Теория

В скриптах есть одна единственная функция, отвечающая за спавн объектов:

alife():create(section,position,levelvertex,gamevertex)

Строго говоря, их две: create и create\_ammo но различия между ними не существенны. Imp 22:45, 23 июля 2007 (EEST)

Первый параметр - секция в конфигурациях, описывающая объект, например "bolt","med\_kit" - это простые секции, простых объектов а есть объекты, которые переходят в онлайн/оффлайн, это неписи, монстры и так далее, например mil\_killer\_respawn\_2 - спавнится снайпер группировки киллеров.

С позицией, думаю объяснять не надо, только существует нюанс - высота это Y, а не Z. Задать позицию можно такой конструкцией vector():set(x,y,z), где x, y и z - координаты точки на уровне, где спавним объект.

Вертекс в игре Сталкер не тоже самое, что вертекс в трехмерной графике. Vertex - определенная зона малого размера за которой закреплен индивидуальный номер в пределах уровня или целой игры. level\_vertex определяет зону на уровне, по которой можно пройти NPC. game\_vertex определяют зоны для свободного хождения, т.е. под руководством ИИ, а также между уровнями. Вертексы имеют одно из первостепенных значений. Точность положения указывается уже через точку спавна. Например, можно получить вертекс ближайший к актору - db.actor:level\_vertex\_id()

game\_vertex\_id нужен для того, чтобы указать на какой карте спавнить объект.

Соответственно, чтобы заспавнить что-нибудь на другой карте, достаточно указать game\_vertex\_id в четвертом параметре Например:

db.actor:game\_vertex\_id()

Итак, чтобы, например, заспавнить болт под ногами актора, пишем:

alife():create("bolt", db.actor():position(), 1, db.actor:game\_vertex\_id())

Почему 1, а не level\_vertex\_id? Проверено - разницы особой нет, какой level\_vertex\_id, хотя в некоторых случаях надо прописывать валидный вертекс, а то предмет может просто заспавнится не там, где планировалось... Но по большей части все проходит нормально и с единицей. (Игнорирование level\_vertex\_id может приводить к проваливанию произведенных предметов/персонажей под землю.) А вот game\_vertex решает все - он указывает на каком уровне спавнить предмет, поэтому его надо указывать. Теоретически можно просто найти для каждого уровня по одному game\_vertex'у и использовать их в скриптах. На самом деле game\_vertex показывает какой фрагмент карты используется (вся карта разбита на кусочки имеющие сквозную нумерацию по всем уровням и game\_vertex выбирает нужный) соответсвенно неправильное использование черевато....

Кроме того - есть еще один параметр - ID объекта, если указать ID NPC или актора - то предмет заспавнится у него в инвентаре.

Пример (спавним артефакт Медуза в инвентаре у актора):

alife():create("af\_medusa", db.actor():position(), 1, db.actor:game\_vertex\_id(), db.actor:id())

Функция спавна возвращает серверный объект, то есть ни NPC, ни монстра ни что-либо еще.

Серверный обьект позволяет свеже созданного NPC или тайник затарить разными рулезами/артефактами. Например, вот так создадим перед входом к Сидоровичу долговца и засунем в него пачку патронов:

local obj

local a = vector() -- Задаем тип переменной

local dir = db.actor:direction()

a.x = -243.61 -- координата X

a.y = -19.52 -- высота Y

a.z = -127.17 -- координата Z

obj = alife():create("bar\_dolg\_respawn\_3", a, 13193, 8, 65535)

alife():create\_ammo("ammo\_9x18\_fmj",

obj.position,

obj.m\_level\_vertex\_id,

obj.m\_game\_vertex\_id,

obj.id,

20) -- число патронов

Кстати, create\_ammo - практически тоже самое, что и create, разница в том, что create\_ammo предназначена специально для спавна патронов и позволяет создавать неполные пачки патронов. Возможно есть еще какие-то отличия. Стоит учесть, что сами авторы игры спавнят патроны исключительно через create\_ammo. Imp 22:38, 23 июля 2007 (EEST)

Просто минимальный набор - координаты, ID, секция,а из него (серверного объекта) обычно нужен только ID, так как по ID можно получить этот самый серверный объект:

(alife():object(id))

Его можно использовать, чтобы поставить метку, например, но я его лично использую для других целей - спавн сложных объектов, конкретно – NPC.

