В этой статье я попытаюсь передать практический опыт скриптового создания точек перехода между уровнями. Файл all.spawn меняться не будет, поэтому такой механизм более удобен для совмещения модов и не требует начала новой игры.

В качестве примера попробуем реализовать режим freeplay - продолжение игры после уничтожения О-Сознания. Для этого нужно создать как минимум две точки — возврат на ЧАЭС-1 со внутреннего дворика (после уничтожения О-Сознания) и переход из ЧАЭС-1 в Припять. К слову сказать, если вернуться на ЧАЭС-1, то там будет продолжать действовать таймер "выброса", который в конце-концов включит "deadzone". Отключить счетчик мне пока не удалось, но зато можно легко отключить сам "выброс" и всласть побегать по уровню.

[править]

Часть 1. Генерация работающего LEVEL\_CHANGER

Стандартной функции alife():create(…) недостаточно для создания полноценного level\_changer. Собственно методика создания сложных объектов описана в статье Один из методов спавна. Сложность заключалась лишь в порядке полей и свойствах Shape. Неоценимую помощь в этом вопросе оказала утилита ACDC (created by bardak).

Ниже я привожу код функции, которая создает и инициализирует level\_changer:

function create\_level\_changer(

p\_story\_id, -- STORY\_ID нового level\_changer (понадобится нам позже)

p\_position, -- вектор, координаты точки, в которой будет располагаться центр нового level\_changer

p\_lvertex\_id, -- level\_vertext\_id - идентифицируют уровень, на котором будет создан level\_changer

p\_gvertex\_id, -- game\_vertext\_id

p\_dest\_lv, -- level\_vertex\_id - идентифицируют уровень, на который level\_changer будет перебрасывать игрока

p\_dest\_gv, -- game\_vertex\_id

p\_dest\_pos, -- координаты точки, в которой на новом уровне окажется игрок

p\_dest\_dir, -- направрение взгляда игрока

p\_dest\_level, -- название уровня, например "L11\_Pripyat"

p\_silent -- следует задать 1, чтобы подавить вопрос о смене уровня (автоматический переход)

)

local obj = alife():create("level\_changer", p\_position, p\_lvertex\_id, p\_gvertex\_id)

level.map\_add\_object\_spot(obj.id, "level\_changer", "")

local packet = net\_packet()

obj:STATE\_Write(packet)

-- свойства cse\_alife\_object

local game\_vertex\_id = packet:r\_u16()

local cse\_alife\_object\_\_unk1\_f32 = packet:r\_float()

local cse\_alife\_object\_\_unk2\_u32 = packet:r\_u32()

local level\_vertex\_id = packet:r\_u32()

local object\_flags = packet:r\_u32()

local custom\_data = packet:r\_stringZ()

local story\_id = packet:r\_u32()

local spawn\_story\_id = packet:r\_u32()

-- свойства cse\_shape

local shape\_count = packet:r\_u8()

for i=1,shape\_count do

local shape\_type = packet:r\_u8()

if shape\_type == 0 then

-- sphere

local center = packet:r\_vec3()

local radius = packet:r\_float()

else

-- box

local axis\_x\_x = packet:r\_float()

local axis\_x\_y = packet:r\_float()

local axis\_x\_z = packet:r\_float()

local axis\_y\_x = packet:r\_float()

local axis\_y\_y = packet:r\_float()

local axis\_y\_z = packet:r\_float()

local axis\_z\_x = packet:r\_float()

local axis\_z\_y = packet:r\_float()

local axis\_z\_z = packet:r\_float()

local offset\_x = packet:r\_float()

local offset\_y = packet:r\_float()

local offset\_z = packet:r\_float()

end

end

-- свойства cse\_alife\_space\_restrictor

local restrictor\_type = packet:r\_u8()

-- свойства cse\_level\_changer

local dest\_game\_vertex\_id = packet:r\_u16()

local dest\_level\_vertex\_id = packet:r\_u32()

local dest\_position = packet:r\_vec3()

local dest\_direction = packet:r\_vec3()

local dest\_level\_name = packet:r\_stringZ()

local dest\_graph\_point = packet:r\_stringZ()

local silent\_mode = packet:r\_u8()

packet:w\_begin(game\_vertex\_id) -- game\_vertex\_id

packet:w\_float(cse\_alife\_object\_\_unk1\_f32)

packet:w\_u32(cse\_alife\_object\_\_unk2\_u32)

packet:w\_u32(level\_vertex\_id) -- level\_vertex\_id

packet:w\_u32( bit\_not(193) ) -- object\_flags = -193 = 0xFFFFFF3E

packet:w\_stringZ(custom\_data)

