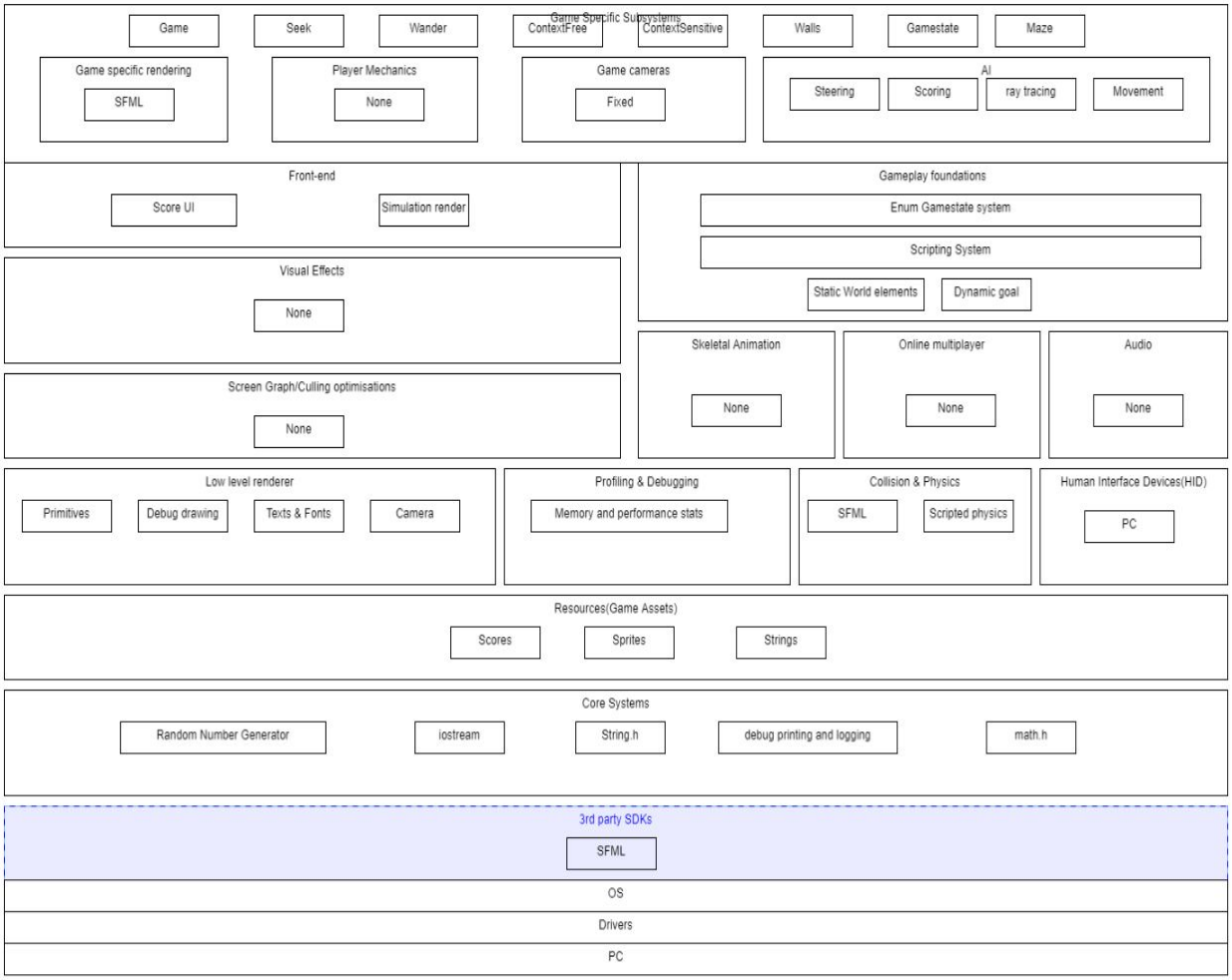
Computer Games Development CW208
Technical Design Document Year III

Student Name: Martin Farrell
Student Number: C00157047

Contents

Game Architecture



Features

Feature: Grid
Tasks:

1. Create a block class that initialises rectangle shapes.
2. Create a 2D array of this block class type.
3. Draw the grid.

Feature: Context-free Steering AI
Tasks:

1. Create a triangle using sf::circleshape from the SFML library.
2. Initialise the triangle and give it a position and velocity vector.
3. Draw the triangle.
4. Calculate a unit velocity vector using the goal's position.
5. Add velocity to the position vector and multiply it by speed and time.
6. Add a rudimentary circle to square collision detection and adjust velocity if a collision happens.

Feature: Context-sensitive Steering AI
Tasks:

1. Create a triangle using sf::circleshape from the SFML library.
2. Initialise the triangle and give it a position and velocity vector.
3. Draw the triangle.
4. Create a danger array, desire array and directional array.
5. Create a move function that takes one of the directions stored in the directional array initially and have the triangle move in that direction multiplied by speed and time.
6. Create a function that checks in 8 directions by a number of pixels and if it detects anything in a given direction it will compute the danger as a float and add it to the corresponding entry in the danger array.
7. Create a function that checks in each direction for the goal and if the goal is found calculate a unit vector towards the goal and set that as the new velocity(times speed and time).