Например надо решить следующую задачу - надо создать наемника, сменить ему группировку и изменить его инвентарь, ну и в нагрузку - сделать другом для игрока.

В определенный момент заспавненый объект переходит онлайн, в этот момент вызывается callback - net\_spawn.

Что мы делаем? Сверяем ID онлайн объекта с сохраненным ID!

Если они совпадают, например так:

if obj:id()==saved\_id then ...

Важно то, что у серверного объекта ID - это параметр, а у онлайнового объекта ID получается с помощью функции. Это важно, а то можно прогореть.

Итак, мы поймали нашего киллера по ID.

Далее все очень просто - вызываем команды для спавна гаусса и патронов к нему в инвентаре NPC (см. выше), меняем группировку специальной функцией, и делаем его другом.

Зачем такие сложности? Просто в оффлайне NPC как бы не существует, есть только косвенное упоминание о нем, и, плюс, все эти функции работают именно с объектом типа "NPC", а не с серверными объектами.

Более простая функция спауна в одну строчку:

Пример:

alife():create("stalker",vector():set(0,0,0),1,135) -- спаунит сталкера на водокачке. (Игра ЧН, на уровне Болото)

alife():create(section,position,levelvertex,gamevertex)

section - это секция предмета

position - Это его позиция (Через vector()) Брать в игре с помощью консольной команды rs\_stats 1 или rs\_cam\_pos 1 (Тока в ЧН)

levelvertex - Пусть равно 1

gamevertex - Его Game Vertex, определяет уровень где будет спаун.

GV можно взять либо скриптом, либо через распакованный all.spawn.

[править]

Практика (часть 1)

1. Чтобы не повторяться в описании создания нового квеста, просто изучите статью по созданию квестов от Fr3nzy – лучшей статьи на эту тему я просто не видел :) Мы просто свяжем все воедино и научимся спавнить объекты из скрипта.

Небольшое отступление:

Почему предпочтительнее делать спавн скриптом, а не через тот же xrSpawner? Программа xrSpawner, при всех своих достоинствах, обладает одним недостатком, а именно – она делает спавн через файл all.spawn, что приводит к:

\* Невозможности совместить два мода, такой спавн использующих

\* Необходимости каждый раз начинать новую игру

При спавне через скрипт ситуация иная: в подавляющем большинстве случаев, ранее сохраненные игры будут работать, что не может не радовать :)

Итак, определимся с квестом.

Задача: после разговора с Сидоровичем спавним зомби на территории фабрики в первой локации. Для того, чтобы не повредить оригинальный сюжет игры, задание будет выдаваться после прохождения квеста с флешкой Шустрого, так как появись там зомби одновременно с бандитами и Шустрым... я думаю, исход боя предрешен :)

Реализация: Постараюсь описать все действия максимально подробно, буквально по шагам. Первым делом запустите игру :)

В консоли введите команду:

rs\_stats on или rs\_stats 1

Тем самым мы включаем вывод информации на экран. Далее вводим еще одну команду:

demo\_record 1

И «летим» на фабрику. Нам нужно выбрать место для спавна объектов и данный режим как нельзя лучше подходит для реализации задуманного. Помещаем камеру в точке предполагаемого спавна и записываем координаты - у меня получились 115, -6, -16.

Для выхода из режима demo\_record нажимаем Esc, в консоли пишем rs\_stats off или rs\_stats 0 (убираем вывод информации).

