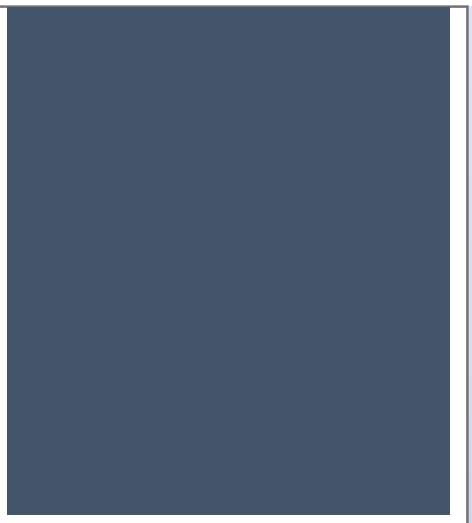

Gold Milestone Report

Resilient Kiwi Game

RKMD Game Programming Team

Contents

Gold Milestone Report

This Gold Milestone Report for *RKMD: Resilient Kiwi, Magnificent Dreams* game created for COMP710 Game Programming Team Project Assignment, will summarise feature completion and responsibilities undertaken by each team member.

Game Summary

Resilient Kiwi: Magnificent Dream
Game Pitch by RKMD Game Programming Team

Resilient Kiwi: Magnificent Dream is a vertical scrolling platformer. You play as a kiwi who dreams of flying higher than any bird has flown before. The aim of the game is to collect as many points as possible before the game is over. The player gets points from both climbing higher and through collecting kiwifruits. Once the player falls, they get another opportunity to collect as many kiwifruits as possible on the way down. High scores are placed on a leader board.

Since kiwis can't fly, so the player must use the platforms in the game's environment to get as high as possible. Screen creep slowly applies pressure on the player to move upward. Kiwi can single jump and double jump-flap to move to each platform. The typical range of platforms are implemented including bouncy platforms, crumbling platforms and moving platforms.

The game world changes as the player ascends upwards. Kiwi starts in a forest environment which changes to a city landscape, to mountains, to clouds and finally, to space. Where possible, the objects and enemies will change with each new environment such as:

- Branch platforms in trees
- Window cleaning platforms in the city
- Cloud platforms etc.

Kiwi will also encounter enemies as he traverses the world. Again, the enemies will be relevant to the current environment such as pests (rats, stoats) in the forest, birds in the clouds etc. Kiwi has the ability to shoot enemies with green laser beams that shoot from his eyes.

Kiwi can collect powerups on the way up which enhance and/or change the gameplay. Such as:

- Kiwifruit – powerups, coins
- Gumboots - spring boots for higher jump
- Hardhat + hi-vis vest - invincibility
- L&P – upside down bottle jetpack

The game art will be vibrant, cartoony and cute, with lots of fun animations and particle effects.

Sounds will have an arcade feel and the music will be upbeat, light-hearted and have an adventurous/triumphant feel to it to match the game's premise.

Target platform: PC
Target audience: Anyone over the age of 5
Genre: Platformer/Arcade/Casual/Action

Unique Selling Points:

- Kiwiana – learn about New Zealand icons and culture
- Simple and easy to play
- Cute and customisable kiwi character
  Reversible gameplay

Kadin Honeyfield

Summary

My responsibilities mostly focused on UI elements and sprites and not so much gameplay apart from creating the base Section, platform classes, making them data driven and travel down the screen to simulate the player climbing. However, we still found ourselves helping each other with our responsibilities by sharing ideas, techniques and peer programming to help solve problems.

Responsibilities

My responsibilities were:

- Creating the data driven sections, the section manager and platform classes
- Creating the Player Animated Sprite Sheet
- Creating the UI Sprite sheet
- Creating Game play user interfaces
- Making sections travel down the screen as the player climbs
- Making Sections respawn and randomize to simulate random platform generation

Features completed

- Section, Platform (and sub classes), Section Manager classes
- INI Reader class
- Data driven sections
- Respawning sections
- Making sections travel down the screen as the player climbs
- Most important Player animated sprites (Standing, jumping, running)
- UI sprite sheets
    - Including Xbox buttons that appear if a controller has been detected
- Platform sprites
- Created Gameplay user interfaces and functionality
- Scoring algorithm – Uses how far the camera has moved = total distance climbed
- Game Over State within play state
- Colour Class
- UserInterface class, InterfaceComponent class and sub classes (rectangle, button, toggle button, label)
- Final main menu background sprite

Devin Grant-Miles

Summary

Responsibilities

Mainly responsible for implementing the game's state machines, particle effects and pickup items (Kiwicoins and Powerups). However, responsibilities naturally overlapped and we took a flexible approach to development, regularly collaborating with the each other and contributing to each other's features.

