Francesco De Marchi

TASK DOCUMENTATION

Unity version 2019.3.3f1

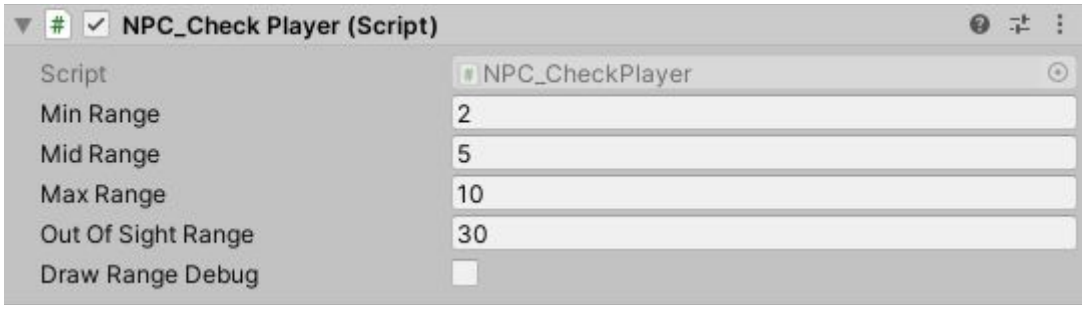
The first idea I had in order to get information on the proximity between an NPC and the player was the trigger interactions, i.e. when the player enters a control zone (sphere/capsule/collider box) it activates a behavior or trigger an action.

The limit of this system is that the NPC behaviour is activated only once, and the distance from a point of interest is not managed.

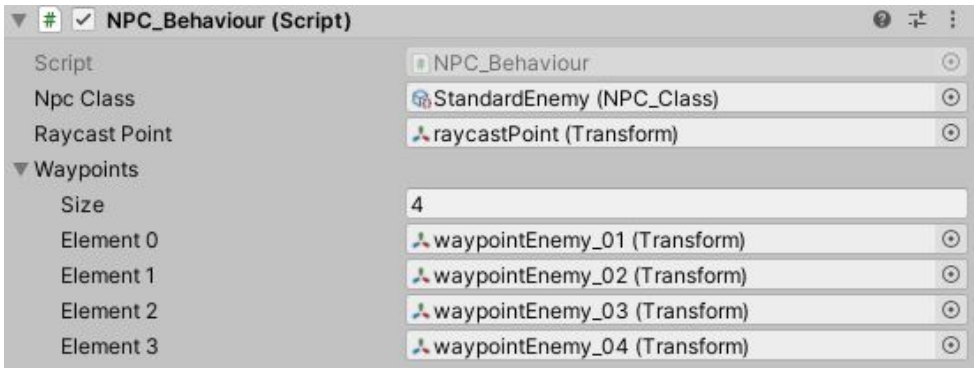
For this reason I have implemented (together with the triggers) a script that calculates the distance from the player (***NPC_CheckPlayer***). To prevent every frame calculation, I didn't manage it in update but inside a coroutine that starts automatically when the scene begins or the NPC spawns, and loops in a certain time frame managed by a variable (***delay***), which it can be recalled and modified at any time.

Inside this script, I managed 4 hypothetical distances between NPC and player (***min-mid-max-out***) where, based on the position of the player, the NPC can have certain behaviors (for example, in ***min***, NPC is in contact with the player while in ***max*** it is in the most distant controlled area).

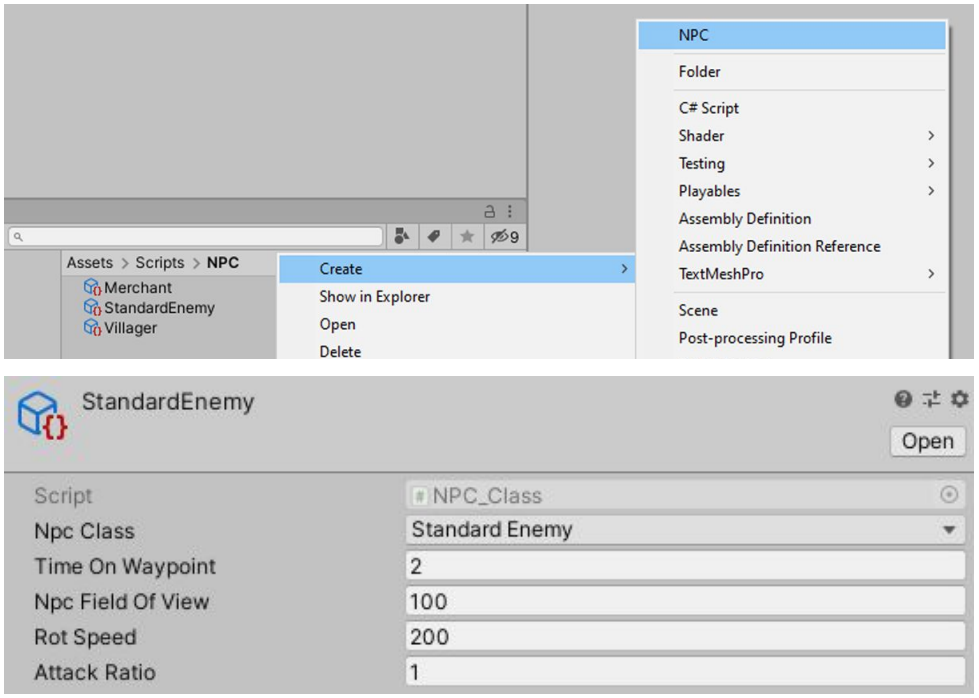
Instead, the range ***out*** defines when the NPC is outside the camera view, and at this moment, it is possible to deactivate its main script (***NPC_Behaviour***) and other components that work in runtime (mesh render, animator, etc.).
The distance calculation still remains active and it will activate all components again if player approaches.



The ***drawRangeDebug*** variable allows you to view the gizmos of the various ranges to visually help those who will have to deal with the implementation of the NPC.



NPC_Behaviour is the script that manages the main AI behaviors: you can choose between 3 NPC class (***villager-merchant-standardEnemy***) using scriptable object ***NPC*** (*right click on project window -> create -> NPC*)



All 3 NPC can move along a path if there are waypoints in the array and stop for a time managed by the ***timeOnWaypoint*** variable.
The variable ***npcFieldOfView*** manages NPC field of view (in degrees) in order to define if the player is in sight or not, and it acts accordingly (*example*: if the villager is looking at the player he can respond to player's interaction , while if an enemy sees the player he can follow or attack him).
rotSpeed is a variable that allows you to manage how quickly the NPC will rotate towards the player (if there is interaction, for example the villager).
attackRatio manages the delay (coroutine) between a possible attack or interaction.
The calculation is made based on the angle of the NPC/Player vector and the Z axis of the NPC, if this is within the range it generates a raycast (***raycastPoint*** is a gameObject that manages where the ray starts from), which if it hits the player makes the NPC perform certain actions, while if it hits a walls he continues it in his path.
