Game 11

Finite State Machines

Finite State Machines ~ Introduction

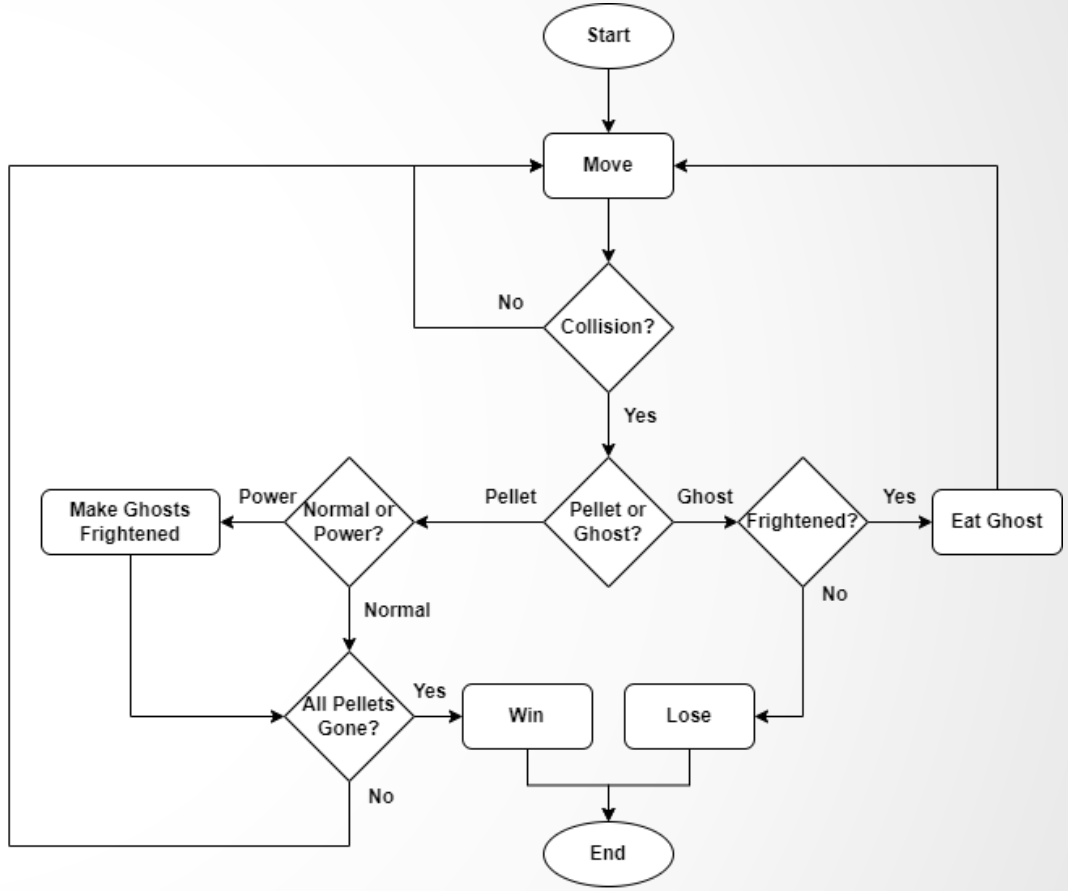
- A Finite State Machine (FSM) is a model used to simulate the behaviour of an object or agent.
- It uses states and transitions to perform actions and respond to certain situations in different ways.
- It is very useful when programming enemies in games, allowing them to respond to the player's input in any way possible.
- One of the earliest well-known uses of FSMs in games is in the 1980s hit arcade game Pac-Man, which made excellent uses of these models to simulate the behaviour of its infamous ghosts.

Finite State Machines ~ AI Explanation (1)

- This recreation of Pac-Man uses a 2D Tilemap to visualise the grid (the walls, pellets, and power pellets).
- Excluding the pellets, there are a total of 5 main Game Objects involved: Pac-Man, and the ghosts, Blinky, Pinky, Inky, and Clyde.
- While Pac-Man makes use of a simple movement script with some win or lose conditions, the ghosts' behaviour is much more complex, using 4 different states to alternate between throughout the game's progress.