Feature: Wander
Tasks:

1. Create a triangle using sf::circleshape from the SFML library.
2. Initialise the triangle and give it a position and velocity vector.
3. Draw the triangle.
4. Generate a random direction as the initial velocity.
5. Implement collision detection so that a random direction 180 degrees away from the object is selected.

6. Add implementation so that when the goal is within 500 pixels the velocity is changed to go towards the goal.

Feature: Seek
Tasks:

1. Create a triangle using sf::circleshape from the SFML library.
2. Initialise the triangle and give it a position and velocity vector.
3. Draw the triangle.
4. Give the the seek AI an initial velocity toward the goal.
5. Calculate a steering force to add to the velocity so it follows a curved path around the wall.
6. When the seek AI reaches a certain point change it's velocity to o towards the goal.

Feature: Walls
Tasks:

1. Get the specific blocks from the grid to be used as the walls.
2. Change their colour to green.
3. Store in an array to be referenced later for positions and global bounds etc.

Feature: Score
Tasks:

1. Create variables to store various elements used to calculate the score.
2. Calculate path using distance formula.
3. Calculate the number of collisions that occur.
4. Calculate time.
5. Use execution start and execution end, subtract from each other and divide by the number of frames to calculate average execution time.
6. Display all of these values as strings between simulations.

Feature: Score UI
Tasks:

1. Create a black rectangle and draw in the centre of screen when in a results gamestate.
2. Draw the strings on top of this rectangle.

Feature: Timer
Tasks:

1. Create a counter that updates every update.
2. Display this counter via a string in the top right corner.
3. Also display on the results screen.
4. Reset between simulations.

Feature: Goal
Tasks:

1. Create a rectangle and give it a position, colour and size.
2. Add a velocity to it and update the position each frame.
3. Place limits on the x and y components of the position vector so it follows the path of a square e.g. if(y <200){velocity = etc}.
4. Add a label to show it is the goal.

Feature: Gamestates
Tasks:

1. Create an enum class.
2. Add all the gamestates.
3. Add logic to game loop for various operations in different states such as what to render and update etc.

CRC Cards

Class: Game
Responsibilities 1. Creating instances of all the other classes. 2. Updates. 3. Handling Gamestates. 4.Rendering.
Collaborators 1.Seek 2. Wander 3. Context-Free 4. Context-Sensitive 5. Gamestates 6. Grid 7. Walls

Class: Seek
Responsibilities 1. returning member variables 2. moving 3. steering
Collaborators 1.Game

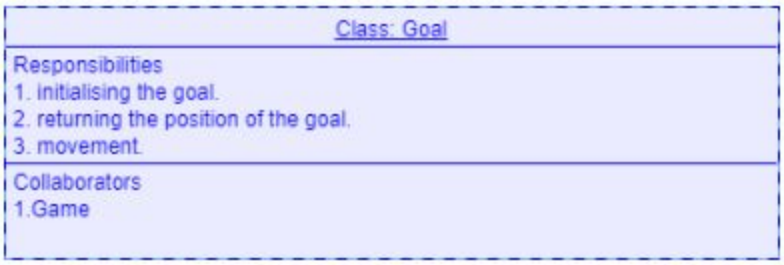
Class: Wander
Responsibilities 1. returning member variables 2. moving 3. steering
Collaborators 1.Game

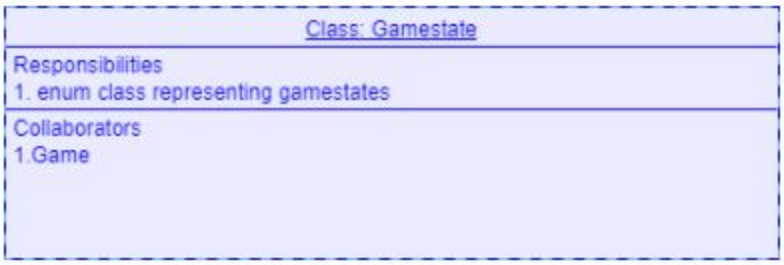
Class: Context-Free
Responsibilities 1. returning member variables 2. moving 3. steering
Collaborators 1.Game

Class: Context-Sensitive
Responsibilities 1. returning member variables 2. moving 3. steering 4. Calculating danger and desire
Collaborators 1.Game

Class: Grid
Responsibilities 1. Generating a 2D array of rectangleshapes 2. initialising the grid.
Collaborators 1.Game

Class: Wall
Responsibilities 1. taking the blocks from grid that will be used as obstacles. 2. changing their colour. 3. storing them in an array.
Collaborators 1.Game





References

TDD template, accessed at: https://onedrive.live.com/view.aspx?resid=5E66DFE2F20FD7ED!357420&ithint=file%2cdocx&authkey=!AJ7181Rim3R1fXI (accessed on Jan 8th 2020).