Руководство по редактированию ИИ в игре Generals Zero Hour.

(версия от 07.12.2006)

Автор – Creator

e-mail: apestryakov@yandex.ru

Содержание

Введение.

В данном руководстве подробно рассказывается о том, как редактировать искусственный интеллект (ИИ) в игре Generals Zero Hour. Даются практические советы, рассматриваются часто совершаемые ошибки, даются ссылки на ресурсы Интернета. Это руководство не является на 100% точным и не претендует на полноту, но, дает ответы на большинство часто задаваемых вопросов.

Небольшая часть материалов, использовавшихся для написания этой документации, была взята с сайта Script Development Initiative, находящегося по адресу http://sdi.cncguild.net/

1. Открытие и сохранение ИИ.

Отредактировать ИИ можно с помощью программы WorldBuilder, поставляющийся в комплекте с игрой Generals Zero Hour. Скрипты, отвечающие за ИИ хранятся в файле SkirmishScripts.scb, который находится в директории Generals Zero Hour / Data / Scripts. Перед редактированием рекомендуется создать резервную копию этого файла на тот случай, если вы его запортите.

Открывается SkirmishScripts.scb следующим образом. Запустите WorldBuilder. В меню выберете Edit->Edit Player List. Внизу окна нажмите кнопку «Add Skirmish Players». Потом нажмите Ok. Далее выберите в меню Edit->Scripts. Откроется редактор скриптов. В его окне нажмите кнопку «Import Scripts», выберете файл SkirmishScripts.scb и нажмите кнопук «Открыть».

Оригинальный SkirmishScripts.scb специально запорчен фирмой EA Games. Он открывается неправильно. В папке с надписью SkirmishGLAStealthGeneral будут находиться скрипты для токсинового генерала, в папке SkirmishGLAToxinGeneral будут находится скрипты для подрывника и так далее, в каждой папке будут не те скрипты, которые надо. Если сохранить только что открытый SkirmishScripts.scb, то в начале игры сразу же будет появляться сообщение «Вы победили». Чтобы такого не было, нужно потратить пару часов на распихивание всех скриптов в нужные места. Но можно этого и не делать. Буржуи уже исправили эту ошибку и выложили исправленный SkirmishScripts.scb в интернет. Скачать его можно отсюда: http://sdi.cncguild.net/downloads/uploads/SkirmishScripts.zip

Для того, чтобы сохранить отредактированные скрипты, надо в окне редактора скриптов нажать кнопку «Export Scripts», в блоке «Include items referenced in the scripts» убрать все галочки, в блоке «Export mode» выбрать «Export all scripts» и только после этого нажать кнопку «Ok».

ВНИМАНИЕ! Никогда, ни после импорта скриптов, ни во время их редактирования, не нажимайте кнопку Esc (на клавиатуре) либо кнопку «Cancel» (в окне). Это приведет к потере всего, что вы сделали либо импотрировали. Всегда нажимайте кнопку «Ok».

После экспорта рекомендуется сохранить скрипты еще и в виде карты, выбрав в меню пункты File->Save. Сохранение скриптов в виде карты позволит в дальнейшем открывать их гораздо проще. Вместо того, чтобы каждый раз добавлять игроков и импортировать скрипты, вы просто открываете заранее сохраненную карту. Это удобнее и быстрее.

Для поиска ошибок существует кнопка «Verify». Она проверяет все скрипты (на это уходит около минуты). Скрипты, содержащие ошибку, помечаются красной пометкой. Конкретное место ошибки внутри самого скрипта помечается тремя вопросительными знаками. Рекомендуется проверить скрипты перед сохранением и исправить найденные ошибки. Однако, не стоит забывать, что и здесь программисты EA Games напортачили. Например, такие области как «Combat Zone» или объекты названные в стиле «Player1Garrison2» не являются ошибочными, хотя WorldBuilder на них упорно ставит красные пометки и знаки вопроса. Для того, чтобы такого идиотизма было поменьше, рекомендуется открывать скрипты с использованием карты AI Scripts Map, которая находится в комплекте «Generals Editing Utilites», изданным Deezire-ом. Комплект можно скачать отсюда: http://www.cncden.com/genx_utilities/GeneralsUtils.zip

После запуска WorldBuilder-а откройте карту AI Scripts Map, и только после этого импортируйте скрипты. Добавлять игроков не нужно – они уже добавлены. Некоторые несуразицы, конечно, останутся, но большинство ошибок пропадет.

2. Состав и принцип действия ИИ.

ИИ в генералах состоит из трех компонент – скриптов, команд и списков строительства. Скрипты отвечают за логику ИИ и находятся в файле SkirmishScripts.scb. Команды отвечают за то, когда и какие боевые единицы строит ИИ. Если хотите, чтобы ИИ производил новые, добавленные вами, единицы, нужно вносить изменения именно в команды. Списки строительства отвечают за то, где и какие здания будет строить ИИ.

Получить доступ к скриптам, можно открыв SkirmishScripts.scb. Сразу же после импортирования в 13 папках появятся скрипты для всех сторон.

Для того, чтобы отредактировать команды, нужно выйти из окна редактирования скриптов, нажав «Ok» (обязательно только «Ok» – это важно). После этого в меню выбрать Edit -> Edit Teams.

Также в WorldBuilder-е на панели инструментов есть кнопки (с краю справа), которыми можно быстро запустить редактор скриптов и редактор команд, минуя использование меню.

Получить доступ к спискам строительства можно, открыв в блокноте файл AIData.ini. Этот файл запакован в архиве INIZH.big. После распаковки архива, искомый файл будет находиться в директории Generals Zero Hour / Data / INI / Default / . Второй файл с таким же именем есть и в директории Generals Zero Hour / Data / INI / , но он не содержит списков строительства.

ИИ функционирует следующим образом. При загрузке карты, игра смотрит на то, какие игроки добавлены в загружаемую карту и активизирует соответствующие папки в скриптах. Если вы добавляете новую сторону в свой мод, надо добавить эту сторону в список игроков (Player List) во всех картах. Иначе ИИ для новой стороны работать не будет. Вот так славно постарались программисты EA Games. Также будьте готовы к тому, что, какой-нибудь «мастер Пепка» сделает новую карту, забудет сформировать Player List, а потом предъявит вам претензии по поводу того, что ваш мод с его картой не работает.

Если под контролем ИИ находятся несколько одинаковых сторон (например, два танковых генерала или три токсиновых), то игра создает копию скриптов для каждой воюющей стороны. При этом, у каждой воюющей стороны, все скрипты, команды и переменные становятся независимыми от других сторон. То есть, при написании скриптов не нужно думать о том, сколько и каких сторон будет на карте. Достаточно описать программу действий для одной стороны.

В процессе игры скрипты взаимодействуют с командами и списками строительства. Подробнее об этом – ниже.

3. Скрипты.

Скрипты бывают 2 типов – собственно, сами скрипты и подпрограммы (subroutine). Разница между ними в том, что скрипты активизируются самой игрой, а подпрограммы – командами или другими скриптами. Игра сама подпрограммы не запускает. Скрипты и команды другой скрипт тоже запустить не могут.

В редакторе скриптов все скрипты и подпрограммы упорядочены с помощью системы групп. Каждая группа может содержать в себе скрипты, подпрограммы и другие группы. В редакторе группы обозначаются желтым цветом, скрипты и подпрограммы – голубым. После голубого значка обычно стоит код вида [ns A D][E N H]. По этому коду можно узнать основные параметры скрипта. Содержимое первой скобки следующее:

1-й знак может быть “S” или “ns”. Если стоит “S” (Subroutine), то это подпрограмма. Если “ns”, то это скрипт.

2-й знак может быть “A” или “na”. Если стоит “A” (Active), то этот скрипт или подпрограмма активны и будут срабатывать во время игры. Если стоит “na”, то этот скрипт или подпрограмма выключены и работать не будут. Также стоит отметить, что скрипты и подпрограммы могут включать и выключать друг друга во время игры. Так что не стоит волноваться, если вы видите большое количество выключенных скриптов. Выключенные скрипты и подпрограммы обычно помечаются красным крестом. Также выключенной может быть вся группа. Если группа выключена (помечена крестом), то все скрипты внутри нее не работают. Не важно включены ли скрипты внутри группы. Если группа не активна, то всё внутри нее тоже не активно. Группы и скрипты можно включать и выключать, нажав на них правой кнопкой мыши. При этом появляется маленькая менюшка с одной лишь надписью – «Active». Нажав на эту надпись, вы можете включить или выключить скрипт или группу.

3-й знак может быть “D” или “nd”. Если стоит “D” (Deactivate on success), то скрипт или подпрограмма будут срабатывать только один раз, после чего будут сами себя выключать. Выключение будет производится только после успешного срабатывания, т.е. в том случае, если выполнилось условие скрипта и все действия, перечисленные в закладке

Actions if true. Если стоит “nd”, то скрипт сможет срабатывать сколько угодно раз, и сам себя выключать не будет.

Содержимое второй скобки – это буквы «E N H», говорящие при какой сложности скрипт будет включен. «E» говорит, что скрипт будет включен на легкой сложности, «N» - на средней, «H» - на тяжелой. Если какие-либо из 3 букв отсутствуют, то данный скрипт не будет включаться на соответствующей сложности.

Создать новую группу можно, нажав кнопку «New Folder». При этом появится окошко, предлагающее отредактировать свойства группы. «Group is Soubroutine» означает, что все скрипты этой группы автоматически будут восприниматься игрой как подпрограммы. «Group is Active» означает, что данная группа активна. Если группа не активна, то ни один скрипт из этой группы выполняться не будет. Если группа активна, то игра будет выполнять те скрипты, из этой группы, которые тоже активны. Ниже, под надписью «Group Name» можно задать название группы. Если требуется отредактировать свойства или название уже созданной группы, то это можно сделать двумя способами – либо выбрать ее и нажать кнопку «Edit», либо сделать на ней двойной щелчок левой кнопкой мыши.

Создать новый скрипт можно, нажав кнопку «New Script». При этом появится окошко, предлагающее отредактировать свойства скрипта. В этом окне можно задать все параметры скрипта. Кроме вышеописанных параметров, есть еще и время срабатывания, обозначенное как «Evaluate script». Скрипт может срабатывать в каждом кадре или 1 раз за несколько секунд. Срабатывание скрипта в каждом кадре будет тормозить игру. Но слишком большое время задержки тоже ставить не следует. Например, вы поставили скрипт, который проверяет, проникли ли на базу враги и при этом поставили время задержки 60 секунд. Тогда можно будет заметить очень глупую ситуацию, при которой скрипт срабатывает, делает вывод о том, что врагов на базе нет, после чего начинает ждать те 60 секунд, которые вы ему задали. За это время приходит враг, начинается битва, битва заканчивается и после нее скрипт опять просыпается, смотрит на базу и говорит: «Надо же! Врагов как не было так и нет. Посплю-ка я еще 60 секунд». Для того, чтобы такого не было, нужно себе представлять, за какое время может измениться та ситуация, которую проверяет условие скрипта.

