File Version: 4.14

In Red there are the most recent changes.
Some important pieces would be always in Orange.
In Green there are changes in the version before the previous one.

Table of contents

Golden Rules

1. Read this document carefully and be prepared to read its updates from time to time.
2. It is very low possibility that the Editor will corrupt the Game Data Files. If at some moment you have the Game crashing using modified Data Files, than you have done something wrong. Recall what you have changed recently and try undoing these operations until the Game stops crashing.
3. Regardless of the Golden Rule 2 backup up your files often.
4. Be very cautious when deleting an Entity that was not created by you in the first place. Why? Because the use of it might be hard-coded in the Game and it will crash or exit with an error.
5. If you want to do something (you have an idea) and don't know how to do it, think if something similar is already made in the original game or in one of the "mods". If yes, go and check how it is done by studying the Entities and parameters using the editor. This is the BEST way to learn about the editor's capabilities.
6. If you encountered a bug or noticed a malfunction, please report it to me. Thanks.
7. All list controls in the Editor support "Copy to clipboard" command 'Ctrl + C'. Only the selected entries will be put to the clipboard. To copy all entries from the list control that has no multi-selection – unselect the currently selected entry (by clicking on the area of the list control that does not contain entries). When no entries selected the 'Ctrl + C' command copies the entire content to the clipboard.
8. Some message boxes that pop up here and there might contain useful information. To copy it to clipboard simply hit 'Ctrl + C' while the message box is open.

Known Game Bugs

1. The last Particle System in a Location is not shown in Game. The Editor automatically creates a Dummy Particle System for you to work-around the problem.
2. The last Halo in a Location is not shown in Game. The Editor automatically creates a Dummy Halo to work-around the problem.
3. The last Dynamic Object in a Location is not shown in Game if there are no Halos in the Location. The Editor automatically creates a Dummy Dynamic Object or Halo to work-around the problem.
4. If a Location has no Path Conditions (which are formed by (or created for) Dynamic Objects) and at least one Dynamic Object is Collidable (Blocks Passage) then the Game might crash while loading the Location. Remove the 'Collidable' flag from the all collidable Dynamic Objects or create Path Condition for at least one collidable Dynamic Object to work-around the problem. During most of the 'Build' command a check is performed so you can see if the Location has this problem.
5. If an NPC uses the last Loot (the Loot with the highest ID) then this Loot won't be dropped when NPC is killed. Make sure that no NPCs use the last Loot (create a Dummy Loot if necessary).

Advanced Capabilities (via CF Agent)

From version 3.60 *Cosmic Forge* is shipped with the pair of binaries which help to bend the hardcoded functionality of the Game. They are CFInjector.exe and CFAgent.dll

In the Editor you use Advanced Capabilities as any other Editor's features – there is no difference and you don't have to do anything special. All Advanced Capabilities (mainly new NPC Script Commands and new Trigger Types) are marked with ** (double asterisk).

To be able to enjoy the Advanced Capabilities in the Game you need to launch the Game with the help of **CFInjector.exe**. To assist you in that there is a batch file **W8LaunchWithAgent.bat** which you should execute in order to launch the Game with the help of **CFInjector.exe**.

You need to edit this batch file in order to make it work. In this batch file you need to specify the path to your W8 exe file two times ("C:\Wizardries\W8\Wiz8_v124_nocd.exe" "C:\Wizardries\W8\Wiz8_v124_nocd.exe" is there by default, just replaces it by your full path to the W8 executable).

Below is the list of Advanced Capabilities that become available when you use CFInjector with CFAgent. It is going to grow. Let me know if you have any cool ideas of what can be done.

New Trigger Types

Check out the Triggers (Planar & Spherical) description here – it might help.

1. **Monster Spawn** – can be used to spawn new Monsters when Trigger is activated. Good for traps and ambushes as well as for some more complicated plot-based events. Parameters:
   a. **Monster** – identifies which Monster(s) to spawn.
   b. **Amount** – how many.
   c. **Disposition** – Monsters' disposition towards Party.
   d. **Condition** – A **Named Position Point** with the same name as the Trigger's should exist in the Location – it would determine the spot of spawning.
2. **Monster Birth/Disposition** – can be used for spectacular Monster appearance (birth - remember El-Dorado?) and/or changing Monster disposition on the fly (remember Arnika Bank Guard?). Parameters:
   a. **Monster** – identifies which Monster(s) to affect.
   b. **Give Birth** – plays 'Birth' animation first.
   c. **Disposition** – changes disposition towards Party.
   d. **Condition** – Monster or Monsters should already exist in the Location. The Trigger will affect all existing Monsters with the specified ID.
   e. **Important Conditions**
      i. Location Monster Group entry should have '*Unborn*' flag checked.
      ii. Location Monster Entry should have '*Unborn*' flag checked.
      iii. Monster itself should have '*Unborn (should be born before acting)*' flag checked.
      iv. Monster Model Script should have '*BIRTH*' entry describing what animation and what sounds to play as the birth cycle.
3. **Effect Spell** – casts the specified spell on the Party when Trigger is activated. Good for traps and some more complicated plot-based events. Without the CFInjector and CFAgent it will act as **Effect Hex**.
   a. **Usage Count** – how many times the Trigger can be used.
   b. **Cool Down** –when the Trigger replenishes the charges.
   c. **Power** – how powerful the spell is.
   d. **Spell** – specifies which spell to cast.
4. **Missile Launcher** – shoots a specified Missile from the NPP with the name as the Trigger's name in the NPP's direction. Plays a specified sound at the NPP's position. Note that some Missile might not work (such as 'Rocket'). Missiles can damage Party and Monsters. Note that since 'To Hit' and 'To Penetrate' aren't adjustable it is possible that high level characters won't be hit by the Missiles at all.
   a. **Damage** – how much damage the Missile will deal.
   b. **Missile** – Missile to launch. The Missile's 3D Model and Negative Effects are used.
   c. **Distance** – specifies the maximum distance the Missile will fly before it disappears.

   d. **Sound** – Sound to play. Note that the sound parameter uses the same slot as the 'Open/Activation Sound', so better don't use 'Open/Activation Sound' for Missile Triggers.
   e. **Condition** – A **Named Position Point** with the same name as the Trigger's should exist in the Location – it would determine the launch position and direction.
5. **Item Spawn** – can be used to spawn a new Item when Trigger is activated. Parameters:
   a. **Item** – identifies which Item to spawn.
   b. **The Item Rotates** – indicates if the Item should be rotating.
   c. **Chance** – specifies the probability of the Item to be spawned.
   d. **Condition** – A **Named Position Point** with the same name as the Trigger's should exist in the Location – it would determine the spot of spawning.
6. **Item Transform** – can be used to give a certain Item or a random Item from certain Loot in exchange for the specified Item. If you specify "Open/Activation Sound" it will be played if transformation succeeded. If Loot option is chosen then a random Item is selected from the Loot based on Items chance weights. Since 'fixed' Items have chance weight 0 they will never appear in this Trigger.
   a. **Item** – identifies which Item to accept.
   b. **Transform to…** – indicates if the Item should be replaced by another specific Item or a random Item from the specific Loot.
   c. **Item/Loot** – specifies what to give back.

New NPC Script Commands

1. **Modify Location Variable** – can be used to control Location Variables from within the NPC Script. You can modify the value of one Location Variable in the command using several methods:
   a. Set value – sets value directly.
   b. Increase by value – increases the Variable's current value by the specified value.
   c. Decrease by value – decreases the Variable's current value by the specified value.
   d. Set bit – set the specified bit in the Variable's current value not affecting other bits.
   e. Clear bit – clears the specified bit in the Variable's current value not affecting other bits.
   f. Bit-wise AND with value – sets all bits that are 1 both in the Variable's current value and in the specified value, other bits are set to 0 (cleared).
   g. Bit-wise OR with value – sets all bits that are 1 either in the Variable's current value or in the specified value, other bits are set to 0 (cleared).
2. **Modify Fact** – is an original command but since 3.91 it is extended and now considered to be Advanced. You can modify the value of one Fact in the command using several methods:
   a. Set value – sets value directly.
   b. Increase by value – increases the Fact's current value by the specified value.
   c. Decrease by value – decreases the Fact current value by the specified value.
   d. Set bit – set the specified bit in the Fact current value not affecting other bits.
   e. Clear bit – clears the specified bit in the Fact current value not affecting other bits.
   f. Bit-wise AND with value – sets all bits that are 1 both in the Fact current value and in the specified value, other bits are set to 0 (cleared).
   g. Bit-wise OR with value – sets all bits that are 1 either in the Fact current value or in the specified value, other bits are set to 0 (cleared).
3. **Activate Location Trigger** – can be used to activate the specified Trigger from within the NPC Script. For instance, an NPC can summon reinforcements when it gets angry… Or open a certain door when given money or an Item.
4. **Modify Attribute** – allows modification of the specified Attribute by specified amount for all Party members. The valid range is [-25, 25]. Limits are [5, 99].
5. **Modify Skill** – allows modification of the specified Skill by specified amount for all Party members. The valid range is [-25, 25]. Limits are [1, 99]. Parameter "Only modify if Skill is Activated" controls which characters will be affected.
6. **Assign Behaviour Script to a Monster** – allows assigning the specified Behaviour Script to the specified Monster (not necessary the NPC you are interacting with at the moment). Allows achieving some special results: Specific Monster animation, Monster disappearance, move Monster to certain NPP, changing Disposition, and so on! See the description of available commands here. You can create as many new MFS files as you wish.
7. **Fact Conditional Jump** – tests one or more Facts on certain conditions and then jumps to the specified Script Block if the conditions are satisfied. Available conditions are:

a. Is equal to value – satisfied when Variable's value equals to the specified value.
b. Is not equal to value – satisfied when Variable's value does not equal to the specified value.
c. Is greater than value – satisfied when Variable's value is greater than the specified value.
d. Is less than value – satisfied when Variable's value is less than the specified value.
e. Has bit set – satisfied when Variable's value has the specified bit set.
f. Has bit cleared – satisfied when Variable's value has the specified bit cleared.
g. Has all bits set – satisfied when Variable's value has all the bits set which are set in the specified value.
h. Has at least one bit set – satisfied when Variable's value has at least one bit set of those that are set in the specified value.

8. **Location Variable Conditional Jump** – just like *Fact Conditional Jump* command tests one or more Facts on certain conditions and then jumps to the specified Script Block, this command tests one Location Variable on a specified condition and jumps to the specified Script Block if the condition is satisfied. Available conditions are:
    a. Is equal to value – satisfied when Variable's value equals to the specified value.
    b. Is not equal to value – satisfied when Variable's value does not equal to the specified value.
    c. Is greater than value – satisfied when Variable's value is greater than the specified value.
    d. Is less than value – satisfied when Variable's value is less than the specified value.
    e. Has bit set – satisfied when Variable's value has the specified bit set.
    f. Has bit cleared – satisfied when Variable's value has the specified bit cleared.
    g. Has all bits set – satisfied when Variable's value has all the bits set which are set in the specified value.
    h. Has at least one bit set – satisfied when Variable's value has at least one bit set of those that are set in the specified value.

Other Changes

1. **Sky for a Location** is now being always reloaded – does not matter if the previous Location had the same Sky Name.
2. **Unique Names for new Locations** – it is now possible to give unique names to new Locations (which were unused in the original game). Use 'rename' button in the Location Dialog or menu ***Edit→Rename Location*** for that.
3. **Ask before teleporting fix** – now Teleporter Triggers ask permission even if they lead to the same Location.

Game Information

Game Information encapsulates characteristics of the Races, Classes and the Skills.

The ***Data*** folder in *Cosmic Forge* directory contains a list of *.cfdat files which the Editor uses as the base source of Game Information. After some bits of information are changed the changes are being saved in the ***Data*** folder in W8 directory. Since that moment *Cosmic Forge* will read Game Information from the ***Data*** folder in W8 directory.

When the Game is launched using injection, CFAgent will read those *.cfdat files and update the Game Information in-game.

Note that some changes will never work. For instance ability ***Unusual Starting Equipment*** will only work for ***Faerie*** Race or ability ***5% Bonus to All Resistances*** will only work for ***Mage*** Profession. This is because they are hard-coded to use not the ability flag, but the specific Race or Profession. Some other parameters might not work as well.

Advanced Capabilities (Lua Scripting)

Since version 3.90 CF Agent supports Lua Scripting. Scripting allows more flexible editing of the special events than the CF Editor allows and much more.

Scripting does not change game data files like the Editor does, instead it interferes with the Game on the fly – during the Game being played.

Setting Up Lua Scripting

The actions below should be done once only. You don't have to do it anymore when downloading new versions of *Cosmic Forge*.

1. **Download and unpack *Cosmic Forge*** files to some folder on your computer. Make sure this folder is not W8 installation folder!
2. **cfagent.lua** file should be present in the same folder with CFAgent.dll. If it is not there, then duplicate **cfagent_clean.lua** and rename the new copy into **cfagent.lua**. When updating *Cosmic Forge* with the new version your **cfagent.lua** will be left intact while **cfagent_clean.lua** and example files in the **LuaExamples** folder might change.
3. **Lua for Windows** should be downloaded (from Google Code) and installed in your system.
4. After installing Lua for Windows you need to **setup the code auto completion** for the functions you can use in scripting. To assist with this the file **cfa.api** is included. To setup the code auto completion do the following:
    i. Open **cfagent.lua** file for editing by right-clicking it and selecting ***Edit Script*** menu command. **SciTE** will open – the editor you'll be using to edit lua files.
    ii. Go to ***Options→Open lua.properties*** menu command and the **lua.properties** file will be opened for editing.
    iii. Locate the line with the **api.$(file.patterns.lua)** string – it should be not far from the beginning of the file.
    iv. Append the following to the end of this line: **;<path_to_cfa_dot_api_file>** where **<path_to_cfa_dot_api_file>** is the full path to the **cfa.api** file which should be near CFAgent.dll. For instance if your CFAgent.dll is in **C:\CosmicForge** folder then **cfa.api** should be there as well and the text to append will be **;C:\CosmicForge\cfa.api**. Note that semicolon (symbol **;**) should be there to separate the appended text from the text that was there before!
    v. Go to ***File→Save*** menu command to save the changes and then quit the SciTE program.

Done! You've set up everything necessary.

When you download a new version of *Cosmic Forge* you simply unpack it over your current version and that is it.

Editing Lua Scripts

1. To **edit a lua script** you right-click it and select ***Edit Script*** command. SciTE will open – the editor you'll be using to edit lua files.
2. When opening a lua file, SciTE is using settings which enable the **syntax highlight** greatly improving understanding the the code.
3. In order to catch silly mistakes before you try the script in W8 game you can **compile script** in SciTE by using menu ***Tools→Compile (Ctrl+F7)***. If it highlights anything then there is an error and you need to fix it.
4. You can use the **code auto completion** feature to assist you in scripting. All functions you can use to control W8 game flow start with ***cfa.*** There are two main subgroups of functions ***cfa.w8*** and ***cfa.util***. In general, to use the auto completion you do the following:
    a. **Type in** any amount of starting symbols (f.e. "cf" or "cfa." or "cfa.w" or "cfa.color.").

b. **Hit *Ctrl+Space*** to drop down the small window with the list of available matching functions. It is resizable so you can resize it if there are a lot of functions.

c. **Select the function** you want and its name will be auto completed for you.

d. **Type in opening bracket** "**(**" and a list of parameters will pop up highlighting the current parameter as you continue typing. Also the description of the function will be displayed.

How Lua Scripting works with CF Agent and the Game

Note: files in the **LuaExamples** folder contain examples of most (if not all) of the functions being used and should be a great source of answers on 'how to' questions.

In short, there are certain events in the Game that CF Agent can intercept and do something about it.
If Lua scripting is supported (you've set it up properly) then CF Agent calls predefined functions in cfagent.lua. These predefined functions are called **handlers** (because they handle certain events).
So far there are several **events** for each of which there is a handler available:

- A Location Trigger is activated.
- An NPC Script Command is executed.
- A Fact state is modified.
- An Item has been acquired (when you buy/receive an Item from an NPC or pick it up at a Location and give it to a Party member or drop it into your Inventory).
- An Item is being used.
- An Item is being picked up at a Location (note that it just the event of picking an Item up and is different from the acquiring Item event).
- A Combat has started.
- A Monster Generator is activated (has just spawned a bunch of Monsters).
- A Monster is killed (by anyone).
- A Location is loaded.

