smallEasyDragon Enemy Prefab

What it does

-This enemy is a small dragon that uses a finite state machine to switch between different states while in the game. The enemy can patrol, wait, or chase the player. On patrol, the enemy will patrol between two points (assuming your game is a platformer), calling the wait state when it reaches either of the patrol point destinations. Once a player enters the enemies line of sight, it will chase the player until it attacks and damages, or until the player outruns/ leaves enemies line of sight.

How to use it

- -First install the prefab and needed sprites and scripts.
- -It will need the scripts:
  - -Enemy.cs
  - -Activity.cs
  - -BaseState.cs
  - -BaseStateMachine.cs
  - -Decision.cs
  - -RemainInState.cs
  - -State.cs
  - -Transition.cs
  - -ChaseActivity.cs
  - -PatrolActivity.cs
  - -PatrolPoints.cs
  - -WaitActivity.cs
  - -DistanceDecision.cs
  - -InLineOfSight.cs
  - -ReachedWaypointDestination.cs
  - -WaitTimeDecision.cs
- -The sprites needed:
  - -dragon.png
- -For ease of use (unless you want custom parameters)
  - -States folder
  - -Transitions folder

After all of these have been downloaded, drag it into the scene and set the player as the object the enemy will look for, and assign appropriate layer mask. The enemy will do the rest of the work.

What it expects

-The enemy will patrol along the designated points until a player comes into line of sight of the enemy. Once this happens, it will chase the player and try to inflict damage until either killed, or the player runs away.

How to modify

-There is LOTS of ways to modify this prefab, thanks to the flexible finite state machine. To change basic stats of the enemy, you can look at the Enemy.cs file and manually assign new values, or in the inspector because all of the fields are serialized. Different states and transitions can also be made and assigned by making respective scriptable game objects and dragging and dropping them. See the States and Transitions folders to see examples.