Любой скрипт состоит из условия и действий. Условие задается в закладке «Conditions». Оно может состоять из 1 простого условия или комбинаций нескольких простых условий. Условия комбинируются с помощью логических операций И (обозначается как AND) и ИЛИ (обозначается как OR). При комбинации с помощью операции И, условие выполняется только в том случае, если выполняются все его составляющие одновременно. При комбинации с помощью операции ИЛИ, условие выполняется в том случае, если выполняется хотя бы одно из составляющих его простых условий.

Действия записываются в закладках «Actions if true» и «Actions if false». В «Actions if true» указывается список действий, которые надо выполнить при исполнении указанного в закладке «Conditions» условия. В «Actions if false» указывается список действий, которые надо выполнить при НЕ исполнении условия. В игре несколько десятков различных действий и условий. Все условия и действия упорядочены по разделам.

4. Таймеры, флаги и переменные.

В скриптах широко применяются таймеры, флаги и переменные. Действия и условия для работы с таймерами, флагами и переменными находятся в разделе «Scripting».

Переменная – это ячейка, в которой можно хранить одно число. В процессе работы к хранимому в переменной числу можно что-либо прибавлять и отнимать, т.е. реализовывать простейшие арифметические операции. Перед использованием переменной, ее нужно обязательно инициализировать. Для этого нужно просто куда-либо записать действие «присвоить переменной значение» и всё.

Флаг – это то же самое, что и переменная, только с тем исключением, что флаг может хранить не число, а 1 бит. Т.е. флаг может иметь одно из 2 возможных значений –

«установлен» (обозначается как TRUE) или «не установлен» (обозначается как FALSE). Флаги инициализировать не обязательно, но желательно.

Помните, что в ваших скриптах может быть условие, которое обратится к флагу или переменной до того, как в них будет записано нужное вам значение. Тогда ждите глюков. Для того, чтобы глюков избежать, надо добавить действия «присвоить переменной значение» или «установить флаг» в блок инициализации, который запускается сразу в начале игры. В существующих скриптах блок инициализации находится в группе Paper Work в скрипте «Initialize variables» в закладке «Actions if true».

Таймеры отсчитывают время по направлению НАЗАД. Соответствующие действия могут установливать таймеры на определенное время, добавлять, отнимать, остановливать время и т.д. Также состояние таймеров можно проверять соответствующими условиями.

Основные условия для работы с таймерами, флагами и переменными следующие:

1) «Counter Compared to a Value» – сравнение переменной с числом.
2) «Flag Compared to a Value» – проверка флага.
3) «Timer expired» – проверка истечения времени в таймере.

Основные действия для работы с таймерами, флагами и переменными следующие:

1) Подраздел «Counters», действие «Increment Counter» – прибавить заданное число к переменной.
2) Подраздел «Counters», действие «Decrement Counter» – отнять заданное число от переменной.
3) Подраздел «Counters», действие «Set Counter to a Value» – присвоить заданное значение переменной. Это действие используется для инициализации. Если какой-либо переменной нигде и ни разу не присваивается значение, то это считается ошибкой.
4) Подраздел «Flags», действие «Set Flag to a Value» – присвоить заданное значение флагу. Это действие используется для инициализации. Если какому-либо флагу нигде и ни разу не присваивается значение, то это считается ошибкой.
5) Подраздел «Timer», действие «Frame countdown timer – set» – установка таймера на заданное число кадров.
6) Подраздел «Timer», действие «Frame countdown timer – set random» – установка таймера на случайное (в заданных пределах) число кадров.
7) Подраздел «Timer», действие «Seconds countdown timer – set» – установка таймера на заданное время в секундах.
8) Подраздел «Timer», действие «Seconds countdown timer – add seconds» – установка таймера на случайное (заданных пределах) время в секундах.
9) Подраздел «Timer», действие «Seconds countdown timer – substract seconds» – отнять заданное количество секунд от таймера. Это действие приближает момент истечения таймера.
10) Подраздел «Timer», действие «Seconds countdown timer – set random» – прибавить заданное количество секунд к таймеру. Это действие отодвигает момент истечения таймера.

Все флаги, таймеры и переменные сугубо индивидуальны. Разные стороны в игре друг на друга с помощью них повлиять не могут. Например, если вы видите переменную ESCALATION в группе токсинового генерала и в группе ядерного генерала, то это не значит, что токсиновый, изменив эту переменную, как-то сможет повлиять на ядерного. Предположим, что ядерный записал в эту переменную число 10, а, через некоторое время, токсиновый записал туда число 5. Как вы думаете, что будет у ядерного в этой переменной? Число 5? Нет. У него будет та 10-ка, которую он сам туда записывал. При этом, если токсиновый обратится к этой переменной, он получит обратно свою пятерку, которую он сам туда положил. То же самое будет и с таймерами и с флагами.

5. Создание алгоритма выбора.

Часто бывают случаи, когда нужно проверить одну переменную и в зависимости от ее значения выбрать одно из действий. Если нужно выбрать одно из двух действий, то всё

просто. Вы создаете скрипт, в условии которого проверяете значение, в закладке «Actions if true» задаете одни действия, а в закладке «Actions if false» задаёте другие действия. Но как быть, если нужно выбрать одно из 3-х или более действий? Это решается просто.

Задача. Пусть требуется проверить значение переменной CHK (которая может принимать значения от 1 до 4) и выбрать одно из 4 действий в зависимости от ее значения. Для этого, создаем скрипт с именем «Check 1». Задаем в нем проверку «CHK=1?», в закладке «Actions if true» задаем действия, которые необходимо выполнить, если CHK=1, в закладке «Actions if false» задаем только одно действие «Run Subroutine» (раздел Scripting->Script). Это действие обозначает запуск подпрограммы. С его помощью запускаем скрипт «Check 2». Далее, создаем скрипт «Check 2», переключаем его в режим подпрограммы (галочка напротив надписи Subroutine), ставим условие «CHK=2?», в закладке «Actions if true» задаем действия, которые необходимо выполнить, если CHK=2, в закладке «Actions if false» опять задаем «Run Subroutine». На этот раз с помощью «Run Subroutine» надо запустить скрипт «Check 3». Далее, создаем наш последний скрипт «Check 3» с проверкой «CHK=3?», в закладке «Actions if true» задаем действия, которые необходимо выполнить, если CHK=3, в закладке «Actions if false» задаем действия, которые необходимо выполнить, если CHK=4. На этом всё. Задача решена.

С помощью этого приема можно выполнять любые проверки (не только переменных) и выбирать одно из действий. Число действий не ограничено. С помощью «Run Subroutine» можно построить цепочку скриптов любой длины.

6. Создание генератора случайных чисел.

Часто бывают случаи, когда надо произвести какое-либо случайное действие. Но генератора случайних чисел нет. Приходится лепить его из того, что есть. В скриптах есть такое действие «Frame countdown timer – set random» из раздела «Scripting->Timers». Оно запускает таймер на случайное число кадров. Вот на нем-то всё и строится. Нужно взять таймер, запустить его на случайное число кадров, а потом посчитать, сколько кадров прошло с момента его запуска. Получится случайное число.

Один из вариантов реализации такой. В одном скрипте запустить таймер на случайное число кадров. Одновременно с этим, присвоить какой-либо переменной значение 0. Далее, понадобится второй скрипт, который будет запускаться в каждом кадре и проверять, а не истекло ли время в таймере. Если не истекло, то к нашей переменной надо прибавить 1. Если истекло, то генерация закончена. Случайное значение находится в нашей переменной.

Второй вариант реализации – запустить несколько таймеров на случайное время, и посмотреть, в каком из них время истечет быстрее. Имя первого завершившего работу таймера будет случайным значением.

Далее можно запустить алгоритм выбора (описанный выше) и выбрать одно из действий в зависимости от сгенерированного случайного значения.

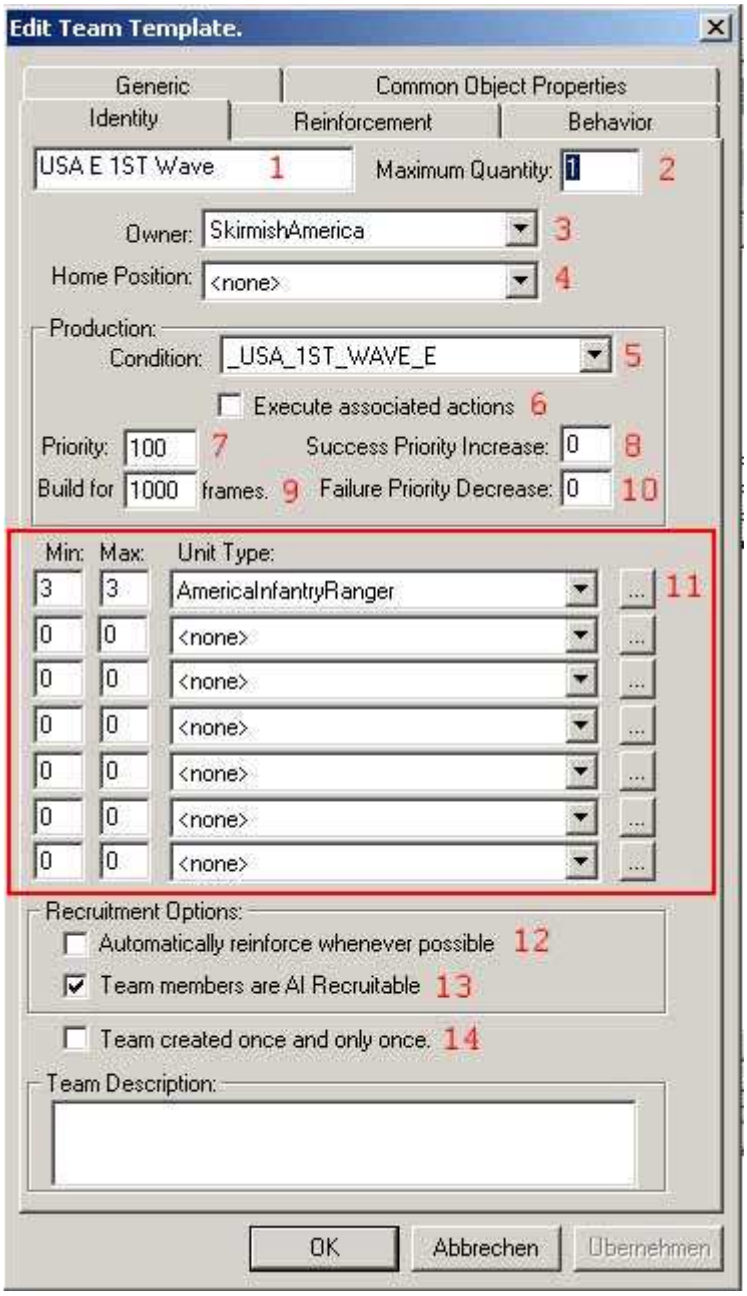
7. Команды.