Finite State Machines ~ AI Explanation (2)

- Pac-Man functions by using a simple movement system.
- He can collide with all pellets, and with the ghosts.
- If he collides with a power pellet, the ghosts are turned into their frightened states.
- If he collides with a ghost in its frightened state, he eats it.
- If the ghosts is not frightened, the player loses the game.
- In order to win, all the pellets must be destroyed by Pac-Man.



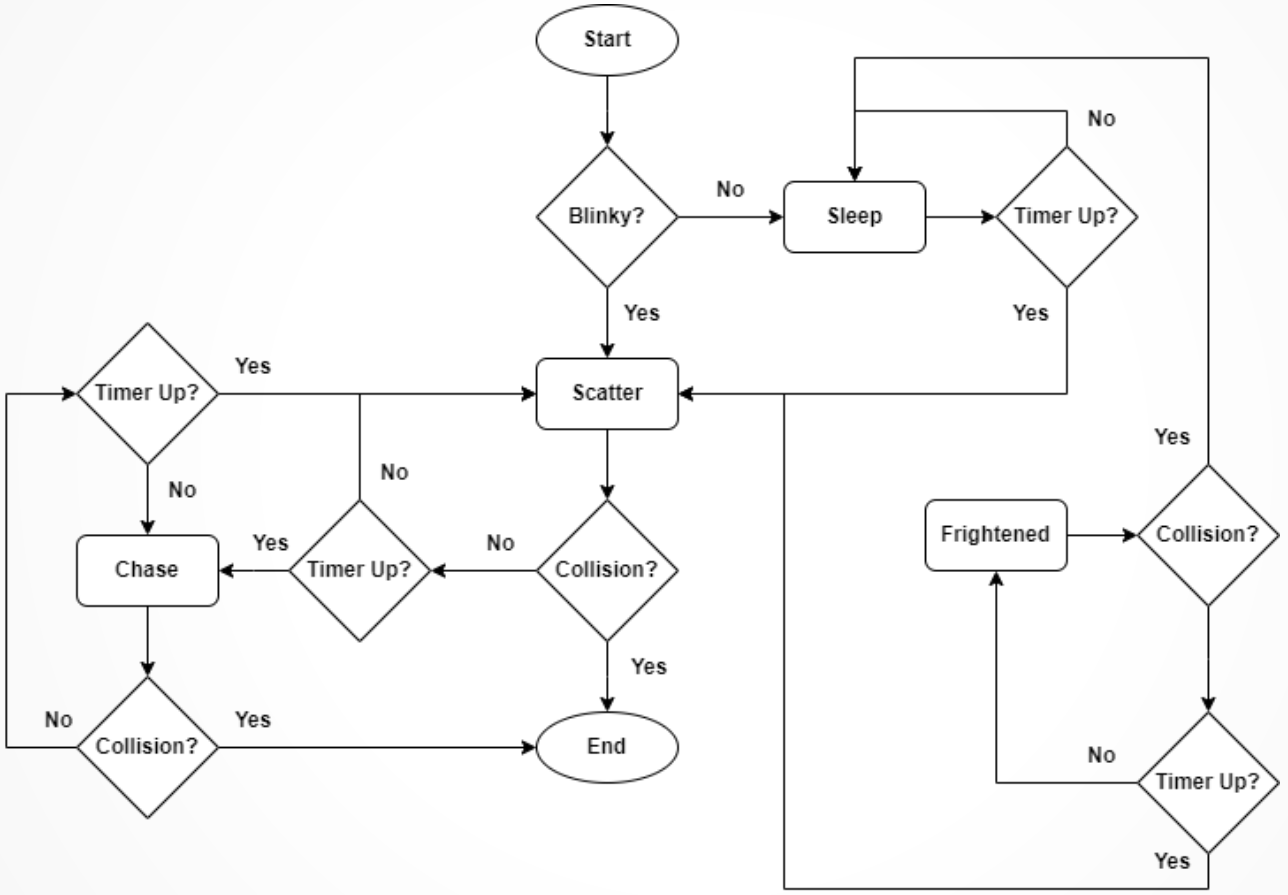
Finite State Machines ~ AI Explanation (3)