packet:w\_u32(p\_story\_id) -- story\_id

packet:w\_u32(spawn\_story\_id)

packet:w\_u8(1) -- количество фигур

-- packet:w\_u8(0) -- тип фигуры: сфера

-- packet:w\_vec3(vector():set(0, 0, 0)) -- sphere\_center

-- packet:w\_float(3.0)

packet:w\_u8(1) -- тип фигуры: box

packet:w\_float(2) -- axis\_x\_x

packet:w\_float(0) -- axis\_x\_y

packet:w\_float(0) -- axis\_x\_z

packet:w\_float(0) -- axis\_y\_x

packet:w\_float(4) -- axis\_y\_y

packet:w\_float(0) -- axis\_y\_z

packet:w\_float(0) -- axis\_z\_x

packet:w\_float(0) -- axis\_z\_y

packet:w\_float(4) -- axis\_z\_z

packet:w\_float(0) -- offset\_x

packet:w\_float(0) -- offset\_y

packet:w\_float(0) -- offset\_z

packet:w\_u8(3) -- restrictor\_type

packet:w\_u16(p\_dest\_gv) -- destination game\_vertex\_id

packet:w\_u32(p\_dest\_lv) -- destination level\_vertex\_id

packet:w\_vec3(p\_dest\_pos) -- destination position

packet:w\_vec3(p\_dest\_dir) -- destination direction (направление взгляда)

packet:w\_stringZ(p\_dest\_level) -- destination level name

packet:w\_stringZ("start\_actor\_99") -- some string, always const

packet:w\_u8(p\_silent) -- 1 for silent level changing

packet:r\_seek(0)

obj:STATE\_Read(packet, packet:w\_tell())

-- news\_manager.send\_tip(db.actor, "LC creation finished", nil, nil, 30000)

end

Для shape типа "box" загрузка координат методом packet:r\_matrix() окончилась неудачей. Я подозреваю, что не был прочитан вектор "offset", но точной уверенности нет, поэтому пока остановился на покомпонентной выборке и сохранении координат.

[править]

Часть 2. Создание точек перехода

Теперь следует написать функции создания нужных точек перехода и подключить их к игре. Сами функции просты:

function exit\_monolit()

if (not has\_alife\_info("freeplay\_activated1")) then

create\_level\_changer(11410, vector():set(-13.26, 47.71, 46.57), 200, 2417,

162109,

2384,

vector():set( 375.615, 0.224, 27.737 ),

vector():set( 0.0, 0.0 , 0.0 ),

"L12\_Stancia",

1)

db.actor:give\_info\_portion("freeplay\_activated1")

end

-- создается переход из ЧАЭС в Припять

create\_chaes2pripyat\_exit()

-- актер перебрасывается в level\_changer, возвращающий его на ЧАЭС, ко входу в бункер

db.actor:set\_actor\_position( vector():set(-13.26, 47.71, 46.57) )

end

function refuze\_o\_sozn()

if (not has\_alife\_info("freeplay\_activated2")) then

create\_level\_changer(21410, vector():set(946.872, 6.0, 167.66), 240852, 2637,

472710,

2280,

vector():set( 1062.15, -0.0982, -3.512 ),

vector():set( 0.0 , 0.0 , -1.0 ),

"L12\_Stancia",

1)

db.actor:give\_info\_portion("freeplay\_activated2")

end

-- создается переход из ЧАЭС в Припять

create\_chaes2pripyat\_exit()

-- актер перебрасывается в level\_changer, возвращающий его к правым воротам ЧАЭС

db.actor:set\_actor\_position( vector():set(946.872, 6.0, 167.66) )

end

function create\_chaes2pripyat\_exit()

-- создается переход из ЧАЭС в Припять

if (not has\_alife\_info( "exit\_chaes2pripyat\_created" )) then

create\_level\_changer(31410, vector():set( 917.35, 0.419, -316.35 ), 403866, 2401,

73868,

2117,

vector():set( 31.3, 3.0, 240.0 ),

vector():set( 0.0, 0.0, -1.0 ),

"L11\_Pripyat",

0)

db.actor:give\_info\_portion("exit\_chaes2pripyat\_created")

end

end

Функцию exit\_monolit я создал исключительно для тестирования, но решил оставить и тут. Вдруг кто-то захочет реализовать более сложный возврат: Меченого грузят в "грузовик смерти" и он снова приходит в себя на кордоне...

Функция exit\_monolit создает «тихий» переход на уровень ЧАЭС-1 и обычный - в начале уровня ЧАЭС-1 для возврата в Припять, после чего перебрасывает актера прямо внутрь созданного перехода. Функция refuze\_o\_sozn делает тоже самое, только игрок появляется перед воротами в правом верхем углу карты (мне кажется, что так более логично). Все телепорты защищаются уникальными info\_portions, дабы избежать их повторного создания, ведь игрок может захотеть «закончить» игру несколько раз.