Когда вы запустите редактор команд, появится окно, в котором слева будут отображены стороны, а справа – команды, которые выбранная сторона использует. Для того, чтобы внести изменения в команду, достаточно щелкнуть на ней левой кнопкой мыши два раза. Появится окно со свойствами команды.

Закладка Identify. Здесь задаются следующие параметры:

1) Название команды.
2) Максимальное количество копий этой команды, которые могут существовать одновременно.
3) Сторона – владелец команды.
4) Стартовая позиция. Этот пункт используется в миссиях. В ИИ он не нужен.

5) Условие, при соблюдении которого начинается постройка команды (Condition). Это условие представляет собой подпрограмму, у которой есть только условие, но нет действий. Действия, как правило, задавать бессмысленно, т.к. они все равно никогда не будут выполняться.



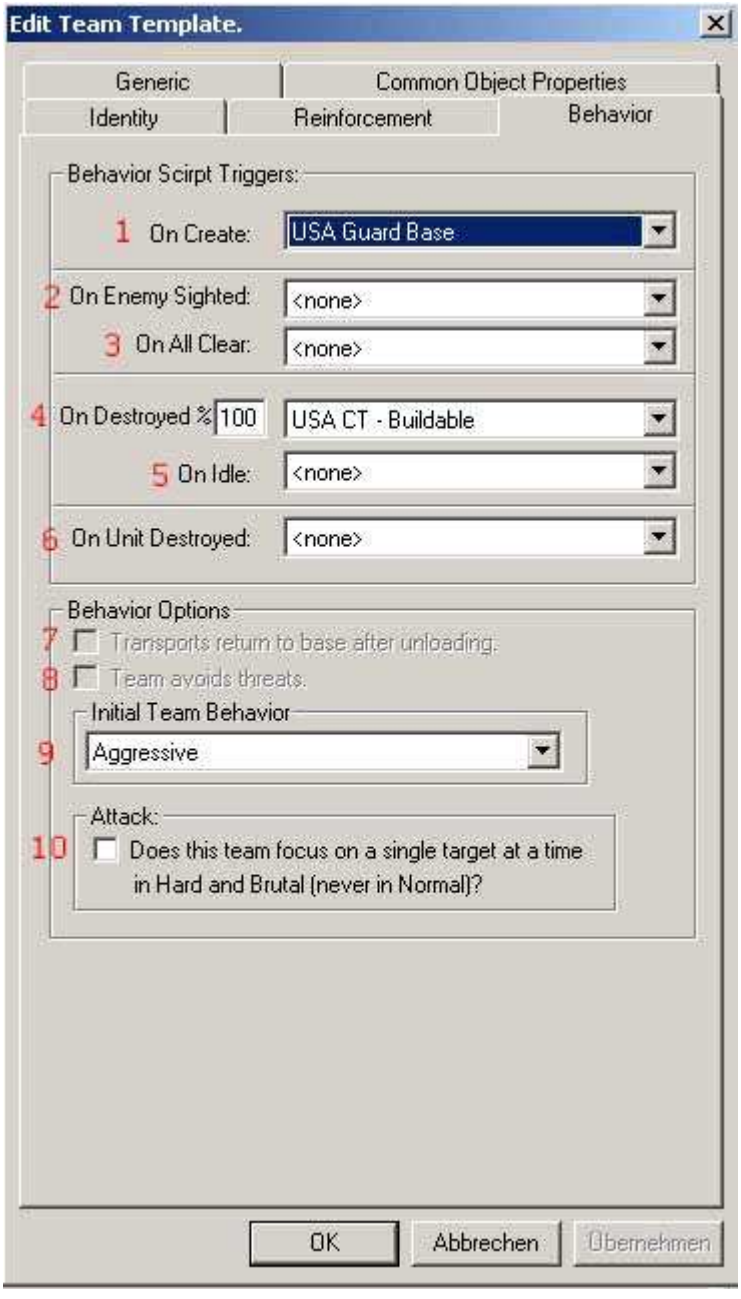
6) На тот случай, если все же надо действия выполнить, существует опция «Execute associated actions». Если она включена, то игра будет выполнять действия подпрограммы, указанной в графе «Condition».
7) Приоритет постройки (Priority). Команды с наибольшим приоритетом будут строиться первыми. Также можно задавать условия изменения приоритета.
8) Пункт «Success priority increase» указывает, насколько будет увеличиваться приоритет, если команда построилась успешно.
9) Строительство команды продолжается указанное количество кадров, после чего компьютер начинает отдавать команде приказы. На ИИ это не действует. Этот пункт нужен только для одиночных миссий.
10) Пункт «Failure Priority Decrease» указывает насколько нужно понизить приоритет, если постройка не удалась. Этот механизм нужен для того, чтобы ИИ не пытался упорно строить те боевые единицы, которые не доступны для строительства. Например, если ИИ хочет построить команду Оверлордов и не имеет при этом центра пропаганды, то после нескольких неудачных попыток строительства приоритет этой команды снизится и ИИ

начнет строить что-либо другое. Когда все более приоритетные команды будут построены, то ИИ опять предпримет попытку построить Оверлордрв. Если к этому времени центр пропаганды уже построят, то сниженный ранее приоритет постройки этой команды восстановится до прежнего уровня.

11) Состав команды. Он задается в таблице. Указываются типы боевых единиц и их количество.
12) Если нужно восстанавливать команду в случае потери нескольких боевых единиц, то ставится галочка напротив «Automatically reinforce whenever possible».
13) Если у команды стоит галочка напротив «Team members are AI Recruitable», то боевые единицы смогут уходить из нее в другие команды.
14) Если команду нужно построить только 1 раз за всю игру, то ставится галочка напротив «Team created once and only once».

Закладка Reinforcement. В основном она используется для одиночных миссий и для ИИ не представляет никакого интереса.

Закладка Behavior. Здесь задаются подпрограммы, которые выполняются при определенных событиях.



1) On Create – создание команды.
2) On enemy Sighted – замечен противник.
3) On All Clear – противника вокруг нет.

4) On Destroyed – уничтожение всей команды или ее части. Тут же задается процент уничтожения команды, необходимый для вызова подпрограммы.
5) On Idle – простаивание без дела.
6) On Unit Destroyed – уничтожение любой одной боевой единицы из команды
7) Никогда и нигде не используется.
8) Никогда и нигде не используется.
9) Пункт «Initial Team Behavior» определяет агрессивность команды. Команды, установленные в Aggressive, не будут садиться в транспорт. Команды, установленные в Normal, Alert или Aggressive будут останавливаться не доезжая до заданной цели затевать бой с врагом на полпути.
10) Если стоит галочка напротив «Does this team focus on a single target …», то команда будет фокусировать огонь на одной цели вместо пальбы во все стороны.

Закладка Generic. Здесь задаются подпрограммы, которые нужно запускать в любом случае. Например, в этой закладке для всех атакующих команд задается подпрограмма, которая проверяет определенный флаг и, если он установлен, то посылает войска в атаку. Также, в этой закладке задаются подпрограммы для строительства дополнительных башенок у Оверлордов и Хеликсов, подпрограммы для строительства зондов у войск США и т.д. Но будьте осторожны. Все подпрограммы, перечисленные в этой закладке, выполняются в каждом кадре. Если большое количество команд запускает по несколько подпрограмм в каждом кадре, то игра будет сильно тормозить.

8. Создание последовательности действий для команды.

Если хотите, чтобы ваша команда выполнила несколько действий одно за другим, то нужно использовать действие «Execute script sequentially - start» из раздела Team-> SequentialScript. Для этого нужно создать скрипт, переключить его в режим подпрограммы и указать в нем необходимые действия (например, пусть команда сначала будет двигаться в одном направлении, потом в другом, потом что-то захватит, а потом пойдет в атаку). После этого нужно создать второй скрипт, который с помощью действия «Execute script sequentially - start» запускал бы первый. Потом в настройках команды надо указать, чтобы команда запускала именно второй скрипт – тот, который с «Execute script sequentially - start». В результате, команда сначала выполнит одно действие до конца, потом приступит ко второму, выполнит его до конца, потом начнет выполнять третье и т.д.

Если для этой цели использовать «Run Subroutine», то команда попытается выполнить всё, что вы ей задали в течение одного кадра и не сможет, т.к. каждое следующее действие будет отменять предыдущее.

Также в наличии есть действие «Execute script sequentially - stop», которое останавливает последовательность, запущенную с помощью «Execute script sequentially - start» и действие «Execute script sequentially - loop», которое запускает последовательность действий как и «Execute script sequentially - start», но, когда последовательность заканчивается, запускает ее снова и так по кругу до бесконечности или пока не остановят принудительно.

9. Пути, зоны и ключевые объекты.

В скриптах используются такие понятия, как пути и зоны. Зоны – это некие области на карте, описанные с помощью многоугольников. Скрипты создавать зоны не могут, но зато могут проверять, а не находится какой-либо объект в какой-либо зоне. Например, с помощью этого механизма ИИ узнает, что на его базу пришли враги. Зоны задаются создателем карты. Игра эти зоны потом переименовывает. Например, автор делает карту для 2 игрокрв и задает зоны InnerPerimeter1 и InnerPerimeter2. В игре эти зоны будут перименованы следующим образом. Для игрока №1 зона InnerPerimeter1 переименовывается в MyInnerPerimeter, а зона

InnerPerimeter2 – в EnemyInnerPerimiter. Для игрока №2 – наоборот. Зона InnerPerimeter2 уже будет MyInnerPerimeter, а InnerPerimeter1 – EnemyInnerPerimiter. Аналогично и для OuterPerimeter.

Пути – это точки на карте, которые используются для навигации. Например, ИИ может прислать свои войска к базе противника тремя путями. Пути должен задавать автор карты. Например, для карты с 2 игроками, должны присутствовать пути с метками: Center1, Cennter2, BackDoor1, BackDoor2, Flank1, Flank2. Пути с пометкой 1 – это пути к базе игрока №1. Пути с пометкой 2 – это пути к базе игрока №2. Если пути не заданы или заданы неверно, то ИИ не будет ходить в атаку. В игре пути будут тоже переименованы. Для игрока №1 пути Cennter2, BackDoor2 и Flank2 переименуются в Cennter, BackDoor и Flank, а Center1, BackDoor1, Flank1 – исчезнут. Для игрока №2 – наоборот. Исчезнут пути Center2, BackDoor2, и Flank2, а Cennter1, BackDoor1 и Flank1 переименуются в Cennter, BackDoor и Flank.

Ключевые объекты – это обычные нейтральные здания, имеющие определённые метки. По меткам здание может быть распознано как ключевое. Обычно, ИИ в них сажает солдат. Ключевые объекты размещаются автором карты.

