РУКОВОДСТВО ПО СКРИПТАМ ГЕРОЕВ ПЯТЬ.
«ОТ ЧАЙНИКА К ЧАЙНИКАМ»

автор: ogo

Вступление

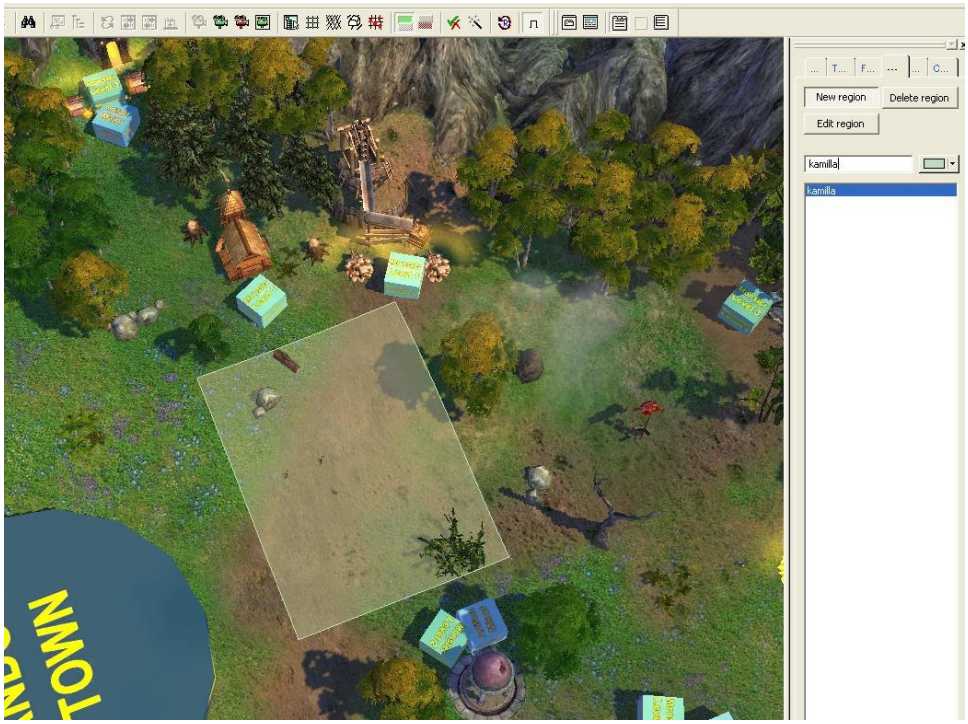
Это руководство предназначено для картостроителей, которые хотели бы пользоваться скриптовыми функциями в своих картах к Героям 5, однако не могут этого делать по незнанию механизма их работы и программного языка.
Не могу сказать, что я сам много знаю, но то, что знаю, попытаюсь донести до читателя: максимально доступно объяснить и привести примеры.

Раньше, в тройке, в редакторе карт были стандартные Event'ы. Они давали достаточно много интересных возможностей и были просты в использовании для любого. В четвёртых героях появилась уже скриптовая система, однако она заключалась в использовании заготовленных пресетов. Это было гораздо тяжелей, однако и возможностей стало гораздо больше. Теперь, в пятой части, нам дали пользоваться практически неограниченными возможностями, однако отобрали малейшие признаки удобства для неподготовленного пользователя.
На самом деле, там ничего страшного нету, в этих ЛУА-скриптах, однако это только, если разобраться. А разбираться можно долго и нудно, как это делал я – путём ошибок и бесконечных загрузок игра <–> редактор карт. Это руководство призвано упростить ваше освоение в мире скриптов к Героям 5.
Настоятельно рекомендую обзавестись руководством по скриптам многоуважаемого Novik65. Практически всё время в этом документе я буду ссылаться на него и на официальное руководство по скриптам от Нивала (HOMM5_Script_Functions_rus).
Делаться это будет на примерах. Педагогических познаний у меня минимум, опыта в написании подобных документов тоже не имеется. Так что не обессудьте, коль чего.

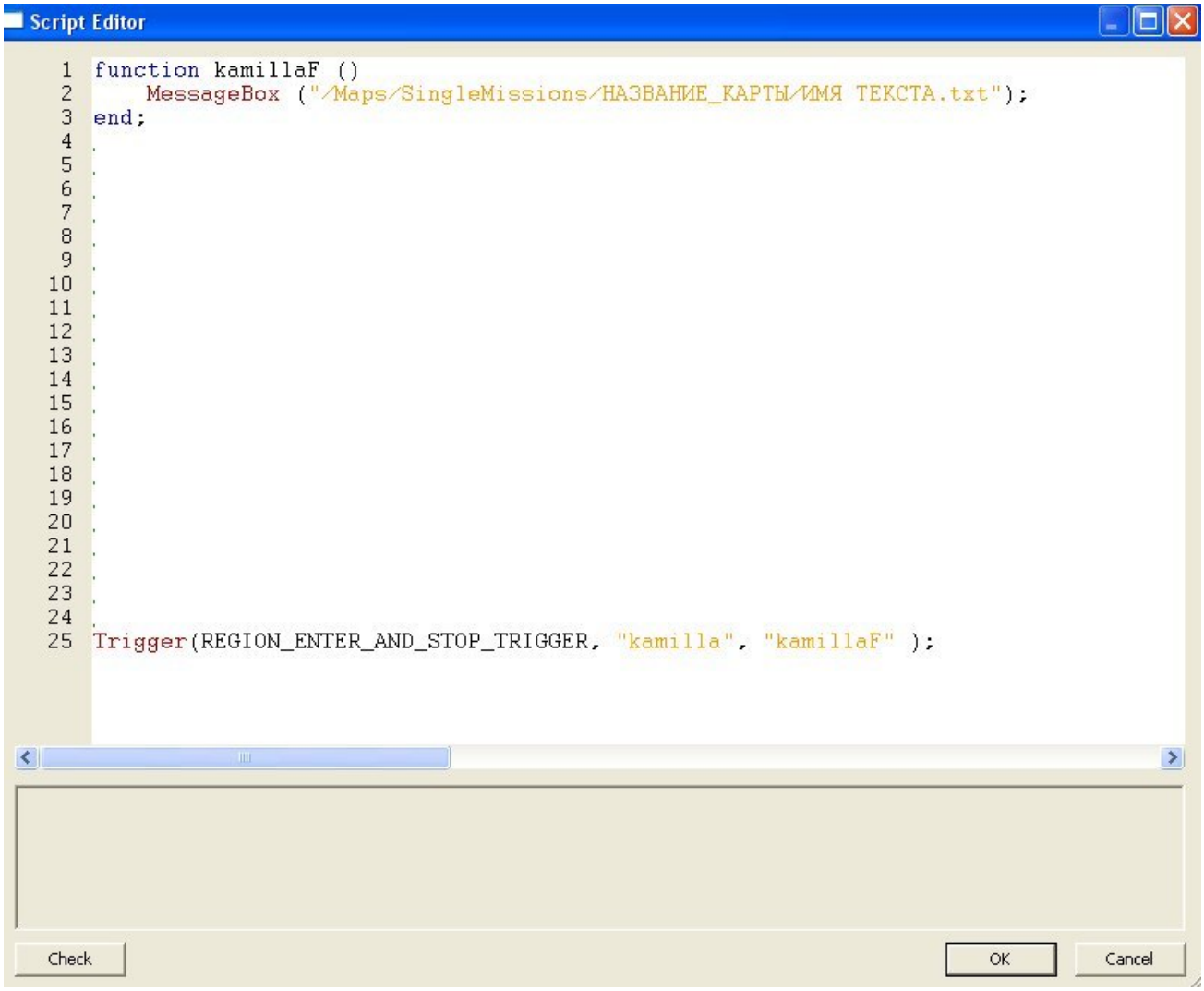
1. Основы

Практически всё в этом мире делается с помощью триггеров. Выглядит, он, к примеру, так:
Trigger(REGION_ENTER_AND_STOP_TRIGGER, "kamilla", "kamillaF");
Триггер будет висеть в игре всё время, и реагировать на те «раздражители», которые вы ему указали. Например, этот будет реагировать на вход в регион "kamilla" героя и вызывать в это время функцию "kamillaF". Рассмотрим этот пример гораздо более подробно.

