Stealthifier

A Stealth system implementation kit for Unity3D game titles.

- *Developed by Dylan Meissner, Joshua Tanner, and Zachary Watson as part of Contrary Scholars.*

Introduction

Welcome to Stealthifier! Stealthifier is a plugin that allows you to add stealth functionality to your title, with a modular system that you can customise by adding or removing any of the components at will.

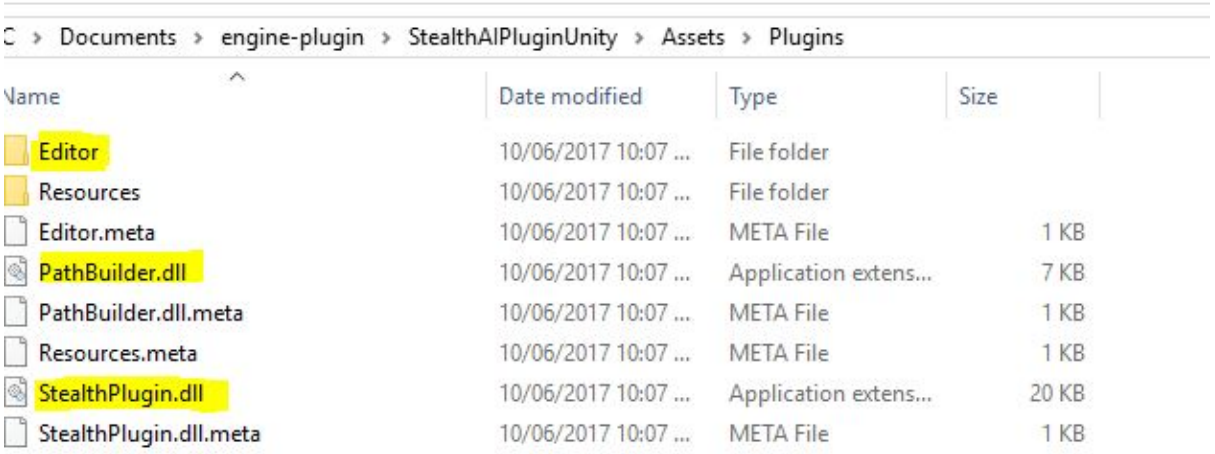
The plugin is broken down into 4 monobehaviour scripts:

- Stealth: This goes on the AI controlled character. This script handles all different types of functionality - from field of view to speed.
- Detection Meter: Creates a detection meter which is filled by the Stealth script.
- Player Detection: Handles the player's detectability - depending on which transforms are passed in here, the AI will be able to see different body components of the player.
- Stealth Manager: Holds references to the AI characters and manages several auxiliary functions.

Initial Setup

This stealth toolkit includes the following files:

- StealthPlugin DLL (handles the enemy AI and player detection)
- StealthEditor DLL (handles the custom inspector windows for StealthPlugin)
- PathBuilder DLL (handles the creation of paths for enemies to follow)
- athEditor DLL (handles the customer inspector windows for PathBuilder)



Note: these editor DLLs must be kept inside an Editor folder within a Plugins folder in Unity, as seen above.

The Stealth and Pathbuilder Plugin can be left in the 'Plugins' folder as above.

Each of Stealthifier's scripts and their properties is described in detail below.

AI Stealth Component

Player Settings

- Player Reference: the player object currently in the scene. If the player is not manually added here, the object tagged with "Player" will be found and added automatically.
- Player Caught: Whether or not the player has been caught, you can use this to trigger your end state.
- Head Transform: Where the AI's 'eyes' are. This is used to raycast from - to make sure that vision checks are accurate. If this is not set, this will use the normal transform of the player.

Detection Settings - Values that affect the AI character with regards to detecting the player.

- Vision Distance: How far the Enemy AI character can see.
- Visual Perception: How quickly the AI character is able to recognise the player character through sight. The larger this value is, the higher the AI's reaction time. If this is 0, the AI is unable to see and can be considered 'blind'.
- Audial Distance: How far the Enemy AI character can hear.
- Ally Alert Range: The Ally Alert range is the AI character's 'shout' range. If an AI character is pursuing the player, they will periodically 'shout'. If another, not-alerted AI character is in this range, they will become 'alerted' and turn to chase the player.
- Detection Decrease Rate: How quickly the AI character's 'detection' value decreases. The higher this value, the quicker the AI's 'detection' will decrease and thus they will return to their original state quicker.
- Suspicion Threshold: At what point the AI will transition from 'suspicious' to 'alerted'. This is represented in the example with the change from the yellow exclamation mark to the red exclamation mark.
- Patrol/Chase/Search FOV values: How wide the AI character's Field of View values are when they are in each state. These can be tuned to make AI characters in specific states are narrower or wider.
- Patrol/Chase/Search speeds: How fast the AI characters move when in their different states.
- Alert Time: How long the AI character will stay 'alerted' for before returning to their previous state.
- Path Parent: The controlling object for the path that the AI will be following. Created when the user clicks the 'Create patrol path' button at the bottom of the stealth script.