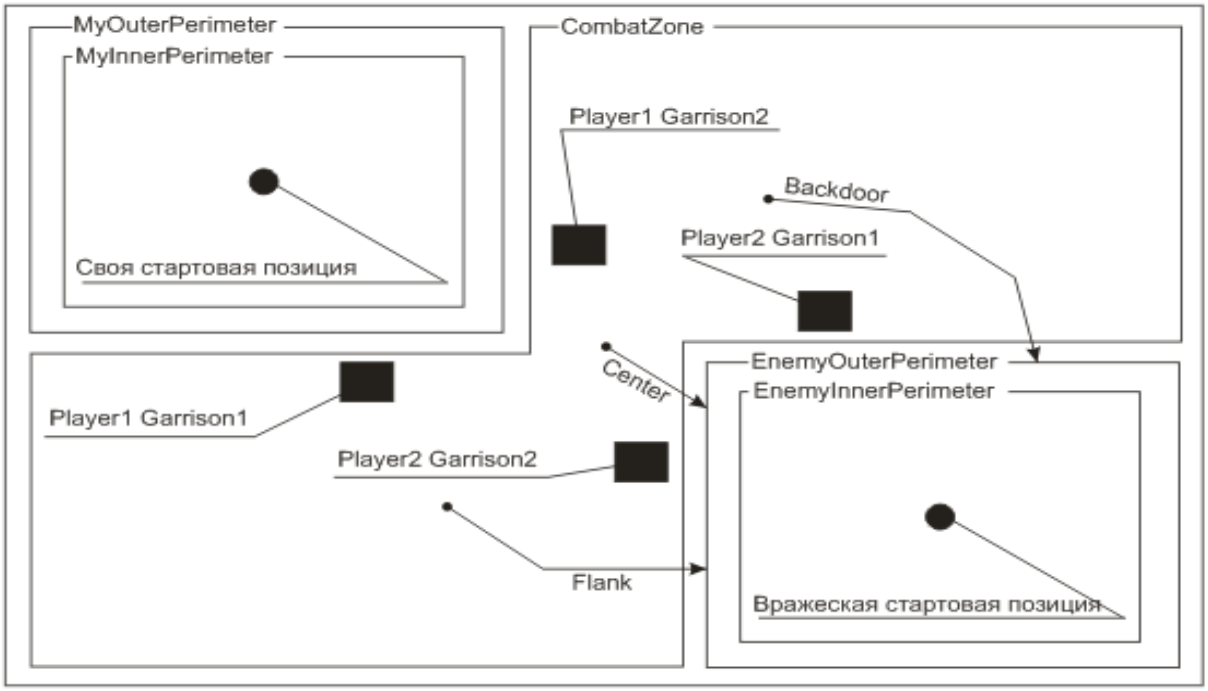
Стандартные зоны, используемые в ИИ следующие:

1) MyInnerPerimiter – центр своей базы.
2) MyOuterPerimiter – своя база целиком.
3) EnemyInnerPerimiter – центр вражеской базы.
4) EnemyOuterPerimiter – вражеская база целиком.
5) CombatZone – всё, что не входит в периметр баз. Как правило, это центр карты.

Стандартные пути следующие:

1) Center – путь к вражеской базе для атаки «в лоб».
2) Flank – путь к вражеской базе с фланга
3) Backdoor – путь к вражеской базе с другого фланга
4) Special – путь неизвестно куда. Нигде не используется.

Часто встречающееся расположение зон, путей и ключевых объектов на карте показано на следующем рисунке:



Обратите внимание, что пути Center, Flank и Backdoor начинаются не от своей базы, а где-то в середине карты. Обратите внимание, что зоны InnerPerimeter находятся внутри зон OuterPerimeter. Т.е. если объект находится в зоне InnerPerimeter, то он одновременно будет и в OuterPerimeter.

Условия для работы с зонами находятся в разделе «Player», подразделе «Area», в разделе «Team», подразделе «Area» и в разделе «Unit», подразделе «Area». С помощью этих

условий можно проверить вхождение команды или боевой единицы в определенную зону. Действия для работы с путями находятся в разделе «SkirmishOnly», в подразделе «Move».

10. Файл AIData.ini.

В INI-коде есть файл, который выполняет часть функций ИИ. Этот файл называется AIdata.ini и находится в директории Generals Zero Hour \ Data \ INI \ Default \. Также следует отметить, что существует еще один файл с таким же именем, который находится в директории Generals Zero Hour \ Data \ INI \ , но он не содержит в себе никакой информации.

В начале файла находятся некоторые параметры, назначение которых не совсем ясно, т.к. их изменение не ведет к каким-либо результатам. Например, есть параметр «MinInfantryForGroup». Он равен 3. В комментарии говорится, что это минимальное количество солдат, необходимое для того, чтобы команда начала движение. Но в игре мы видим, что ИИ захватывает нефтяные вышки командами по 2 солдата. Т.е, «MinInfantryForGroup» вообще на ИИ никак не влияет. Есть параметр «TeamsWealthyRate», который говорит о том, что ИИ должен строить команды быстрее, если у него много денег. Но строительство команд целиком и полностью регулируется скриптами. Т.е. этот параметр тоже ни на что не влияет. И т.д. Под большим вопросом стоит и назначение других параметров.

Далее идут 12 блоков «SideInfo» (для каждой стороны – по одному). Эти блоки описывают некоторые параметры сторон. Задаваемые параметры следующие:

ResourceGatherersEasy, ResourceGatherersNormal и ResourceGatherersHard – количество сборщиков ресурсов, которое надо строить для каждого ресурсного центра на легкой, средней и тяжелой сложностях.

BaseDefenseStructure1 – основное защитное сооружение. Здание, указанное в этом параметре будет строиться при запуске действий «Build one additional base defenses on the flank» и «Build one additional base defenses on the front» из раздела «Skirmish Only».

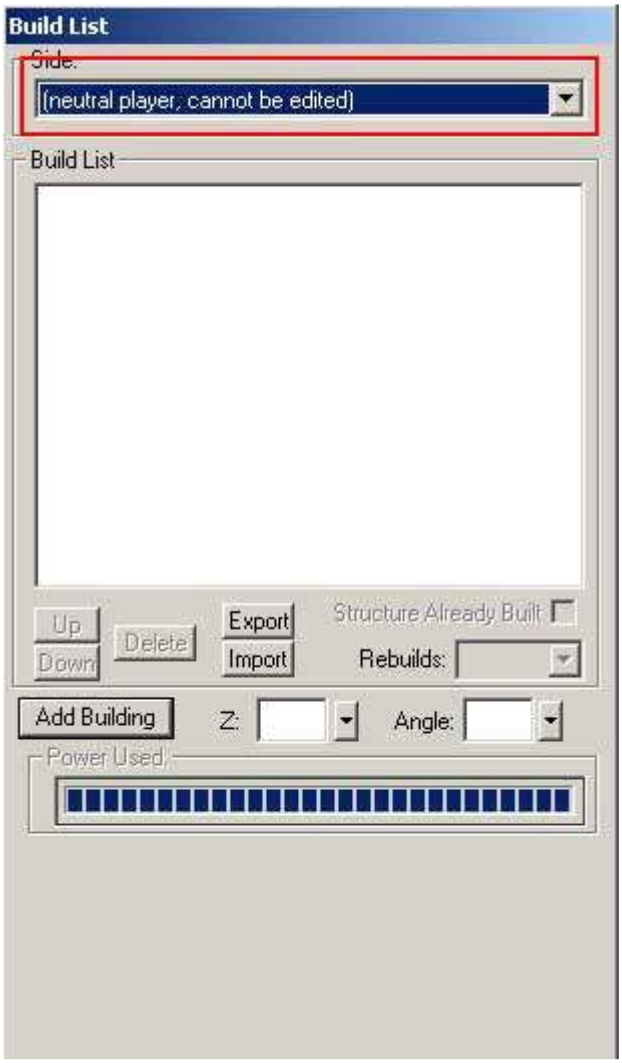
Блоки SkillSet1 и SkillSet2 указывают, какие генеральские возможности следует приобрести при получении повышений. В начале игры блок выбирается случайным образом - либо SkillSet1, либо SkillSet2. В дальнейшем, ИИ приобретает новые возможности в том порядке, в котором они указаны в выбранном блоке.

Далее содержатся 12 блоков «SkirmishBuildList» (для каждой стороны – по одному). Эти блоки являются списками строительства и отвечают за расположение зданий на базе. Когда какой-либо скрипт из группы «Base Building» пытается построить здание, игра обращается к соответствующему списку и строит здание в заданном месте. Если в заданном месте обнаруживается препятствие, то ИИ автоматически двигает здание и строит его рядом с указанными координатами. Если здания в списке строительства нет, то оно не строится. Будьте внимательны! Если вы используете действие «Build a building» из раздела «Skirmish Only», то ИИ будет строить только те здания и только в том количестве, в котором они указаны в списках строительства. Если вы хотите, чтобы ИИ построил какое-либо еще здание, то нужно добавить его в соответствующий список строительства.

Чтобы обойти списки, существуют такие действия, как «Build structure on flank perimeter» и «Build structure on front perimeter». Эти два действия будут строить здания либо на фланге базы, либо спереди базы. Но злоупотребление этими действиями приведет к тому, что ИИ построит множество зданий «в куче», т.е. вокруг одной точки очень близко друг к другу.

Вернемся к спискам строительства. Редактировать их можно только вручную. Но это неудобно. К сожалению, WorldBuilder не может их открывать. Но зато с его помощью такие списки можно создавать. Для этого на панели инструментов WorldBuilder-а есть кнопка «Build List Tool». Вот эта кнопка:

Нажав ее, вы увидите два новых окна. Одно - «Pick a unit» в центре экрана. Другое – «Build List» справа экрана. Сначала в окне «Build List» в пункте «side» выберете сторону. Сторону выбирать обязательно!



Потом в окне «Pick a unit» можно выбрать любые здания и расставить их на карте в любом месте. Когда вы закончите со «строительством», в окне «Build List» нажмите кнопку «export». При нажатии этой кнопки, в директории, где установлена игра, создается файл с именем вида «SkirmishAmerica_BuildList.ini». Также может быть «SkirmishChina_BuildList.ini» или «SkirmishGLADemolitionGeneral_BuildList.ini» в зависимости от того, какую сторону вы выбирали. Нажатие кнопки «export» только создает файл и никаких сообщений на экран не выводит. После создания файла, остается скопировать его содержимое и вставить в AIdata.ini ВМЕСТО какого-либо старого блока «SkirmishBuildList». После создания списка строительства, рекомендуется сохранить карту, чтобы в дальнейшем можно было ее открыть, отредактировать список и снова его экспортировать.

11. Структура скриптов.

В данной главе рассказывается о том, как сделаны скрипты в существующем ИИ от EA Games. Это только описание сделанного и ни в коем случае не является руководством к действию. Вы можете придумать свою структуру скриптов и свою логику их работы.

Перед тем, как начать рассказ, разберемся с обозначениями. В скриптах и командах очень много повторяющихся вещей, выполняющих одни и те же функции. Например, есть скрипты «_USA_1ST_WAVE», «_CHINA_1ST_WAVE» и «_GLA_1ST_WAVE». Они выполняют одно и то же действие, но для разных сторон. Поэтому в дальнейшем я буду их называть просто «1ST_WAVE», пропуская слова «USA», «CHINA» и «GLA». И так далее.

