Statemachine – Exercise

Exercise 1 – enum

Create an enum "RabbitState" with the following States:

- Idle
- RunToward
- RunAway

The Rabbit-NPC should have an instance-variable of this state called _curState (=Current State).

Exercise 2 - State-Logic

Create a seperate Update-Method for each state which should be called in Unity's Update-Method every frame (i.e. void UpdateIdleState() for the Idle-State).
In the RunToward-State the Rabbit should move toward the player and in the RunAway-State it should move away from the player.
Note: You can just use the Method *SetVelocityX(…)* and the variable *runningSpeed* to set the movement-Speed of the Rabbit.

In RabbitAI.Update() you should call the Updatemethod of the state, which the Rabbit is currently in. For that you can use a switch-statement or an if/else-if Block.

Exercise 3 - State-Transitions

Implement the State-Transitions as shown in the graphic. You can use the already existing distance – Variable to determine the distance to the player.

Final Note: Implement the State-Change only in the specific Update-Method of every state and not in the general Update-Method of the Rabbit.

Exercise 4 – Bonus for Completionists

We could also implement Animations which trigger in the different states. You can just call

animator.SetIngeger("anim", 0)

- anim=0 is the Idle-Animation
- anim=1 is the run-Animation
- anim-2 is the run-backwards-Animation