Другой способ получения тех же сведений - прийти в нужное место и запустить там скрипт, который выдаст все нужные координаты. Я пользуюсь следующим скриптом (вызываю общеизвестным способом, через main\_menu):

function main\_menu:main\_cheat\_f3()

-- Выдадим сообщение о нашем местоположении

local text

local vid

local gvid

local a = vector() -- Тип переменной

local text

a = db.actor:position() -- Наше положение в координатах

vid = db.actor:level\_vertex\_id()

gvid = db.actor:game\_vertex\_id()

text = "Позиция:\\nX= "..a.x.."\\nY= "..a.y.."\\nZ= "..a.z.."\\nlevel\_vertex= "..vid.."\\ngame\_vertex\_id= "..gvid

news\_manager.send\_tip(db.actor, text, nil, nil, 30000)

end

В результате не нужно эксперементировать мы сразу получаем все, в том числе и level\_vertex и game\_vertex. Imp 22:38, 23 июля 2007 (EEST)

Выходим из игры, идем в папку с установленной игрой и создаем каталог gamedata (предполагается, что «лепим» свой «мод» на «чистую» игру, без установленных модов, и имеем распакованные ресурсы игры в папке, скажем, gamedata source).

В папке gamedata создаем папку config, а в ней - папку creatures. Скопируем из оригинальной папки файл m\_zombie.ltx и откроем его на редактирование.

В файлах игры присутствуют 5 моделей гражданских зомби:

файлы zombi\_1.ogf, zombi\_1\_ghost.ogf, zombi\_2.ogf, zombi\_trup.ogf, zombi\_trup\_2.ogf

Вернем в игру их всех :)

Уже имеются секции:

[zombie\_weak]:m\_zombie\_e, [zombie\_normal]:m\_zombie\_e, [zombie\_strong]:m\_zombie\_e и [zombie\_immortal]:zombie\_strong.

Два последних типа используют одну и ту же модель zombi\_trup.ogf, хм... непорядок, исправляем. Последняя секция выглядит теперь так:

[zombie\_immortal]:zombie\_strong

$spawn = "monsters\zombies\zombie\_immortal"

visual = monsters\zombi\zombi\_trup\_2

panic\_threshold = 0.05

Добавим пятую модель.

Для этого в конце файла создадим секцию:

[zombie\_ghost]:zombie\_strong

Это означает, что наш пятый зомби наследует все параметры zombie\_strong, мы добавим лишь визуальное представление.

Пишем дальше:

$spawn = "monsters\zombies\zombie\_ghost"

visual = monsters\zombi\zombi\_1\_ghost

Все. Сохраняем изменения и закрываем файл.

2. Пишем скрипт спавна. В папке gamedata создаем новую папку scripts, в ней создаем новый текстовый документ и называем его esc\_zombie.script.

Отступление третье:

При написании статьи использовался оригинальный скрипт zombie\_story.script из horror-mod’а. Концепция спавна перенесена практически без изменений, поэтому на авторство этого способа спавна я никоим образом не претендую :)

Итак, открываем наш пустой файл на редактирование, первой строкой объявляем переменную, в которой хранятся наши зомби:

local zombie\_types = {"zombie\_weak", "zombie\_normal", "zombie\_strong", "zombie\_immortal", "zombie\_ghost"}

Далее пишем функцию:

function spawn\_zombies( position, total )

local zombie\_index -- тип зомби из массива zombie\_types

local new\_pos, x\_offset, z\_offset -- объявляем переменные

for zombie\_index=1, total do -- крутим цикл столько раз, сколько задает переменная total

x\_offset = math.random(5) -- случайное (рандомное) x от 1 до 5

z\_offset = math.random(5) -- случайное (рандомное) z от 1 до 5

new\_pos = position -- передаем координаты в функцию

new\_pos.x = new\_pos.x + x\_offset -- прибавляем к указанной нами координате x полученное выше рандомное x

new\_pos.z = new\_pos.z + z\_offset -- прибавляем к указанной нами координате z полученное выше рандомное z