Для каждой стороны набор скриптов и групп стандартный. Отличия в основном только в названиях скриптов и в их содержании. Роль, которую они выполняют, остается неизменной. Например, у GLA есть скрипт «GLA Build Arms Dealer», а у USA есть скрипт «USA Build War Factory». Название и содержание этих скриптов разные, а роль одна и та же – построить завод. Далее, рассмотрим стандартные группы.

Группа «Paper Work» отвечает за инициализацию. В ней всем переменным и таймерам придаются начальные значения. Боевые единицы классифицируются и делятся по группам. Например, есть такие группы как «танки», «самолеты», «артиллерия» и другие. Т.е. чтобы проверить, с каким врагом ИИ имеет дело, он смотрит, а не входит ли враг в какую-нибудь группу. Если вражеская единица ни в одну группу не входит, то она воспринимается как «не опасная» со всеми вытекающими отсюда последствиями. Например, не срабатывают некоторые скрипты, которые предназначены для организации обороны. Так что при добавлении новой единицы в игру, она обязана быть добавлена в скрипт «Define Object Lists» из группы «Paper Work». Программисты EA Games тут уже ляпнули - в «Define Object Lists» забыли добавить техника ГЛА (он строится в 3-х вариантах) и Sentry Drone-а.

Группа «Sell Off» содержит скрипты, отдающие команду продажи всем зданиям, когда ИИ проигрывает. Эта команда отдается, когда у ИИ нет бульдозеров, фабрик и командного центра. Это не очень логично, т.к. если нет бульдозеров, фабрик и командного центра, то ещё можно пользоваться супер оружием. Проверки на наличие супер оружия там нет.

Группа «Base Expansion» используется для постройки и укрепления ресурсных центров вблизи источников ресурсов. Попытка построить новый ресурсный центр предпринимается 1 раз в 10 минут при нормальной сложности, либо 1 раз в 2 с половиной минуты при тяжелой сложности независимо ни от чего. При такой попытке всегда строится одна новая группа «Replace Dozer» не зависимо от того, можно ли вообще построить ресурсный центр (тупость).

Группа «World State Detection» содержит откровенные cheat-ы для ИИ – деньги с неба.

Группы серии «Build Conditions» отвечает за строительство атакующих команд. В большинстве скриптов этой группы отсутствуют какие-либо действия – прописаны только условия. Такие скрипты используются как условия постройки команд. Им не нужны никакие действия.

Группа «Attack priorities» содержит приоритеты атаки. Т.е. скрипты дают командам возможность атаковать более приоритетные цели в первую очередь и менее приоритетные – во вторую очередь. Например, можно сделать, чтобы танк первым делом стрелял в другой танк, а только потом в стоящего рядом пехотинца. Еще можно сделать, чтобы прорвавшиеся на базу врага силы первым делом сносили командный центр. Но повторяю – это всё из разряда «можно сделать». На самом же деле, эти приоритеты НЕ применяются для атакующих команд, а применяются только для защитных команд, и то не для всех. Видимо, разработчики про них совсем забыли.

Группа «Attack Waves» отвечает за переключение «волн». ИИ в Генералах атакует «волнами» - сначала строит команды первой волны, посылает их в атаку, потом – команды второй волны, и т.д. Переключение на каждую следующую волну атаки осуществляется скриптами этой группы. Величина, говорящая какая волна атаки активна в данный момент, хранится в переменной ATTACK_WAVE. Посылание уже построенных команд в атаку производится одним из скриптов серии «Attack Wave X Execute», где вместо X – цифра от 1 до 5. Эти скрипты находятся в группе «Attack Waves» и занимаются только тем, что устанавливают флаг «_LAUNCH_ATTACK» в значение «TRUE». После того, как этот флаг был установлен в «TRUE» , он сразу же сбрасывается в «FALSE» в следующем кадре. За 1 кадр все команды успевают среагировать и начать двигаться в атаку.

Группа «Escalation» содержит скрипты, определяющие развитие ИИ. Скрипты этой группы увеличивают переменную «ESCALATION». На этой переменной завязано множество всевозможных манипуляций. Например, значение этой переменной проверяется перед строительством определенных зданий и переключении «волн» для атаки. Можно сказать, что

эта переменная определяет уровень развития ИИ. Чем она больше, тем более дорогие здания и более мощные боевые единицы строит ИИ.

Группы серии «Complimentary» отвечают за строительство команд усиления. Скрипты этих групп «подсматривают» за своим противником и строят команды серии «CT». Например, если у противника много авиации, строится команда «CT – Air», содержащая ПВО. Если у противника много пехоты, то строится команда «CT – Infantry», содержащая боевые единицы, эффективные против пехоты и т.д. Далее все эти команды серии CT отправляются в атаку.

Группа «Generic Attack» содержит скрипты, используемые атакующими командами при наступлении.

Группа «Generic Behavior» - это сборник разных подпрограмм, используемых различными командами. Например, в ней есть подпрограммы посадки в транспорт и высадки, подпрограммы переключения оружия и т.д.

Группа «Alert Team Attacks» - еще один сборник подпрограмм, как и «Generic Behavior».

Группа «Combat Zone» содержит скрипты, отвечающие за действия в зоне «CombatZone». Эти скрипты используются командами серии «Guard Combat Zone».

Группы серии «Base Building» содержат скрипты, строящие базу. Строительство базы завязано со списками строительства, находящимися в файле AIdata.ini. При срабатывании скриптов этой группы ИИ сначала смотрит в список строительства. Если он в нем находит соответствующее здание, то здание помещается в очередь постройки. Потом бульдозерам или рабочим раздаются задания из очереди постройки в той последовательности, в которой они туда записывались. Если у ИИ не хватает денег на постройку 1-го здания в очереди, то он просто ждет, пока они наберутся и только потом начинает его строить. Второе здание в очереди НЕ может быть построено до того, пока не начата постройка первого.

Группа «Tech Buildings» отвечает за захват нейтральных зданий в начале игры. Скрипты этой группы строят апгрейд на захват здания, ждут, пока он завершится, после чего строят команды серии «Tech Building Capture» и посылают их на захват зданий. Здесь есть ошибка. В скрипте, отвечающим за захват здания, прописано действие «Team '<This Team>' will move towards the nearest 'Tech Buildings' within area 'CombatZone'», которое означает, что ИИ будет захватывать только те здания, которые находятся за пределами его базы. Нейтральные здания, стоящие непосредственно на его базе, захватываться не будут.

Группа «Upgrades» содержит скрипты, отвечающие за строительство большинства (но не всех) апгрейдов. Если нужно добавить новый апгрейд, то достаточно скопировать один из существующих скриптов этой группы и изменить в его копии название апгрейда.

Группа «Generals Powers» отвечает за использование сил генералов, таких как высадка десанта или ковровая бомбардировка. Следует отметить, что на то, куда будет нанесен удар, влияет параметр «enum» в INI-коде, встречающийся в описаниях супероружий и сил генералов. Например, если генеральская сила имеет enum «SPECIAL_CLUSTER_MINES», то удар будет нанесен по своей базе.

Группа «Super Weapon Use» является полным аналогом группы «Generals Powers», за исключением того, что она управляет только одним супер оружием (ядерной ракетой, лучевой пушкой или скад-штормом). Если нужно добавить новое скпер оружие или новую силу генералов, достаточно скопировать 2 скрипта из этой группы или из группы «Generals Powers» и адаптировать их под новое оружие. Логика их действия такова. За запуск каждого супер оружия отвечают 2 скрипта. У одного в конце названия стоит «AI», у другого – «Fire». Скрипт с пометкой «AI» проверяет заряжено ли оружие. Если заряжено, то он активирует скрипт с пометкой «Fire». Скрипт с пометкой «Fire» запускает супер оружие и сразу же деактивирует себя. Эта схема немного сложна для такого простого действия, как запуск супероружия. Но никак по-другому не работает. Уже проверяли.

Группа «Use Battle Plans» существует только у сторон, относящихся к США, и отвечает за использование боевых планов стратегического центра. В ней только 2 скрипта, которые запускают только 1 план «Hold The Line» (т.е. «держать позиции»). Выбирать

другие планы ИИ не умеет. Но это можно реализовать с помощью генератора случайных чисел.

Группы «Garrison Builds» и «Garrison Orders» отвечают за размещение гарнизонов в нейтральных зданиях. Для их размещения на карте должны быть объекты (нейтральные здания) с метками в стиле «PlayerX GarrisonY», где вместо X стоят цифры от 1 до 8, а вместо Y – от 1 до 3. Если таких объектов нет, то ничего не происходит, если есть, то строятся команды серии «Garrison» и направляются в эти здания. Но и здесь есть ошибка. Если посмотрите в группу «GarrisonBuilds», то увидите там условия «Unit 'Player1 Garrison1' exists and is alive.» Такое условие прописано в 8 скриптах подряд. Везде – «Player1 Garrison1». А должно быть «Player1 Garrison1», «Player2 Garrison1», «Player3 Garrison1» и так далее до «Player8 Garrison1», Потом идет такая же ерунда с «Player1 Garrison2» и «Player1 Garrison3» - там опять номера игроков не меняются. А должны.

12. Структура команд.

В данной главе рассказывается о том, как сделаны команды в существующем ИИ от EA Games. Это всего лишь описание сделанного и ни в коем случае не является руководством к действию. Вы можете придумать свою структуру команд и свою логику их работы.

Перед тем, как начать рассказ, разберемся с обозначениями. В скриптах и командах очень много повторяющихся вещей, выполняющих одни и те же функции. Например, есть скрипты «_USA_1ST_WAVE», «_CHINA_1ST_WAVE» и «_GLA_1ST_WAVE». Они выполняют одно и то же действие, но для разных сторон. Поэтому в дальнейшем я буду их называть просто «_1ST_WAVE», пропуская слова «USA», «CHINA» и «GLA». Так же я буду поступать с командами типа «USA E 5TH WAVE Tanks F» или «USA H 5TH WAVE Artillery B». Подобные команды выполняют одну и ту же роль, поэтому команды такого типа я буду обозначать просто как «5TH WAVE». И так далее. Например, если я пишу о команде «Garrison». То я имею в виду все команды, в названии которых встречается слово «Garrison».

Итак, все команды, используемые в игре, условно можно разделить на атакующие, защищающие и служебные. Такой классификации в игре нет, но её целесообразно ввести для лучшего понимания.