Features completed

1. IntroState and splash screens (apart from RKMD splash screen)
2. Creating the game state machine and contributing to the implementation of the various states
3. Working with Rob implementing the player state machine and contributing to the various state transitions, methods etc.
4. Creating the power ups and Kiwicoins and implementing their effects + integrating with the Section and SectionManager classes with PickUpFactory (reusing Maya's enemies code)
5. Creating the particle effects
6. Implementing game controller input and other features of the input handler
7. Implementing sounds with Maya (using Maya's SoundManager class)

Robert Dumagan

Summary

My main responsibilities consisted of implementing the animated sprites, reviewing everyone's completed features and small modifications and quick fixes. Since we have decided to take on a Kanban approach towards the development of the game, I have been involved in feature's that required touching up or bug fixing.

Responsibilities

- Attending each stand up meeting regularly.
- Communicating any problems or bugs with everyone in the team.
- Consistently updating with everyone's changes.
- Adding quick fixes and modifications on features.
- Creating and implementing the animated sprites to read in a sprite sheet.
- Refactoring the old player state algorithm with Devin.
- Updating documentation.
- Constant testing and debugging with every commit to the game.
- Making sure that there are no memory leaks or warnings before committing.
- Using correct naming conventions and standards throughout the code.

Features completed

- Implementing animated sprites to read in multiple rows with the help of Kadin.
- Integrating the animated sprites with each player state process.
- Creating an animated sprite manager which uses the sprite manager
- Involved in refactoring the previous method of ENUM's for the player states into a more manageable approach with Devin and Kadin.
- Adding additional debug features such as capturing the players feet hitbox and assigning a new hotkey to give the player the invincibility power up.
- Modified the player to change direction towards the way he is shooting.
- Fixed the crumbling platform not respawning properly.
- Added a crumbling platform respawn timer that allows the crumbling platforms to respawn after a given amount of time.
- Fixed the player screen wrap around bug along the left side of the screen.

Maya Ashizumi-Munn

Summary

Main responsibilities included applying scrum to our planning and development process (taking meeting minutes, organising backlog, stand-ups), and also developing Creature entities (player and enemies) and their interactions.

Responsibilities

My team responsibilies included:

- Taking meeting minutes for all of our meetings
- Performing stand-ups at our development meetings

My development responsibilies included:

- Assisting Kadin to create some sprites for the game
- Creating the RKMD team logo
- Creating base entity classes for creatures (player and enemies)
- Basic enemy AI
- Basic player functionality
- Working with the team to implement input handling
- Configuring the release mode for Gold milestone

Features completed

- Player movement implementation – physics and input handling for moving left, right and jumping
- Collision methods:
    - Player/Platform – Ensuring player can stand on platforms (Rectangle/Rectangle collison)
    - Player/Enemy – Collision with an enemy will cause game over (Circle/Circle collision)
    - LaserBeam/Enemy – Reduce hitpoints off an enemy if laserbeam collides (Rectangle/Rectangle collision)
- Enemy class and its subclasses – GroundPassiveEnemy, GroundAggressiveEnemy and FlyingAggressiveEnemy. Implemented different AI for each type and created different sprites for each enemy type in each level (15 total)
- Manager classes
    - SoundManager – grabbed from individual assignment
    - SpriteManager – created to automate and simplify sprite object deletion
- Bullets to attack enemies – LaserBeam class, ripped from SpaceInvaders exercises, modified for use in RKMD game
- Level changes – 5 different levels: Forest, City, Mountain, Cloud and Space. Made some methods to switch backgrounds and made different enemy sprites for each level to imitate level changing.