- As for the ghosts, each one behaves a bit differently but they all share the same states: Chase, Scatter, Frightened, and Sleep.
- Most of the ghosts begin in their sleep state, apart from Blinky, who begins in his Scatter state.
- Each ghost has a unique duration for their sleep state, ranging from 5 seconds for Inky, to 15 seconds for Clyde.
- However, the ghosts share the same durations for their other states, with the Scatter state lasting for 10 seconds, and the Chase and Frightened states lasting for 15 seconds.

Finite State Machines ~ AI Explanation (4)

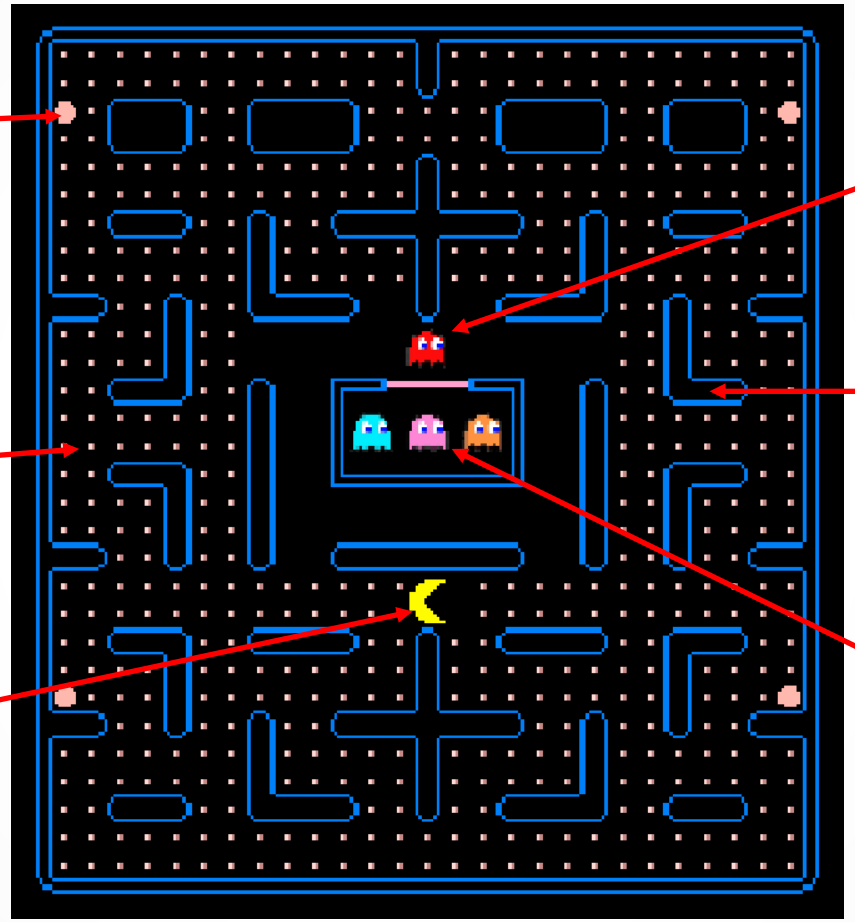
- The states function as follows:
  - **Chase:** Through the implementation of an A* pathfinding algorithm, the ghost moves towards Pac-Man's position and tries to destroy him. Once the timer ends, it alternates to the Scatter state.
  - **Scatter:** Once again using A*, the ghost moves to a pre-set path in a corner of the map using waypoints. Each ghost has its own set of waypoints in different corners of the map. Once the timer ends, it alternates to the Chase state.
  - **Frightened:** When Pac-Man eats a power pellet, the ghost swaps its sprite to its blue, scared sprite. It moves to random points on the map, using colliders to ensure it doesn't conflict with the walls. If the ghost is eaten by Pac-Man, the state ends early and it alternates to the Sleep state, resetting its sprite. Otherwise, if the timer runs out, the ghost simply resets its sprite and alternates to the Scatter state.
  - **Sleep:** At the start of the game, or after being eaten, the ghost stands still in a closed-off box at the centre of the grid. Once the timer ends, it teleports to the starting position and alternates to the Scatter state.