Атакующие команды, как вы уже догадались, предназначены для атаки. Они строятся так называемыми «волнами». Т.е. по прошествии определенного времени срабатывает условие в скрипте «1ST_WAVE», которое запускает процесс строительства команд. ИИ строит все команды серии «1ST WAVE». Потом, через определенное время, срабатывает другой скрипт, который посылает всё, что построено в атаку. Далее срабатывает скрипт «2ST_WAVE». ИИ строит все команды серии «2ST WAVE», потом посылает их в атаку и так далее до «5TH WAVE». Развившись до определенного уровня, ИИ строит только команды серии «5TH WAVE», посылает их в атаку и опять строит. Т.е. на 5-й «волне» он зацикливается. Однако, следует отметить, что бывают случаи, когда команды какой-либо волны кроме пятой могут построиться и пойти в атаку 2-3 раза подряд. Такие случаи тоже бывают. Если скрипт переключения на другую волну не сработает, то команды предыдущей волны могут построиться еще раз.

Атакующие команды в своем имени имеют обозначения E,N и H. Например, «H 2ND WAVE» или «E 2ND WAVE». Эти буквы обозначают сложность, на которой будут строиться команды. E – на легкой, N – на нормальной, H – на тяжелой. Сами буквы не являются указателем на сложность. Если изменить E на H, то ничего не изменится. Если хотите поменять сложноть, на которой строится команда, надо менять скрипт постройки, указанный в команде в графе «Build Conditions».

Скрипт, отвечающий за постройку команд, зависит от переменной ESCALATION. А ESCALATION, в свою очередь, зависит от времени и от того, каких зданий ИИ понастроил. Проще говоря, скрипт, отвечающий за постройку новой «волны» сработает либо после постройки определенного здания, либо от времени. При этом компу абсолютно до фени,

какие здания в данный момент имеются на базе, и может ли он вообще и построить какую-либо команду. Скрипт, отвечающий за посылание команд в бой, срабатывает 1 раз в 60 секунд и ему по барабану, что было построено и построены ли какие-либо команды вообще. Например, если из всех атакующих команд построен только 1 танк, то он тупо отправляется в атаку.

Когда команда идет в атаку, она использует фиксированный путь. Т.е. одни команды атакуют только «в лоб», другие – только фланга, третьи – только с тыла. Существующий ИИ выбирать направление для атаки не умеет. Посмотреть, с какой стороны будет атаковать команда, можно в ее свойствах в закладке Generic. Если там будет присутствовать скрипт «Attack enemy sequence front» - то команда пойдет «в лоб». Если «Attack enemy sequence flank» - то с фланга. Если «Attack enemy sequence back» - то с тыла. На самом деле, понятие «с тыла» означает «с другого фланга». Но, видимо, для кривых американских программистов, писавших этот ИИ, другой бок – это зад. Поэтому и написано «back».

Защищающие команды – это все команды серий «Defender» и «Alert Invasion». Они начинают строиться только тогда, когда на базу пришел враг. ИИ умеет классифицировать врага и принимать соответствующие меры. Например, если на ИИ напали одни танки, то он начнет строить только команды, состоящие из солдат с РПГ.

Команды серии «CT» являются командами усиления атаки. Алгоритм действия следующий. Сначала ИИ строит одну или несколько команд серии «CT» в зависимости от необходимости. Команды серии «CT» каждого типа могут быть построены только в одном экземпляре. Далее, все построенные команды серии CT сливаются в одну команду – Complimentary team. Эта команда в списке присутствует, но она пустая. Таки и должно быть, т.к. она создана для того, чтобы объединять в себе другие команды. Далее, Complimentary team присоединяется в любой атакующей команде. У атакующих команд в закладке Generic есть скрипт «CT – Join My Team». Этот скрипт как раз и отвечает за присоединение к себе команды Complimentary team.

Команды серии «Garrison» ИИ размещает в нейтральных зданиях. Для их размещения на карте должны быть объекты (нейтральные здания) с именами в стиле «PlayerX GarrisonY», где вместо X стоят цифры от 1 до 8, а вместо Y – от 1 до 3. Если таких объектов нет, команды серии «Garrison» не строятся. Строительство и действия этих команд управляются скриптами из групп «Garrison Builds» и «Garrison Orders». В командах серии «Garrison», как правило, 1-2 солдата. Это очень мало. Игроки сажают в здания от 4 до 8 солдат.

Команды серии «Guard Combat Zone» занимаются охраной зоны, обозначенной как «Combat Zone». Как правило, эта зона охватывает всё, что не входит в периметры баз. Было бы ошибкой полагать, что «Combat Zone» - это поле боя. Никакое это не поле боя. Бой может быть где угодно, а «Combat Zone» всегда там, где ее нарисовал автор карты. Центр этой зоны почти всегда находится в центре карты. Если команде задать действие охранять какую-либо зону, то она пойдет в центр зоны и переключится в режим охраны. В режиме охраны команда будет атаковать только тех врагов, которых увидит, а не тех, которые зашли в зону. Так что фактически, команды серии «Guard Combat Zone» и занимаются охраной центра карты. Зачем – не ясно. На некоторых картах, где в центре находятся нефтяные вышки, это может быть целесообразным. Но ведь не все карты одинаковые.

Команды серии «Guards» занимаются обороной базы. Для каждой «волны» атаки строится своя охранная команда. Как правило, такие команды просто стоят около командного центра и сторожат. В этом и заключается «оборона».

Команды серии «Replace Dozer» содержат только 1 бульдозер или 1 рабочего и строятся либо при попытке возведения нового ресурсного центра, либо в начале игры, когда у компьютера много денег и ему надо построить много всего и сразу. Данные команды не имеют триггера. Их постройка вызывается в каком-либо скрипте с помощью действия «Start building a team» (раздел Team->AI). Здесь опять очередной ляп. Некоторые команды этой серии переключены в режим «Aggressive», из-за чего ИИ может начать постройку какого-либо здания и оставить ее в степени готовности 0% до конца игры.

Команды серии «Tech Building Capture» содержат только 2 пехотинцев (либо рейнджеров, либо повстанцев, либо красных стражей). Они используются для захвата нейтральных зданий (нефтяных вышек, артиллерийских платформ т т.п.) в начале игры. ИИ сначала ждет, пока не завершится апгрейд на захват здания, а потом строит эти команды.

Команды серии «Alert Capture Neutral» содержат только 1 солдата и используются для захвата нейтральной техники. Например, после того, как поработал Жармен Келл или пушка с нейтронными снарядами, остается много нейтральной техники. Она захватывается. Недочет состоит только в том, что ИИ не может захватывать несколько машин одновременно. Он строит только 1 пехотинца, захватывает им 1 машину и смотрит, есть ли еще нейтральная техника. Если есть, то он строит еще 1 пехотинца и т.д.

Команды серии «Base Expander» и «Tech Capture Big OFF» - назначение неизвестно. Скорее всего, эти команды никогда не строятся.

Кроме описанных выше команд также существуют и команды, включающие в себя все здания и боевые единицы, которые ИИ имеет в распоряжении. Эти команды называются teamSkirmishAmerica, teamSkirmishAirforceGeneral, teamSkirmishChina и т.д. Нигде в редакторе команд они не прописаны. Они доступны только в редакторе скриптов. Эти команды очень удобно использовать, когда надо что-либо сделать со либо всеми боевыми единицами сразу, либо с теми единицами, которые в обычные команды никак не входят. Например, построить боевых зондов можно, приказав команде teamSkirmishAmerica нажать на кнопку постройки зондов. Это делается с помощью действия «Use CommandButton Ability» из раздела Team->CommandButton. Таким же способом можно делать апгрейды для электростанций и строить башенки для Оверлордов.

13. Отладчик скриптов.

Отладчик скриптов представляет из себя один файл DebugWindow.dll, который можно найти в комплекте «Generals Editing Utilites», изданным Deezire-ом. Для инсталляции отладчика надо просто переместить DebugWindow.dll в директорию, в которой установлена игра. Для запуска отладчика нужно в INI-коде в файле GameData.ini найти строку «AIDebug = no» и заменить её на «AIDebug = yes». Если такой строки нет, то её надо добавить. Далее, нужно запустить игру с ключами win и scriptdebug. В Windows-е для этого нужно отредактировать свойства ярлыка, которым запускаются Генералы и в строку запуска вместо строки типа

"C:\Games\Generals Zero Hour\generals.exe"

написать

"C:\Games\Generals Zero Hour\generals.exe" -win -scriptdebug

Обратите внимание на кавычки. Я их поставил там, где они и должны стоять.

При запуске Генералы будут работать в окне. Но появится еще одно окно, в котором будут отображаться результаты работы скриптов. Данный отладчик позволяет только пронаблюдать за работой скриптов, но не позволяет что-либо менять в процессе работы. При запущенном отладчике игра работает раз в пять-десять медленнее.

Также можно уменьшить размер окна, в котором будет работать игра, добавив в ярлык запуска игры следующие параметры в конец строки запуска:

-xres 640 -yres 480

Еще можно уменьшить время запуска игры, убрав заставку, которая показывается на фоне меню. Для этого надо добавить параметр:

–noshellmap

Время загрузки можно уменьшить еще больше, убрав заставку «EA Games». Для этого в INI-коде в файле «Video.ini» надо найти строки «Filename = EA_LOGO» и «Filename = EA_LOGO640», и заменить их на «Filename = NONE».

14. Создание динамических путей наступления.

Как известно, ИИ не умеет выбирать пути для атаки. Его можно этому научить. В игре есть 3 стандартных пути – «в лоб», «с фланга» и «с тыла». По этим путям можно пустить разведку (Техников или Хаммеров) и посмотреть что с ними станет. Хорошая оборона разнесет разведку задолго до того, как та сможет приблизиться к базе, а плохая – пропустит супостата. По этому критерию можно определить, с какой стороны оборона сильнее, а с какой – слабее, и пустить основные силы в слабо защищенное место.

Для реализации этой идеи понадобятся 3 команды разведчиков. Каждая команда – это 1 Техник, либо 1 Хаммер, либо 1 танк. Обязательно нужно поставить разведкоманды в режим «passive», а не то они будут нападать на любого врага, попавшегося на дороге, и никогда не доедут до цели. Одна из команд должна иметь скрипт «Move enemy base front», другая - «Move enemy base flank», третья - «Move enemy base back». Новые скрипты серии «Move enemy base» нужно создать самим на базе уже имеющихся скриптов серии «Attack enemy base». Различие в том, что скрипты «Attack enemy base» двигают команду к базе противника, а потом запускают действие «begins hunting», которое означает «бей всё что движется». В новых скриптах «begins hunting» нужно заменить на «Move team towards the nearest object of a specific type», т.е. двигаться внутрь базы противника по направлению к какому-либо зданию. Разведчикам воевать не надо. Им надо двигаться вглубь базы чтобы разведать какая там у врага оборона.

Разведка обязательно должна посылаться к базе врага только тогда, когда все 3 разведкоманды построены. Только тогда можно рассчитывать на правильный результат. Постройку можно контролировать с помощью флагов или переменных. Пусть, каждая разведкоманда в графе «on create» запускает скрипт, который выставляет флаг «разведкоманда номер такой-то готова». Когда все 3 флага установлены, можно пускать команды в разведку. Проверку этих 3 флагов можно запихнуть в условия созданных вами скриптов серии «Move enemy base». В перспективе, чтобы сделать ИИ еще умнее, можно запускать разведку тогда, когда построены все атакующие команды и компьютер полностью готов к атаке.