So whenever one of the aforementioned events happens, CF Agent intercepts it and calls the handler associated with the event.
That is when you can do something! You can do it by calling one or more of the functions available to you.

One example is better than a 1000 of words. So here we go:
Let's place an Item with ID 200 in the narrow corridor somewhere.
Let's place a Missile Launcher Trigger with the name "deadly_missile" in the corridor, so the missile will fly along the corridor and hit anyone near the Item.
Let's place a tiny one-shot button with the Generic Trigger "deadly_missile_disarm" somewhere on the wall.
Let's create a Fact "FACT_DEADLY_MISSILE_TRAP_DISARMED" with ID 950 (for instance).
Now we want the Player to be hit by a missile when he picks up the item, but if he instead pushes the button then the trap will be disarmed and he can picks up the item unharmed.
You might be able to do it solely in the Editor (I'm not sure), but let's do it in the scripting:

1. During the gameplay Party **picks up an Item with Id=200** at some Location. CFAgent intercepts this event.
2. CFAgent calls Lua handler ***function onItemPickedUp(itemID)*** in the cfagent.lua supplying 200 in the itemID parameter.
   Suppose the code of the handler is the following:

```
function onItemPickedUp(itemID)
    if itemID == 200 then
        cfa.w8.locationActivateTrigger("DEADLY_MISSILE")
    end
end
```

3. In Lua the check "itemID equals to 200" succeeds and the call ***cfa.w8.locationActivateTrigger("DEADLY_MISSILE")*** is made which reaches CFAgent.
4. CFAgent calls Game's function(s) to activate the specified trigger. Trigger is being activated. CFAgent intercepts trigger activation event.

5. CFAgent calls Lua handler ***function onLocationTriggerActivated(trigName, trigType)*** in the cfagent.lua supplying "DEADLY_MISSILE" in the trigName parameter.
   Suppose the code of the handler is the following:

```
function onLocationTriggerActivated(trigName, trigType)
    if trigName == "DEADLY_MISSILE" then
        if cfa.w8.factGetState(950) == 1 then
            return false
        else
            return true
        end
    end
end
```

6. In Lua the check *trigName == "DEADLY_MISSILE"* succeeds and then depending on the state of the Fact 950 the function returns either true or false. If the false is returned then the Trigger activation is stopped and the trigger does not get activated. If the true is returned then the Trigger gets activated as usual.
7. The return value (true or false) reaches the CFAgent.
8. CFAgent either allows the Trigger to be activated or not.
9. Now to process the button which can disarm the trap we should add the following code to the handler ***function onLocationTriggerActivated(trigName, trigType)***:

```
function onLocationTriggerActivated(trigName, trigType)
    if trigName == "DEADLY_MISSILE_DISARM" then
        cfa.w8.factSetState(950, 1)
        return true
    end
end
```

10. So when the button is pressed CFAgent intercepts trigger activation event.
11. CFAgent calls Lua handler ***function onLocationTriggerActivated(trigName, trigType)*** in the cfagent.lua supplying "DEADLY_MISSILE_DISARM" in the trigName parameter.
12. In Lua the check *trigName == "DEADLY_MISSILE_DISARM"* succeeds and then the state of the Fact 950 is set to 1.
13. The final handler ***function onLocationTriggerActivated(trigName, trigType)*** now has the following code:

```
function onLocationTriggerActivated(trigName, trigType)
    if trigName == "DEADLY_MISSILE" then
        if cfa.w8.factGetState(950) == 1 then
            return false
        else
            return true
        end
    end

    if trigName == "DEADLY_MISSILE_DISARM" then
        cfa.w8.factSetState(950, 1)
        return true
    end
end
```

Examples of Lua Scripting

Check out the plethora of examples in the several lua files in the **LuaExamples** folder. They contain examples of most (if not all) of the functions being used and should be a great source of answers on 'how to' questions.

Handlers' Description

cfa.result.* onLocationTriggerActivated (trigName, trigType)

Called whenever a Trigger is activated in the game (before it is actually activated).

trigName - contains the capitalized (upper-cased) name of the Trigger being activated.

trigType - numerical type of the Trigger being activated (probably not useful).

cfa.result.ok or ***cfa.result.cancel*** can be returned from this handler allowing or disallowing the Trigger activation - this allows complex checks to be made before deciding to activate the Trigger.

int onNPCScriptCommandExecuted (cmdType, NPCID, scriptBlock, commandIndex, scriptBlockPrev, scriptFlag1, scriptFlag2)

Called whenever a Command in an NPC Script is executed in the game.

cmdType - string explaining what type the command is.

NPCID - the ID of the NPC which script is being executed.

scriptBlock - index of the Script Block where the Command is.

commandIndex - index of the Command in the Script Block.

scriptBlockPrev - previous Script Block (in case a jump was made).

scriptFlag1 - flag 1.

scriptFlag2 - flag 2.

cfa.result.ok, ***cfa.result.cancel*** or ***Script Block Index (non-negative number)*** can be returned from this handler indicating if to perform Unconditional Jump to another Script Block. If ***cfa.result.ok*** is returned (should be by default) then the command execution proceeds as usual. If ***cfa.result.cancel*** is returned then the command is not executed. If a non-negative number is returned (0 and greater) then instead of executing the command, Unconditional Jump will be performed to the Script Block specified in the returned number. See lua examples.

onFactStateModified (factID, newState)

Called whenever a Fact's state is modified in the game.

factID - ID of the Fact being modified.

newState - number reflecting the new Fact's state.

onItemAcquired (itemID)

Called whenever an item is added to the Party's inventory or acquired by a Character.

itemID - ID of the Item added.

cfa.result.* onItemUsed (itemID, characterIndex, itemSlotGroup, itemSlotIndex)

Called whenever an item is used or attempted to be used by the Party.

itemID - ID of the Item added.

characterIndex - character who used the Item.

itemSlotGroup - specifies where the Item is (backpack, equipped, party or cursor)

itemSlotIndex - specifies the slot index where the Item is.

cfa.result.ok or ***cfa.result.cancel*** can be returned from this handler allowing or disallowing the Game using the Item in question. Useful to return ***cfa.result.cancel*** when you either managing the item yourself in the script (such as remove it from the inventory upon usage and when the Item does not have any other usage other than the one coded in the script) or you want to block the Item usage unless some condition is satisfied (like some item can be used only at night).

onItemPickedUp (itemID)

Called whenever an item is picked up at a Location.

itemID - ID of the Item being picked up.

onCombatStarted ()

Called whenever a Combat begins.

onMonGenActivated (monGenName)

Called whenever a Monster Generator spawns a group of Monsters.

onMonsterKilled (monsterID)

Called whenever a Monster is killed in the Game.

onLocationLoaded ()

Called whenever a Location is loaded (loading the saved game, through the teleporter, etc.).

Multiple Lua Files

Since version 3.92 it is possible to have any amount of lua files all used by the main cfagent.lua file. This can be a necessary way to split code into compact groups when the code grows too big to be viewed and understood comfortably in a single lua file.

Function **cfa.util.includeLuaFile(luaFile)** should be used in the beginning of the cfagent.lua file for all additional lua files you split your code into.

Here is the example where we split trigger related code into different files based on the current Location:

Fragment of the code in cfagent.lua:

```
cfa.util.includeLuaFile("triggerActivatedLowerMonastery.lua")
cfa.util.includeLuaFile("triggerActivatedArnika.lua")

function onLocationTriggerActivated(trigName, trigType)
    if cfa.w8.locationGetCurrentID() == 8 then
        return OnTriggerActivatedInLowerMonastery(trigName)
    elseif cfa.w8.locationGetCurrentID() == 1 then
        return OnTriggerActivatedInArnika(trigName)
    end

    return true
end
```

Fragment of the code in triggerActivatedLowerMonastery.lua:

```
function OnTriggerActivatedInLowerMonastery(trigName)
    if trigName == "DOOR01" then
        -- do something here
    elseif trigName == "DOOR02" then
        -- do something here
    elseif trigName == "DOOR03" then
        -- more code
    end
end
```

Fragment of the code in triggerActivatedArnika.lua:

```
function OnTriggerActivatedInArnika(trigName)
    if trigName == "DOOR_LM01" then
        -- do something here
    elseif trigName == "DOOR_LM02" then
        -- do something here
    elseif trigName == "DOOR_LM03" then
        -- more code
    end
end
```

Lua Namespaces

Apart from the handlers which are the functions that are called by the CFAgents and should not be called by you, the rest of the functions and constants are used by .lua scripts and can be called by you.

All functions and constants are somewhat grouped into **namespaces**.

The top level namespace contains all subordinate namespaces and is named **cfa**. That is, anything you call or use (except for raw numbers and strings) starts with **cfa** (i.e. ***cfa.w8.factGetState(10)***).

Under **cfa** there are few more grouping namespaces:

- **util** – contains functions not directly related to W8 and helper functions (i.e. ***cfa.util.setBit(100, 3)***)
- **w8** – contains functions directly related to W8 (i.e. ***cfa.w8.gameGetDay()***). In turn the functions under the cfa.w8. namespace are divided into groups using prefixes:
    - **location** – functions executing something in the current Location (i.e**.** ***cfa.w8.locationShowMessage(30)***).
    - **fact** – functions dealing with Facts (i.e. ***cfa.w8.factGetState(359)***).
    - **item** – functions dealing with Items (i.e. ***cfa.w8.itemGetName (202)***).
    - **npc** – functions dealing with NPCs (i.e. ***cfa.w8.npcAddItemToTradeInventory(20, 456, 1, 0)***).
    - **loot** – functions dealing with Loots (i.e. ***cfa.w8.lootGetRandomItem(13)***).
    - **party** – functions dealing with the Party (i.e. ***cfa.w8.partyAddItemToInventory(401, true)***).
    - **character** – functions dealing with a character in the Party (i.e. ***cfa.w8.characterModifyAttribute(3, cfa.attr.piety, 5)***).
    - **game** – functions dealing with some general game concepts (i.e. ***cfa.w8.gameGetDay()***).
- **color** – contains a list of color constants which can be used in some functions (i.e. ***cfa.w8.gamePrintInterfaceStringInGeneralTextArea(1457, cfa.color. pink)***).
- **attr** – contains a list of attribute constants which can be used in some functions (i.e. ***cfa.w8.characterModifyAttribute(4, cfa.attr.speed, -1)***).
- **skill** – contains a list of skill constants which can be used in some functions (i.e. ***cfa.w8.partyModifySkill(cfa.skill.scouting, 2, true)***).
- **operation** – contains a list of operation constants which can be used in some functions (i.e. ***cfa.w8.factModifyState(618, 0, cfa.operation.setBit)***).
- **condition** – contains a list of constants for negative conditions (effects) that a character could suffer (such as fear, hex, nausea, missing, etc.). The constants can be used in some functions (i.e. ***cfa.w8.characterAddCondition(7, cfa.condition.insanity, 20, 0, true)***).
- **itemslotgroup** – contains a list of constants describing the location of an item in Party possession (character's backpack, character's equipment, party inventory or is currently held by the cursor). The constants can be used in some functions (no examples yet).
- **gender** – contains a list of gender constants which can be used in some functions (i.e. ***cfa.w8.partyGetRandomCharacter(cfa.gender.any, -1, false, false)***).
- **itemprop** – contains a list of constants used to access Items' properties(i.e. ***cfa.w8.itemGetProperty(63, cfa.itemprop.name)***).
- **monsterprop** – contains a list of constants used to access Monsters' properties(i.e. ***cfa.w8.monsterGetProperty(270, cfa.monsterprop.name)***).
- **disposition** – contains a list of constants used to control Monsters' disposition (i.e. ***cfa.w8.locationChangeDispositionAndGiveBirthToMonsters(252, true, cfa.disposition.friendly)***).
- **result** – contains a set of used for return value from the handlers (***cfa.result.ok, cfa.result.cancel***).
- **charprop** – contains a list of constants used to access Characters' properties(i.e. ***cfa.w8.characterGetProperty(2, cfa.charprop.attribute_base, cfa.attr.strength)***).
- **spellsphere** – contains a list of constants used to specify a spell sphere (i.e. ***cfa.w8.characterGetProperty(2, cfa.charprop.spMax, cfa.spellsphere.fire)***).

Shipping a Mod with Lua scripts

If you have your Mod ready, then highly likely you are using injection to have access to advanced functionality. So you've got to supply CFInjector.exe, CFAgent.dll and some sort of .bat file to launch W8 with injection.

Just keep all .lua files along CFAgent.dll and it should work. CFAgent expects all .lua files to be in the same folder with itself.

Debugging Lua scripts

Sometimes or even quite often your scripts might not work. There are several things you can do about it.

1. Check ***CFAgent.log*** file after you quite the game for any errors. It might contain some error that will point you to incorrect syntax in the script or other mistakes.

2. Use ***cfa.util.showMessageInGame(...)*** function extensively to see in game if your script code is reached at all and if reached then what values the variables have. Check the supplied examples on how to use the function - there are plenty. The format can be learned in the internet: look for ***sprintf*** format.

3. If some complex logic does not work - split it into simple pieces and see if simple parts work. For instance you want to remove a certain item from a certain NPC's inventory when a certain Fact takes a certain value **and** the party has a certain item. And it does not work. Try removing the item only when the Fact takes its value (without checking the party's item) and see if it works. After checking it you will have more information and will narrow down the possible cause. Well, it is not the best example, but I hope you got the gist. Try simple stuff first!

Location Editor

Important Notes

1. Almost everything you change in a Location will **only** appear in the Game when you arrive to this Location for the **First Time**!
2. When you change something in a Location (say add an Item or a Monster) and load a saved game in which you have already visited that Location than you changes will unlikely be noticed by you.
3. Due to the Important Notes 1 and 2 you are facing a problem: how do I test my changes on various Locations? Do I have to play the half of the game each time? No, you don't. Just follow the following steps to test a Location:
   - Read the Making a Teleporter tutorial.
   - Using the acquired knowledge create a 'place of arrival' Teleporter in the Location you want to test (preferably close to the point where you'll be testing).
   - Create a Teleporter on the beach in Lower Monastery Location leading to the just created 'place of arrival'.
   - Launch the Game, start new Game and head strait to the Teleporter to be instantly transferred to your Location.
   - Perform any tests.
   - Later before shipping the mod do not forget to delete all temporary Teleporters and other helper Objects.
4. If you have accidentally deleted something from a Location that causes the Game to crash or exit with an error, **do not panic**.
   - Exit the Editor.
   - Copy all the files related to the Location to some safe place (back the files up).
   - Then delete those files from the Game installation directory (GID).
   - Run the Editor.
   - Now as the modified files are gone from the GID, the editor will pick up the original files where the deleted Entity still exists (and will exist forever).
   - Select any other Location and create a new Entity using the one you have deleted (it does exist in the original data files; it is deleted only in your data files that you have backed up).
   - Save your changes. Exit the Editor.
   - Copy Location's backed up files back into the GID.
   - Run the Editor. Select the Location.
   - Create a new Entity using the one you have created in another Location at step f.
   - Save your changes. Now the deleted Entity is back.
   - Delete the Entity from another Location if you don't need it.
5. If you are going to add new Textures it is advisable that they aren't too big in dimensions. Sticking with dimensions less or equal 256 is a good idea though you may use larger Textures. Also I've heard that it is best to have dimensions equal to a degree of 2 (2, 4, 8, 16, 32, 64, 128, 256, 512, etc.).

Terminology

Camera - The point in the 3D View from which the Location is shown.

Location - Independent set of **Terrain**, objects, monsters etc.

Level - Same as **Location**.

Terrain - Visual static part of the **Level** (walls, water, etc.).

Terrain Mesh - Piece of the **Terrain**.

Mesh - Same as **Terrain Mesh**.

Terrain Mesh's Face - Part of the **Terrain Mesh**. A triangle, actually.

Face - Same as **Terrain Mesh's Face**.

Vertex - One of the three vertices for a triangular **Face**.

Material - Descriptor of a substance (contains information on opacity, colouring, etc.)

Texture - An image, a pattern. This can see on the walls: the surface of water, etc. Do don't mix it with the **Material**!

