IRoamer

aka InstaGIB roamer Instigator

Description and Content:

- This document;
- IRoamer.u File this is main mutator file;
- IRoamer.INT - making mutator visible in preferences menu and mutators list;
- A Sample INI file - even this can be used if it's renamed to **IRoamer.ini**;
- A MP4 small video captured after a LED screen - just for having an idea about functionality;
- A miserable English :/;
- Bot Haters should remove this mutator/archive in cause.

This is a mutator usable as Server-Side usually addressing DeathMatch types in game UT'99 when Bots are part of the game in company of mutators a la InstaGIB which are removing items from Levels - Bot goals for roaming. Don't ask me about the logic behind this... brain-fart ideea of InstaGIB.

Expanded Explanations:

When bot pawn is loaded in such a DeathMatch, usually it will attempt to navigate using an internal randomized solution - or crawling for JumpBots which can be found in such cases. The problem is that Not all time bot will move well, it will camp randomly for a small time, I could see Bots in such a ”Insta” server taking a lot of breaks and being very easy targets, game having no challenge and no flow.

Taking in account that Bot is motivated to move by ITEMS not by null content, this mutator somehow can solve this problem in big parts. InventorySpot actors left behind items removed are completed with fake goals - the number of these goals is configured in INI file which is created by mutator at first launch or using ”Preferences” command for lauching advanced configuration menu in game. Ini file created can be copied in server or used in Stand-Alone games. Goals are depleted taking in account default Pathing rules, which means they will have preserved space at least 100 UU (UnrealUnits) each-other - there are maps with a bunch of InventorySpot types in the same spot, a stupidity after all. In maps poorly loaded or without paths mutator does mainly nothing, it is addressing InventorySpot actors (any - even subclasses, see XC_Engine stuff by Higor) - all stock maps should be good at this point and similar maps.

Assuming we do have such good spots, goals are depleted and distracting Bot around, when are touched, or bot is closer they go inactive for some time leaving other goals to distract Bot and then state roaming is more properly engaged. After a configurable time, goals are activated again, and Bot can use internal native function ”FindBestInventoryPath” to travel around these actors. Other Bot classes more smarty coded might not need such mutators, they will proceed to hunt pawns if items are not found but they are weaponized properly. By doing various combinations between Inactive time and Number of items Bot can be motivated in different ways to roam around rather than randomly camping undecided.

Iterating through NavigationPoint actors for adding items it's done in a state code. If some troubles are happening, here can be used ”SlowMode”, mutator will get into a sort of background process breaking cycles in order to prevent a game crashed when engine might reach at

iterations limit.

The next problem. If some map (or many) will need a high tech paths hacking doable in time, mutator has a delay which is configurable, I mean goals for Bot are depleted after a time interval and not at loading Level. It will wait the action operated by other mutators and then will process Paths. Time used is counted in seconds. For preventing over and over again the same completions of spots which later can be predictable, mutator uses a randomizer - how many chances are to see an InventorySpot completed 0 to 1. 0 means mainly no chance, 0.5 means 50%, 0.9 means around 90% chances, in this way multiple sessions in the same map are mapped different, a server might use the same map with dedication so it should not be the same all time.

Ini Explained:
It do looks like here:

```
[IRoamer.IRoamer]
StartDelay=3.000000
DesireRateTime=15.000000
GoalChance=0.920000
bUseSlowMode=True
bDebug=True
GoalsUsed=22
```

StartDelay - seconds (sleep time) until mutator starts doing its job.

DesireRateTime - Bot engaged for this item and crawling around it, will set this item inactive, Item will be reactivated AFTER this time in seconds. Items inactive will not distract Bot to roam around, unless Bot has a native route to this spot according to internal PickDestination function, or another goal is active around.

GoalChance - this is the lottery of goals appearance in a said spot, value here means 92% chances. Values suitable here probably are 0.7 to 0.9 but you can test other ones.

bUseSlowMode - will be indeed slow in maps with a bunch of paths and if **GoalsUsed** value is a big one. Here purpose is to break number of iterations for UE1. Values are TRUE for slow Mode, FALSE for default mode.

bDebug - this is a debugger, not for mutator because IT DOESN'T include bugs found so far :P, but you can mess up at INI setup stage. When this is **True**, goals are visible in Stand_Alone games (you should not mess your awesome server so first would be good to check it privately) for figuring suitable values. Goals are Red when are not desired, triggered by Bot (default Bot the mostly), and they are Green when are in account for roaming, desired goals. Pick your values and see how do bot's roaming works. My goal was to see my BotyMan3 tester mutator moving ME constantly.

GoalsUsed - this is the limitation, the number of goals won't get over this value but number of goals can be lower based on **GoalChance** setting. Scenario -> map might have 10 InventorySpot actors and value is set for 30. For sure you will not have more than 10 Goals located and if you are decreasing GoalChance you might have only 5 spots used as targets. If these are inactive, Bot will not be interested in a powerful roaming unless they are activated again according to **DesireRateTime**.

Setup:
Mutator is called in chain as

IRoamer.IRoamer

through the rest of mutators and will not need a super duper priority, it can be the last. Being an authoritative thing it **won't need** to go in ServerPackages because only Server do needs it, so to speak machine handling game and not clients connected. Client doesn't need new useless files.

Archive contains a small External Video Capture with these items and mod set **bDebug=True** for figuring them and Bot's activity around. I'm not bother with too much quality at this point, it's just for having some idea about this tool and how do looks like a testing-setting up session. Values used here have been like this:

```
[IRoamer.IRoamer]
StartDelay=3.000000
DesireRateTime=15.000000
GoalChance=0.720000
bUseSlowMode=True
bDebug=True
GoalsUsed=28
```

That's all, folks!