Вот есть регион:



Он назван нежным именем kamilla. Что нужно сделать, чтобы он реагировал на то, что в него кто-то вошёл и при этом выдалось текстовое сообщение? Правильно, повесить триггер.
Смотрим в официальное руководство по скриптам от Нивала (находится по адресу: "папка с игрой\Editor Documentation") или в руководство Новика и находим наиболее (и единственно) подходящий триггер - Trigger(REGION_ENTER_AND_STOP_TRIGGER, sRegionName, sProc) (Процедура (sProc) вызывается, когда какой-либо герой останавливается в регионе с именем sRegionName).
Триггеры записываются обязательно в том виде, котором они предложены. То есть этот мы запишем так: Trigger(REGION_ENTER_AND_STOP_TRIGGER, «теперь в кавычках пишем название региона, которое мы ему дали», «имя функции, которая будет вызываться, когда мы туда войдём»), и после этого это должно выглядеть так -
Trigger(REGION_ENTER_AND_STOP_TRIGGER, “kamilla”, “kamillaF”);
Название функции можете давать любое, но я для удобства даю то же, что и у региона (в данном случае), только добавляя буковку «F».
Откуда взять функцию, которая будет вызываться? Всё очень просто – написать её.
Вот как это выглядит:



(Скрипты вписываются только в Map Properties -> Scripts)

Это обязательная надпись function, затем название, отмеченное в триггере, затем пустые скобки. Зачем? Не знаю, но без них иногда не работает.

Потом, в следующей строке, идёт собственно команда. **MessageBox ()** вызывает окошечко с описанным вами текстом. Для того чтобы игра могла прочитать текст, текстовый файл в формате Unicode должен находиться в папке с картой по тому адресу, на который ссылается скрипт (/Maps/SingleMissions/НАЗВАНИЕ_КАРТЫ) Как туда файл положить? Откройте карту любым архиватором - это обычный zip-архив. Вложите файл в конечную папку. Как сделать файл в формате Юникод? Скопируйте любой текстовый документ с карты (к примеру, name) и сделайте с него копию, а потом вложите обратно уже с новым текстом и названием, как было описано выше. Все имена следует давать латинскими буквами.

Вместо **MessageBox** можно вписать любую другую команду или хоть десять **MessageBox**ов, все они будут выполняться по очереди.

!Внимание – как именно пользоваться командами достаточно подробно описано у Новика и в официальном руководстве по скриптам от Нивала. Поэтому это не входит в задачу данного руководства.

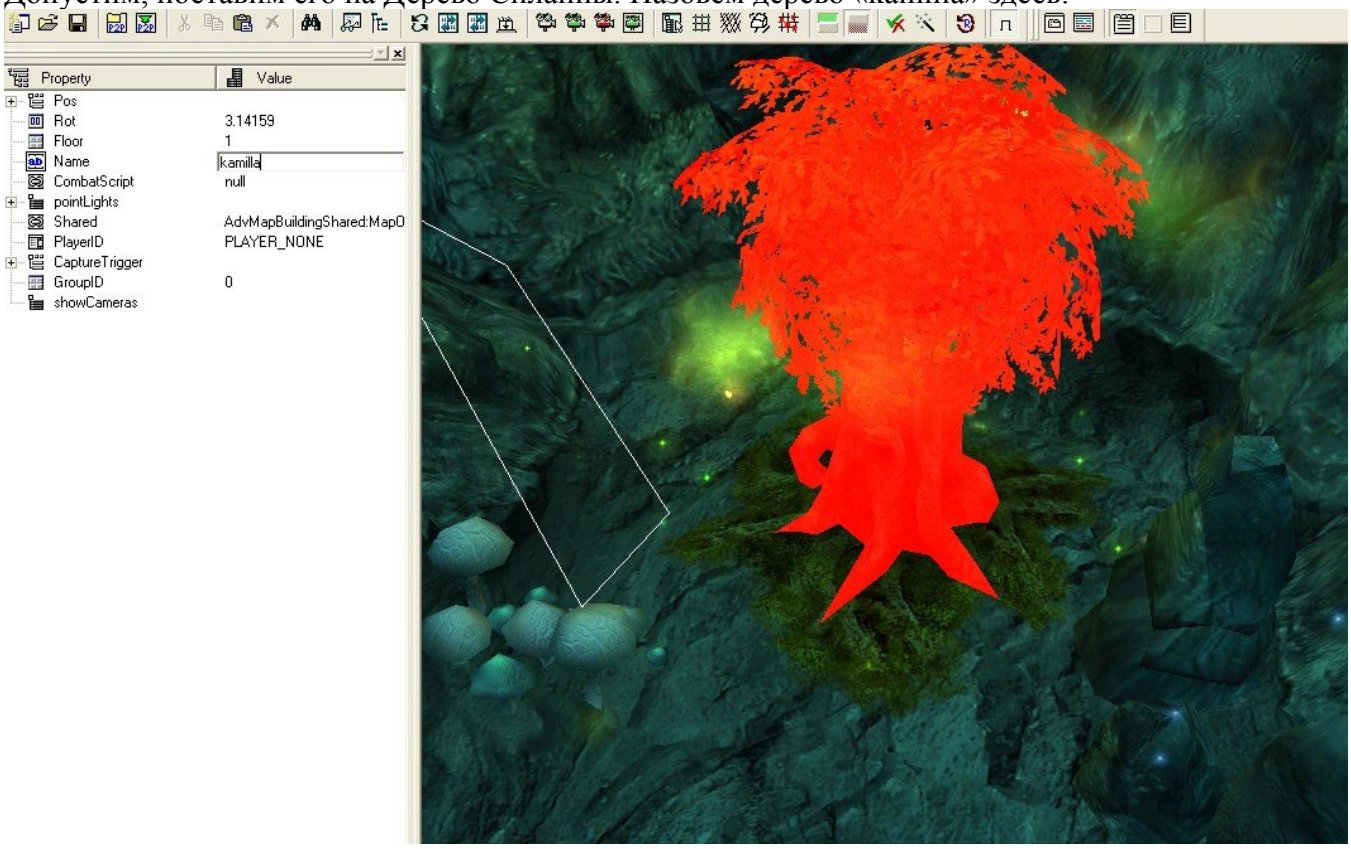
Обязательно следите за тем, чтобы команды были написаны точно так, как и в подано в руководствах. **!**Помните, большая и маленькая буквы – это разные символы! Так же не забывайте после каждой команды ставить точку с запятой.