Floating Point RGB Colour - Colour represented by three floating point numbers of red, green and blue. Each value is ranged from 0.0 to 1.0.

Texture Coordinates (TCs) - Two floating point numbers usually called a pair of **Texture Coordinates** (this is NOT a pair of two floating point numbers). Those numbers are 2D coordinates defined for some **Vertex**. They determine what **Texture**'s pixel will be displayed at the **Vertex**. Usually a triangular **Face** has a set of 3 such coordinates for each of its three **Vertices**, thus completely defining how the **Texture** will be mapped onto the triangular **Face**.

Please, read some articles on 3D in the Internet if you have no idea what all that means.

Location Editor Dialog's Layout

Menu

File

Apply Changes – becomes enabled when something has been changed. This command Saves all current changes to the selected Location data files.

Discard Changes – becomes enabled when something has been changed. Discards all current changes to the selected Location and reloads the Location data.

Edit

Rename Location – allows renaming Location. Note that Location names are stored separately from Locations – in the Strings Database file and thus require separate saving when changed.

Undo – undo the previous action.

Redo – redo the undone action.

Copy All <Entities> – copies all entities from the currently selected Tab into the global Location Buffer.

Copy Selected <Entities> – copies the selected entities from the currently selected Tab into the global Location Buffer.

Paste N <Entities> from <Location> – pastes entities from the global Location Buffer into the currently selected Location. **Note that for correct 'paste' operation all entities in the global Location Buffer should exist in the source Location and be saved into the Location files.**

3D View – checkable menu items to toggle the visibility of Location Entities in 3D View.

3D Options

Change View Distance – objects and/or their faces which are farther from the Camera than the View Distance are not shown in the 3D View thus improving editor's performance on slow PCs.

Go to Coordinates – simply moves Camera to specified position.

Enable Animation – toggles animation.

Sync 3D View with Selected Entity – toggles automatic focusing of the Entity selected on one of the Dialog Tabs.

Go to Selected Entity – Focus on the Selected Entity.

3D Gizmos

Move (Entity Local Orientation) – *Move Gizmo* is shown representing three Entity's Axes. Left clicking on an Axis and dragging mouse will move the Entity along that Axis.

Move (World Global Orientation) – *Move Gizmo* is shown representing three World's Axes. Left clicking on an Axis and dragging mouse will move the Entity along that Axis.

Rotate (Entity Local Orientation) – *Rotation Gizmo* is shown representing three Circles of Rotations around three Entity's Axes. Left clicking on a Circle and dragging around will rotate the Entity accordingly.

Rotate (World Global Orientation) – *Rotation Gizmo* is shown representing three Circles of Rotations around three World's Axes. Left clicking on a Circle and dragging around will rotate the Entity accordingly.

Scale (Entity Local Orientation) – *Scaling Gizmo* is shown representing three Axes of Scaling (Resizing) along three Entity's Axes. Left clicking on an Axis and dragging mouse will scale the Entity along that Axis. Also left clicking on a central white cube and dragging mouse will scale the Entity uniformly (i.e. along all three Axes simultaneously).

None – no 3D Gizmo is shown.

Build

Rebuild OCTree – This command deletes all OCTree data and generates it from scratch. You should be prepared that many Terrain Meshes will change their sequence. This command should be used as soon as you have some visibility problems in game or even when the game crashes or you fall through the floor, etc. Or just to make sure that everything is all right in the Location.

Rebuild OCTree (Terrain Only) – This command deletes OCTree data that relates to Terrain Meshes only and generates it from scratch. You should be prepared that many Terrain Meshes will change their sequence. This command should be used as soon as

you have some visibility problems in game or even when the game crashes, etc when you have not changed physical faces.

Generate Pathfinding Database... – Generates Pathfinding Database using all Walkable Physical Faces having 'Generate Pathfinding Database over the Face(s)' flag set. Use this command only when you have altered Physical Walls (as a part of Terrain modification) or altered Dynamic Objects that generate Path Conditions.
Now the algorithm supports Walkable objects (those that have 'Footsteps Sound' specified). For them it creates additional nodes which in the end will enable Monsters to walk where you can walk (like bridge in Lower Monastery).

Generate Automap Nodes… – This command generates Automap Nodes used for in-game Automap obviously. Use it when you have modified Physical Walls.

Relight Terrain... – Recalculates Lighting throughout the Location. Though the Lighting is updated whenever you modify/delete/add a Light Source it is wise to use this command occasionally to correct possible Lighting discrepancies. Also should be used when you have altered Terrain and want to see the correct Lighting from Light Sources.

Recalculate Sun Light over Terrain... – Recalculates Sun Light throughout the Location. This command should be used when you have altered Terrain or the Sun parameters and want to see the correct Sun Lighting.

Recalculate All Lighting… – This command recalculates all lighting in the Location.

Rebuild All… – This command rebuilds everything from scratch (OCTree, lighting, sunlight, Pathfinding Database, Automap Nodes). Use it at least once before releasing the Location to public.

Tools

Recombine Meshes (OCTree-wise)... – This command is designed to redistribute Terrain Faces into the Terrain Meshes in the best possible way. It is highly recommended to use it from time to time as you change your Terrain.

Recombine Meshes (Material-wise)... – This command redistributes Terrain Faces into completely different set of Meshes using Texture, Material and relative positions of the Faces. It creates more logical unions of Terrain Faces into the Terrain Meshes albeit at some cost. The amount of Meshes might rise dramatically, though this should not be much of a problem. Use this command when you want to fetch some Terrain structure (house, cliff, etc.) from a Location. After this command all you have to do is merge two or more Meshes into one and you have a one-piece Mesh representing something.

Find unused Texture Files... – This searches for the Texture files in the current Location "bitmaps" folder and checks if the Location data files use these Texture files. It lists all unused Texture files and later proposes to delete them.

Clear Location... – This command invokes a dialog which allows selecting types of Entities to delete from the current Location. You can delete everything (by default) or only certain things. This option can be used as 'Delete All' command which is not implemented anywhere (for instance when you want to delete all Terrain Meshes quickly). Undeletable things: single Terrain Mesh with the single Face, 'Sun' Light Source (if it existed) and few Materials used by the survived Terrain Mesh. It is recommended to use **Rebuild OCTree** command immediately after this command.

Recreate Location... – This command allows you to completely replace the current Location's content by the content of a selected Location. Can be used to move a Location to another slot, for instance when you realise that a Location cannot have a sky and the sun (because it occupies the 'sky-less' slot) while you really want it to have them.

Create Maze... – This command invokes Maze Creation Dialog which contains a lot of options to generate a random (though fully controllable by you) working Maze.

Export Terrain... – This command exports the whole Terrain into one big **obj** file which further can be modified in many 3D Editors. After that it can be imported back to W8 as the Terrain.

Note that Rebuild OCTree command now (from version 3.14) works completely different from what it was before (and I believe somewhat slower). Hopefully better.

Toolbar

The left toolbar duplicates the '3D View' menu. The right toolbar duplicates the '3D Options' menu.

Location Selector Controls

Edit control which contains the name of the currently selected Location and a button that allows selecting of another Location.

Entity Tabs

Each Tab contains information about part of the Location and usually allows alteration of Location Entities of certain type.

3D View

A control displaying the Location almost as it would be displayed in the Game. Also serves for navigating the Location, selecting Entities and their Faces (if applicable).

The Entities that have their own 3D models are shown using those models. Other Entities are shown using special 3D models and/or shape models. Entities having orientation may have an additional arrow model attached which indicates the orientation. The jut on one of the arrow's sides indicates 'UP' side.

Entities can be moved, rotated (orientated) or scaled (resized). Depending on the Entity type one or more Active Entity Control Groups can be shown thus allowing changing Entity's positioning, orientation or size.

Active Entity Controls

Those display the Active (Selected) Entity's name and provide controls (buttons and others) for positioning, rotating and scaling the Entity (if allowed).
To make a single action just click left mouse button on the control button once. To make repetitive action click and hold left mouse button on the control button until the Entity takes up the required position/shape.

Positioning group

This group contains six buttons for moving entity Forward/Backward, Right/Left and Up/Down. 'Step' parameter determines the distance of the single move action.
If 'Align Move with Axes' check-box is checked the movements will be performed along the Axes (North/South, East/West) otherwise right and left will be determined by the Camera's facing.
Up/Down are always up/down regardless of the Camera's facing.

Also contains two additional buttons: 'Move entity to the camera' and 'Floor Entity'. First one simply moves Entity to the Camera's current position. The second one tries to put Entity to the nearest Terrain surface below.

Rotation group

This group contains six buttons for rotating: **Yaw**/**Pitch**/**Roll**.
'Step' parameter determines the amount of rotation per single action (in degrees).
The seventh button (the one with the stop sign) is designed to reset all rotation data for the Active Entity. This could be useful when you became confused rotating/orientating the Entity.

Scaling group

This group contains three edits which control the scaling of the Entity along three Axes.

Dimensions group

This group contains three edits which control the dimensions of the Entity along three Axes. Usually this option is used for entities having area or shape in the form of rectangular parallelepiped.

Radius group

This group contains one parameter which is basically Entity's radius (if applicable).

Undo/Redo Functionality

Since version 2.95 the Editor supports undo/redo functionality in Location Editor Dialog. Undo/Redo commands could be accessed by ‘Undo’ & ‘Redo’ buttons just below the toolbar or using the ‘Edit’ menu or using the well-known Windows combinations of keys ‘Ctrl + Z’ or ‘Alt + Backspace’ for Undo, ‘Ctrl + A’ or ‘Ctrl + Y’ for Redo.

Supported Tabs:	Unsupported Tabs:
Terrain	**General Parameters**
Physical Walls	**Sky**
Materials	
Lights	
Items	
Monster Generators	
Cubes	
Named Position Points	
Camera Routes	
Particle Systems	
Monsters	
Dynamic Objects	
Halos	
Triggers	

Note that Undo buffer is reset whenever you Discard Changes, Select another Location, Relight Terrain, Apply Sun Light to Terrain, Fix or Rebuild OCTree or Clear Location!

Location Editor Dialog Tabs

General Terms

Usually all available functionality is represented by a context menu which is invoked by right mouse click on the list item in the list control or just in the list control (if there are no items). So this point will not be described below.

The most abundant commands in the list controls are duplicated by hot keys: for Delete hit ‘Delete’ key, for Add hit ‘Insert’ key and for Modify hit ‘Ctrl + E’ combination or double click (if it is not reserved for Additional Functionality).

Sometimes additional controls are provided on the Tabs and in this case they will be described.

Usually each modification dialog for an Entity contains Coordinates which determine Entity’s position within the Location. You can change Entity’s position using them or alternatively you can use Active Entity Positioning Group Controls.

In each Tab section in this document you will find several clauses:
Description – describes what that Tab is for.
Functionality – describes what you can do in the Tab and how.
Entity Controls – describes if you can use Active Entity Controls.
3D Model – describes what 3D Model is used to show an Entity in the 3D View.
Parameters – describes the parameters of the Entity of the given type.
Modification Dialog Tabs – for complex Entities describes which Tabs are available.
Modification Dialog Parameters – describes the parameters of the Entity of the given type.

General Parameters Tab

Description

This Tab displays some parameters that do not fit into any other Tab.

Functionality

You can Change parameters on this Tab.

Parameters

- ***Underwater Location*** – specifies whether or not the whole Location is under water. In an Underwater Location you party is always drowning and the graphics is 'dizzy'.
- ***Fog Options*** parameters group.
  - ***View Distance*** – everything beyond that distance won't be visible in-game.
  - ***Near Fog Border*** – ranges from 0.0 to 1.0. Specifies where the Fog starts to be visible. This value is tied to the **View Distance** – for instance if VD is 110 then setting **NFB** to 0.4 means that the Fog starts to be visible at 44, setting **NFB** to 0.65 will 'move the Fog further' to 71.5
  - ***Far Fog Border*** – ranges from 0.0 to 1.0. Specifies where the Fog gets its maximum density. **FFB** just like **NFB** is tied to the **View Distance**.
  - ***F1*** – unknown parameter. Have not noticed any difference in changing it.
  - ***Ambient Light*** – not sure if I named it correctly but it seems to affect the overall illumination of the Location.
- ***Music Setup*** parameters group.
  - ***List of MP3 files*** – shows what mp3 files (music files) will be played in the Location. Also shows their 'weights' if they are not zero. A 'weight' reflects probability of the music to be played. It is not 100% based probability but rather the accumulative one. In the list you can delete music, add new one and modify music's weight using context menu.
  - ***Chance for pause after music*** – a probability for a pause to happen between a music finished and a new music started.
  - ***Min*** – minimum time for a pause.
  - ***Max*** – maximum time for a pause.
  - ***Play Music*** – plays the music currently selected in the list.
  - ***Stop Music*** – stops the music currently played.

Items Tab

Description

This Tab displays all Items scattered over the selected Location. These Items your party might find when exploring the Location.

Functionality

You can ***Add / Modify / Delete*** Items via the context menu.

Double click left mouse button on the Item in the item list to view the Item's attributes.

Entity Controls

Positioning is to change the selected Item's position within the Location.

3D Model

An Item's 3D Model is used.

Modification Dialog Parameters

- ***Item*** from the Game's Item Database.
- ***Coordinates*** at which the Item will be positioned.
- ***The Item is Hidden*** property, if checked than the Item should be searched for.

- ***The Item Rotates*** property, if checked than the Item should be searched for.
- ***Unknown Parameters*** – advise me if you get any idea about them.

Monsters Tab

Description

This Tab displays all Monsters initially lurching in the selected Location. These Monsters your party will encounter when exploring the Location, but they won't re-spawn if you kill them (see Monster Generators Tab for re-spawns).

It is so that be that a single Monster or a bunch of them you will need a Monster Group. So you will operate with Monster Groups in the upper list control and with the Monsters within a Monster Group in the lower list control. You can select several Monsters in a Group – list control and the 3D Window support multi-selection.

Functionality

You can ***Add / Modify / Delete*** Monster Groups via the context menu in the upper list control.

You can ***Add / Modify / Delete*** Monsters in the selected Monster Group via the context menu in the lower list control.

Double click left mouse button on the Monster in the lower list or on the Monster Group in the upper list to view the Monster's attributes.

Entity Controls

Positioning to change the selected Monsters' positions within the Location.
Rotation to change Monster's facing.

3D Model

Monsters' 3D Models are used.

Monster Group Modification Dialog Parameters

- ***Monster*** from the Monster list (this is the 'Main Monster' of the Group, even if the Group consists of different Monsters the game would show the single name for the Group and it will be taken from the Group's Monster) .
- ***Coordinates*** at which the Monster Group will be positioned (this is the 'general' coordinates since for each Monster there are separate coordinates which will be used in game).
- ***Chief Monster Group*** – don't really know what it actually does, no time to test.
- ***Subordinate Monster Groups*** – don't really know what it actually does, no time to test.
- ***Unborn*** – checked when certain action should happen to 'give birth' to the monster – basically when the unborn monster is activated its special 3D Model is replayed to show its 'birth'. Examples are – Marten and El Dorado.

Monster in Monster Group Modification Dialog Parameters

- ***Monster*** from the Monster list (may differ from the Group's Monster).
- ***Initial Attitude to the Party*** – this could be overridden by Monster's Faction's attitude, by monster's Minimal Neutrality Distance, by the Behaviour Script and probably other factors.
- ***Unborn*** – checked when certain action should happen to 'give birth' to the monster – basically when the unborn monster is activated its special 3D Model is replayed to show its 'birth'. Examples are – Marten and El Dorado.
- ***Coordinates*** at which this particular Monster will be positioned.
- ***Unknown Parameters and Flags*** – advise me if you get any idea about them.
- ***Script parameters*** including ***Behaviour Script*** and a couple of unknown parameters. (This actually must be assigned to the Leader monster to take effect, determines the behaviour of the whole Monster Group: would it roam the area or stand on the place or patrol between NPPs).
- ***Some coordinates*** – Advise me if you get any idea about them.
- ***Some more coordinates*** – Way Points? It seems that those are Way Points for the monster, but I'm not sure if they are used. But I have an idea: Behaviour Script could contain some references to some coordinates at the Location: FACE North, POINTPATROL Southwest_corner and so on. I think that those commands in the Script should have corresponding 'Way Points'.