-- Ниже, собственно и вызывается функция спавна случайного типа зомби zombie\_types[math.random(5)] привязанного к нашим координатам

alife():create(zombie\_types[math.random(5)],new\_pos,db.actor:level\_vertex\_id(),db.actor:game\_vertex\_id())

end

end

И последнее:

function zombie\_story\_1( actor, npc )

-- десять зомби на фабрике (Кордон)

local spawn\_point = vector():set( 115, -6, -16 ) -- здесь указываем координаты,

-- выбранные нами для спавна, когда «летали» камерой :)

spawn\_zombies( spawn\_point, 10 ) -- собственно вызов предыдущей функции

-- с передачей ей координат и количества объектов

end

Все. Сохраняем и закрываем файл.

Продолжаем разговор :)

Для того, чтобы игра не вылетала после того, как мы добавили новый тип монстров, их нужно добавить в файл xr\_statistic.script. Итак, скопируем этот файл из папки игры scripts в нашу папку к файлу esc\_zombie.script и откроем на редактирование.

Добавим в local killCountProps к монстрам строчку:

zombie\_weak = 1, zombie\_normal = 2, zombie\_strong = 3, zombie\_immortal = 4, zombie\_ghost = 5

В local sect\_alias после строки:

zombied\_novice = 1, zombied\_experienced = 2, zombied\_veteran = 3, zombied\_master = 4,

Допишем эти строки:

zombie\_weak = "zombie\_weak", zombie\_normal = "zombie\_normal", zombie\_strong = "zombie\_strong",

zombie\_immortal = "zombie\_immortal", zombie\_ghost = "zombie\_ghost",

А ниже в monster\_classes строчку:

[clsid.zombie\_s ] = "zombie"

В функцию getNpcType(npc) добавляем конструкцию:

elseif npc:character\_community() == "zombie" then

community = "zombie"

Сохраняем изменения и закрываем файл.

Все будет работать на ура, пока мы не попробуем обыскать убитого зомби. Как только мы это сделаем, игра вылетит с примерно такой ошибкой.

Expression : fatal error

Function : CInifile::r\_string

File : D:\xray-svn\xrCore\Xr\_ini.cpp

Line : 351

Description : <no expression>

Arguments : Can't find variable icon in [zombie\_weak]

Все верно – игра не знает какую иконку нам показывать для зомби. Иконки монстров хранятся в файле ui\_npc\_monster.dds. Здесь есть два варианта:

\* Если дружите с Фотошопом, отредактировать этот файл (нарисовать, добавить иконки);

\* Взять готовый из любого мода, естественно, с разрешения авторов мода. Сейчас мы пропустим данный аспект и присвоим нашим зомби иконки контролера :)

Вернемся к файлу m\_zombie.ltx и в секцию [m\_zombie\_e]:monster\_base впишем параметр

icon = ui\_npc\_monster\_kontroler

Все. Вылетов не будет.

3. Тема данной статьи не предусматривает подробного описания того, как сделать новый диалог. В начале статьи я упомянул источник, где можно найти исчерпывающую информацию по созданию диалогов, могу также привести в пример статью по созданию диалогов от BAC9-FLCL.

Нам нужно просто проверить работоспособность скриптового спавна, поэтому я приведу просто собственно сам измененный диалог из файла dialogs\_escape.xml:

<dialog id="escape\_trader\_talk\_info">

………

<phrase id="999">

<text>escape\_trader\_talk\_info\_999</text>

<next>7770</next>

<next>9991</next>

<next>9992</next>

<next>9993</next>

<next>9994</next>

<next>9995</next>

<next>9996</next>

</phrase>

<phrase id="9992">

<text>escape\_trader\_talk\_info\_9992</text>

<next>99922</next>

</phrase>

<phrase id="99922">

<text>escape\_trader\_talk\_info\_99922</text>

<next>9996</next>

<next>9995</next>

</phrase>

<phrase id="9993">

<text>escape\_trader\_talk\_info\_9993</text>

<next>99933</next>

</phrase>

<phrase id="9995">

<text>escape\_trader\_talk\_info\_9995</text>

</phrase>

<phrase id="3121">

<text>escape\_trader\_talk\_info\_3121</text>

<next>9996</next>

<next>9995</next>

</phrase>

<phrase id="3131">

<text>escape\_trader\_talk\_info\_3131</text>

<next>9996</next>

<next>9995</next>

</phrase>

<phrase id="41">

<text>escape\_trader\_talk\_info\_41</text>

<next>9996</next>

<next>9995</next>

</phrase>

<!------Наш диалог: Начало------->

<phrase id="7770">

<text>escape\_trader\_talk\_info\_7770</text>

<next>7771</next>

</phrase>

<phrase id="7771">

<text>escape\_trader\_talk\_info\_7771</text>

<next>7772</next>

<next>7773</next>

</phrase>

<phrase id="7772">

<text>escape\_trader\_talk\_info\_7772</text>

<next>7777</next>

</phrase>

<phrase id="7773">

<text>escape\_trader\_talk\_info\_7773</text>

<next>7779</next>

</phrase>

<phrase id="7779">

<text>escape\_trader\_talk\_info\_7779</text>

<next>9996</next>

<next>9995</next>

</phrase>

<phrase id="7777">

<text>escape\_trader\_talk\_info\_7777</text>

<action>esc\_zombie.zombie\_story\_1</action>

<next>9996</next>

<next>9995</next>

</phrase>

<!------Наш диалог: Конец------->

<phrase id="51">

<text>escape\_trader\_talk\_info\_51</text>

<next>9996</next>

<next>9995</next>

</phrase>

……

</dialog>

И также связанный с ним файл stable\_dialogs\_escape.xml. В самом начале файла пишем следующее:

<string id="escape\_trader\_talk\_info\_7770">

<text>Происшествий никаких не было?</text>

</string>

<string id="escape\_trader\_talk\_info\_7771">

<text>Да знаешь... Вроде как тихо все у нас. Хотя, вот, вспомнил! Говорили мне

на днях, что на фабрике, ну, там, где бандюки околачиваются постоянно, видели какиих-то то ли

людей, то ли призраков... Мало ли что спьяну почудится - я и сказал этим паникерам, мол,

закусывать надо! Хех, блин, алкаши...</text>

</string>

<string id="escape\_trader\_talk\_info\_7772">

<text>Дык мне по любому мимо фабрики топать - заодно и посмотрю на этих

"людей-призраков".</text>

</string>

<string id="escape\_trader\_talk\_info\_7773">

<text>Да я как-то не собирался в ту сторону...</text>

</string>

<string id="escape\_trader\_talk\_info\_7779">

<text>Ну, смотри сам, все равно будь осторожен.</text>

</string>

<string id="escape\_trader\_talk\_info\_7777">

<text>Ага. Сходи, проветрись. Потом зайдешь, расскажешь, что там и как.</text>

</string>

<string id="esc\_bridge\_soldiers\_start\_11">

<text>Здесь проход воспрещён, сталкер.</text>

</string>

Все. Можно запускать игру, идти на Кордон, после разговороа с Сидоровичем, в зависимости от выбранного Меченным решения, бежим на фабрику и … смотрим сами :)

Готовые файлы примера(перезалито)

Spawn Lib

Домашнее задание - вернуть в игру 6-ой тип гражданского зомби :)

Продолжение следует…

[править]

Практика (часть 2)

4. Сегодня мы закончим с зомби в полном объеме – добавим их описания в энциклопедию, добавим иконки, и разберемся с «домашним заданием»:) Думаю, что внимательно изучив эту статью, вы сами сможете через скриптовые функции восстановить любого персонажа, не вошедшего в финальный релиз игры. Если у кого хватит времени и желания, те могут даже написать что-то типа «Ночи живых мертвецов» :) - серию квестов, связанную своим собственным сюжетом. Итак, «домашнее задание» - добавляем в игру шестого зомби.