Finite State Machines ~ AI Explanation (5)

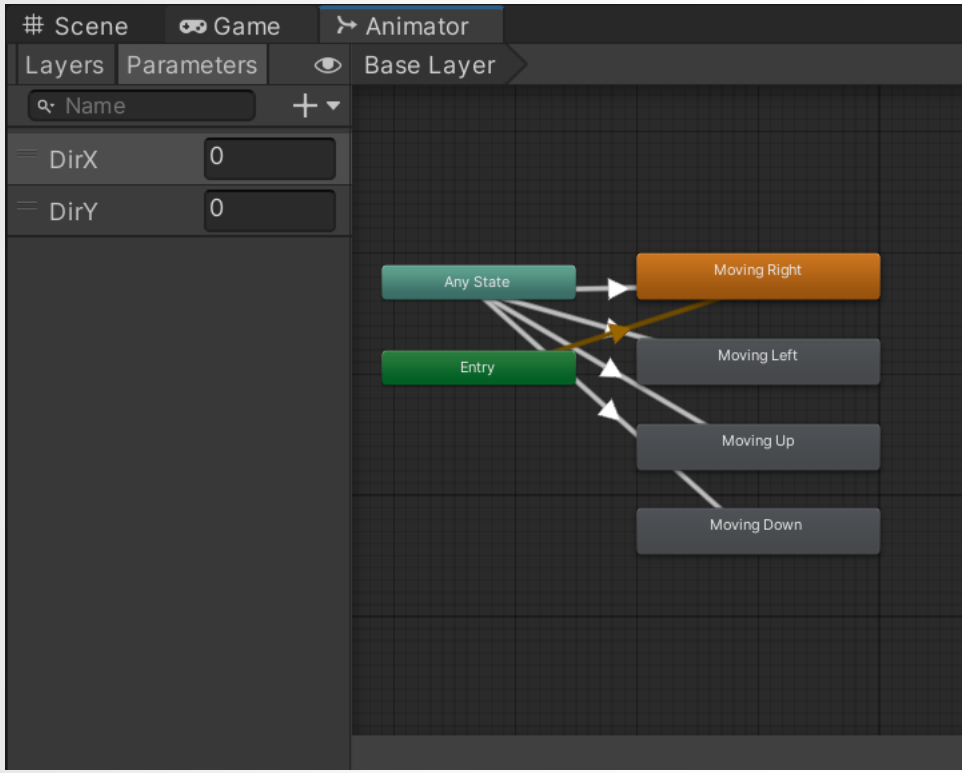


Finite State Machines ~ Mini-Game Implementation (1)

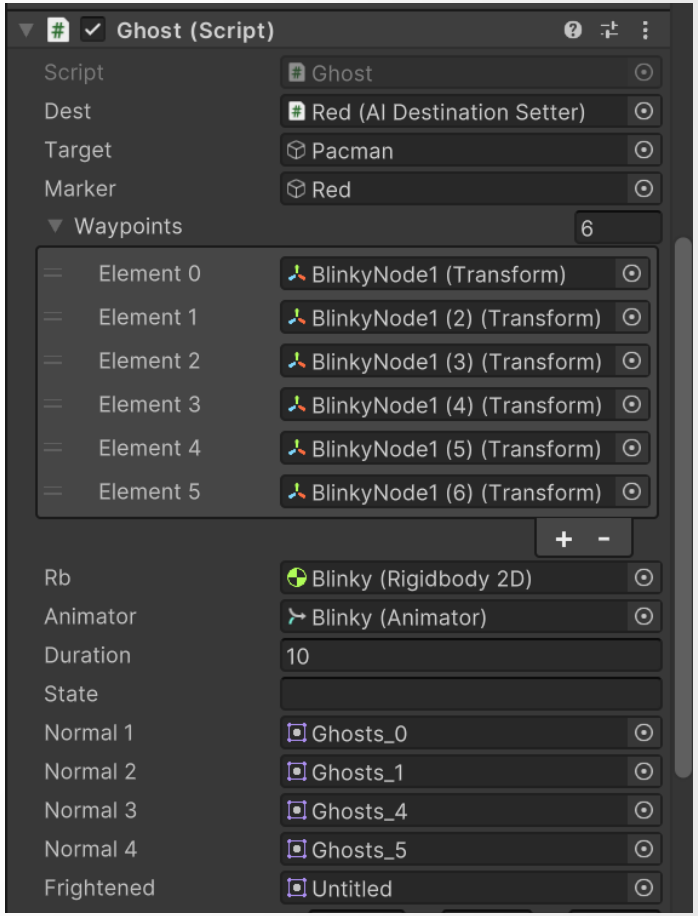
Playable Area:



Finite State Machines ~ Mini-Game Implementation (2)

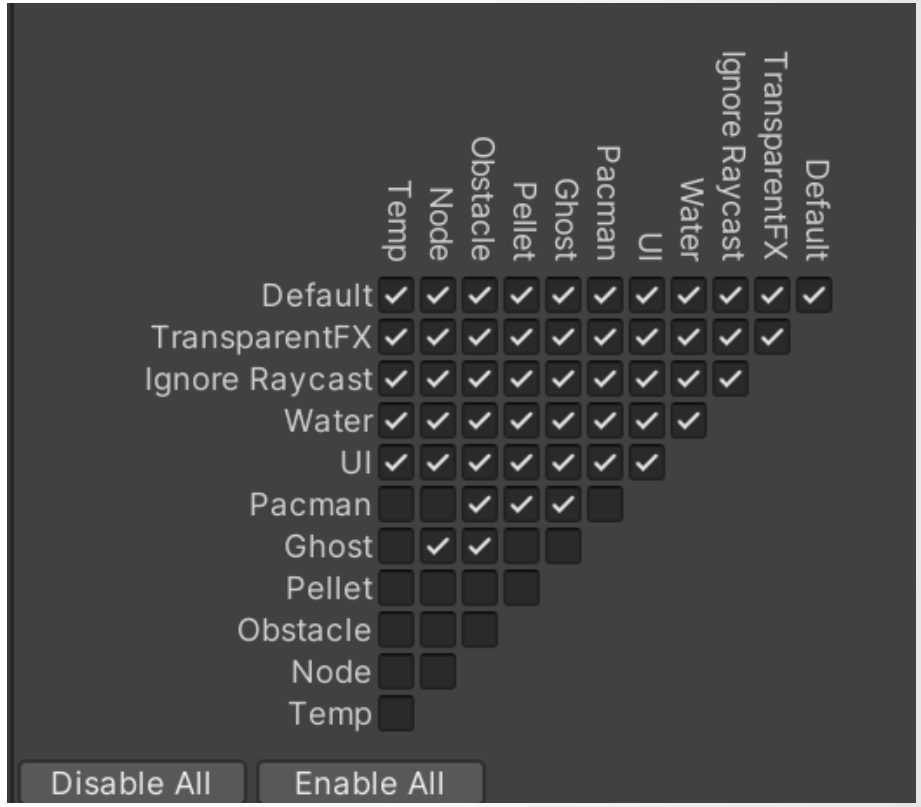


- (Left): The animation states for Pac-Man. The ones for the ghosts are similar.
- (Right): The parameters of Blinky. The ones for the other ghosts are similar.



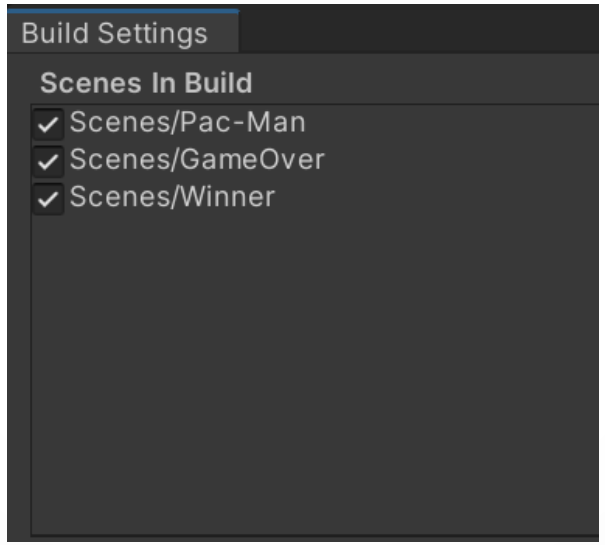
Finite State Machines ~ Mini-Game Implementation (3)