Monster Generators Tab

Description

This Tab displays all Monster Generators in the selected Location. These will generate new Monsters occasionally while you are exploring the Location.
The lower list control contains the list of Monster that can be generated by the selected Monster Generator.

Functionality

You can ***Add / Modify / Delete*** Monster Generators via the context menu.

Double click left mouse button on the Monster Generator in the Monster Generators list to view the Encounter Table's attributes. You can also use button ***Encounter Table Editor…*** for that.

General Parameters are available for edit on the Tab. Their meaning is not clear and they influence (if at all) all the Monster Generators within the selected Location.

Entity Controls

Positioning is to change the selected Monster Generator's position within the Location.

3D Model

A Special Model is used – Rapax Mannequin.

Modification Dialog Parameters

- ***Name***.
- ***Encounter Table*** from the Game Encounter Table Database (determines what Monsters will be generated).
- ***Coordinates*** at which Monsters will be generated.
- ***Respawn Frequency*** – probably means how often it tries to respawn Monsters.
- ***Respawn Chance*** – probably means the chance when time comes**.**
- ***Unknown Parameters*** – advise me if you get any idea about them.

Named Position Points Tab

Description

This Tab displays all Named Position Points (NPPs) in the selected Location. These are used as return points for RPCs, as land markers for monster scripts and for various internal purposes (like creating Items or Monsters).

Functionality

You can ***Add / Modify / Delete*** NPPs via the context menu.

Double click left mouse button on the Item in the item list to view the Item's attributes.

Entity Controls

Positioning to change the selected NPP's position within the Location.
Rotation to change the selected NPP's orientation.

3D Model

A 'Pavilion' model is used with an orientation indicating Arrow is inside it.

Modification Dialog Parameters

- ***Name*** – used by other Game Entities to refer to this particular NPP: by NPCs, by Behaviour Scripts, by the game code.
- ***Coordinates*** at which the NPP is positioned.

Dynamic Objects Tab

Description

This Tab displays all Dynamic Objects in the selected Location. Dynamic Objects (DOs) are the interactive filling of a Location. Levers, doors, bridges, curtains, ropes, traps, buttons, chest lids, weeds and many other objects are Dynamic Objects. Some of them you can activate, others are activated by other Dynamic Objects or Triggers and the rest is simply environment.

Functionality

You can ***Add / Modify / Delete*** Dynamic Objects via the context menu.

Additionally you can create new Dynamic Objects using 3D Models of Monsters, Items, Spells and Missiles via ***Add New Dynamic Object based on*** command.

Recreate Dynamic Object from another Dynamic Object... command will replace the selected Dynamic Object with another Dynamic Object.

Create Path Condition for Dynamic Object command will mark the selected Object so the Path Condition for it will be generated during the subsequent Pathfinding Database generations. If the Path Condition for the Object already exists then this menu item will be disabled. Path Condition is a mechanism that marks certain Path Nodes impassable for Monsters and is widely used for doors, bridges, invisible obstacles etc, effectively prohibiting Monsters from walking through closed doors, chasms with absent bridges and such.

Delete Path Condition for Dynamic Object command will mark the selected Object so the Path Condition for it will not be generated during the subsequent Pathfinding Database generations. If the Path Condition for the Object is already deleted then this menu item will be disabled.

Create a New Terrain Mesh from Dynamic Object command will duplicate the selected Dynamic Object as a Terrain Mesh at the same place.

For some Dynamic Objects the context menu might contain ***Go to an Entity Affected by This Dynamic Object*** and/or ***Go to an Entity Affecting This Dynamic Object***. The first one unfolds into the menu containing the list of Entities affected by this Dynamic Object. The second one unfolds into the menu containing the list of Entities that affects this Dynamic Object. Choosing any Entity in those two menu items results in its selection.

Object List Control supports ***Multi-selection***. Only limited number of operation can be performed if more than one Object is selected though.

Copy command stores a reference to the selected Object(s) into the Editor Buffer.

Paste command can be used in this or another Location Editor Dialog to instantly create copies of the Objects from the Editor Buffer.

Do not delete the Object(s) referenced by the Editor Buffer or you won't be able to paste them afterwards. Commands are available in the corresponding List's context menu.

Entity Controls

Positioning is to change the selected Dynamic Object's position within the Location.
Rotation is to rotate the selected Dynamic Object.
Scaling is to change the selected Dynamic Object's size.

Additional

Many DOs are … 'dynamic' – doors swing open, buttons get pressed and depressed, weeds are swaying, etc. Such objects can be called multi-frame objects and in this case a small control ***Show and apply Positioning…*** in the bottom of the Tab will be enabled. By changing (selecting) the frame you can see in 3D how the object will look like in action. Also when certain frame is selected **Positioning/Rotation/Scaling** changes will be applied ONLY to this frame.

Also apply rotation to Frames' Shifts – this check-box plays an important role. When it is checked than rotating an object (via the Active Entity Rotation controls) will also rotate the Shifts of its Frame(s). When it is unchecked than object's Frame(s)' Shifts are not rotated.
What does it mean? Imagine a button (Lower Monastery has one that activates the Bridge). When you press it, you can see as it is actually pressed and automatically depressed. You can see this behaviour because the button has 30 frames. 15 of them 'press' the button and the rest of them 'depress' it. The effect is achieved by the Shifts.

Shift is a 3 floating-point numbers which determine where the object will be moved in 3D Space. By modifying Frame(s)' Shifts we can control where and how the object is moved.
Each Frame has its own Shift. In the case of our button its Frames shift it to the South and back. This is because the button is on the wall that faces North.

Let's imagine that we have to place the button on the wall that faces South (the wall that across). For that we have to move our button using Active Entity Positioning Controls and than to rotate it by 180 degrees using Active Entity Rotation controls.

If we have rotated the button with the **Also apply rotation to Frames' Shifts** check-box checked, than our button now pressed and depressed correctly (to the North and Back). But if we have rotated the button with the unchecked check-box than our button now behaves very strangely – it is shifted South and back, thus effectively moving from the wall and only than back to the wall.

3D Model

A Dynamic Object's 3D Model is used.

Modification Dialog Tabs

- ***General Parameters***.
- ***3D Model Parameters*** (see 3D Model Editor Dialog).
- ***Trigger Parameters*** (see Location Common Triggers).
- ***Lock/Trap Information***.

General Parameters Tab

- ***Name***.
- ***Coordinates*** at which the Object will be positioned.
- ***Sunlit*** – determines if the object is subjected to sun's light.
- ***Blocks Passage (collidable)*** – determines if the Object blocks the way, like doors.
- ***Start Moving on Activation Only*** - might work differently in some cases.
- ***Stop Moving after the Whole Frame Cycle is Done*** – might work differently in some cases.
- ***Segments***. Segments used to separate Object's movement into two or more Segments (pretty pointless to create just one Segment). The very good example of the Segmented Object is the 'microdial' Object at Upper Monastery. Look at it and you must get the idea of the Segments.
  - It seems that for the Segments to work you need your Object to have a Trigger with type **Segmented**.
  - Segmented Objects (SOs) set Location Variables to values 0-9, depending on what Segment the Transformation/Movement of the Object has stopped while usual Objects set Location Variables to TRUE/FALSE values (note that 0 != FALSE and 1 != TRUE).
  - To modify/add/delete a Segment right-click the list and use appropriate menu item.
  - Note that Segment 0 is a kind of 'dummy' segment. It seems that initially SO is supposed to be in the Segment 0 state though it is actually not. When activating the SO for the first time it passes the Frames for both the Segment 0 and Segment 1. For that reason all original SOs have Segment 0 covering only one Frame so you cannot tell that SO actually passes two Segments and not just one.
- ***Footsteps Sound***, ***Substance Type*** and ***Environment*** controls if and what sound is used for footsteps when you walk over the object. Useful for bridges, lifts, carpets and so on. Similar to Physical Faces footstep sounds.

Note that Footsteps Sound now controls the generation of Walkable Path Nodes for a Dynamic Object. For each collidable Dynamic Object that has Footsteps Sound specified the Editor tries to create additional Path Nodes during the Pathfinding Database generation. This allows Monsters to be able to walk over bridges or supports what might not have been at place some time ago.

3D Model Parameters Tab

(see 3D Model Editor Dialog)

Trigger Parameters Tab

(see Location Common Triggers).

Lock/Trap Information Tab

- ***Type*** – **Lock**, **Trap** or **Unlockable by an Item**.

- ***Difficulty*** – from 0 to 10.
- ***Item*** – it will unlock if the type is set to **Unlockable by an Item**. Read the note on the dialog.
- ***Unknown Parameters*** – advise me if you get any idea about them.

Halos Tab

Description

This Tab displays all Halos in the selected Location. Halos are simple flat rectangles with a single texture usually semi-transparent that are used mainly for showing the glowing of the light sources (like candles and lamps).

The main property of the Halo is that it is always perpendicular to the view direction. That means that although Halos are flat rectangular you always see their front regardless of the side you approach to them.

Functionality

You can ***Add / Modify / Delete*** Halos via the context menu.

Recreate Halo from another Halo... command will replace the selected Halo with another Halo.

Object List Control supports ***Multi-selection***. Only limited number of operation can be performed if more than one Object is selected though.

Copy command stores a reference to the selected Halo(s) into the Editor Buffer.

Paste command can be used in this or another Location Editor Dialog to instantly create copies of the Halos from the Editor Buffer.

Do not delete the Halo(s) referenced by the Editor Buffer or you won't be able to paste them afterwards. Commands are available in the corresponding List's context menu.

Entity Controls

Positioning is to change the selected Halo's position within the Location.
Scaling is to change the selected Halo's size.

3D Model

A Halo's 3D Model is used.

Modification Dialog Tabs

- ***General Halo Parameters***.
- ***3D Model Parameters*** (see 3D Model Editor Dialog).

General Halo Parameters

- ***Name***.
- ***Coordinates*** at which the Halo will be positioned.
- ***Sunlit*** – determines if the object is subjected to sun's light.

3D Model Parameters Tab

(see 3D Model Editor Dialog)

Terrain Tab

This is the Important Briefing about Terrain Meshes.
Please read before proceed with Terrain modification.

Terrain Mesh is a part of visible terrain (walls, ground, water, etc). Something static which cannot change shape dynamically like a Dynamic Object might do. Terrain though can have multi-textures (such as water).

Why have so many Meshes when we can have the whole Terrain as a one Huge Mesh? And why Meshes are split in so bizarre manner? Well, I strongly suspect that in the editor Sirtech artists were using they had is as one big piece. Here we deal with the Terrain broken into pieces for performance benefit.

Why are we still working with many pieces instead of just one piece now, when we know so much about all this? Because the Editor is designed in this way and in the beginning I did not know a thing about that.

Since the Game requires the Location being in a certain format the Editor should behave and create a nice structure. Even now we are having some problems such as disappearance of some Meshes. It is because I actually don't know the algorithm to generate the 'correct' level. The Editor is using the algorithm I reverse engineered and which is highly likely somewhat different from the original. But it is constantly being improved.

Not long time ago you had to deal with Base and non-Base Meshes and with Regions manually to avoid some problems. Now it is all done automatically every time you build your level.

Now if you have any problems with Mesh visibility (it appears & disappears from view depending on your position and orientation in Game) the question is: how to fix that.

The first thing to try (it is available since version 3.40) is ***Tools→Recombine Meshes (OCTree-wise)…*** command which will rearrange Terrain Faces into Meshes in the best way possible.

Then make sure you ran ***Build→Rebuild OCTree*** command in the Location Editor for the Location in question and after that you saved the changes.

If you don't want to run ***Tools→Recombine Meshes (OCTree-wise)…*** command for some reason then follow this approach: usually the problem is that the troublesome Mesh (or its Parent Mesh) is too big. Somehow you should split that Mesh into two or more smaller Meshes and then ***Rebuild OCTree***.

How to split a Mesh? You can use ***Split Mesh in Half*** command. If it does not work (the Mesh has huge Faces) you can perform some workaround trick: use ***Carve Mesh…*** command from the Mesh List, but before that create a duplicate of that Mesh. So you carve original Mesh leaving everything, which is inside, and carve the duplicate Mesh leaving everything, which is outside, effectively slicing the Original Mesh into two pieces, which together compose the whole. For cutting it is better to use either cube or surface Mesh (which are easily created).

Note that it is HIGHLY RECOMMENDED to use ***Tools→Recombine Meshes (OCTree-wise)…*** command!

Note that now when you did some changes to Terrain Meshes, Dynamic Objects, Halos, Particles, it is better to run ***Build→Rebuild OCTree (Terrain Only)*** or ***Build→Rebuild OCTree*** command before testing it in-Game. If you changed physical faces use ***Build→Rebuild OCTree*** command.

Description

This Tab displays Terrain Meshes and controls to manage Terrain Meshes' Faces and Vertices of the Terrain Meshes' Faces.

Functionality

Add New Terrain Mesh – button with Green Plus invokes the following options:

- You can ***Add New Terrain Mesh*** using one of the several geometrical figures as templates. For each geometrical figure there are some parameters determining its size and texture mapping. Worth to note the ***Surface (Landscape)*** shape. You can specify a Height Map file (right now it is a greyscale bitmap file) and a Surface will be generated using that Height Map. I believe Height Maps could be found anywhere in the Internet. But just in case you can take a **free** program capable of generating height map files here: http://hme.sourceforge.net/ Don't forget that at the moment Editor will support ONLY greyscale bitmap files as Height Maps.
- You can import a ***3DS***, ***MS3D*** or ***OBJ*** file into a new Terrain Mesh.
- You can import ***Item***, ***Monster***, ***Spell*** or ***Missile*** model into a new Terrain Mesh

Delete – deletes the selected Mesh. You will also be prompted to delete Physical Faces corresponding to the Faces of the Terrain Mesh being deleted. Look for small button just below the Mesh List with a red cross on it.

Recalculate Lighting and Sun Light – small button with a Bulb image on it does that for the selected Mesh.

Replicate Mesh – instantly creates a new Mesh identical to the selected one.

Adjoin Mesh to Another Mesh – invokes a dialog where you can adjoin the selected Mesh to another Mesh using one of the assorted methods. In each method you got to specify the index of the Mesh to which the selected Mesh would be adjoined. In some methods you got to specify Vertices. Here are the available methods:

- ***Shift the whole Mesh using single nearest Vertex*** – finds the two closest Vertices – one from each of the two Meshes and shifts the selected Mesh so these two Vertices coincide.
- ***Shift the whole Mesh using specified Vertex*** – you specify the two Vertices – one from each of the two Meshes and the selected Mesh is shifted so these two Vertices coincide.
- ***Rotate the whole Mesh using specified Vertex around another Vertex*** – you specify the two Vertices – one from each of the two Meshes and the selected Mesh is rotated around the additionally specified Rotation Centre Vertex so these two Vertices appear on the same line going from the Rotation Centre Vertex. This method is designed to adjoin corridors for instance. You select two Vertices in each corridor Mesh that must coincide. First you use 'Shift the whole Mesh using single nearest Vertex' method to Adjoin the first Vertices in the pairs and then you use this rotation method to rotate the selected Mesh around the already adjoined Vertex so that the second Vertices in the pair get on the one line going from the first Vertex (already adjoined). Thus you can perfectly adjoin two corridor Meshes. Then you might have to use 'Resize the whole Mesh using specified pair of Vertices' if two vertices did not coincide.
- ***Rotate the whole Mesh using specified Face*** – you specify the two Faces – one from each of the two Meshes. The selected Mesh is rotated so that its specified Face becomes parallel to the specified Face of another Mesh.
- ***Resize the whole Mesh using specified pair of Vertices*** – you specify the four Vertices – two from each of the two Meshes. The selected Mesh is then resized along the line made up by two specified Vertices so that the distance between them becomes equal to the distance between the two specified Vertices of another Mesh.
- ***Shift specified Vertex (Mesh distortion occurs*** – you specify the two Vertices – one from each of two Meshes and the Vertex in the selected Mesh is shifted so it coincides with the Vertex in another Mesh. Other selected Mesh's Vertices remain at their positions so the selected Mesh gets distorted. Good for adjoining rock and mountain Meshes.
- ***Shift all near Vertices (Mesh distortion occurs*** – you specify a fractional number (an Epsilon) and for each Vertex in the selected Mesh it is checked if it is lies from some Vertex of another Mesh not far than the Epsilon. If such Vertex in another Mesh is found than the Vertex in the selected Mesh is shifted so it coincides with that Vertex in another Mesh.

