Aerisarn's Spawn System

Documentation release 1

Introduction:

The spawn system presented in this paper aims to provide efficient tools for builders to create a multiplayer-oriented module. The proposed solution has the following characteristics:
1) Ability to create dynamically-placed creatures, objects and traps in any area
2) Creating these objects on-demand, or only when there are actually playing characters in area
3) Automated destruction process of any created object after a configurable despawn time after last player leaves area (default 60.0 seconds, see ais_ss_inc for details and reconf).

Installation:

To activate the system is necessary to execute in the events onEnter, onExit and onuserdefined of every desidered, calls to the respective functions AISSS_OnAreaEnter (), () AISSS_OnAreaExit and AISS_HandleAreaEvents (). If you do not have these scripts, you may use those provided with demonstration module called "ais_spawn_system.mod".

Operation and interface for builders:

The system operates through a series of local variables placed on the areas or on specific-tagged waypoints and, if you're not familiar with the creations of such variables, you may need to acquire such expertise first to customize the system; it expects to find the following local variables on the desidered areas:

iAISS_selectedEvent (integer): integer that represents the number of events to be handled at the entrance of the first character in the area in which is located. This whole goes in pairs with strings ***sAISS_selectedEvent_<i>,*** specifying precisely the events to manage. The possible events, and hence the possible values for the above strings are:
1) "**wandering**": generation of selected monsters in random positions
2) "**fixed**": generation of selected monsters in fixed positions (through the use of waypoints)
3) "**rp_traps"** generation of selected traps in random positions
4) "**fp_traps**" generation of selected traps in fixed positions
5) "**rp_plac**" generation of selected placeable objects in random positions
6) "**fp_plac**" generation of selected placeable objects in fixed positions

For each of these events additional local variables are needed to specify desidered parameters; Here is presented the list of parameters available for each event:

1) requests the following variables:

sAISS_resref_group_ <i> (string): this holds the blueprint of i-th group of monsters to be spawned. <i> must be replaced by an integer; for example, to create a group of standard goblin set sAISS_resref_group_1 to "nw_goblina". To add another group of goblin type b, set the variable sAISS_resref_group_2 to "nw_goblinb". In all the following variables, substitute <i> with appropriate natural number. If confused, refer to the demonstration module
iAISS_size_group_ <i> (int): Specifies the size of th group; following the example above, for 10 goblin a-typed, set iAISS_size_group_1 = "10", and to provide 5 goblin b-typed, set iAISS_size_group_2 = "5".

2) To specify the positions of groups of monsters, it is necessary to place waypoints appropriately tagged as the following string: "aiss_spawn_<area_tag>" where <area_tag> must be replaced with the current area tag. After that, simply put on the waipoints variables like the previous case. When the event occurs, each waypoint so tagged will mark the spawning location for monsters groups with the configuration explained above. Consequently, every tagged waypoint must contain the variables ***sAISS_resref_group_ <i>*** and ***iAISS_size_group_ <i>*** configured properly.

3) As in the case of random monsters, you must set variables over the area. The variables, and then the options available are as follows:
SAISS_trap_group_type_ <i> (string): type of the i-th group of traps, admissible values for this string are as follows: acid, acid_splash, electrical, fire, frost, gas, holy, negative, sonic, spike, tangle

SAISS_trap_group_level_ <i> (string): level of the i-th group of traps, and options are: minor, avarage, strong, deadly, epic (in order of trap power). NOTE: the epic one is only available for the following types: acid, electrical, fire, frost.

iAISS_trap_group_size_ <i> (int): how many to-place-traps of the i-th type.
fAISS_trap_group_covered_area_ <i> (float): size of the areas of i-th group of traps (surface ground covered).

bAISS_trap_group_not_detectable_ <i> (int): If this value is set to 1, the traps of i-th group will not be identifiable. If not specified, the traps will be identifiable.
iAISS_trap_group_detect_dc_ <i> (int): If traps are identifiable, this value indicates the DC to locate traps belonging to the i-th group. If you do not specify a value, default trap one will be used.
bAISS_trap_group_not_disarmable_ <i> (int): As with the detectability, but refers to the ability to disarm traps;
i***AISS_trap_group_disarm_dc_ <i>*** (int): As with disarmable traps, for the detectability
sAISS_trap_group_keytag_ <i> (string): If set, this indicates the tag necessary to remove the traps belonging to i-th group
bAISS_trap_group_not_recoverable_ <i> (int): As with the detectability, but for the recovery of the trap.
sAISS_trap_group_disarm_ <i> (string): ADVANCED: This string allows you to associate a custom script to the onDisarm event of the i-th traps group.
sAISS_trap_group_triggered_ <i> (string): ADVANCED: same meaning of onDisarm, but related to onTriggered.

4) as in the case of the monsters in fixed positions, it is necessary to include waypoints with tags necessarily equal to aiss_trap_spawn_ <tag_area>, where <tag_area> must be replaced with the current area tag. On each waypoint you can specify variables presented earlier for random traps, which will be this time only concerning the trap be placed in the waypoint location. Consequently variables are:
sAISS_trap_type, ***sAISS_trap_level***, ***fAISS_trap_covered_area***, ***bAISS_trap_not_detectable***, ***iAISS_trap_detect_dc***, ***bAISS_trap_not_disarmable***, ***bAISS_trap_disarm_dc***, ***sAISS_trap_keytag***, ***bAISS_trap_not_recoverable***, ***sAISS_trap_disarm***, ***sAISS_trap_triggered***. For dercription see the above explanation.

5) As in the case of traps and monsters in random position, you must specify on the following variables over the area:
sAISS_plac_resref_group_ <i> (string): blueprint of (resref)-th group positioned
iAISS_plac_size_group_ <i> (int): number of positioned-th type (resref)

6) as in the case of monsters and traps spawned in fixed positions, it is necessary to include waypoints with tags necessarily equal to aiss_plac_spawn_ <tag_area> where <tag_area> must be replaced with the current area tag. Unlike for traps, each location may be associated with more than one placeable, but that choice is essentially due to the need to provide visual effects with positioned. To avoid side effects, it is advisable to not associate more than a solid placeable for each waypoint. Each waypoint must be provided with the following variables:
sAISS_plac_resref_group_ <i> (string): blueprint(resref) of i-th group of placeables.
iAISS_plac_size_group_ <i> (whole): size of the i-th group.

Considerations

Unfortunately neverwinter does not generate random "valid" positions for placeables and traps Therefore is possible that some objects may end up in places unreachable by the characters, or may not be usable.

Credits:

My special thank to:

Kesda, for helping with some functions

Sparda, for beta-testing

B.M. for understanding