Searching Settings - Values that affect the AI character when they are in the search state.

- Rotate Speed: How quickly the AI character will rotate while they are doing their 'sweep' procedure when they are looking for the player.
- Search angle: How far the AI character will rotate upon reaching its search position.

Manager: The scene's stealth manager, used for things like Ally Alert Range.
Draw Gizmos: Whether the user wants the gizmos for pathing etc. to be visible.

The stealth script is the core of this plugin. When added to an AI character, this stealth script will be able to allow AI characters to patrol a marked area, that can be set up very easily.

How to set up a patrol path.

- Create an AI character by adding a stealth script component to the GameObject.
- Create a patrol path (click the Create patrol path button at the bottom of the script)
- Click on the new "Path Controller" object, which is a child of the selected character.
- Choose the path type - loop, or pingpong. Pingpong causes the AI to move along the route, then move back. Loop causes the AI to follow the path repeatedly.
- Create a path point. A Gizmo with the + and S symbols will be created.
- Use the Z (blue) and X (red) axis arrows to move the object around (make sure the 4-point movement option is selected in the top left)
- When another path point is required, create another one by clicking the 'add next point' button, or clicking the + button gizmo on the GameObject.
- Use the S button gizmo to select a particular path point.
- Adjust the rotation values to the preferred angle (this is the angle that the AI character will rotate to before moving to the next point if they have a wait time).
- Adjust the wait time (this is how long the AI character will wait at a particular point, use this for the AI interacting with an object, for example)
- The path will auto complete itself, and the process is complete.

Encountering Issues?

- Make sure that the navmesh is created before the scene is run. Ensure that non-walkable and walkable areas are specified.
- Ensure that there is a "Player" object present in the scene. The plugin looks for an object with the tag "Player" and will not work if there is no "Player" object present.
- Ensure that there exists only one Stealth manager object in the scene.

Detection Meter

The detection meter is a graphical representation of the AI's alertness of the player and is totally optional for the user. A detection meter is added by pressing the 'Add Detection Meter' button at the bottom of the script.

- Unaware/Suspicious/Alert meter values: Sprites used for the indicators. This is an exclamation mark in the example scene.
- Detection Pos/scale values: Changes the position above the AI character, as well as the scale of the image directly above the AI character.

Player

The player component contains a comparatively small amount of information.

- Player Detection: This script handles percentage detection, which allows AI detection speeds to vary based on the amount of the player that they are able to see. Components like arms and legs can be passed in, and then AI can detect individual body parts or the user as a whole. This Player Detection component also includes the currently active disguise value, which allows the systems to recognise which disguise the character currently has equipped.

Stealth Manager Component

The Stealth Manager component is used to hold all of the Stealth scripts in the scene. This holding of game objects is used when creating sounds primarily. The stealth manager is also used for adding and removing disguises, and registering/deregistering enemies that are created in the scene after awake() has been called. Finally, the system includes the AlertNearbyGuards function, which functions as a 'shout' for guards.

Functions: A basic guide

Functions in Stealthifier covers a variety of actions. Here is a description of the functions you might find useful.

Stealth

Set Speed/WaitingTime/ Destination: Sets the relevant values for AI pathing. The destination is the current point that the AI character is moving to. It is recommended that you let the Stealth script handle this.

GetDestinationReached: Returns true when the AI character has reached a destination point.

SetState: Sets the current state of the AI character to the specified state. It is recommended that you let the Stealth script handle this.

CheckPlayerInView: Returns true if the player is currently visible to the NPC. This is not a percentage tick.

Stealth Manager

CreateStepSound: Create footsteps. These act as noise generation when the player is moving. If the player is making step sounds, AI characters will hear them. Sounds cannot be heard through a wall, but they can be heard around corners and objects.

CreateSoundOneShot: Create a singular noise. This can be considered as a ‘shout’ and can be used to attract NPCs. This can be used to intentionally draw enemies to particular places.

ApplyDisguise: Applies the disguise (disguiseName) to the player object. This only occurs if the disguise that needs to be set is present in the disguises array on the object that has the StealthManager script attached.

RemoveDisguise: Removes the current disguise from the player, and also adjusts their detectionModifier back to 1. Consider this ‘normal clothing’.

RegisterEnemy: Adds the specified enemy to the stealthlist if they are not currently in it.

UnregisterEnemy: Removes the specified enemy from the stealthlist if they are present.

AlertNearbyGuards: This is the ‘shout’ function. This is called periodically while the AI is chasing the player character, and will alert previously not alerted characters.