Mirror Terrain Mesh – mirrors the selected Mesh using one of the 3 coordinate planes.

Merge Mesh's Two Vertices... – allows you to specify two Vertices of the selected Mesh and the first Vertex will be deleted, Faces that were using it will use the second Vertex instead and one or Two Faces will be deleted for good (those that were using both the first and the second Vertices and now would have been transformed into a 'square-less' lines if not deleted). You can also access this functionality from the **Vertices of the single Selected Face** list.

Reduce Face (Polygon) Count... – command allows you to simplify the selected Mesh by applying a polygon reduction algorithm to it. Good for Meshes imported from external models (though you can do that just before the import). Also good after Carving a Mesh with another Mesh (see the command description below) when the Carving option 'Reduce Face Count after Carving' was off for some reason.

Look here for parameters description: Polygon Reduction Parameters Dialog.

Split Mesh Faces – underlying commands allow you to split one or More Mesh's Faces into pieces thus increasing the Face and Vertex count and at the same time possibly improving Terrain Lighting. The commands are accessible from the Mesh or from the Face(s). There are three methods of Face splitting available: I believe they are quite understandable if you look at the menu commands representing them.

When activated from the Mesh the split commands will ask you for either a 'Minimum Face Square' or 'Minimum Face Longest Side Length'. 'MFS' means that all Mesh's Faces having

square less than specified number will not be split during the command processing. Likewise 'MFLSL' means that all Mesh's Faces having the longest side shorter than specified will not be split during the command processing. 'MFLSL'is used for the third splitting method while 'MFS' for the first two methods.

'MFS' and 'MFLSL' are made in an attempt to allow splitting of only large Faces and not touching already small ones. The initial 'MFS' or 'MFLSL' are taken from the first of currently selected Faces or from the first Mesh's Face if none selected – thus you can select a face which size you think is 'almost good' so you don't want the smaller Faces to be split and you don't need to modify 'MFS' or 'MFLSL' – the editor will pick them up from the selected Face.

When activated from the Face(s) the split command won't ask you for 'MFS' or 'MFLSL' – all the selected Faces will be split regardless of anything.

The fourth Face splitting method ***Using Foreign Vertex on a Face Side***" scans for every selected Face if any Vertices lie in very close proximity to its sides and then uses found Vertices to split the Face. This method is designed to eliminate possible tiny gaps between Faces of adjoined Meshes or between adjoined Faces of the same Mesh. You are asked to specify the Maximum Distance value so the Editor will know how close a Foreign Vertex should be to the Face side in order to be used to split the Face.

Note that splitting Faces by itself won't improve the lighting. The lighting will be improved only when the large Face is partly covered from the one or more light sources by other Terrain pieces – in that case you can even attain more or less good shadow.

Note also that heavy Face splitting will significantly increase total Location Face count, which will reduce performance. Not that you might notice that using modern computers...

Note also that it was noticed that two of the three splitting methods (that split into 4 and 2 faces) could produce gaps between Meshes.

Merge Mesh into Another Mesh... – allows you to specify a Mesh into which the currently selected Mesh would be merged. During this operation the currently selected Mesh will be deleted and all its Faces, Vertices, etc will be added to the target Mesh. The command is intended to facilitate already available (through new Mesh creation) Mesh merge process. It is advised to use whenever you have a lot of small Meshes.

Note that now you can perform Merge Meshes operation using Drag & Drop functionality in the Mesh List.

Carve Mesh with Another Mesh... – allows you to carve the selected Mesh by another Mesh. Note that if the Carving Meshes is not convex then result can be different of what you expected. Though you can cut with Planes hemispheres and other 'semi-convex' bodies like a box with some Faces removed or half a cilinder. The Carving Mesh always remains untouched, only the Target Mesh is changed. You can either leave all what is inside or outside of the Carving Mesh. Mutual Carving designed to supplement the Target Mesh with the Faces from the Carving Mesh (the option you will probably use often, so it is On by default).

Another option is **Reduce Face Count after Carving** designed to simplify the resulting Mesh greatly – it uses **Face Reduction Algorithm** with option **Keep Geometry Intact** turned on and **New desired Face count** is 1, so it will remove as many redundant Faces as possible.

Sometimes it is inappropriate to use **Reduce Face Count after Carving** option because (due to the imperfection of the Carving Mesh or other reason) some of the Faces that should be carved out remain. Then you can switch off this option, delete those remaining Faces manually (that should not be hard) and then apply **Reduce Face (Polygon) Count …** command to the resulting Mesh to simplify it.

Split Mesh in Half – allows you to split Mesh in half (what else it can do?). It can be used when some large Mesh has visibility issues. You just divide it, rebuild OCTree and see if the situation improved.

Change Mesh's OX Axis... – allows you to manually specify a Mesh's local OX Axis which will be used for scaling afterwards. The two remaining Axes are calculated automatically. To specify OX Axis you supply two Mesh's Vertices – the OX Axis will be parallel to the line running through those two Vertices.

Retexture Landscape (Surface) Mesh... allows you to apply a set of specific Texturing Conditions to a Mesh, so various layers of it could be textured with various Textures.

Note that if you have retexture a Mesh and not satisfied with the result and going to change the Height Range for one or more Texture Conditions do not perform the retexturing straight away! First perform the Undo operation to cancel the previous retexturing. Otherwise your Mesh ends up with incredibly large amount of Faces.

In the **Texturing Landscape Dialog** you can do the following:

- ***Modify / Add New / Delete Texturing Conditions*** – see about them slightly below.
- ***Elevate / Lower all Texturing Conditions*** – adds a specified number (can be negative) to all the 'Height From' & 'Height To' parameters of all Conditions.
- ***Load*** – loads Texturing Information from a 'cfti' file.
- ***Save as*** – saves the current Texturing Information to a 'cfti' file.
- ***Create Default*** – creates Texturing Information with default settings.
- ***Ok*** – perform the retexturing of the selected Mesh. Note that Texturing Information will be automatically saved upon leaving the dialog.
- ***Cancel*** – do not perform the retexturing of the selected Mesh. Note that you'll be asked to save Texturing Information in case you have altered it.
- ***Do Vertices Shifting to reduce Face count*** – this option is recommended to be kept 'On' unless you want your Landscape Mesh to be precise for some reason. The Max Distance from a Condition Height parameter determines when a Mesh Vertex has to be shifted to the nearest Condition Border (0.3 is a default value and means that when a Vertex is at the distance of 0.3 or closer to a Condition Height it will be shifted to that Condition Height thus eliminating the need to split the Faces using that Vertex.

A **Texture Condition** is comprised of

- ***Name*** – just for your convenience.
- ***Texture*** – you select if from the list which is build of the Materials of your current Location, so there should be a Material using a specific Texture if you want to use this Texture in retexturing.
- ***Height Range*** – consists of two values (From & To). If a Mesh's Face is above this range or below then this Face does not get retextured using this Texturing Condition.
- ***Slope Range*** – consists of two values (From & To). If a Mesh's Face's tilt angle is more than 'to' angle or less than 'from' angle then this Face does not get retextured using this Texturing Condition.
- ***Texturing Parameters*** – define how exactly Texture is applied to the Faces that satisfy Height & Angle Ranges.
  - ***U Offset*** and ***V Offset*** just regulates the shift of the texture (if you want it to fit precisely somewhere).
  - ***U Scale*** and ***V Scale*** control the Texture scaling (if you want if to be smaller or bigger).
  - ***Rotation angle*** allows you to change the direction of Texture application (useful when you have a Texture that needs to be applied upside-down or at some peculiar angle.
  - ***Texturing Method*** determines how the texture coordinates are calculated for the Faces so they look more or less nice.
    - ***Slope-Based Texturing*** – is good for anything – it is automatically determines if a Face should be textured like a floor or a wall piece.
    - ***Up-Down Transition Texturing*** – this is useful for texturing transitions between two other Textures (such as snow above and grass below, since W8 does not support texture blending we need to introduce those). The big difference from other methods is that despite of the 'V Scale' parameter, this method always scales the Texture so it fits exactly the specified Height Range. So 'V Scale' is ignored here.
    - ***Floor Texturing*** – can be used for texturing all the Faces like floor pieces. Be careful – if there are very steep or vertical Faces they will be textured incorrectly.
    - ***Wall Texturing*** – can be used for texturing all the Faces like wall pieces. Be careful – if there are very low steep or level Faces they will be textured incorrectly.

Currently Selected Mesh's Faces… – Since version 2.91 there is no more Faces list on the page. Instead you have a bunch of controls to select Faces in a different ways and to perform operations on the selected Faces. This has been done to improve performance and with the assumption that you don't really need to look at this Faces list all the time.

The following controls related to Terrain Faces management exist:

- ***Two-line text control*** – shows how many Faces are currently selected and the Material and Texture of the first selected Face.
- ***Do not deselect already selected Faces*** – this checkbox controls if the currently selected Faces get deselected when you select a bunch of new Faces using selection buttons. If it is checked then the bunch of new Faces is added to the currently selected Faces otherwise the currently selected Faces get deselected (this does not apply to the selection from the 3D Window).
- ***Select all Faces*** – simply selects all Mesh's Faces.
- ***Custom Selection…*** – shows a menu allowing you to:
  - ***Select a Face with specified Index***.
  - ***Select all Faces using specified Vertex***.
  - ***Select all Faces having a Texture as per selected Face*** (Texture is shown in the text control above).
- ***Select selected Faces' siblings*** – button adds all sibling Faces for every selected Face to the selection.
- Button with the ***Magnifying Glass*** – invokes a Faces List dialog, which shows a list of Faces from which you can select the Faces you want.
- ***Select all Faces with Texture…*** – button allows you to select all Faces having a specific Texture. The button to the right with the ***Blue Arrow*** switches to the Material Tab and selects the Material with the Texture that the first selected Face is using.
- ***Select all Faces with Material…*** – button allows you to select all Faces having a specific Material. The button to the right with the ***Blue Arrow*** switches to the Material Tab and selects the Material used by the first selected Face.
- ***Change Texture*** and ***Change Material*** – functionality allow you to change the Texture or Material for the currently selected Faces.
- ***Reverse*** – reverses selected Faces.
- ***Split…*** – shows a menu allowing splitting the selected Faces (see **Split Mesh Faces** information above).
- ***Create from Selected Faces…*** button shows a menu with some commands which allow you to create:
  - ***A New Dynamic Object…*** - useful when you want to make non-static piece of environment (a barrel, which can be broken, or a chair, which can be kicked).
  - ***New Physical Walls…*** - very useful for new Terrain Faces when you want to 'solidify' them using Physical Walls.
  - ***A New Terrain Mesh*** – is useful to duplicate pieces of Terrain: trees, furniture, houses and for making new landscapes.
  - ***Physical Rectangular Parallelepiped around Selected Faces*** – is very useful for certain pieces of Terrain (barrels, furniture, etc.) The achieved effect – you or monsters cannot traverse into the area occupied by the Selected Faces. Sometimes better than creating Physical Walls from Selected Faces because adds much less Physical Faces and Vertices. Example: creating Physical Rectangular Parallelepiped around a barrel adds 12 Physical Faces and 8 Vertices while creating New Physical Walls from all Barrel Terrain Faces adds 75 Physical Faces and many Vertices.
- ***Red Cross (Delete)*** button allows you to delete the selected Faces. You will also be prompted to delete Physical Faces corresponding to the Terrain Faces being deleted.

Face Buffer – holds controls to work with the Face Buffer. Face Buffer serves to accumulate Terrain Faces in order to create something from them at a later stage.

Face Buffer is now global (starting from version 3.41), so you can place Faces of one or more Terrain Meshes from one Location in one Location Editor Dialog and use them to create something in another Location in another Location Editor Dialog. In other words Face Buffer works between different opened Location Editor Dialogs.

Do not delete any of the Terrain Faces you put into the Buffer – this might lead to a crash or undefined behaviour when you try to create something from a Face Buffer at a later stage.

The following controls related to Face Buffer management exist:

- ***Small Text Box*** – describes the current content of the Face Buffer.

- ***Plus (Add) Button*** – command adds all selected Faces to the Face buffer. You can safely add Faces more than once – the editor will actually add each Face only once.
- ***Red Cross (Delete) Button*** – clears the Face Buffer.
- ***Create from Face Buffer*** – button allows you to create:
  - ***A New Dynamic Object…***
  - ***New Physical Walls…***
  - ***A New Terrain Mesh***

To sum up you can create a Terrain Mesh using the following methods:
- Selecting one or more Faces of the existing Terrain Mesh and using ***Create from Selected Faces→A New Terrain Mesh*** command.
- Using Face Buffer by adding to it Faces from different Terrain Meshes and then using ***Create using Faces from Face Buffer→A New Terrain Mesh*** command. Note that you can fill Face Buffer using one Location and than you can select another Location and create a new Terrain Mesh there.
- Using ***Create a New Terrain Mesh from Dynamic Object*** command on a Dynamic Object on Dynamic Objects Tab will duplicate it as a Terrain Mesh at the same place.
- Using ***Add Terrain Mesh*** button under the Mesh list and picking up one of the geometric figures or using one of the import functions (MS3D, 3DS and OBJ formats are supported at the moment) or creating new Mesh from Item, Monster, Spell or Missile model.

Entity Controls

Positioning is to change the selected Mesh position within the Location.
Rotation is to rotate the selected Mesh.
Scaling is to change the selected Mesh.

Additional

Texturing Mini-Tab provides controls for managing Texture placement placement over one or more Faces of the selected Terrain Mesh.
- ***Generate TCs…*** button opens a dialog when you can specify the aforementioned shift, contraction and rotation angle as well as several more settings for generating the Texture placement. Among them:
  - ***U/V Contractions*** and ***U/V Shifts*** control general Texture placement.
  - ***Texture Coordinates Anchor group*** – allows specifying of the Texture placement anchor – a point from where the texture placement starts (in other words a point which will have Texture Coordinates set to (0, 0). Using this you can specify a common Anchor for different Meshes.
  - ***Faces General Orientation group*** – allows specifying the general orientation of the bunch of Faces you want to lay your Texture over. Basically, the three options within the group determines the plane into which all Faces' Vertices would be projected in order to determine the Texture Coordinates – OYZ plane, OXZ plane or OXY plane. This group also contains two additional options that allow more convenient Texture placement in some cases where the first three methods perform badly:
    - ***This Mesh's Face*** – the plane of the specified Mesh is used for projecting all Vertices into it in order to determine the Texture Coordinates. The first edge of the Face serves as OX axis, OY axis is calculated based on the first edge and the plane normal.
    - ***Another Mesh's Face*** – likewise ***This Mesh's Face***, but allows using any Mesh in the Location.
  - ***Texture Rotation group*** – allows specifying the Rotation angle of the Texture before laying it onto the Faces. ***Along the Edge*** option allows you aligning Texture along a specific Edge (in case when you don't know the angle).
  - ***Make use of Faces' Average Gradient relative to the General Orientation Plane*** – this checkbox controls if the algorithm regards the two Average angles between the Faces' plane and the plane to which the Faces' Vertices will be projected. For instance, it can be used when you have the piece of floor not parallel to the ground, but say descending/ascending by 30 degree.
- ***Detach TCs*** button does special function. 'TCs' means 'Texture Coordinates'. What does it mean? First read about Faces and Texture Coordinates. So each Terrain Face references to 3 Texture Coordinates pairs within a set of Terrain Mesh's Texture Coordinates pairs.

Thus two different Faces can reference to the same Texture Coordinate pair and when you change Texture Mapping for the first Face it is also changed for the second Face. So you can 'detach' the selected Face's Texture Coordinates making them 'exclusively-for-the-selected-Face' so no other Face will reference them and thus you can change Texture Mapping only for the selected Face.

- ***Share TCs…*** button does somewhat opposite to the **Detach TCs** button. You specify a Face you want your selected Face(s) to pick TCs from. And if any of the selected Faces has a different TC on the Vertex it shares with the specified Face, this TC is replaced by the one from the source Face.
- ***Retexture Landscape…*** button does the same as Retexture Menu Command.

Texture Adjustment Mini-Tab currently contains ***U-Mirror*** and ***V-Mirror*** buttons which mirror the Texture on the selected Faces and controls to shift Texture on the selected Faces.