В конце обязательно поставьте **end;**

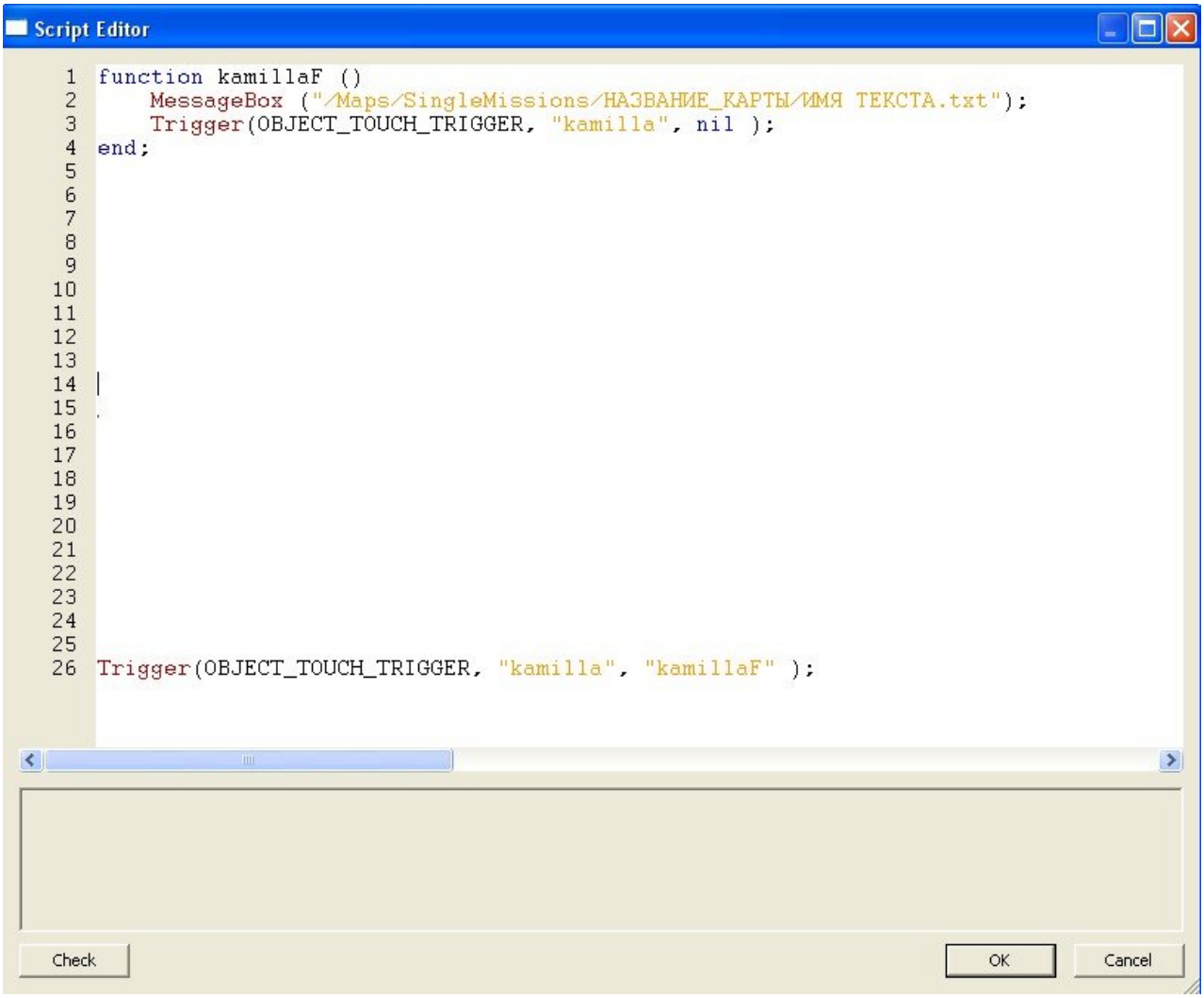
Так же вы можете использовать вместо REGION_ENTER_AND_STOP_TRIGGER другой триггер: REGION_ENTER_WITHOT_STOP_TRIGGER – разница в том, что в первом варианте герой войдёт и остановится в регионе, во втором случае он останавливаться не будет.
Вместо **MessageBox** бывает удобней использовать **ShowFlyingSign** — эта команда отображает отлетающее от объекта сообщение.
Чаще всего нужно, чтобы функция срабатывала лишь однажды, когда герой впервые зайдёт в обусловленный регион. Для этого обнулите триггер и, желательно, запишите это в самом начале функции (однако можно и в конце – где запишете, там и обнулится). Обнуляется он просто: Trigger(REGION_ENTER_AND_STOP_TRIGGER, “kamilla”, nil); то есть вместо имени функции вписывается значение nil… Выглядит теперь функция так:

```
function kamillaF ()
  Trigger(REGION_ENTER_AND_STOP_TRIGGER, "kamilla", nil );
  MessageBox ("/Maps/SingleMissions/ НАЗВАНИЕ_КАРТЫ / НАЗВАНИЕ_ТЕКСТА.txt");
end;
```

Давайте рассмотрим другой случай. Вместо триггера на то, чтобы герой входил в регион, поставим другой, который реагирует на прикосновение к объекту на карте. Ищём. Находим - OBJECT_TOUCH_TRIGGER — взаимодействие с интерактивным объектом.
Допустим, поставим его на Дерево Силанны. Назовём дерево «kamilla» здесь:



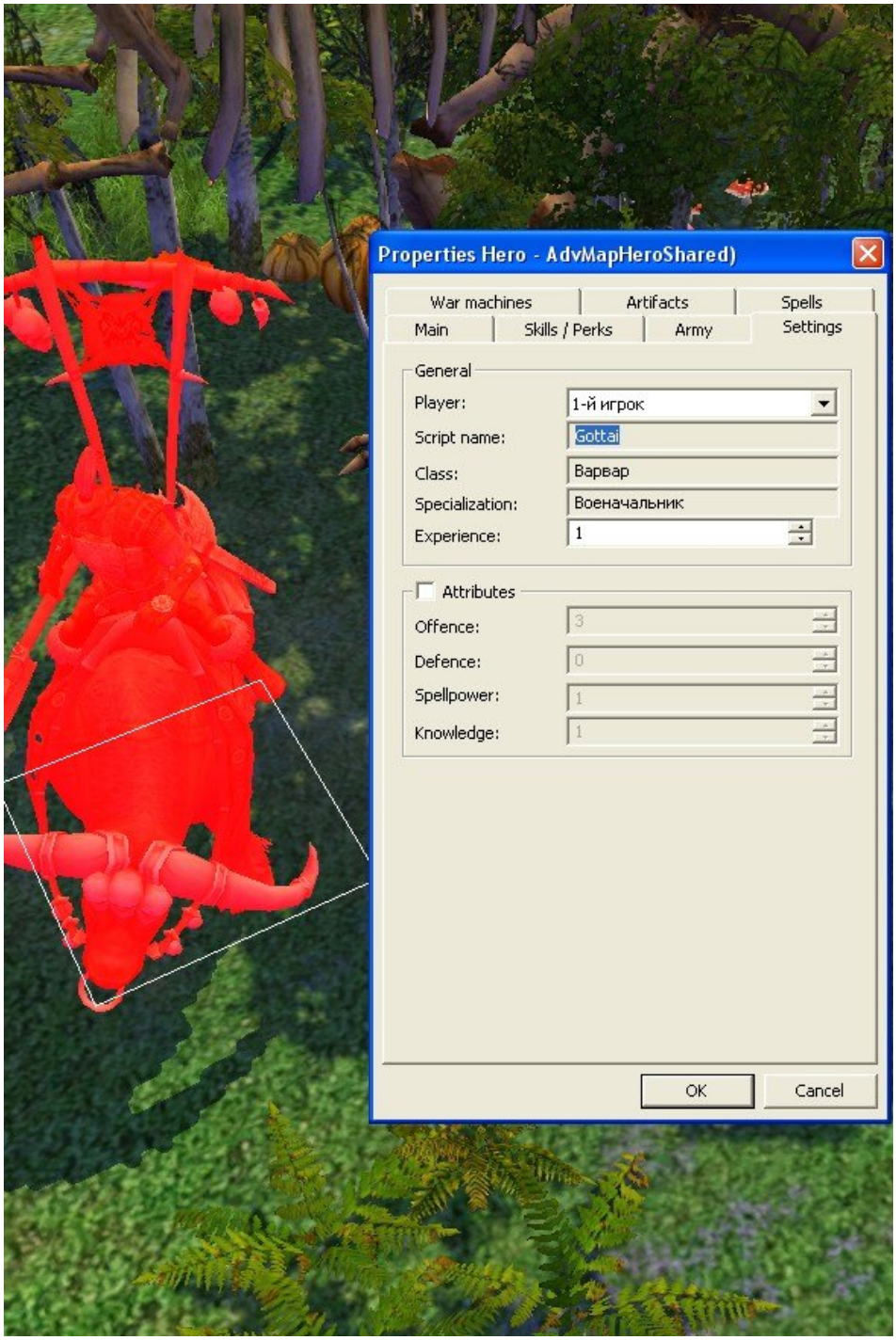
и впишем значения в триггер - Trigger(OBJECT_TOUCH_TRIGGER, “kamilla”, “kamillaF”);



И что теперь?
Теперь, когда вы заходите в дерево, срабатывает функция kamillaF. Правда дерево ещё и опыт приносит. Ну что ж… Если вас это не устраивает, запишите в скриптах (не в функции, а в отдельном поле) следующую строчку:
SetObjectEnabled("kamilla", false); - что можно прочитать как УстановитьОбъектРабочим (камиллу, гнусная ложь!). Теперь дерево не будет предлагать повысить опыт.