\* Создаем в папке gamedata папку meshes, в ней папку monsters, а там – папку zombi. В папке meshes хранятся модели персонажей, объектов, окружения, присутствующих в игре. Как я уже говорил выше, в игре представлено 5 моделей гражданских зомби, а вот текстур – 6. Также немного смутила нумерация моделей – 1, 2, 4... А где третий? Наверное, сбежал :)

\* Скопируем из папки с оригинальными файлами игры в созданную нами папку zombi файл zombi\_2.ogf и переименуем его в zombi\_3.ogf. Откроем файл нашей новой модели любым HexEditor’ом, я использую BiEd (Binary Editor 1.00). Так как данная статья рассчитана не только на «продвинутых юзеров», но и на «обычных чайников», которые, тем не менее, хотят «что-нить замутить» и при этом не сильно «парить моск», я не буду рассказывать здесь про адресацию, двухбайтовую запись и т.д., я просто наглядно покажу, где что поправить :)

Файл модели до редактирования

Файл модели после редактирования

На скринах видно, что мы просто изменили для этой модели путь к текстуре. Все. Без использования 3D-редакторов и затраты кучи времени на обработку модели в них, мы получили абсолютно новую (на внешний вид) модель. :)

5. Теперь пропишем нашего нового зомби во все файлы, которые мы создали ранее. В файл m\_zombie.ltx в самый конец добавляем секцию:

[zombie\_old]:zombie\_normal

$spawn = "monsters\zombies\zombie\_old"

visual = monsters\zombi\zombi\_3

в файле esc\_zombie.script изменяем массив в первой строке:

local zombie\_types = {"zombie\_weak", "zombie\_normal", "zombie\_strong", "zombie\_immortal",

"zombie\_old", "zombie\_ghost"}

В функции spawn\_zombies изменяем строку спавна:

alife():create(zombie\_types[math.random(6)],new\_pos,

db.actor:level\_vertex\_id(),db.actor:game\_vertex\_id())

в функции zombie\_story\_1 меняем число объектов на кратное 6-ти (необязательно):

spawn\_zombies( spawn\_point, 12 )

Всё. Сохраняем и закрываем.

6. Копируем в папку gamedata\config\gameplay\ файл encyclopedia\_mutants.xml, добавляем описание зомби в энциклопедию:

<!-------------------------------- Zombieg ----------------------------->

<article id="mutant\_zombieg\_general" name="Zombieg" group="Mutants">

<texture>ui\_npc\_monster\_zombieg</texture>

<text>enc\_mutant\_zombieg\_general</text>

</article>

И в связанный с ним файл string\_table\_enc\_mutants.xml в папке gamedata\config\text\rus\ добавляем:

<string id="Zombie">

<text>Зомби, гражданский</text>

</string>

<string id="Zombieg">

<text>Зомби-призрак, гражданский</text>

</string>

<string id="enc\_mutant\_zombieg\_general">

<text>Зомби-привидение отличается от обычного зомби лишь тем, что

воздействие Выжигателя мозгов полностью разрушило не только структуру личности, но и

тело, поэтому зомби-призрак несколько более живуч по сравнению с обычным зомби.</text>

</string>

<string id="enc\_mutant\_zombie\_general">

<text>Воздействие Выжигателя мозгов полностью разрушает структуру личности,

оставляя только телесную оболочку.\n\n Побродив немного по Зоне, лишённые разума тела

начинают превращаться в настоящих зомби. Из рефлексов у них остаются лишь самые примитивные,

оружие и экипировка скоро приходят в негодность. В результате зомби становятся собой ни чем

иным, как медлительными полутрупами, которых наличествуют лишь два эффективных раздражителя:

еда и сон. Зомби совершенно неразборчивы в выборе пищи и питья, поэтому их тела буквально

пропитаны радиацией и токсинами. Как правило, эти существа бесцельно бродят по Зоне или,

словно трупы, валяются внутри заброшенных построек. Однако, лишь только зомби почует близкое

присутствие живого человека, он сразу же пытается атаковать. Умудрённые опытом сталкеры

стараются обходить эти неуклюжие опустошённые оболочки.</text>

</string>

Копируем сюда же файл stable\_statistic\_caption.xml и изменяем в нем 3 строчки:

<string id="zombie\_normal">

<text>зомбированный, гражданский</text>

</string>

<string id="zombie\_strong">

<text>зомби-призрак, гражданский</text>

</string>

<string id="zombie\_weak">

<text>зомби, гражданский</text>

</string>

Сохраняем и закрываем.