После того, как разведка запущена, нужно ждать, пока она дойдет до базы противника, либо будет уничтожена. Для проверки уничтожения нужно в графах «on team destroyed» прописать скрипты, которые будут устанавливать флаги «разведкоманда номер такой-то уничтожена». Для проверки того, достигла ли команда цели, нужно в закладке Generic задать скрипт, который в каждом кадре будет проверять, находится ли команда в зоне «SkirmishEnemyInnerPerimeter». Если находится, то надо устанавливать флаг «разведкоманда номер такой-то достигла цели».

Также нужен еще такой скрипт, который хотя бы раз секунду будет обращаться ко всем флагам, чтобы проверить, уничтожены ли все разведкоманды, и все ли они достигли цели. Если все 3 команды либо уничтожены, либо достигли цели, то разведка закончена, и нужно приступать к принятию решения об атаке. Если какая-то команда достигла цели, то можно сделать вывод о том, что путь, по которому она шла, слабо защищен. Если команда была уничтожена и не достигла цели, то путь, по которому она шла, защищен сильно. На основании этих данных можно сделать вывод о том, где у противника сильная защита, а где слабая. Если сильная защита оказалась только с 2-х сторон, то все ясно. Оставшееся незащищенная сторона – это и есть путь атаки. Если защита оказалась сильной только с одной стороны, то надо запустить генератор случайных чисел, который выберет какую из слабых сторон атаковать. Если защита везде слабая, либо везде сильная, то тоже нужно обратиться к генератору случайных чисел. В результате всех изысканий должна получиться переменная (назовем её «ATTACK_PATH»), в которой должно быть записано число от 1 до 3, обозначающее путь для атаки.

Теперь по выбранному пути нужно запустить все атакующие команды. Для этого надо посмотреть в закладку Generic всех атакующих команд и все скрипты с именами «Attack enemy base front», «Attack enemy base flank» и «Attack enemy base back» заменить на новый скрипт «Attack enemy base dynamic». Новый скрипт будет проверять значение переменной

«ATTACK_PATH» и посылать атакующие команды по нужному пути. Скрипту нужно будет выбирать один из 3 путей. Как это сделать – смотрите в главе «Создание алгоритма выбора».

Остается только напомнить, что в скриптах есть флаг «_LAUNCH_ATTACK». Когда этот флаг устанавливается в значение «TRUE», все атакующие команды начинают наступление. Значение флага проверяется в скриптах серии «Attack Sequence» из группы «Generic Attack». Проверьте всё внимательно – этот флаг должен устанавливаться в «TRUE» только после того, как разведка завершена, ее результаты обработаны, и переменная «ATTACK_PATH» до конца сформирована. В противном случае, войска пойдут в атаку перед тем, как разведка завершит работу, и обязательно выберут неправильное направление. Также следует напомнить, что флаги и переменные, используемые при разведке, нужно устанавливать в их начальные значения сразу же в начале игры и потом повторять инициализацию каждый раз после запуска очередной атаки. В противном случае, данные от следующей разведки смешаются с данными от предыдущей и получится ерунда.

15. Добавление новой боевой единицы.

Для начала новая боевая единица должна быть добавлена в INI-код. Важно добаыить кнопку для строительства новой единицы. Когда вопрос с кодом решен, можно приступить к скриптам.

Первым делом надо классифицировать новую единицу и занести ее в списки. Классификация и занесение в списки происходит в скриптах в группе «PaperWork» в скрипте «Define Object Lists». Нужно добавить новую единицу в соответствующий список. Такое добавление нужно сделать 12 раз – у каждого генерала свои списки.

Потом нужно добавить новую единицу в команды. Для этого надо отредактировать все команды серии «1ST_WAVE», «2ND_WAVE» и т.д. до «5TH_WAVE» и в них добавить новую единицу. Нужно отдавать себе отчет в том, что команды серии «1ST_WAVE» и «2ND_WAVE» строятся в самом начале игры, а «5TH_WAVE» - в конце. Если новая единица будет доступна в начале игры, то логично внести ее только в «1ST_WAVE» или «2ND_WAVE», если в конце, то логично внести ее только в «5TH_WAVE» или «4TH_WAVE». Далее, если нужно, можно внести новые единицы в команды серии «Guard Combat Zone». Тогда ИИ будет строить их, и ставить в центре карты. Можно внести новые единицы в команды серии «Guards». Тогда ИИ будет охранять ими базу. Можно внести новые единицы в команды серии «Garrison». Тогда ИИ в начале игры будет запускать их в нейтральные здания. И т.д. Если хотите, можете сделать одну или несколько команд, в которых будут только новые единицы.

16. Добавление новой стороны.

Начать надо с того, что описать новую фракцию в файлах Faction.ini и PlayerTemplate.ini. Далее, надо распаковать все карты генералов и в WorldBuilder-е добавить в них нового игрока (меню «Edit», пункт «Edit Player List», кнопка «Add new player»). Новому игроку нужно присвоить новую фракцию. Нужно добавить нового игрока в КАЖДУЮ КАРТУ. Иначе, ничего не будет работать. Когда все карты отредактированы, откройте SkirmishScripts.scb. Перед открытием или после него, еще раз добавьте нового игрока. На этот раз, игрок добавляется уже в сам ИИ.

Структуру скриптов и команд для новой стороны лучше не создавать заново, а скопировать и адаптировать копию для новой стороны. Редактор не позволяет копировать несколько скриптов или команд сразу – только по одной. Поэтому приходится выкручиваться. Чтобы скопировать огромное количество скриптов, нужно сделать их экспотрт, а потом – импорт. Для этого надо выделить группу скриптов (например, SkirmishGLAStealthGeneral), нажать кнопку «Export Scripts», внизу выделить «Export selected

scripts» и нажать «Ok». Потом надо выделить группу для новой стороны, нажать кнопку «Import Scripts» и выбрать файл, который вы только что сохраняли.

Потом нужно накопировать команды. Идем в редактор команд. Много команд сразу скопировать не удастся. Поэтому выбираем любую сторону и копируем команды по одной. После нажатия кнопки «Copy Team» копия команды появляется в самом низу списка. После того, как создадите копии всех команд, нужно будет щелкнуть мышью 2 раза на каждой копии и изменить сторону-владельца в пункте «Owner». Разумеется, ставим новую сторону.

Далее открываем редактор скриптов, открываем группу «PaperWork», открываем скрипт «Define Object Lists» и добавляем туда все новые здания, и боевые единицы. Новый скрипт «Define Object Lists» должен быть у всех сторон в игре. Логично скопировать только что отредактированный скрипт, раздать его копии всем сторонам, не забыв стереть его старые версии.

Далее, откройте группу для новой стороны и детально изучите каждый скрипт на предмет наличия имен каких-либо объектов. Например, в некоторых скриптах явно прописаны названия зданий. Эти названия нужно заменить новыми, которые соответствуют новой стороне.

Далее, откройте группу «Upgrades» и пропишите все апгрейды для новой стороны, удалив старые данные.

Далее, откройте группы «Generals Powers» и «Super Weapon Use» и пропишите все супер оружия и силы генералов для новой стороны.

Далее, идем в редактор команд. Нужно отредактировать каждую команду, заменив прописанных там боевых единиц на новые.

Далее, надо построить базу для новой стороны. В данном руководстве в главе «Файл AIData.ini» объясняется как это сделать. В файл AIData.ini нужно добавить новый список строительства и новый блок «SideInfo» для новой стороны.

На этом всё. Теперь остается проверить скрипты на наличие ошибок, протестировать и поправить найденные недочеты.

17. Добавление новой музыки и исправление существующей ошибки с музыкой.

Во-первых, ваши MP3 файлы должны находиться в директории «Generals Zero Hour \Data\Audio\Tracks\». Также новую музыку надо прописать в INI-коде в файле Music.ini.

Теперь вернемся к скриптам. В группе «PlyrCivilian» можно найти скрипты, которые отвечают за музыкальное сопровождение. Логика их работы такова.

1) Определить сторону игрока и установить переменную «Music Track» в 0.
2) Начать играть мелодию и запустить таймер.
3) Когда время в таймере истечет, начать играть трэк «Silence» (тишина) – это пауза между мелодиями. При этом запускается еще один таймер.
4) Когда время в таймере истечет, прибавить к «Music Track» 1 и перейти к пункту 2 Т.е. начинается следующая мелодия и запускается следующий таймер. И так далее пока не проиграются все мелодии. Потом «Music Track» сбрасывается в 0 и играется опять первая мелодия.

Недостаток этой логики в том, что таймер отсчитывает игровые секунды, которые не всегда равны реальным. На некоторых компьютерах игра работает медленнее – на них игровые секунды больше реальных. Поэтому после окончания мелодии будет большая пауза. На некоторых компьютерах игра работает быстрее – на них игровые секунды меньше реальных. Поэтому после мелодия оборвется не доиграв до конца.

Чтобы устранить эту ошибку, нужно запускать следующий трек только тогда, когда предыдущий закончится. Это делается с помощью условия «Music track has completed some number of times» из раздела «Multimedia». Данное условие можно использовать по-разному. Я, лично, предпочел убрать все паузы («Silence») и запускать новый трек сразу же по завершении предыдущего. Хотя, паузы можно и сохранить.

Чтобы добавить свою музыку, нужно либо просто записать имена новой музыки на место старой в INI-коде, либо создать в INI-коде новые блоки с именами новой музыки и, соответственно, прописать эти новые блоки в скриптах. Если вы исправили ошибку с таймерами, то на этом можно закончить. Если не исправляли, то придется отредактировать все музыкальные таймеры и проставить в них продолжительность мелодий.

18. Исправление существующей ошибки с апгрейдами.

Если вы посмотрите в скрипты группы «Upgrades», то увидите интересную картину. ИИ запускает апгрейд и сразу же выключает соответствующий скрипт, не дожидаясь завершения апгрейда. Это приводит к тому, что ИИ может сделать не все апгрейды.

Например, ИИ заказывает 2-3 апгрейда в стратегическом центре, после чего вы этот центр бомбите, и все начатые и не начатые апгрейды теряются. Человек в таком случае отстраивает стратегический центр заново и заново запускает все апгрейды. ИИ этого не делает. Он просто отстраивает стратегический центр и всё. Скрипты, отвечающие за апгрейды, уже отключены и НИКОГДА больше не включатся. Поэтому, если ИИ 2-3 раза потерял стратегический центр или военный завод, то будьте уверены в том, что половину апгрейдов ИИ никогда не сделает.

Для того, чтобы это исправить, надо для каждого апгрейда делать 2 скрипта. Первый будет пытаться запустить апгрейд каждые 20-40 секунд. В этом первом скрипте не должно быть галочки напротив «Deactivate on success». Второй скрипт каждые 20-40 секунд будет проверять, а не завершен ли соответствующий апгрейд. Проверка осуществляется с помощью условия «Player built an upgrade» из раздела Player->Upgrades. Если завершен, то скрипт отключает самого себя (т.е. у него есть галочка напротив «Deactivate on success») и отключает скрипт, запустивший апгрейд с помощью действия «Disable Script» из раздела Scripting->Script.