Подобным образом вы можете использовать любые из описанных в названых мною руководствах триггеры и вписывать любые команды.

К примеру, там, где заскриптован вход в определённый регион, к всплывающему сообщению можно добавить **StartCombat** (); (начать битву – описание команды в руководстве Нивала или Новика) и тогда это станет похоже на стандартный Event из тройки. Правда тогда вам понадобится вписать скриптовое имя героя. Взять его нужно здесь:



!Если вы не хотите использовать конкретного героя, а рассчитываете скрипты на случайного, тогда вам следует вписывать в скобки возле названия функции что-то вроде **heroname**, а вместо скриптового имени героя впишите то же самое (**heroname)** – всё без кавычек (вместо **heroname** вы можете вписать любое другое слово – в функции оно будет обозначать имя активировавшего её героя). Например:

```
function kamillaF (heroname)
	MessageBox ("/Maps/SingleMissions/ НАЗВАНИЕ_КАРТЫ / НАЗВАНИЕ_ТЕКСТА.txt");
	sleep (5);
	StartCombat (heroname, nil, 5, 74, 6, 74, 4, 81, 4, 74, 4, 74, 6, nil);
	Trigger(REGION_ENTER_AND_STOP_TRIGGER, "kamilla", nil );
end;
```

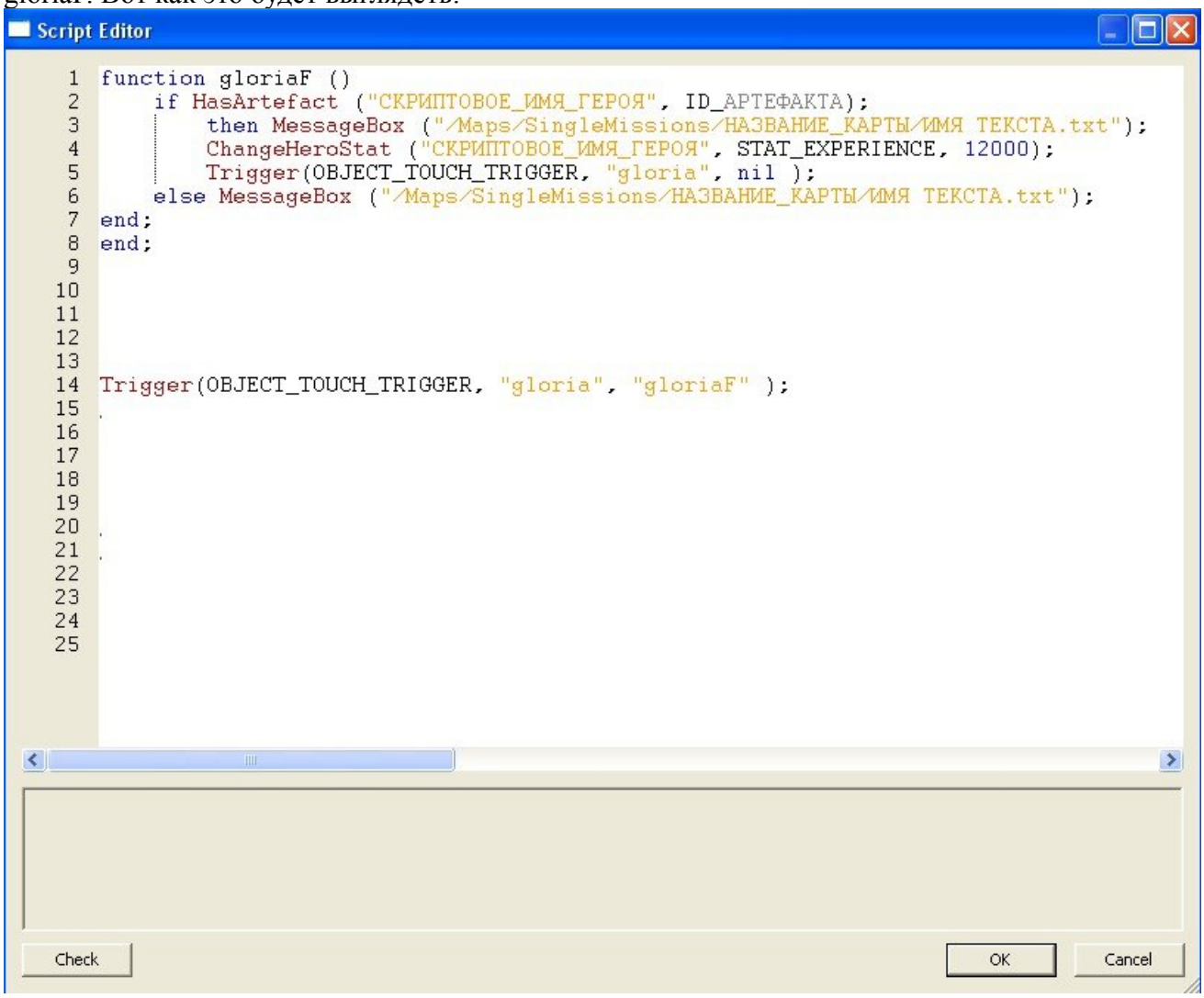
Trigger(REGION_ENTER_AND_STOP_TRIGGER, "kamilla", 'kamillaF');

Функция sleep (5); здесь особо не нужна. Я её вписал, чтобы рассказать о ней. Она играет роль паузы. В данном случае она означает, что после того, как вы нажмёте ОК на текстовом сообщении, будет пауза в одну секунду и лишь тогда начнётся бой. sleep (15); - будет означать паузу в три секунды (200мс на один sleep). Бывает очень полезно, вставлять такие вот паузы – честное слово.

Итак, я думаю с основами покончено. Из этой части можно вывести уроки, как пользоваться триггерами, куда вписывать команду, как обнулять триггеры, писать функции – то есть сами основы для скриптования карты. У вас уже есть большое поле для действий. Переходим на следующий уровень.