- **How to Set Up Package (1):**
  - After importing package, navigate to the top right of Unity Editor and do the following:
    - **Layers > Edit Layers.**
    - **Click the Slider button at the top right of the Inspector, in between the question mark and the three dots.**
    - **Select Preset.**
  - Now navigate to the top left of Unity Editor and do the following:
    - **Edit > Project Settings > Physics 2D > Layer Collision 2D.**
    - **Set the collision as follows:**



Finite State Machines ~ Mini-Game Implementation (4)

- **How to Set Up Package (2):**
  - After importing package, navigate to the top left of Unity Editor and do the following:
    - **File > Build Settings.**
    - **Drag the scenes GameOver and Winner into the build index like the picture below.**



Finite State Machines ~ Exercise (1)

- **Your turn!**

1. Find the script Ghost and navigate to the first state function Chase.
2. For the Chase function:
   - The A* has already been configured for you. Use the AIDestinationSetter field to set the ghost's target to Pac-Man's transform.
   - Update the animation for the ghost by using the x and y coordinates of the direction it is moving in.
   - Create a timer so that after a pre-determined duration, the ghost's state changes to Scatter.
3. For the Scatter function:
   - Re-use the code from the Chase function but now set the ghost's target to continue following its waypoints.
   - The ghost should return to the Chase state after Scatter's duration runs out.

Finite State Machines ~ Exercise (2)

4. For the Sleep function:
    - Set the ghost's target to the transform of the Game Object which the ghost is attached to. This object contains the A* navigation system which moves the ghost.
    - Like before, set a timer for Sleep, but when the timer runs out, simply teleport the ghost to the position of the parent Game Object.
5. For the Frightened function:
    - Check if the sprite of the ghost is still normal. If it is, alternate to the frightened sprite. You may need to play around with the scale of the new sprite.
    - Check if the ghost has arrived at its target node using a simple bool variable. This should always be true at the start of the state.
    - If it has, create a new Game Object and assign it a circle collider and a custom tag.
    - Find a random Vector3 position in the range of the grid and assign it to the Game Object you created. Use Physics.OverlapSphere() to make sure that the random position isn't inside a wall.
    - Set the ghost's target to the random position.

Finite State Machines ~ Exercise (3)

6. Almost done! Create a trigger function for the colliders.
7. In this function, you must check for three different types of collisions:
    - Using tags, check if the collision is with a waypoint node and update the next waypoint target accordingly.
    - If the ghost is in its Sleep state and it collides with the Start node, alternate it to the Scatter state.
    - If the ghost is in its Frightened state and it collides with the Game Object you previously created, set the arrive bool variable to true.

Finite State Machines ~ Conclusion

- Finite State Machines are a very useful tool to incorporate in games with NPCs.
- The more complex they are, the more realistic and immersive the game becomes.
- They are very easy to implement and can help make your enemy AI even more challenging and fun to play against.

Finite State Machines ~ References

[1] – Prof. A. Dingli, ICS2211: "Level4_FSM" [Online]. Available: https://www.um.edu.mt/vle/pluginfile.php/1116065/mod_resource/content/1/Level4_FSM.pdf [Accessed: 18-May-2023]

[2] – noobtuts, Unity 2D Pac-Man Tutorial, [Online]. Available: https://noobtuts.com/unity/2d-pacman-game [Accessed: 18-May-2023]

[3] – Zigurous, How to make Pacman in Unity (Complete Tutorial), 2021 [Online video]. Available: https://www.youtube.com/watch?v=TKt_VlMn_aA [Accessed: 18-May-2023]