Raw TCs Mini-Tab shows raw Texture Coordinates of the three Vertices of the single selected Face. So you can tweak a single TC in order to fine-tune the Texture placement. The controls are hidden when no Faces or more than one Face is selected in the Mesh's Face list.

Vertices Mini-Tab shows two list controls – one displays three Vertices of the single selected Face and the other is a set of Vertices that can be used to create a new Face. Use button '>' to add existing Vertices to the right list and then use right mouse button to invoke a menu which would allow you a few options.

Set of Vertices to create a new Face List can contain up to three Vertices and is used to create a new Face based on them. When there are less than 3 Vertices in this list additional new Vertices are generated to complement the amount to 3 Vertices.

- You can ***Clear Vertex Set***.
- You can ***Remove Vertex From the Set***.
- You can ***Add New Mesh Face using Vertices from the Set***.

Note that you can create a Face based upon no Vertices – in this case all 3 Vertices would be generated based on the first Vertex of the Selected Mesh.

Note that you can create a Face using Vertices from different Meshes.

Transformations Mini-Tab controls Terrain Mesh or Faces transformations (positioning / rotation / scaling):

Apply Positioning / Rotation to… – combo box determines how entity controls (positioning/rotation) affect the selected Mesh.

- The default option ***Apply Positioning/Rotation to the whole Terrain Mesh*** is when the whole Mesh is affected – in this case no distortion occurs and Mesh geometry will never suffer due to floating point precision loss.
- The second option ***Apply Positioning/Rotation only to the selected Mesh Face(s)*** specifies that only Vertices of the selected Faces will be affected.
- The third option ***Apply Positioning/Rotation only to the selected Vertices*** specifies that only the selected ***Vertices of the single Selected Face*** in the ***Vertices*** mini-tab will be affected.

- The two latter options will lead to Mesh distortion and possibly to some tiny permanent distortions due to floating point precision loss if you going to fine-tune changes positioning/rotating/scaling them back and forth. **Undo** operation reverts the changes back without those losses.

Note that rotation/scaling are performed around the centre of the Vertices selected (it is being recalculated each time you start rotation/scaling).

Use Vertices' Centre/Use Mesh's Centre – combo box determines what centre to use when rotating only Mesh Faces or Vertices. It is disabled when the whole Terrain Mesh is affected. It has two options:

- ***Use Mesh's Centre*** – the centre (origin) of the Mesh is used.
- ***Use Vertices' Centre*** – the centre is calculated for the Vertices that are being affected (it is default option).

Also transform corresponding Physical Faces – when checked, tells Editor to transform corresponding Physical Faces as well.

3D Model

A Terrain 3D Model built from all Terrain Meshes is used.

See Location's Terrain Structure chapter for additional information.

Materials Tab

Description

This Tab displays all Materials from the Location Data File. Materials are required by Terrain Meshes and are required when you want to add a new Texture and use it on the Terrain.

Functionality

You can ***Modify*** any Material except for the first one which is reserved.
You can ***Delete*** a Material that is not used by any Terrain Mesh (Terrain Face).
You can ***Add*** a new Material.
Go to First Utilizing Terrain Mesh command will select the first Terrain Mesh containing at least one Face that uses the selected Material.

Modification Dialog Parameters

- ***Name.***
- ***Texture File(s)*** – You usually specify one of those. 2 and 3 are actually never used. You specify a Texture in the Slot 1 for most cases. Slot 4 serves to designate that the part of the 3D object having this material is specially processed – it becomes translucent and shining, I guess. Though it is hard to say for sure. Nevertheless Slot 4 used for Halos, Particles, some glass things and other similar stuff.
- ***Diffuse Colour*** – can be used to modify white (by default) Sun colour affecting Terrain Meshes. So you can make Red, Blue or whatever you want Sun now. Does not seem to have any effect on other 3D objects (affects only Terrain).
- ***Ambient Colour*** – looks like used for non-Terrain 3D objects (items, dynamic objects, etc.).
- ***Emissive Colour*** – Does not seem to do anything.
- ***Specular Colour*** – Does not seem to do anything.
- ***Alpha*** – Determines 3D object's translucency/opacity.
- ***Shininess*** – Controls 3D object's Emissive property – the object starts to 'shine'.
- ***Material Flags*** determining the special material effects.
- ***Texture Change Frequency***. This one is used for Material having an 'image file list' (IFL) as a Texture. IFL is a text file containing the list of Textures and those Textures replace one another endlessly to produce a Slide Show effects like water, monitor screen, etc. The parameter controls the frequency of Texture Change.
- ***Few unknown parameters***.

Notes

In general, Material structure needs to be re-researched and a LOT of tests and observations done in-game (that's what actually stops me from doing it – will take heaps of time).

See Location's Terrain Structure chapter for additional information.

Triggers Tab

Description

This Tab displays Triggers. Trigger is basically an area in the Location that triggers something as soon as it is detected that Party has entered that area. For example, all Teleporters are actually spherical triggers.

There are two types of Triggers: Common and Sound. Each type can be of two shapes:

- **Spherical Sound Triggers** (Play some Ambient Sound as long as Party is within the Sound Trigger's area. Area is determined by the sphere centre and radius.)
- **Rectangular Sound Triggers** (Play some Ambient Sound as long as Party is within the Sound Trigger's area. Area is determined by the Width, Depth and Height of the rectangular parallelepiped and the position of that rectangular parallelepiped in 3D World).
- **Spherical Common Triggers** (Triggers something. Area is determined by the sphere centre and radius. Also have orientation). Note that unless the Trigger has 'Works only once' checked, it will continue 'triggering' as long as Party is within it.
- **Planar Common Triggers** (Triggers something. Area is determined by two coplanar triangles usually making a rectangle.) Note that unless the Trigger has 'Works only once' checked, it will 'trigger' only once when Party steps on it, to make it 'trigger' once again Party has to come off it and then step on it again. This makes Planar Triggers ideal for multi-shot traps, because if you use Spherical Trigger you'll get a machine gun.

Functionality

You can ***Add / Modify / Delete*** Triggers via the context menu.

For some Triggers the context menu might contain ***Go to an Entity Affected by This Trigger*** and/or ***Go to an Entity Affecting This Trigger***. The first one unfolds into the menu containing the list of Entities affected by this Trigger. The second one unfolds into the menu containing the list of Entities that affects this Trigger. Choosing any Entity in those two menu items results in its selection.

Command ***Set Planar Trigger to match Face Buffer*** allows you adjusting a Common Planar Trigger's shape to match the bunch of Terrain Faces that are currently in the Face Buffer.

Command ***Add Common Planar Trigger matching Face Buffer...*** allows you adding a new Common Trigger as usual, but then it forces the Trigger's shape to Planar and matches it against the bunch of Terrain Faces that are currently in the Face Buffer.

Trigger List Control supports ***Multi-selection***. Only limited number of operation can be performed if more than one Trigger is selected though.

Copy command stores a reference to the selected Trigger(s) into the Editor Buffer.

Paste command can be used in this or another Location Editor Dialog to instantly create copies of the Triggers from the Editor Buffer.

Do not delete the Trigger(s) referenced by the Editor Buffer or you won't be able to paste them afterwards. Commands are available in the corresponding List's context menu.

Entity Controls

Positioning and ***Radius*** to change the selected Spherical Sound Trigger's position within the Location and Sphere Radius.

Positioning, ***Rotation*** and ***Scaling*** to change the selected Rectangular Sound Trigger's position, rotation and size.

Positioning, Rotation and ***Radius*** to change the selected Spherical Common Trigger's position, orientation and Sphere Radius.

Positioning and ***Rotation*** to change the selected Plane Common Trigger's position and rotation.

3D Model

For Sound Triggers a Special Model is used – 'Lyre'.

For Common Triggers a Special Model is used – 'Demon in a Box'.

When a Trigger is selected and is a Current Active Entity its semi-transparent Area is added to the 3D Model (a sphere for Spherical Triggers, a rectangular parallelepiped for Rectangular Sound Triggers and a flat 2D rectangle for Plane Common Triggers)

For Spherical Common Triggers an orientation indicating Arrow is attached to the sphere.

Modification Dialog Sound Trigger Parameters

- ***Name*** – should be unique within the Location scope.
- ***Sound*** – determines the sound file from 'ambients\' folder.
- ***Volume*** – I believe it is taken randomly from the interval each time the sound is activated.
- ***Periodical flag*** – if it is set the sound is not played in a loop, but played just once and than played in a randomly determined time period defined by the two period parameters. This is used for birds' and frogs' sounds, for instance.
- ***Trigger Area Shape*** – Can be either **Spherical** (centre and radius) or **Rectangular** (rectangular parallelepiped position, dimensions and scaling).

Modification Dialog Common Trigger Parameters

- ***Name*** – should be unique within the Location scope.
- ***Type*** – determines the type of the trigger. Somewhat elaborated below.
- ***Type 2*** and ***Type 3*** – advise me if you get any idea about them.
- ***Palpable*** – for Common Triggers this means that the Trigger detects Party intrusion into its Area, for Triggers within Dynamic Objects this means that you can activate the Object using the mouse cursor.
- ***Works only once*** – when set the Trigger works only once and cannot be activated afterwards.
- ***Positioning and Shape***. Either **Spherical** determined by a centre or **Plane/Planar** (determined by four vertices upon which two triangles are constructed).
- ***Radius*** – for Spherical Triggers this is the radius of the sphere, for Triggers within Dynamic Objects this is the maximum distance from which the Dynamic Object is reachable).
- ***Invokes NPC*** – serves to make the Triggers of **NPC Invoker** type. The Trigger must not be any of particular type to be an NPC Invoker. The specific name determines if a Trigger invokes an NPC or not. If a Trigger invokes an NPC that NPC's name is shown in the 'Invokes NPC' edit box. To select an NPC click on 'Magnifier' button and select an NPC or click 'None' to make the Trigger not to invoke any NPCs. Other Trigger's properties might not work when it invokes an NPC. Note that some NPCs may have identical names and it can be that the edit box will show the one NPC while another NPC is actually invoked. So give all your NPCs distinct names.
- ***Open/Activation Sound*** – if specified this sound will play when the Object is activated or opened, like a door.
- ***Close/Deactivation Sound*** – if specified this sound will play when the Object is deactivated or closed, like a door.
- ***Contains Message(s)*** – if set to something than in Game the mouse icon takes magnifier form as opposed to the usual hand form and activation/deactivation action produces a Level Message.
- ***Open/Activation Message*** – the Level Message is shown upon Object's activation or opening.
- ***Close/Deactivation Message*** – the Level Message is shown upon Object's deactivation or closing.
- ***Contains Loot*** – if Loot is specified that this object is a source of a Treasure. Note that it does not work for some objects.
- ***Requires an item*** – indicates that an item is required in order to activate the object.
- ***Consume Item flag*** – the item is removed from inventory when it was used to activate the object.
- ***Affect following Entities*** – when set it means that this Trigger (or a Dynamic Object having a Trigger) affects other Location Entities upon activation.
- ***List of Affected Entities*** – here you can add/remove Location Entities to be affected by Adding/Removing Links to them.
- ***Entities affecting this one*** – Just shows what Dynamic Objects or Triggers affect this Trigger (or a Dynamic Object having this Trigger). It is informational only.
- ***Depends on Variables*** – list of Variables and their values required for this Trigger (or a Dynamic Object having this Trigger) to be activated.
- ***Affected Variables*** – list of Variables set by this Object or Trigger. Note that you do not need to add a Variable ahead of time in order to use it here. You can add here either an existing variable or add an absolutely new one.
- ***Parameters 1, 2 and 3*** – Sometimes they are used by other parameters and get disabled or hidden. If they are enabled they could mean something. Parameter 1, for instance, controls doors (0/1 – auto-closing or not). Let me know if you find other usages.
- ***Damage*** – Controls damage dice for the Pain Triggers.
- ***Unknown Parameters*** – advise me if you get any idea about them.
- ***Door-like*** – a group of parameters that is usually used by various types of doors.

Modification Dialog Common Trigger Specific Parameters

- ***Usage Count*** – for Healing-type Triggers this determines the amount of usages.
- ***Infinite flag*** – for Healing-type Triggers this sets the amount of usages to infinite.

- ***Power*** – for Healing-type Triggers this determines the Heal power.
- ***Teleports to*** – for Teleporter-type Triggers this specifies that this Trigger is the Teleporter, you must than select the Destination using the small button below.
- ***Ask before teleporting*** – for Teleporter-type Triggers this flag indicates that you will be asked before teleportation otherwise you will be teleported immediately.
- ***Place of arrival*** – for Teleporter-type Triggers this specifies that this Trigger is the Teleporter Destination and does not teleport your Party anywhere. Note that in this case it is important to set the 'facing' of the Trigger's Sphere (in 3D View the Sphere's 'facing' is shown by the arrow), thus arriving party will face the specified direction.

Cubes Tab

Description

This Tab displays Location Cubes. Location Cube is basically a rectangular parallelepiped in the Location that may trigger a Location Message to appear, trigger random PC to say a certain quote or maintains certain conditions within its area (Anti-Magic area, for instance).

Right now there are three types of Cubes are known:

- **Location Message Cube** – A Location Message is displayed once the Party entered the Cube's area.
- **No Magic Cube** – Party members cannot use magic while within the Cube's area. A Location Message is displayed each time a Party member tries to use magic.
- **PC Quote Cube** – Random PC says the certain quote once the Party entered the Cube's area).
- **Other Types** are unknown at the moment advise me if you get any idea about them.

Functionality

You can ***Add / Modify / Delete*** Location Cubes via the context menu.

Entity Controls

Positioning is to change the selected Location Cube's position within the Location.
Dimensions are to change the selected Location Cube's size.

3D Model

Semi-transparent rectangular parallelepiped model is used to indicate Cube's area.

Modification Dialog Parameters

- ***Name***
- ***Type*** – see above for details.
- ***Unknown Flag*** – advise me if you get any idea about it.
- ***Position*** of the rectangular parallelepiped in 3D World.
- ***Dimensions*** of the rectangular parallelepiped.
- ***Location Message*** or ***PC Quote***.

Camera Routes Tab

Description

This Tab displays Camera Routes. Camera Route is basically a description of the route the camera should move and the speed of moving. Practically each Camera Route has Speed parameter and a set of Camera Route Points each of which determines the point in the 3D space and the orientation (facing) of the Camera at the point.
Do you remember underwater jumps at Bayjin Shallows? Or ride in Mine Tunnels? Yes, these are examples of Camera Routes.

Functionality

You can ***Rename*** Camera Routes and ***Change their Speed Factor***.
You can ***Add / Delete*** Camera Routes.

You can ***Add / Delete*** Camera Route Points in the lower list control.
Multi-selection works in the lower list control so you can alter several points at once.
When inserting new Point(s) their position and orientation are interpolated based on the position and orientation of the preceding Point (the one you insert new Point(s) after) and the subsequent Point (the one you insert new Point(s) before).
If there is no subsequent Point (you insert new Point(s) after the last Point) the position and orientation of the new Point(s) are extrapolated based on the position and orientation of the two preceding Points.

Entity Controls

Positioning is to change the selected Camera Route Point(s)'s position.
Rotation is to change the selected Camera Route Point(s)'s orientation.

Additional

Check-box '**Apply Positioning and Rotation to the whole…**' controls if the whole selected Camera Route is altered or only the selected Camera Route Points.

3D Model

A set of Arrows (one Arrow for each Point) is used. Each arrow indicates the position and orientation of a Point.

Camera Route Parameters

- ***Name*** – used for references.
- ***Speed Factor*** – how fast camera will move.
- ***Set of Route Points***.

Camera Route Point Parameters

- ***Position***.
- ***Orientation*** – where the Camera will be facing at that point.

Particle Systems Tab

Description

This Tab displays all Particle Systems in the selected Location. Particle Systems are what their name actually implies – the system of particles with a particular Material all moving according the particular rules. With particles various effects are simulated: fire, smoke, magic, etc.

Functionality

You can ***Add / Modify / Delete*** Particle Systems via the context menu.

Entity Controls

Positioning is to change the selected Particle System's position.
Rotation is to change the selected Particle System's orientation.
Dimensions are to change the selected Particle System's Particle Birth Area.