2. Следующий уровень

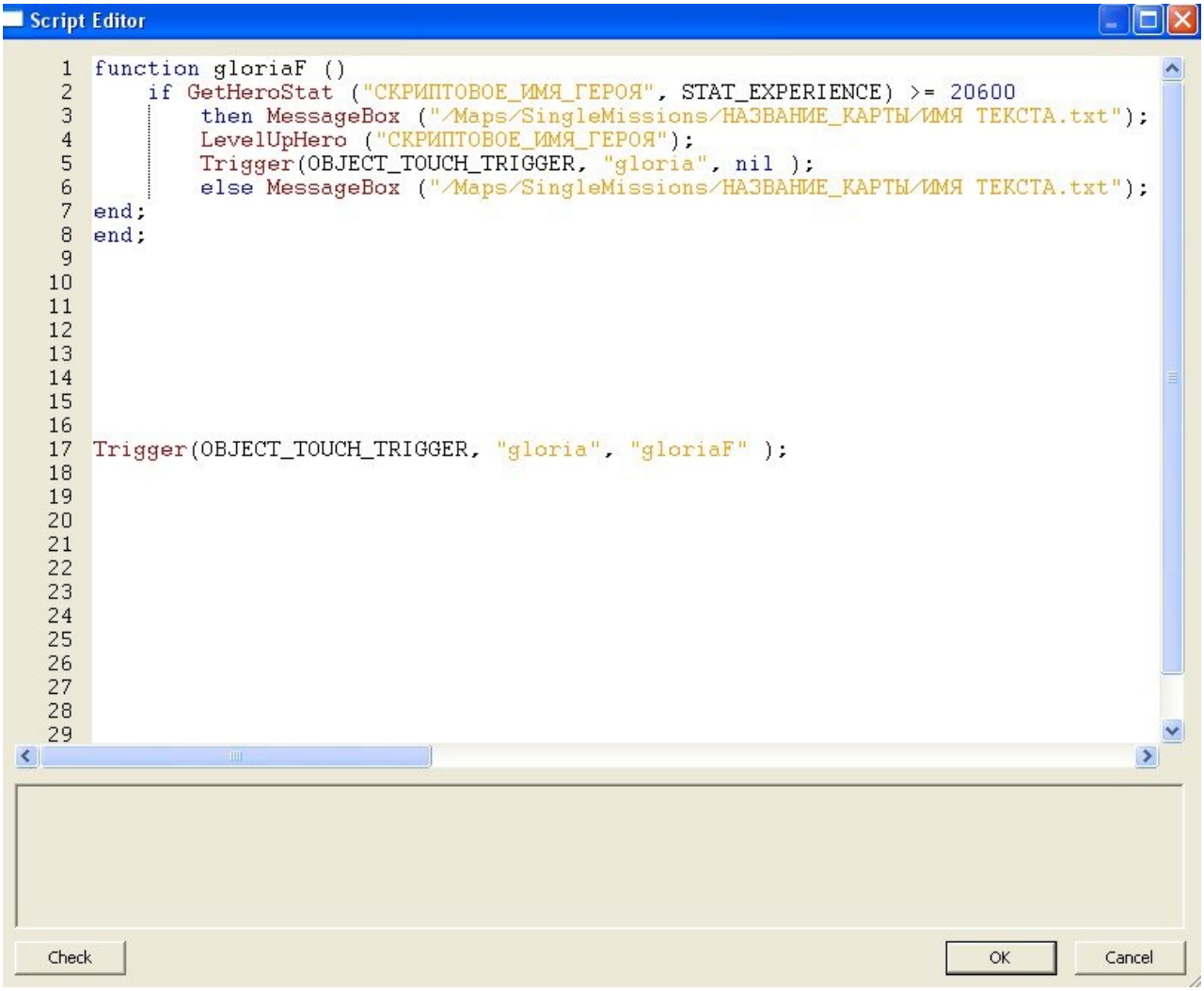
К примеру, вы хотите, чтобы герой зашёл в хижину и ему давали опыт, но только при определённых условиях. Условие – наличие определённого артефакта у героя. Называем хижину gloria и пишем, как и раньше триггер OBJECT_TOUCH_TRIGGER и функцию gloriaF. Вот как это будет выглядеть:



Появляется загадочное **if** (если). Его мы в данном случае пишем, ибо подразумеваем: Если у героя есть арт (**HasArtefact** и значения в скобках – у кого и какой именно (все **ID** описаны у Нивала в документе HOMM5_IDs_for_Scripts_rus), тогда (then) выдаём ему текстовое сообщение, затем повышаем ему характеристику опыта на 12000 единиц. Вместо **HasArtefact** () вы можете вписать множество других интересующих вас параметров: **HasHeroSkill (), HasHeroWarMachine ()** и др.
else (по-другому, иначе) использовано для того, чтобы когда герой заходит в здание без артефакта, игра выдавала ему простое текстовое сообщения, мол нету у тебя этого добра, а жаль.
Уже знакомое обнуление триггера, заметьте, использовано только для того случая, если герой выполнил задание (т.е. имел артефакт). Сделано это для того, чтобы не повторялась бесконечная выдача опыта. Однако, в то же время, герой, до тех пор пока не придёт с артом, будет сколько душе угодно натыкаться на обламывающее сообщение.
По желанию можно убрать у героя артефакт командой **ReamoveArtefact(heroName, artefactID)**.

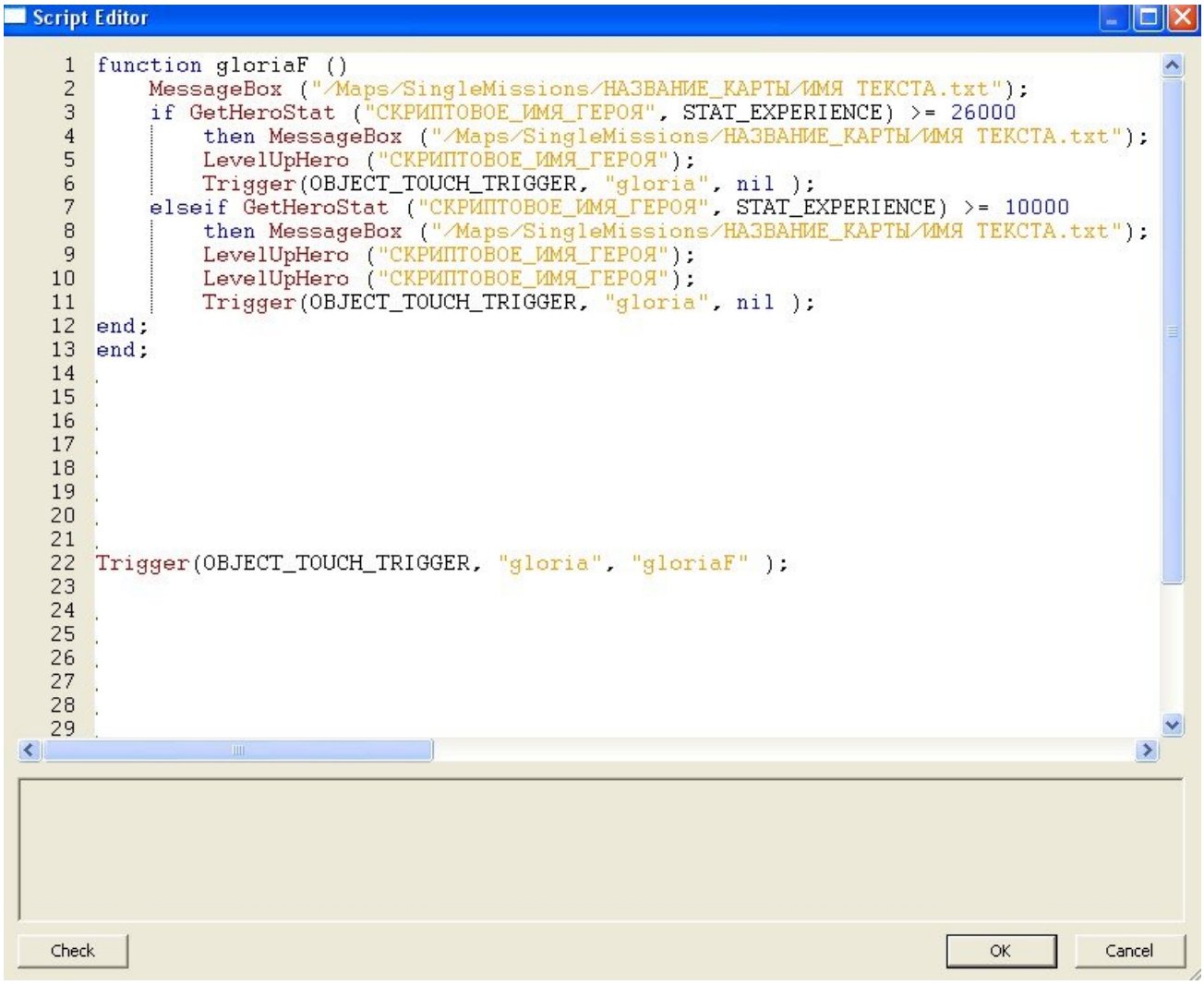
!Обратите внимание, что в конце стоит два конца (**end; end;**). Один, как и раньше, закрывает function, другой закрывает **if**. Если вы используете функции со значением **if –** не забывайте об этой детали.