Теперь подключение. Во-первых надо добавить новые info-portions. Я решил не изменять оригинальные файлы игры, а сделал для них (ну и для других тоже) отдельный файл

config\gameplay\\_info\_sa.xml

следующего вида:

<?xml version="1.0" encoding="windows-1251" ?>

<game\_information\_portions>

<info\_portion id="freeplay\_activated1"></info\_portion>

<info\_portion id="freeplay\_activated2"></info\_portion>

<info\_portion id="exit\_chaes2pripyat\_created"></info\_portion>

</game\_information\_portions>

Теперь в этот файл можно будет добавлять новые info\_portion, которые вы будете использовать в своих сюжетах. Подключается этот файл в system.ltx в секции «info\_portions»:

[info\_portions]

;список xml файлов, содержащих info\_portions

files = \_info\_sa, info\_portions, ....................

Кстати, именно так я рекомендую добавлять новые диалоги и новых персонажей. Это упростит процессы совмещения модов и аддонов.

Следующий шаг - подключение наших скриптов к игре. Для этого откройте файл

config\ui\ui\_movies.xml

найдите элементы «mov\_desire\_» (их пять, по количеству ложных концовок игры - все варианты общения с Монолитом). Внутри каждого элемента есть дочерние элементы «function\_on\_stop», которые задают функцию, запускающуюся сразу после ролика. Стандартное содержимое:

<function\_on\_stop>xr\_effects.game\_credits</function\_on\_stop>

Функция xr\_effects.game\_credits запускает финальные титры. Именно ее и нужно заменить на вызов нашей функции \_freeplay\_sa.exit\_monolit. Вот что должно получиться:

<function\_on\_stop>\_freeplay\_sa.exit\_monolit</function\_on\_stop>

<!-- original: <function\_on\_stop>xr\_effects.game\_credits</function\_on\_stop> -->

Старый элемент я советую не удалять, а закомментировать.

Чуть ниже «mov\_desire\_5» находится тэг ролика для концовки «Присоединение к О-Сознанию». Его мы трогать не будем - Меченый станет медузой. А вот после него - тэг для ролика «Отказ от О-Сознания»: «mov\_refuse\_osoznanie». Функцию завершения в нем заменим следующим образом:

<function\_on\_stop>\_freeplay\_sa.refuze\_o\_sozn</function\_on\_stop>

<!-- original: <function\_on\_stop>xr\_effects.game\_credits</function\_on\_stop> -->

Можно запускать и, если есть сохранения перед монолитом и/или о-сознанием, тестировать. Первые переходы срабатывают «тихо» - запрос на смену уровня не выдается (в параметре p\_silent задана 1). До перехода в Припять можно успеть добежать, пока не сработал выброс... Но это как-то неправильно, выброс надо остановить (он же произошел, пока мы были внутри станции). «По-честному» (скриптом, без модификации all.spawn) таймер выброса отключить не получится, но зато выброс можно подавить небольшим «хаком». Найдите файл xr\_logic.script, а в нем - функцию switch\_to\_section. Ее нужно модифицировать следующим образом:

-- Выполняет переключение на указанную секцию, если задана.

-- Если section == nil, остается работать старая секция.

function switch\_to\_section(npc, st, section)

if section == nil or section == "" then

return false

end

-- 15.03.2008 by SA:

-- отключает "смертельные зоны" на ЧАЭС после запуска режима FREEPLAY

-- таймер не отключается, но сам "выброс" не происходит

if (section == "sr\_aes\_deadzone") then

if (has\_alife\_info("freeplay\_activated1") or has\_alife\_info("freeplay\_activated2")) then

return false

end

end

... далее без изменений ...

end

Данный способ оставляет висящий на нулях таймер и все эффекты начала выброса, но сам «выброс» отключается.

И последнее: переход ЧАЭС1-Припять необходимо отметить на карте. Для этого существует вполне «легальный» механизм. Находим файл level\_tasks.script и в конце функции add\_lchanger\_location дописываем следующее:

-- aes (++16.03.2008 by SA):

obj = sim:story\_object(31410)

if obj then

level.map\_add\_object\_spot(obj.id, "level\_changer", "To Pripyat")

end

Здесь 31410 - story\_id нашего level\_changer, который создается нашей функцией create\_chaes2pripyat\_exit.

На этом пока все (мелкие огрехи типа направления взгляда игрока после смены уровня исправлю позже). Прошу тестировать и дополнять.

С уважением, sarthur.