3D Model

Transparent rectangular parallelepiped model is used to indicate Particle Birth Area and actual working Particle System.

Modification Dialog Parameters

- ***Name***
- ***Particle Material*** – defines the appearance of the particles.
- ***Particles' Size*** – the size of each particle.
- ***Always render behind semi-transparent objects*** – controls Particles rendering, does what it says.

- ***Position*** – centre of the Particle System in the Location.
- ***Rectangular Parallelepiped Dimensions*** – these determine the Particle Birth Area if Particle Birth Area is set to **Rectangular Parallelepiped**. Parallelepiped is centred at the Position.
- ***Scaling?*** – must be scaling parameters but I did not notice any difference when changed them.
- ***Particle Birth Area*** – either **Single Point** or **Rectangular Parallelepiped**. In the first case all particle are created at the **Position** as opposed to the second case when particles are created in the random point within the **Rectangular Parallelepiped**.

- ***Initially Active*** – if checked than the Particle System is active and works when the Location is loaded, otherwise it is inactive.
- ***Particles' Lifetime*** – each particle will die when this time elapsed since the moment of its birth. Also there are could be additional conditions for a Particle to die (see below).
- ***Maximum Particles*** – when the current amount of Particles has reached this value and it is time to create a new Particle then if the **Particles' Death Cause: Slot for a new Particle is required** checkbox is checked then the oldest Particle is killed in order to create new one.
- ***Particles' Creation Delay*** – the System won't create a new Particle until this time is passed after the previous Particle creation moment. Put zero if you want all the particles to be created simultaneously (up to **Maximum Particles** though). It also looks like for Particle Systems with **Single Point** Birth Area **Creation Delay** cannot be lower than 30 (if it is lower than it is treated like 30 anyway).
- ***Shut down after producing N particles*** – if checked then after creating N particles the system will stop working.
- ***Active only within Frames N to M*** – determines the Frame range within which the System will be creating Particles (this is only applicable to Particle Systems within MON models which can have Frames). Set to [-1, -1] to specify 'always active'.

- ***Particles Death Cause*** parameters. Any Particle that has its Lifetime expired dies. Additionally you can specify that when the current amount of particles has reached the **Maximum Particles** value and it is time to create a new particle then the oldest Particle(s) is killed in order to create new one(s). Also you can specify one of three options: **Only when Lifetime is expired** – does nothing special; **Distance from birth position** – a Particle which has moved away from the Particle System's position far enough (the distance can be specified) is killed; **Unknown** – not sure what is does (could do something in conjunction with other unknown parameters).

- ***Initial Velocity*** can be **Constant** (each particle starts with this velocity), **Random Velocity in Range** (each particle starts with random velocity between two values) and **Inert** (each particle starts immobile).

- ***Velocity Direction*** can be **Inert** (each particle starts immobile regardless of **Initial Velocity**), **Straight** (each particle starts moving along with Particle System's direction), **Two-axial Deviation** (each particle starts moving along the line which is randomly deviated from the Particle System's direction), **Random** (each particle starts moving in completely random direction).
- ***Particle System's Direction*** – cannot be changed in the dialog, but can be changed by rotating the Particle System in the 3D Window of Location Editor using the Rotation Entity Controls.
- ***Has Acceleration*** – if checked than Acceleration vector is applied to each particle. Can be used to simulate gravitation, wind, and the lightness of smoke or any other force.
- ***Unknown Parameters*** –certainly affect the Particle System in a way I have not yet figured out. If you have any idea about them or some suspicions just let me know.

Light Sources Tab

Description

This Tab displays all Light Sources in the selected Location. Light Source is an invisible entity which emits light.

Functionality

You can ***Add / Modify / Delete*** Light Sources via the context menu.

Entity Controls

Positioning is to change the selected Light Source's position within the Location.
Radius is to change the selected Light Source's Radius.
Rotation is to rotate the selected Light Source's frames around the 1st frame (this option is available for Animated Mobile Light Sources only).

Additional

A Light Source (LS) can be Animated and/or Mobile. The only example of a Mobile Light Source is the 'Omni01' at 'Mt. Gigas Upper Caves'. Check it to see that it has several frames which basically determine the positions of the Light Source through which it repeatedly passes. To set up these frames using control **Show and apply Positioning…** in the bottom of the Tab will be enabled. By changing (selecting) the frame you can see in 3D how the Light Source has taken up the position corresponding for the selected frame. When certain frame is selected **Positioning** changes will be applied ONLY to this frame.
The special Light Source 'Sun' is prohibited from deletion, but allowed to be modified.

3D Model

A special 3D Model is used. For the active (selected) Light Source also its area (determined by the radius) is showed using semi-transparent coloured sphere.

Modification Dialog Parameters

- ***Name*** – should be unique.
- ***Initially Active*** – determines if the LS is initially 'On' or 'Off'.
- ***Coordinates*** at which the LS will be positioned.
- ***Light Colour*** – determines the colour of the LS.
- ***Flag 1*** – I have no idea (advise me if you get any idea about them).
- ***Light Radius*** – Determines the radius of the LS. Terrain and Objects beyond the radius are not affected.
- ***Intensity*** – the 'power' of the LS.
- ***Dynamic Light*** – when checked then turning it 'On' or 'Off' in-game affects Terrain. Otherwise only Items, Monsters and Object are affected. The majority of Light Sources are not **Dynamic**.
- ***Animated Light*** – Animated Lights are those that move or/and change their colour. Animated Lights should be Dynamic otherwise they won't affect Terrain and there will be no point in that.
- ***Mobile (has frames)*** – This specifies that the light moves. Manage Frames just like you manage them in 3D Model Editor Dialog).
- ***Change Rate*** –at least it affects the speed of LS moving from one Frame to the next. Maybe it also affects the rate of colour change. Need testing.
- ***Unknown Parameters*** – those are for Animated Light Sources (advise me if you get any idea about them).

Other Notes

When moving a Light Source or changing it Radius using Active Entity Controls the re-lighting of the affected portion of Terrain happens after a second of stopping moving or changing the radius.
Dynamic Light Sources do not affect Terrain lighting in the Editor because they affect Terrain in-Game.

Sky Tab

Description

This Tab displays current Location's Sky (if the Location can have a Sky at all). Some Locations cannot have Sky. We can do nothing about it because this is hard-coded. Let's better see what we **can** do with the Location's Sky.

The Sky consists of the single Firmament object, Sky Objects and Celestial Bodies. Celestial Bodies behave like Location Halos and are always perpendicular to the view direction.

Functionality

You can ***Modify*** the Firmament using button ***Modify the Firmament***.
You can ***Add / Modify / Delete*** Sky Objects in the Sky Objects list control.
You can ***Add / Modify / Delete*** Celestial Bodies in the Celestial Bodies list control.

Entity Controls

Positioning is to change the selected Sky Object's or Celestial Body's position.
Rotation is to rotate the selected Sky Object.
Scaling is to change the selected Sky Object's or Celestial Body's size.
Dimensions are to change the size of the Firmament.

Additional

Local Sky File Name – see the Note below. It is never changed.
Create Local Sky – used for creating a separate Local Sky for the Location. See Creating a Local (Separate) Sky for a Location tutorial.
Current Sky File – shows which Sky is currently being used by the Location.
Modify the Firmament and ***Select the Firmament in 3D Window*** – do exactly what they tell.

3D Model

The Firmament, Sky Objects and Celestial Bodies use their own Models.

Notes

There are only two different Skies in the Original Game: the 'DefaultSky' and the 'CircleSky'. The first one is used by every Location that can have Sky and the latter is used by the 'Cosmic Circle' location.

Note that the 'Local Sky File Name' uniquely identifies the Sky that Game loads when you enter the Location. The Game will not reload Sky when going from one Location to another if those Locations have identical 'Local Sky File Names'. This is regardless the fact that Skies for those Locations may actually be different. The Sky will be reloaded only when 'Local Sky File Names' of the Location you are leaving and the Location you are arriving at are different. Keep that in mind.

Note that starting from the version 3.61 when using CFInjector.exe to run Wizardry 8 this limitation is removed – the sky will be reloaded regardless of the 'Local Sky File Names'. Note though that Locations that share location folder (such as Lower Monastery & Upper Monastery) cannot have different Skies.

Note that the Firmament, Sky Objects and Celestial Bodies are now shown in 3D Window in the way they are shown in Game. Unlike other Location Entities Sky Entities are displayed as if the (0,0,0) is always where the player camera is. Hence you can never 'approach' a moon, nebula or a cloud plane in the Editor's 3D Window.

Physical Walls Tab

Description

This Tab is designed to manage Location Physical Walls (Physical Faces). Physical Walls is a set of invisible triangular Physical Faces that do not allow you to fall through the floor or to pass through the Terrain walls.
Yes, Terrain is visual-only and won't prevent you from falling into the Abyss. For that purpose Physical Walls are used.

Functionality

This Tab differs in its functioning from other Tabs. It does not show all Physical Faces in the main list control. Instead it shows only the selected Faces and you can select Faces in 3D View only or through some helper controls.
The list control with the Physical Faces is only shown if you have checked the **Show Physical Faces currently selected in the 3D View in the List** checkbox. This is made to increase the performance of the page and because you don't really want to see all the selected Faces in the list all the time.

The following commands are accessible via main list control's menu:
Unselect Highlighted Faces – unselects only highlighted Physical Faces.
These commands affect ALL Selected Faces in the list control regardless of their highlight status.
Reverse – reverses Physical Faces so they will block passage from the other side.
Modify / Delete – do obvious things

Modify, ***Delete*** and ***Reverse*** operations are duplicated in the ***Operations*** group box in case the Physical Face List is not shown.

Several selection options are available at the moment:
- ***Select selected Faces' siblings*** – adds all sibling Faces for every selected Face to the selection.
- ***Select all Faces*** – selects all Faces.
- ***Select all Faces with Substance Type...*** – allows you to select Faces with a specific **Substance Type**.
- ***Select all Faces with Environment...*** – allows you to select Faces with a specific **Environment**
- ***Deselect all Faces*** – deselects all Faces.
- ***Custom Selection*** – provides a few more selection options:
    - ***Select a Face with specified index*** – that is obvious.
    - ***Select all Faces having parameters as per selected Face*** – selects all Faces, which have the same parameters as the first selected Face.
    - ***Select all Walkable Faces*** – selects all Faces that have **Walkable** flag set.
    - ***Select all Non-Walkable Faces*** – selects all Faces that have **Walkable** flag unset.
    - ***Select all Faces with Pathfinding Flag*** – selects all Faces that have **Pathfinding** flag set.
    - ***Select all Faces with Flag 6*** – selects all Faces that have **Flag 6** flag set (which probably means 'Party takes no damage because of the fall onto that Face')

Vertices of the selected Face list shows 3 vertices only when a single Face is selected. From this list you can ***Add Vertices to the Face Creation Set*** using small button on the right.

Set of Vertices to create a new Face list can contain up to three Vertices and is used to create a new Face based on them. When there are less than 3 Vertices in this list additional new Vertices are generated to complement the amount to 3 Vertices.
You can do the following here:
- ***Add New Physical Face using Vertices from the Set*** – does what it says. Note that you can create a Face based upon no Vertices – in this case all 3 Vertices would be generated based on the current camera position.
- ***Remove Vertex From the Set –*** removes highlighted Vertex from the list.
- ***Clear Vertex Set -*** removes all Vertices from the list.

Entity Controls

Positioning is used either to move Selected Physical Faces or to move Selected Vertices of a Single Selected Physical Face. In either case a set of Vertices is actually moved thus altering the Physical Terrain.
Note that 'Floor Entity' button does not work with this Tab.
Rotation is to rotate the selected Faces or Vertices.
Dimensions are to change the selected Faces' or Vertices' size.

Additional

Apply Positioning / Rotation / Dimenstions to – specifies to what the Positioning, Rotation or Dimension changes will be applied to – either to the all the selected Faces or to the selected Vertices of the single selected Face.

3D Model

A Model is constructed using Physical Faces and special Textures.

Modification Dialog Parameters

- ***Face(s)*** – shows what Face(s) will be affected.
- ***Slope*** – indicates how difficult the Face is to traverse (would the Party simply slide down the Face or could Climb it). Technically it is a cosine of the angle between the normal to the Face and the 'UP' vector. When the Face is level the angle is zero and the cosine is 1.0 and when the Face is perpendicular to the floor the angle is 90 and the cosine is 0.0.
- ***Calculate Automatically*** – select this for the **Slope** to be calculated automatically for every Face.
- ***FootSteps Sound*** – Determines the ***Substance Type*** (Material) and the ***Environment*** (Room Type) of the Face. Based on these parameters certain sound files are used to playback footsteps sound.
- ***Walkable*** – it seems that all Faces where the Party could more or less normally walk must have this Flag set. Note the text just below the Flag.
- ***Generate Pathfinding Database over the Face(s)*** – this Flag should be set for the Walkable Faces over which Monsters could walk. Otherwise don't set this Flag.
- ***Other Parameters*** – Unknown Parameters.
- ***Flag 6*** – allegedly means 'Party takes no damage because of the fall onto that Face'.

Note that Location Physical Face modification dialog differs from any other dialog in the editor. Firstly it allows modifying several Faces simultaneously.
When modifying several Faces you may see a word '<distinct>' instead of the actual parameter. It means that this parameter varies in the Selected Faces. If you leave it unchanged the Faces retain each its own parameter. If you specify a certain value the Faces will pick up that value.
The similar situation is with the Flags (check boxes). If a certain check box is in the 'grey state' then this parameter varies in the Selected Faces. If you leave it unchanged the Faces retain each its own parameter. If you specify a certain value (check it or uncheck it) the Faces will pick up that value.

Way Points Tab

Description

This Tab displays current Location's Way Points (WPs) and Way Point Links (Links). It seems that wandering monsters use WPs to navigate through the location. WPs play important role for the Monsters Generators – if there are no WP near a Monster Generator then the generated Monsters might immediately disappear just after the creation.

Several things I noticed while implementing WPs in my editor:

- WP is usually connected with one or more nearby WPs via Link. Links are one way. Usually two WPs are connected (if connected at all) by two Links: from WP1 to WP2 and from WP2 to WP1, but it is not prohibited to create a one way only Link between two WPs.
- WPs in a Location are usually divided into groups so no WP from one group is linked to a WP from another group. I believe this is done for patrolling purposes so a Monster Group will patrol only the part of the Location and won't travel to other parts. So making separate groups of WPs will ensure that a patrolling Monster Group will stick to the route determined by those WPs and Links between them.
- When a Monster Group is hunting for/chasing the Party it travels to the Party ignoring WPs.
- When in combat Monsters do not use WPs either.

Way Point's Links list shows all Forward links from the selected WP and all Backward links to it.
Forward Links are indicated by the blue arrow icon.
Backward Links are indicated by the green arrow icon.
The number in the first column indicates to what (or from what) WP this Link is. The 'distance' column is calculated automatically.

Functionality

You can ***Modify / Add / Delete*** Way Points in the Way Points list using context menu.
Add Way Point linked to the selected Way Point command adds a new WP that is automatically linked to the currently selected WP.
You can ***Modify / Add / Delete*** Way Point Links in the 'Way Point's Links' list using context menu.

Note: it is probably important that any Link between two WPs does not intersect any Terrain piece or a collidable stationary object to better the Monster's Path Finding performance, though it is not critical.

Way Point Modification Dialog Parameters

- ***Coordinates*** at which the Way Point will be positioned.
- ***Flags*** – a bunch of unknown Flags.

Way Point Link Modification Dialog Parameters

- ***Passes Through a Door*** – select this is the Link is going through an area which can be potentially blocked by a Dynamic Object (such as Doors).
- ***Flag 1*** – an unknown Flag.
- ***Flag 6*** – an unknown Flag.

Automap Layers Tab

Description

This Tab displays additional Automap information like **Automap Layers** and **Automap Texture Exclusions**.

Automap Layers serve as cutting planes when you invoke Map View in the Game. The default Layer shows everything. Adding additional Layers will allow you to filter out the things that above them when the Map is shown in the Game. Automap Layers Very useful for Locations with two or more layers of Terrain places above/below each other (like Monastery, Dungeon, etc.).