Предположим, что вы хотите использовать другое условие. К примеру, если герой выше 12-ого уровня, дать ему ещё один уровень. Без вопросов, пишем:



Мы, в строчке **if GetHeroStat** («СКРИПТОВОЕ_ИМЯ_ГЕРОЯ», STAT_EXPERIENCE) >= 20600 определяем следующее: Если Герой Имеет Показатель (имя героя, опыта) больше-равно 20600 единиц, то (следующая строчка) тогда Поднять Уровень Героя (имя героя), (следующая строчка) обнулить триггер ну и так далее.
Узнать сколько очков опыта нужно для определённого уровня можно в документе HOMM5_Hero_Level_and_Experience_rus.

Возможно, вы захотите, чтобы если у героя есть 12-ый уровень, жилец хижины повышал ему уровень на один, но если у героя нет 12-ого уровня, однако есть 8-ой, то повышал сразу на два уровня. Не беда. Пишем:



Появляется **elseif** (иначе если). Выходит, у нас записано следующее: Если у героя есть 12-ый уровень опыта, тогда мы выдаём текстовое сообщение, поднимаем уровень героя, обнуляем триггер. Иначе если у героя есть 8-й уровень опыта, тогда мы выдаём текстовое сообщение, поднимаем уровень героя, поднимаем уровень героя, обнуляем триггер.
В начале для красоты добавим текстовое сообщение, которое предупреждает о том, как дальше могут развиваться события…

Естественно такого рода пример я привёл, чтобы продемонстрировать работу **if**, **else**, **elseif**, double **end;** и обозначений больше-равно. Ещё один такой пример:

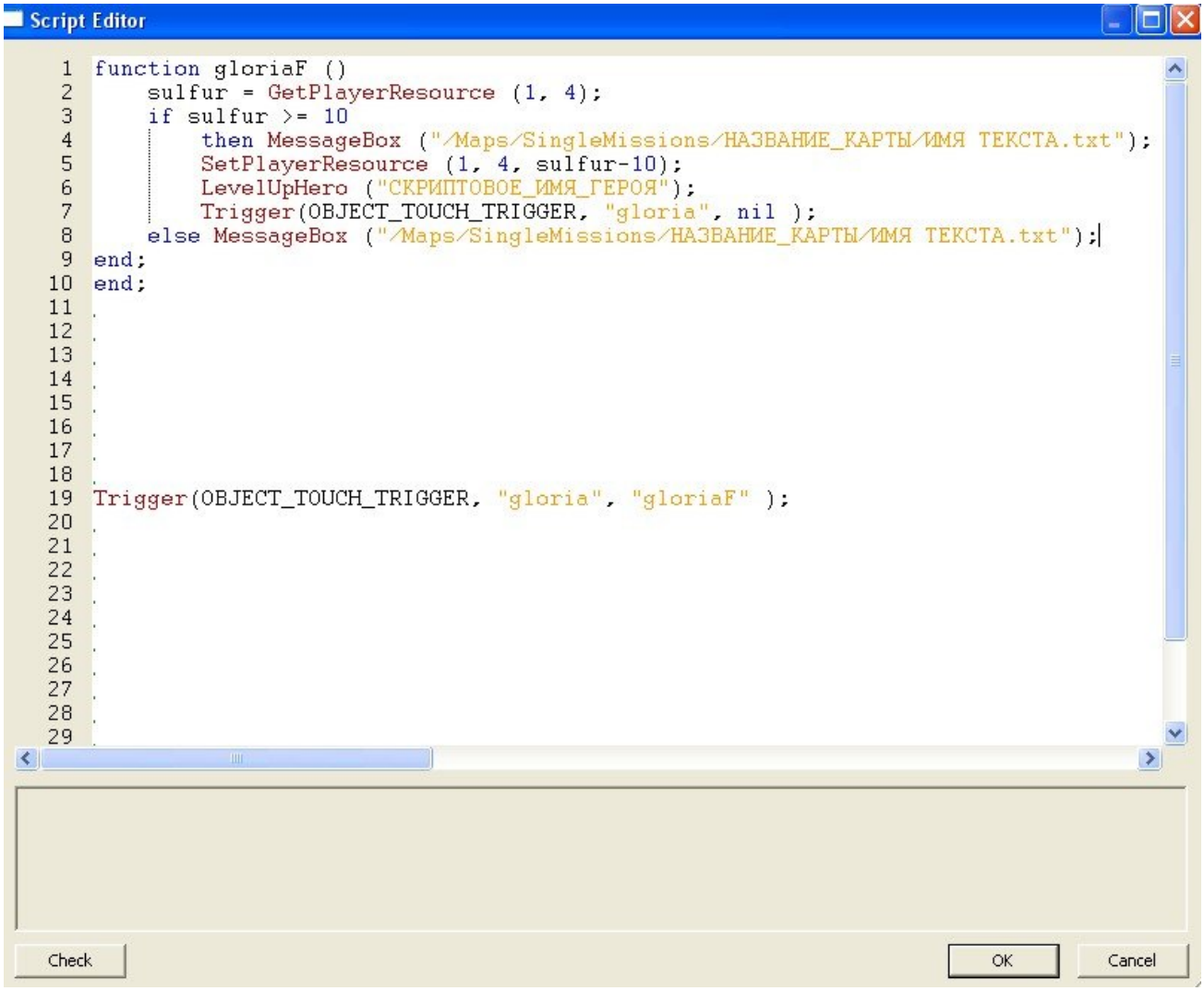
```
Trigger( NEW_DAY_TRIGGER, "hello" )

function hello()
        if GetDate(DAY) == 2 then
                MessageBox("/Maps/SingleMissions/777/TEXT05.txt");
                OpenCircleFog( 122, 58, 0, 20, 1 );
                OpenCircleFog( 38, 25, 0, 30, 2 );
                sleep(3);
                MoveCamera( 122, 60, 0, 40, 1, 1.145, 0, 0);
                SetObjectEnabled( 'green', false );
        elseif GetDate(DAY) == 3 then
        QuestionBox("/Maps/SingleMissions/777/TEXT13.txt", 'chiose1', 'choise2');
        elseif GetDate(DAY) == 4 then
        MessageBox("/Maps/SingleMissions/777/TEXT07.txt");
        elseif GetDate(DAY) == 5 then
        MessageBox("/Maps/SingleMissions/777/TEXT26.txt");
        elseif GetDate(DAY) == 6 then
        MessageBox("/Maps/SingleMissions/777/TEXT27.txt");
        elseif GetDate(DAY) == 7 then
        MessageBox("/Maps/SingleMissions/777/TEXT28.txt");
        elseif GetDate(DAY) == 8 then
        MessageBox("/Maps/SingleMissions/777/TEXT29.txt");
        elseif GetDate(DAY) == 9 then
        MessageBox("/Maps/SingleMissions/777/TEXT30.txt");
                end;
end;
```

Слегка видоизменённый фрагмент, который взят из моей карты «Одни хлопоты». Здесь мы запускаем NEW_DAY_TRIGGER, который будет на начале каждого геройского дня запускать функцию hello. Вписывать там нужно только имя функции. Функция hello имеет в себе команду **GetDate**(DAY), которая определяет какой сегодня день и если он **= =** тому дню, который вы желаете, скрипты выполнят записанное вами действие. Я выдал на второй день текстовое сообщение, открыл туман (неизведанную область) в двух местах, выдержал паузу в три десятых секунды, повёл камеру по заданным координатам и сделал неактивным на карте объект, который перед этим назвал 'green'. Далее, на каждое, указанное иным от первого, значение даты, выскакивают другие разнообразные сообщения игроку.

Главное, что можно вынести из этой части – это создание определённых условий, без которых команда(ы) не будут работать.

3. Денежные дела

Представим себе ситуацию, вы, как герой, заходите в хатку, а жилец предлагает поднять ваш уровень, но только за оплату. Такое бывает сплошь и рядом. Но что делать, чтобы это записать скриптами?
Итак, имеем команду **GetPlayerResource**(player, resourceKind); Будем считать, что торговец опытом неравнодушен к сере. Осуществляем:



Написав, что sulfur = GetPlayerResource (1, 4); мы уравняли эти значения в правах. То есть, если вы запишите, что imiamojejsobachki = GetHeroStat (“СКРИПТОВОЕ_ИМЯ_ГЕРОЯ”, STAT_EXPERIENCE); то в функции можно будет подставлять вместо GetHeroStat (“СКРИПТОВОЕ_ИМЯ_ГЕРОЯ”, STAT_EXPERIENCE); то, что вы напишете перед знаком “=”.