7. И последнее – добавим иконки. Скажу сразу, воспользовался готовым файлом, уже содержащим иконки зомби и других «восстановленных монстров» (спасибо Fr3nzy). Поэтому просто скопируйте файл ui\_npc\_monster.dds из архива в папку gamedata\textures\ui\, а файл ui\_npc\_monster.xml – в папку gamedata\config\ui\. Если вы хотите сделать собственные - прочитайте урок по изменению текстур.

Вкратце, что описывает файл ui\_npc\_monster.xml: в нем задаются координаты иконок, расположенных в файле ui\_npc\_monster.dds, применительно к каждому типу монстров в игре.

Заключительный штрих. Откройте файл m\_zombie.ltx и в первой секции замените строку

icon = ui\_npc\_monster\_kontroler

На:

icon = ui\_npc\_monster\_zombie

В секцию [zombie\_ghost] добавьте строку:

icon = ui\_npc\_monster\_zombieg

Сохраняйте изменения. Всё.

Вот готовые файлы примера. Удачи и спасибо за внимание :)

[править]

Спавн NPC

Эта часть статьи написана Arhet и создана на примере того, как в SRP Mod были созданы NPC группировки "Грех".

Задействуем файлы:

\* gamedata\config\gameplay\character\_desc\_escape.xml

\* gamedata\config\gameplay\npc\_profile.ltx

\* gamedata\config\creatures\spawn\_sections.ltx

Начнем с character\_desc\_escape.xml. Описание что значит каждая строка писать не буду, так как все до меня уже сделано.

Тут создадим нового персонажа:

<specific\_character id="esc\_dark\_stalker\_1" team\_default = "1">

<name>GENERATE\_NAME\_bandit</name>

<icon>ui\_npc\_dark\_1</icon>

<map\_icon x="0" y="0"></map\_icon>

<bio>sim\_stalker\_novice\_bio</bio>

<class>esc\_dark\_stalker\_01</class> - вот тут наш новый класс кстати.

<community>dark\_stalker</community> <terrain\_sect>stalker\_terrain</terrain\_sect>

<money min="200" max="600" infinitive="0"></money>

<rank>198</rank>

<reputation>-24</reputation>

<visual>actors\dark\_stalker\dark\_stalker\_1</visual>

<snd\_config>characters\_voice\human\_01\monolith\</snd\_config>

<crouch\_type>-1</crouch\_type>

<panic\_treshold>0</panic\_treshold>

<supplies>

[spawn] \n

wpn\_dark\_colt1911 \n

ammo\_11.43x23\_hydro \n

#include "gameplay\character\_items.xml" \n

#include "gameplay\character\_drugs.xml" \n

#include "gameplay\character\_food.xml"

</supplies>

#include "gameplay\character\_criticals\_3.xml"

<start\_dialog>dm\_hello\_dialog</start\_dialog>

<actor\_dialog>dm\_cool\_info\_dialog</actor\_dialog>

</specific\_character>

Впишем наш код после какого-нибудь </specific\_character>

Теперь идем в npc\_profile.xml и туда вгоняем:

<character id="esc\_dark\_stalker\_1">

<class>esc\_dark\_stalker\_01</class>

<specific\_character>esc\_dark\_stalker\_1</specific\_character>

</character>

Теперь надо заняться spawn\_sections.ltx. Скрипт будет «брать NPC» как раз из этого файла. Пишем туда:

\* [esc\_dark\_stalker\_1]:stalker – имя секции для скрипта..