Также следует отметить многочисленные ошибки в перечне тех апгрейдов, которые ИИ пытается запустить:

1) Пехотный генерал пытается сделать апгрейд «Nationalism», а должен делать «Fanaticism», потому что «Nationalism» ему дается с самого начала игры.
2) Генеральша пытается сделать апгрейд «Composite armor», хотя этого апгрейда по условиям игры у неё нет.
3) Ни летчик, ни генеральша никогда не получают апгрейд «Advanced training», т.к. в запускающих его скриптах прописано условие того, что «Advanced training» можно начинать делать только тогда, когда сделан апгрейд «Composite armor». А апгрейда «Composite armor» у этих генералов нет.
4) Все американские генералы пытаются сделать апгрейд «MOAB» после получения генеральской силы «Daisy Cutter». Но проверка на наличие генеральской силы работает со сбоями, поэтому апгрейд «MOAB» в 90% случаях они заполучить не могут. Для исправления этой ошибки нужно действовать через переменную или флаг. Т.е. один скрипт проверяет наличие генеральской силы и устанавливает флаг. Второй скрипт проверяет флаг и запускает апгрейд.

19. Исправление существующей ошибки с бомбовым грузовиком.

В игре у бомбового грузовика есть возможность маскироваться. Но ИИ эту возможность никогда не использует. Его можно этому научить.

Во-первых, каждому бомбовому грузовику ГЛА в INI-код нужно добавить этот модуль:

```
Behavior = CommandButtonHuntUpdate BTDisguiseHunt
End
```

Принцип действия такой. Сначала грузовику нужно выполнить действие «Team begins

hunting using Ability 'Command_DisguiseAsVehicle'». От этого он замаскируется. Потом нужно выполнить действие «Team begins hunting». Тогда он пойдет в атаку. Проблема заключается в том, что после первого действия надо подождать.

Один из способов реализации такой. Создаем два новых скрипта и ставим их в режим подпрограммы. Первый скрипт должен иметь условие – «True» и действия:

1) Team begins hunting using Ability 'Command_DisguiseAsVehicle'.
2) Set timer «Bombtruck_Disguise_Done» to expire in 5.00 seconds.
3) Set Flag «Bombtruck_Is_Disguised» to TRUE

Второй скрипт должен иметь условия:

1) Timer «Bombtruck_Disguise_Done» has expired.
2) *AND* Flag «Bombtruck_Is_Disguised» IS TRUE

и действия

1) Team begins hunting.
2) Set Flag «Bombtruck_Is_Disguised» to FALSE

Первый скрипт нужно прописать в команде в графе «On Create», второй – в закладке «Generic». При этом грузовик будет маскироваться сразу же после постройки, и через 5 секунд пойдет в атаку. Остается только напомнить, что флаг «Bombtruck_Is_Disguised» и таймер «Bombtruck_Disguise_Done» должны быть обязательно проинициализированы в начале игры. Таймер нужен для того, чтобы подождать определенное время. Флаг нужен для того, чтобы выполнить действие «Team begins hunting» только 1 раз. Без этого флага действие «Team begins hunting» будет выполняться в каждом кадре и сильно тормозить игру.

20. Исправление существующей ошибки с боевым автобусом.

В игре у ГЛА есть боевой автобус, в который нормальные игроки сажают пехоту. ИИ так делать не умеет. Он просто строит боевой автобус, 8 ракетчиков и посылает их в атаку отдельно. Причина этой ошибки кроется не в скриптах, а в коде. Для ее исправления нужно в INI-коде у всех боевых автобусов в строку «KindOf» добавить слово «TRANSPORT».

Еще нужно проверить команды. Команды, использующие боевой автобус, НЕ должны быть установлены в режим «Aggressive». (Проверяется в закладке Behavior).

21. Исправление существующей ошибки с американским ракетчиком.

В игре у американских ракетчиков есть возможность использовать лазерное наведение. Но ИИ его никогда не использует. Его можно этому научить.

Для этого, в INI-коде всем американским ракетчикам нужно добавить следующий блок кода:

```
Behavior = CommandButtonHuntUpdate ModuleTag_Hunt01
End
```

Далее, нужно создать скрипт «USA Laser Lock» в группе «Generic Attack». Скрипт должен быть таким:

Пункт «Deactivate upon success» – без галочки.

Пункт «Script is Subroutine» – с галочкой.

В закладке «Condition» – Только одно «True»

В закладке «Actions if true» - только одно действие – «Set to hunt using commandbutton ability». В этом действии вместо «Team ???» поставить «This Team» и вместо «Ability ???» поставить «Command_AmericaMissileDefenderLaserGuidedMissiles».

Далее, в командах, в которых присутствуют ракетчики можно задать новый скрипт «USA Laser Lock» для какого-нибудь события. Например, «enemy sighted» или «on create».

22. Как ИИ использует кнопки.

ИИ, как ни странно, нажимает кнопки так же, как и игрок. Для того, чтобы ИИ что-нибудь построил, или использовал какую-либо специальную возможность, в наличии должна быть соответствующая кнопка. Но у ИИ есть 3 особенности:

1) В CommandSet.ini кнопкам присваиваются номера. Всего номеров 14, т.к. у игрока на экране отображаются только 14 кнопок. Но в этом файле можно прописать и 15-ю, и 16-ю кнопку и так далее. Номера можно использовать без ограничений. У игрока кнопки с номерами больше 14 на экране не будут появляться, но ИИ ими сможет спокойно пользоваться. Для чего это надо – читайте следующий абзац.

2) Если у строителя или у завода слишком много возможностей, то прибегают к смене CommandSet-а. У рабочего ГЛА в Zero Hour как раз есть такая смена – переключение между строительством обычных и поддельных зданий. Но ИИ не умеет переключать CommandSet-ы сам. Для него нужно либо писать какое-то нагромождение скриптов, чтобы он нажимал на кнопку смены CommandSet-а, либо сделать проще – задать 15-ю, 16-ю и другие кнопки.

3) ИИ может пользоваться кнопками с пометкой SCRIPT_ONLY. Если в CommandButton.ini в коде кнопки написано Options = SCRIPT_ONLY, то для игрока эта кнопка будет не видна, но зато ИИ сможет ее нажимать.

23. Как ИИ реагирует на ReplaceObjectUpgrade

В коде встречается тэг ReplaceObjectUpgrade, который при после апгрейда заменяет объект на другой. ИИ на него реагитует очень специфически.

Случай первый – ReplaceObjectUpgrade в боевой единице. В Zero Hour этот случай не встречается, но в своих модах вы можете использовать этот прием. Протестировано и достоверно определено, что подмена объекта никаким образом не влияет на команду. ИИ воспринимает новый объект как «родной». Новый объект остается в команде и подчиняется всем скриптам, которые в описании команды прописаны. Здесь могут возникнуть только 2 возможные ошибки. Первая – новый объект, прописанный в ReplaceObjectUpgrade, может быть не внесен в список ObjectLists из группы Paper Work (тогда ИИ противника будет воспринимать новую боевую единицу как что-то незначимое и в некоторых случаях не будет на нее никак реагировать). Вторая – сразу же после замены старого объекта на новый, боевая единица «забывает» про свой последний приказ. Например, если боевая единица куда-то ехала, то она обязательно остановится, если в дороге сработает ReplaceObjectUpgrade. Остановившаяся боевая единица будет стоять до тех пор, пока скрипты не дадут команде следующий приказ.

Случай второй - ReplaceObjectUpgrade в здании. В Zero Hour это используется для апгрейда поддельных зданий ГЛА до настоящих. Дело в том, что ИИ не зависимо от скриптов пытается сам восстанавливать утраченные здания. Если срабатывает ReplaceObjectUpgrade, то для ИИ это равносильно потере здания. А, значит, он будет пытаться построить его снова. В результате ИИ будет делать примерно следующее. Сначала он построит здание, потом сделает апгрейд для него. Вследствие апгрейда сработает ReplaceObjectUpgrade и заменит здание на другое. ИИ подумает, что он потерял здание и построит его еще раз рядом. Потом опять сделает апгрейд, подумает, что он опять потерял здание и построит еще одно. И так далее. Когда вокруг уже не будет места, он начнет строить здания «в куче» - т.е. одно поверх другого. Он не угомонится до тех пор, пока не у него не кончатся деньги.

24. Часто совершаемые ошибки.

1) Скрипт «Set to hunt using commandbutton ability» не работает, если у боевой единицы в INI-коде нет блока «CommandButtonHuntUpdate».

2) Все три скрипта из раздела «Team» -> «Transport» (загрузка в транспорт) не работают, если у транспорта в INI-коде в строке KindOf нет слова TRANSPORT. Также загрузка в транспорт не работает, если команда переключена в режим «Aggressive».
3) Скрипт «Build a building of type» не работает, если в файле AIData.ini в соответствующем списке строительства отсутствует здание, указанное в скрипте. Также следует обращать внимание на количество зданий в списке строительства.
4) Запуск генеральских сил не работает, если ИИ не приобрел соответствующую возможность. Приобретение возможностей можно изменить в файле AIData.ini в блоке «SideInfo».
5) ИИ внезапно прекращает строительство своей базы, если не соблюдено дерево развития. Например, если сначала дать команду построить стратегический центр, а потом – военную фабрику, то ИИ не построит ни того, ни другого. К строительству военной фабрики он не приступит до тех пор, пока не выполнит предыдущий заказ на строительство стратегического центра, а стратегический центр не сможет построить, т.к. нет военной фабрики.
6) ИИ не ходит в атаку, если автор карты забыл добавить зоны и пути со специальными метками.
7) ИИ вообще ничего не делает, если автор карты забыл сформировать список игроков (Player List).
8) Игра сильно тормозит, если какая-либо команда в каждом кадре запускает действие «hunt» или «begin hunting».
9) Действия «Run Subroutine» (раздел Scripting->Script) и «Execute script sequentially» (раздел Team-> SequentialScript) не работают, если вызываемый в них другой скрипт не переключен в режим подпрограммы.
10) Иногда скрипты могут импортироваться и экспортироваться некорректно, если в директории «My Documents / Generals Zero Hour Data /» есть файл с именем «_tmpchunk.dat».
11) Если вы одновременно открыли редактор скриптов и редактор команд, то все изменения, сделанные в редакторе команд, потеряются.
12) Если вы редактируете свойства команды и задаете ей какой-либо скрипт, то имя скрипта нужно обязательно выбрать мышкой. Если вы где-то скопировали имя скрипта и вставили его, то WorldBuilder это проигнорирует. Чтобы не игнорировал, нужно выполнять следующее шаманское действие – вставляем заранее скопированное имя скрипта, жмем на стрелку и в появившемся меню мышкой выбираем этот же скрипт. После чего можно жать на «ОК».