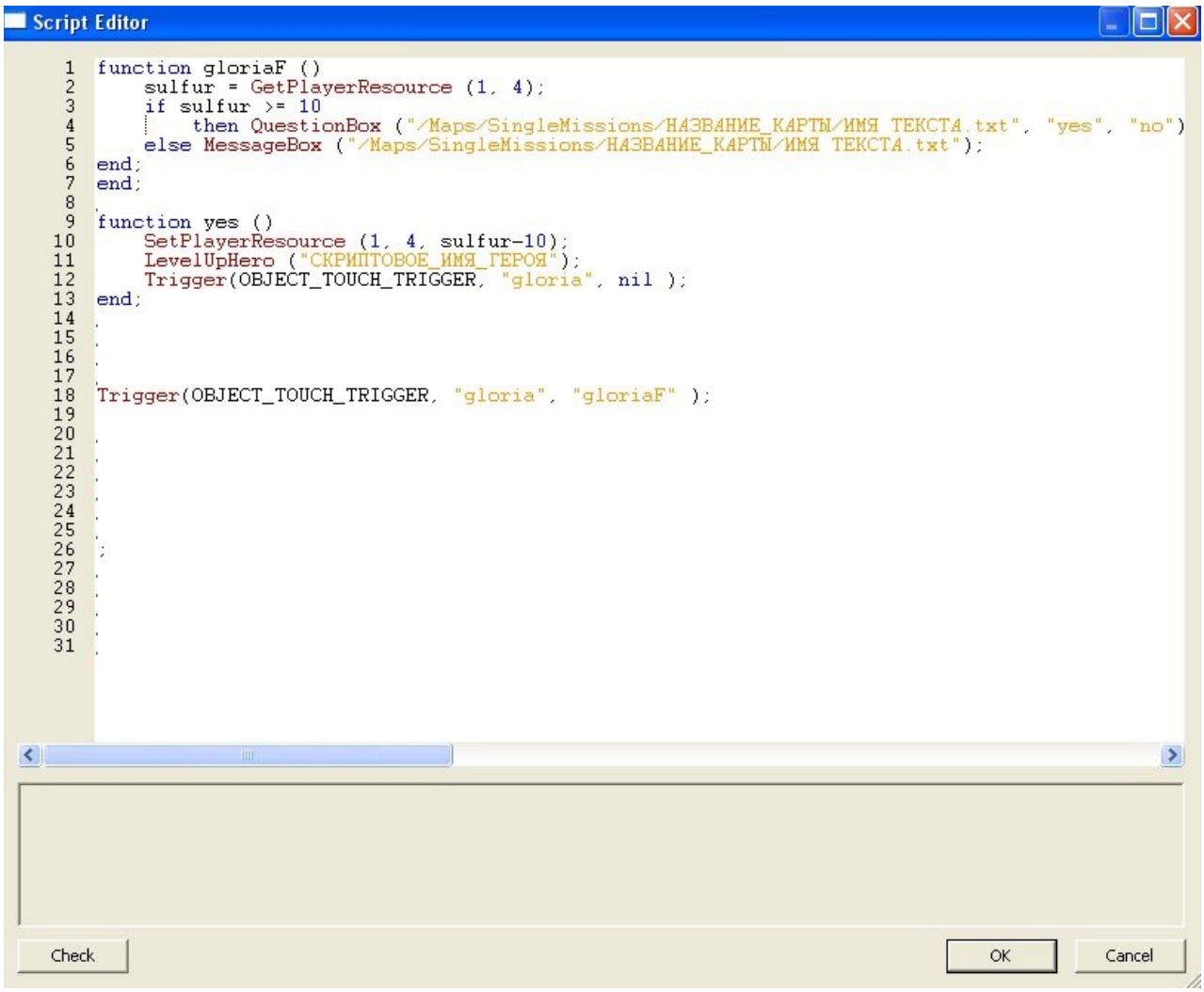
Значит теперь в нашем случае sulfur равняется значению количества определённого ресурса (серы) на момент срабатывания функции (**GetPlayerResource ()** как раз призвана определять значение – читайте рекомендуемые мной руководства).

Далее, строчкой **if** sulfur >= 10 мы определяем, есть ли у игрока сульфур в достаточном количестве и только тогда (чтобы не вышло конфуза) отнимаем нужное количество, вписывая, что УстановитьИгрокуКоличествоРесурса (какому игроку, какой ресурс, сколько (добытое число сульфура минус десять).

То же самое можно делать с любым другим ресурсом. Числовые ID ресурсов есть в руководствах.

Так же важно отметить такой момент: если вы запишете, что local sulfur = GetPlayerResource (1, 4); непосредственно в функции, то уровняловка эта будет справедлива только для данной функции. Без приписки local , после срабатывания функции должно записаться значение sulfur для всех скриптов. Если же вы впишете её отдельно от всего, то с самого начала игры определится количество серы у игрока (на сложности «Ветеран» - 10 мер) и уже всю игру sulfur будет равен десяти (естественно, если вы не перебьете это значение записью sulfur = GetPlayerResource (1, 4); в самой функции).

Приведённый выше пример имеет явный игровой недостаток. По геймплею он скучный и непонятный. По скрипту: мы заходим в хижину, у нас незаметно проверяют значение серы, отбирают её, если находят, и повышают уровень. Конечно, там есть несколько сообщений, но этого мало.



В этом примере всё гораздо интереснее. Теперь после успешной проверки на количество, задаётся вопрос, а не сразу же повышается уровень. Мол, уважаемый, согласны ли вы на такую сделку? Если вы нажимаете ОК, срабатывает функция “yes” (в которой мы старательно записали вычитание из казны 10-ти мер серы, лэвэлап героя и обязательное обнуление триггера). Вместо “yes” можно писать, что угодно. Первое значение отвечает за согласие, второе – за отмену. *(Идентичным способом можно вызывать функцию используя* ***MessageBox ();*** *, только вот вариант ответа будет, само собой, один. Например:* **MessageBox (Maps/SingleMissions/ НАЗВАНИЕ_КАРТЫ / НАЗВАНИЕ_ТЕКСТА.txt", 'yes');** *)*
Если же у нас нет столько серы (10 мер), то вопрос не выскочит, сработает **else** и выскочит **MessageBox** с сожалениями о вашей серной несостоятельности.
Для того, чтобы установить отличное от стандартных, количество ресурсов в начале игры, используйте **SetPlayerStartResources ();**

В этой части мануала, главной стала возможность вести манипуляции с ресурсами при помощи скриптов.

ПРИМЕЧАНИЕ: В каждой команде, в скобках, как вы успели заметить, прописываются определённые значения. Например, GetPlayerResource (1, 4) - здесь первая единичка означает, что кол-во ресурсов мы узнаём у первого игрока, а четвёрка, обозначает то, о каком ресурсе мы узнаём. То есть используются прописанные ID. Однако, всё это надёжней прописать, как GetPlayerResource (PLAYER_1, SULFUR), то есть использовать текстовые идентификаторы (а вдруг ID, которые отвечают этим идентификаторам, поменяются, к примеру, в патче?!). Я вот всё равно использую ID – мне так удобней, а потому и объясняю в мануале соответственно.

4. Вопрос – ответ

Как мне контролировать действия АИ?

Всмотритесь внимательней в команды **SetAIPlayerAttractor**(objectName, playerID, priority); и **SetRegionBlocked**(regionName, status, playerID = -1); К примеру, если вы хотите, чтобы компьютерный игрок стремглав мчался к какому-то объекту на карте и только по определённому пути, заблокируйте ему другие и повесьте наибольший приоритет на тот объект. Однако не забудьте снять приоритет, как только игрок до него дотронется, иначе АИ до конца своих дней будет в него самым непристойным образом тыкаться.