\* $spawn = "respawn\esc\_dark\_stalker\_1"

\* character\_profile = esc\_dark\_stalker\_1 – ссылка на описание нашего NPC

\* spec\_rank = regular – спец ранг

\* community = dark\_stalker – группировка.

Теперь берем любой скрипт спавна NPC, вгоняем туда имя секции из spawn\_sections.ltx и ву-а-ля.

Ну к примеру создаем свой файл со скриптом спавна, назовем его - spaw\_new\_npc.script и запишем туда вот такую функцию (взял из статьи):

function spaw\_n\_n() --тут задается сама функция которая в последующем будет активирована к примеру с помощью диалога

local a = vector() -- db.actor:position() -- тип переменной, которая будет задействована в спавне, вместо кооржинат будет писаться a

a.x = -220.21

a.y = -19.93

a.z = -158.98

alife():create("wpn\_ak47",a,33447,47,65535) -- здесь сам спавн объекта, a - тип переменной которую мы указали выше

end

Вот и все весь скрипт! Теперь надо его активировать, ну к примеру через диалог, после диалога вставляем:

<action>spaw\_new\_npc.spaw\_n\_n</action>

распишу что значит:

<action>название скрипта.функция которую необходимо активировать</action>

Тока забыли что данному НПС будет присвоенная дефолтовая логика (т.е. тупо хожу куда сам не знаю) забыли упомянуть про (ххх - лока):

ххх gulag.script (действия НПС! Где можно: где можно присвоить статус кампера; волкера; и.д.т)

ххх gulag.ltx (дублер логики т.е. дублирует кратко действия НПС)

[править]

Авторы

Статья создана:

\* xStream

\* OffBar

\* Fr3nzy

\* Imp

\* Arhet

Статья исправлена:

\* Участник:Rekongstor

В статью добавили:

\* Участник:Byurrer

"спавн несколько нпс одновременно"

1. Сначала создаешь одного НПС по шаблону из 1 сообщения.

2. Далее меняешь свой скрипт на этот:

function название\_функции ()

for i =1, 1000000 do --за место 100000 поставь то число сколько сталкеров ты хочешь заспавнить

alife():create("айди npс из spawn sections - типа agr\_stalker\_regular",vector():set(x,y,z),lv,gv)

end

end

3. вставляешь в диалог <action>ваше\_название\_скрипта.ваше\_название\_функции</action>

"как заспавнить полукругом(или в ряд) несколько НПС"

local angle = 0 --это первый вариант спавна

function spawn\_new()

local pos = vector():set(-241.74,-19.54,-127.23) -- нужные координаты

local dir = vector():set(0,0,1)

local lv = 14471 --левел вертекс позиции точки

local gv = 8 --гейм вертекс позиции точки

for i=1, 10 do --кол-во NPC

if (angle > 360) then

angle = angle - 360

end

local new\_dir = vector\_rotate\_y (dir, angle)

pos = pos:add(new\_dir:mul(5)) -- 5 - это дистанция от центра точки

alife():create("agr\_stalker\_regular", pos, lv, gv)

angle=angle+20 --диапазон изменения угла

end

end

local angle = 0 -- это второй вариант функции

function spawn\_new\_2()

--local pos = vector():set(-241.74,-19.54,-127.23) -- здесь они не нужны

local dir = vector():set(0,0,1)

local lv = 14471

local gv = 8

for i=1, 10 do

if (angle > 360) then

angle = angle - 360

end

local new\_dir = vector\_rotate\_y (dir, angle)

local lvid = level.vertex\_in\_direction(lv,new\_dir,5)

local lpos = level.vertex\_position(lvid)

alife():create("agr\_stalker\_regular", lpos, lvid, gv)

angle=angle+20

end

end

чуть не забыл - во второй функции собственно координаты точки вообще не нужны - а нужны только вертексы, так как позиция высчитывается по ним.

то есть строчка local pos = ... лишняя