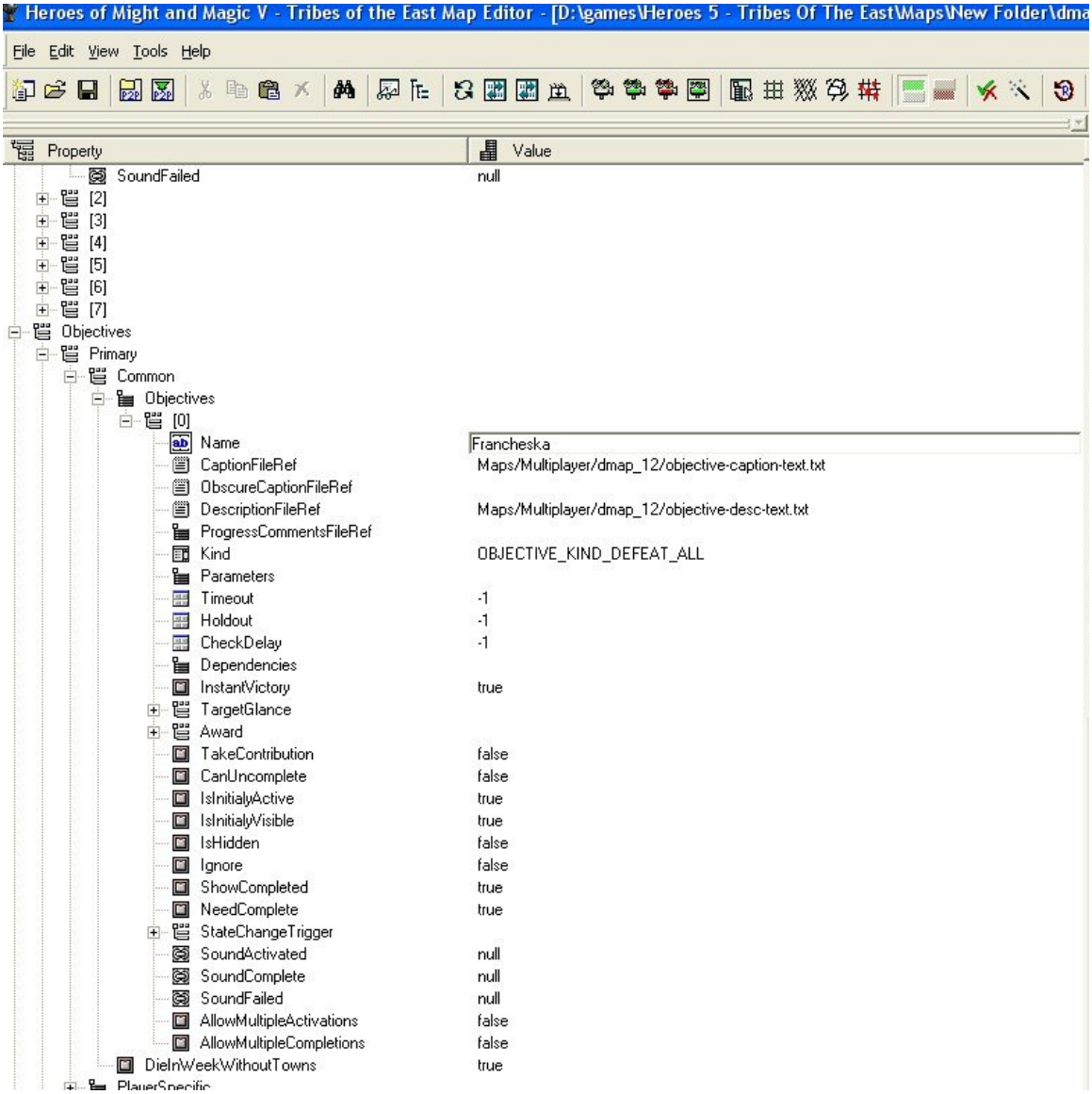
Как скриптами узнать сложность, которую выбрал игрок?

Для этого есть команда **GetDifficulty(),** которая выдаёт цифровые значения (Рекрут = 0… … Герой = 3).

Как работать с выдаваемыми заданиями?

GetObjectiveState(sObjectiveName, nPlayerID = PLAYER_1), **SetObjectiveState**(sObjectiveName, nState, nPlayerID = PLAYER_1), **GetObjectiveProgress**(sObjectiveName, nPlayerID = PLAYER_1), **SetObjectiveProgress**(sObjectiveName, nProgress, nPlayerID = PLAYER_1), **IsObjectiveVisible**(sObjectiveName, nPlayerID = PLAYER_1), **SetObjectiveVisible**(sObjectiveName, bVisible, nPlayerID = PLAYER_1).

Все они описаны в руководствах. Всюду фигурирует **sObjectiveName –** его мы записываем там (назван «Francheska»):



Как сделать так, чтобы только если определённый герой зайдёт в регион, выдавалось сообщение и начиналась битва?

Зная детали, о которых говорилось выше, можно придумать следующее:

Регион назовём по старинке – kamilla:

```
function kamillaF (heroname)
        if heroname == “Gottai”
        then MessageBox ("/Maps/SingleMissions/ НАЗВАНИЕ_КАРТЫ /
НАЗВАНИЕ_ТЕКСТА.txt");
        sleep (5);
        StartCombat (“Gottai”, nil, 5, 74, 6, 74, 4, 81, 4, 74, 4, 74, 6, nil);
        Trigger(REGION_ENTER_AND_STOP_TRIGGER, "kamilla", nil );
        else sleep (1);
end;
end;


Trigger(REGION_ENTER_AND_STOP_TRIGGER, "kamilla", 'kamillaF' );
```

Ранее уже упоминалось, что в скриптах, напрямую использующих имя героя, нужно писать heroname, однако не вспоминалась возможность данной проверки **if** heroname == “Gottai” **then.**

! Полезный момент: во всех примерах скриптов, приведённых в этом мануале, путь к текстовым файлам записывался примерно так: **"/Maps/SingleMissions/777/TEXT13.txt"**, однако это дело можно упростить благодаря команде **GetMapDataPath().** Она полностью заменяет длинное и порой неудобное **Maps/SingleMissions/777/...** В данном примере одиночная миссия называется многозначительно – «777» и пути к ней так и прописаны, однако если вы пожелаете сохранить карту под другим названием, они (пути) работать прекратят и скрипт выдаст ошибку (к примеру, если вы сохраните эту карту под именем «888», то тексты уже будут лежать в папке Maps/SingleMissions/888/… и игра не сможет их достать, используя старую строку). С **GetMapDataPath()** этого не произойдёт – с ней команда **MessageBox** () будет выглядеть так:

MessageBox (GetMapDataPath().."TEXT07.txt"**);**

Можно ещё больше упростить себе задачу, сравнив для игры команду **GetMapDataPath()** с чем-то совсем коротким, например: **dir = GetMapDataPath()**
И тогда достаточно записать: **MessageBox (dir..**"TEXT07.txt"**);**

Послесловие

1. Руководство будет дополнятся в соответствии с вашими пожеланиями, замечаниями, обидами и жалобами (ogo-i-o@ya.ru, ogo-i@ukr.net, http://ogo-maps.org.ru/).
2. Настоятельно рекомендую проверять все скрипты программой HoMM5MapScriptsEditor.[RC8] (автор: Sergei A. Klochkov (aka HSerg)). Если она не покажет ошибки, это не является гарантией того, что их вообще нет, однако большинство недочётов она вылавливает.
3. Внимательно читайте руководство по скриптам Novik 65, официальное от Нивала и заглядывайте на форум Гуголка в тему «Редактор карт – Обсуждение».
4. В выведенных мною скриптах я допускаю неточности. Я не профи – ошибаюсь часто. Однако цель у этого руководства – не предоставить вам готовые шаблоны, но погрузить в процесс, облегчить адаптацию, поделится знаниями, объяснить механику и принципы действий.
5. Спасибо за внимание.

Версия руководства 1.04