Automap Texture Exclusion list contains the list of Textures and the Terrain that uses those Textures will not be drawn while in the Map mode thus drastically increasing Map loading speed and potentially preventing the Game from crashing by trying to show too much stuff. What Textures you want to place in Exclusions list? Well, could be Textures **solely** used for billboard trees, bushes, plants or grass texture or some fences, basically anything vertical (which you won't see on the Map anyway) or simply unimportant (like tiny stones or some junk). Just make sure that the Texture you want to exclude from being drawn in the Map mode is not used by something horizontal and important.

Functionality

You can ***Modify / Add / Delete*** Automap Layers in the main list using context menu.

You can ***Add / Remove*** Automap Texture Exclusions in the 'Texture Exclusions' list using two buttons below the list.

You can shift the selected Automap Layer up and down using ***Positioning*** controls.

Modification Parameters

- ***Height*** – is the only Automap Layer's parameter.
- ***Texture Name*** – is the only Automap Texture Exclusion's parameter.

Way Point Link Modification Dialog Parameters

- ***Passes Through a Door*** – select this is the Link is going through an area which can be potentially blocked by a Dynamic Object (such as Doors).
- ***Flag 1*** – an unknown Flag.
- ***Flag 6*** – an unknown Flag.

Location's Terrain Structure

Each **Location** has a set of **Materials**.
Each **Location** has a set of **Terrain Meshes**.

Each **Material** describes a substance and has one or more **Textures**.

Each **Terrain Mesh** contains a set of **Vertices**, a set of **Texture Coordinates** and a set of **Faces**.

Each **Face** has references to the 3 **Vertices** for each of its 3 apices. Two or even more different **Faces** can use the same **Vertex** (it happens when each of those triangular **Faces** have a common apex in the 3D space).

Each **Face** has a reference to the 3 pairs of **Texture Coordinates** for each of its 3 apices.

Each **Face** has a reference to a **Texture**. Technically it is a reference to a **Material**, but the substance description is ignored by the **Face** and only a **Texture** is used.

Each **Vertex** has a reference to a **Material** which determines the substance at that **Vertex** (the **Texture** is ignored). In some **Terrain Meshes** all Vertices refer to the same **Material** so instead of having many references for each **Vertex** such **Meshes** has only one reference for the whole **Mesh**.

Each **Vertex** has **Additional Light** property. This is a **Floating Point RGB Colour** which means that some light source illuminates that **Vertex** (this light source can be of any colour).

Each **Vertex** has **Light Intensity** property. This basically means how much Sun's light reaches the **Vertex**.

Capabilities of the 3D window

You should left-click on the 3D window to move focus on it in order to get access to the capabilities described below.

3D window is capable of highlighting the selected object by drawing its wire frame in white colour. Additionally, for **Terrain Meshes** (and in few other cases) it will highlight the selected **Faces** in red outline as well as with ***Selected Faces Highlight Color*** (you can change this color in the main menu ***Setup→3D Settings…***).

To select/unselect a **Face** use Ctrl + Left Mouse Button or select/unselect **Faces** in the Face List.

With Arrows, 'Home' and 'End' keyboard buttons you can move camera in the 3D world. Alternatively you can use 'standard' FPS controls (**'W', 'A', S', 'D'**).

Holding Right Mouse Button and moving the mouse you can change camera view direction.

If you are lost in 3D world you can use 'R' (Return, Reset) button to reset camera's position.

Below is the list of the capabilities (you can always see this on the location dialog by clicking small blue 'i' button or invoking a 3D Window menu clicking on the tiny button in the upper left corner of the 3D Window and selecting '**Help**' command.
Note that not all of them are accessible in every 3D Window.

Here is the controls summary table:

'Up', 'Down', 'Left', 'Right', 'Home', 'End' or **'W', 'A', S', 'D', 'Q', 'E'**	**Moves Camera** (5 times faster with 'Shift' pressed).
'Mouse Wheel'	**Moves Camera** forward/backwards (5 times faster with 'Shift' pressed).
'Right Mouse Button' + 'Mouse Move'	**Rotates Camera**.
'R'	**Resets Camera** position and orientation.
'Shift' + 'R'	**Resets Camera** orientation only.
'X', 'Y', 'Z'	**Rotates Entity** around axes (inversely with 'Shift' pressed).
'Left Mouse Button' click & release	**Select an Entity**. **Select an individual Face** (if allowed).
'Ctrl' + 'Left Mouse Button' click & release	**Toggle Selection of an individual Face** (if allowed). **Toggle Selection of an Entity** (if multi-selection is allowed for them).
'Shift' + 'Left Mouse Button' click & release	**Add an individual Face to selection** (if allowed). **Add an Entity to selection** (if multi-selection is allowed for them).
'Left Mouse Button' click, drag & release	**Select a bunch of Faces** (if allowed). **Select a bunch of Entities of one type** (if multi-selection is allowed for them).
'Ctrl' + 'Left Mouse Button' click, drag & release	**Toggle Selection of a bunch of Faces** (if allowed). **Toggle Selection of a bunch of Entities of one type** (if multi-selection is allowed for them).
'Shift' + 'Left Mouse Button' click, drag & release	**Add a bunch of Faces to selection** (if allowed). **Add a bunch of Entities of one type to selection** (if multi-selection is allowed for them).
'Left Mouse Button' on Entity's Moving Axis + 'Mouse Move'	**Move the selected Entity along the Axis**. Only supported when the *Move Gizmo* (*3-coloured Axes*) is visible. Left-click on an Axis, it will become yellow and you can drag the selected Entity along this Axis.
'Left Mouse Button' on Entity's Rotation Circle + 'Mouse Move'	**Rotate the selected Entity around an Axis**. Only supported when the *Rotate Gizmo* (*3-coloured Circles*) is visible. Left-click on a Circle, it will become yellow and you can rotate the selected Entity

	accordingly.
'Left Mouse Button' on Entity's Scaling Axis + 'Mouse Move'	**Scale the selected Entity along the Axis**. Only supported when the *Scale Gizmo* (*3-coloured Axes*) is visible. Left-click on an Axis, it will become yellow and you can scale the selected Entity along this Axis. White cube allows uniform scaling.
'P'	**Toggles** polygon fill mode.
'H'	**Toggles** Highlighting of the selected Entity.
'U'	**Toggles** 'Outline Selected Entity Through Others' mode.
'O'	**Toggles** 'Show only selected Entity' mode.
'T'	**Toggles** Terrain rendering mode.
'F', 'V'	**Focuses** on the selected Entity.
'C'	**Changes** Camera's speed.
'B'	**Toggles** 'Also Select Backfaces' option affecting Block Selection (Rectangle Selection).
'1'	**Sets 3D Gizmo** into the' Move Local' mode.
'2'	**Sets 3D Gizmo** into the 'Move Global' mode.
'3'	**Sets 3D Gizmo** into the 'Rotate Local' mode.
'4'	**Sets 3D Gizmo** into the 'Rotate Global' mode.
'5'	**Sets 3D Gizmo** into the 'Scale' mode.
'0'	**Sets 3D Gizmo** into the 'None' mode.

3D Model Editor Dialog

3D Model Editor consists of the following parts:

- A set of sub-models in the Sub-Models list control. Usually each model consists of the only one Sub-Model so you won't be seeing this list control very often. But if you see it then each Sub-Model acts like a Frame in the Model.
- A set of Faces for the selected Sub-Model is shown in the Faces list control. Supports multi-selection. Here you can ***Delete Selected Faces***, ***Change Material*** for the selected Faces or ***Reverse Selected Faces***.
- The Frames of the 3D Model are shown in the Frames list control. Frames allow the object to be 'alive' – to move, rotate, to chance its size or even to change its very shape (as 'Sea Weeds' you can find in the Lower Monastery). You can ***Remove*** the selected Frame or ***Add*** / ***Insert*** new Frame(s). Note that when you Add or Insert new Frames, the Editor tries to interpolate them so they more or less fit the animation.
- 3D View window which will show the selected Frame or the slide-show of all the Frames if the 'All Frames' item is selected in the Frames list control.

The following Tabs exist in the Dialog (note that not all of them are always available – the set depends on what Model you are editing):

- **Materials Tab** – contains the list of the Materials and a Texture preview control. Here you can ***Modify*** an existing Material or ***Add*** a new one using list context menu.
- **Texturing Tab** – contains some controls that might help with the texturing of the model. Here you can modify the texture mapping by shifting, contracting or rotating it. The changes applied to the currently selected Faces. You can also use ***Generate TCs*** tool for auto-generation of texture mapping. ***U-Mirror*** and ***V-Mirror*** buttons allow you mirroring texture mapping. ***Raw Texture Coordinates*** are accessible only for a single Face.
- **Parameters** – contains some unknown parameters and Animation FPS which determines how fast the frames will be changed (Frames per second). Note, that it is possible to override this parameter in MLS script file. See MLS Tutorial.
- **Transformations** – allows shifting and rotating single Frame (or all Frames when 'All Frames' is selected in the Frame list). Also Allows ***Mirroring*** and ***Axes Swapping***.

Dialog now contains ***Export*** button which will export the 3D Model into *.obj file that can be edited in an external application. In case of more than one frame multiple *.obj files are exported.

3D Model Editor Dialog – Particle Systems Tab

A ‘Particle Systems’ Tab has been added to the Modify Animated 3D Model Dialog for Monsters, Spells and Missiles Model.

- You can ***Add / Modify / Delete*** Particle Systems inside the Model.
- You can ***Bind*** the selected Particle System to a model Vertex or ***Unbind*** it from the Vertex.
- You can ***Rotate*** the selected Particle System using Rotation controls.
- Use Particle Systems List or 3D window to select a Particle System.
- Use 3D Window to select a Face if you want to Bind the Particle System to one of its Vertices.
- In the 3D Window each Particle System is represented by a small cube textured using the Texture taken from the Particle System’s Material. The centre of the cube is in the exact centre of the Particle System.
- The Particle Systems here are the same as in Locations except it has two additional parameters: ***Active from Frame*** and ***Active to Frame*** which determine during which Model’s Frames this Particle System is active.

3D Model Editor Dialog – Light Sources Tab

A ‘Light Sources’ Tab has been added to the Modify Animated 3D Model Dialog for Monsters, Spells and Missiles Model.

- You can ***Add / Modify / Delete*** Light Sources inside the Model.
- Use Light Sources List or 3D window to select a Light Source.
- In the 3D Window each Light Source is represented by a small sphere coloured using the colour taken from the Light Source. The centre of the sphere is in the exact centre of the Light Source.
- Note that for versions 2 and 3 Light Sources the values specified for parameters in the ‘Base Parameters’ group might do nothing. Experiment with other parameters. If you have the strong tested confidence that these parameters are not used (or used) in versions 2 and(or) 3 Light Sources, mail me to remove them from the Dialogs or to update this document.

3DS/MS3D/OBJ Model Importer Dialog

Model Importer Dialog provides the functionality for importing 3D models from binary 3DS, MS3D or OBJ files. The dialog is accessible from the Main Menu: ***Tools→3DS Model Importer...*** or ***Tools→MS3D (Milkshape) Model Importer...*** or ***Tools→OBJ (Wavefront) Model Importer...***

It is also accessible from Location Dialog when you can create a new Mesh by importing an external 3D Model.

You can open a model file. If model contains too many Frames you will be asked if you want to load all Frames or only part of them.

Dialog allows importing Animated Model using multiple OBJ files. For this you should have a set of OBJ files that are named uniformly in the following pattern: "<some_name>XXXX.obj". XXXX here is a number.

It is very important that amount of digits in XXXX is the same in all OBJ files: "0000, 0001, 0002, …, 0950" or "000, 001, 002, …, 064". If it is not the case then order of the loaded OBJ files is undetermined.

The Editor detects the presence of the multiple OBJ files when you open any one of them and asks you if you want to load them as a single Animated Model.

How to get multiple OBJ files for an Animated Model? It can be done by running a small script in 3DSMax. Load in 3DSMax an Animated Model which you cannot directly import using CF (for instance a Model of MAX format). Then run the small script and you get a bunch of OBJ files.

Now here is the script:

```
for i = animationRange.start to animationRange.end do (
slidertime = i -- forcing the slider as some elements don't react well to 'at time'
numStr = subString ((10000 + i) as string) 2 -1
exportFile ("c:\\filename_" + numStr + ".obj") #noprompt
)
```

Note that this script will create all the files in the "C:\" folder and they will be named "filename_0000.obj", "filename_0001.obj" and so on. You might want to change this behaviour by modifying the 4th line of the script.

Controls:

- ***Frames list*** – contains the loaded Frames. Allows multi-selection. 3D Window plays the selected Frames when more than one Frame is selected. Note that if you select too many Frames 3D window might fail to show the Animation and/or the Editor might crash.
- ***3D Window*** – shows the Model and allows selecting individual Face(s).
- ***Initial Faces / Initial Vertices*** – shows initial amount of Faces and Vertices in the Model.
- ***Current Faces / Current Vertices*** – shows current amount of Faces and Vertices in the Model. They might differ from the initial amounts if you performed some actions (like ***Reduce Face Count***).
- ***Look*** – centres the Model in 3D Window.
- ***Remove Duplicated Vertices during Loading*** – if checked then an algorithm is ran on the loaded Model to remove duplicated Vertices. Recommended by default, turn it off only if you experience some problems (i.e. Model deformations).
- ***Materials list*** – contains all distinctive Materials the Model uses. Allows multi-selection. Double click invokes Material Modification Dialog. Note that when you select one or more Materials then all the Faces using them are highlighted (selected) in the 3D Window.
- ***Materials Tab*** – hosts controls for Material management.
    - ***Modify Texture for Selected Material(s)…*** – allows changing Texture for one or more selected Materials with one operation.
    - ***Modify Material…*** – invokes Material Modification Dialog for a single Material.
    - ***Merge Selected Material(s)*** – tries to merge the selected Materials, effectively deleting the duplicates. Does not merge Materials that differ from each other.
- ***Model Tab*** – hosts controls for the top-level Model manipulation.
    - ***Change X-Y, Change X-Z, Change Y-Z …*** – allow swapping two axes to orientate the Model as required.
    - ***Mirror X, Mirror Y, Mirror Z…*** – mirror the Model along an axis.
- ***Faces Tab*** – hosts controls for the Faces manipulation.

- ***Select Selected Faces' siblings*** – finds and adds to the selected set the Faces which are sibling to the currently selected Faces. Useful for selecting an entire separate piece of geometry (such as a bone) in order to delete it.
- ***Delete Selected Faces…*** – deletes the selected Faces.
- ***Reduce Face Count***… – executes Face Reduction algorithm on the Model.
- ***Return Original Face Count*** – *since this dialog does not have Undo/Redo stack, this serves as the only one Undo point after the Model has been loaded.*

- ***Import Tab*** – hosts controls allowing importing the Model as one of the W8 Models.
  - ***Create a New Monster Model*** – creates a Monster Model using the selected Frames.
  - ***Create a New Missile Model*** – creates a Missile Model using the selected Frames.
  - ***Create a New Spell Model*** – creates a Spell Model using the selected Frames.
  - ***Create a New Item Model*** – creates an Item Model using the first selected Frame.
  - ***Create a New Terrain Mesh*** – creates a Terrain Mesh using the first selected Frame (only available when invoked from the Location Editor Dialog).
  - ***Scale Model during Import*** – allows specifying scaling in case the Model is too large or too small.

Polygon Reduction Parameters Dialog

The dialog is used to specify Polygon Reduction parameters. Reduction is used to reduce amount of Faces usually without much 3D object distortion. Used in 3D Model Importer dialog and in Location Editor.

- ***New desired Face count*** – the algorithm will continue reduction until the specified amount of faces is reached.
- ***Try to Keep Geometry Intact*** – if this flag is set then the algorithm will try to avoid distorting the Mesh in any significant way.
- ***Error*** – used when **Try to Keep Geometry Intact** flag is ON. Changes to the Mesh which lead to the Error greater then specified are not performed. Error measured by comparing the altered Faces' squares before and after the change.
- ***Support 2-sided surfaces (ribbons, tree billboards, etc.)*** – Turn this on when your object has at least one pair of faces which share Vertices but are facing in opposite direction. Wizardry 8 trees are the great example of this.
- ***Generation of Texture Coordinates*** – you can select either ***Uniform*** or ***Individual*** method.
    - **Uniform** method calculates the same TCs for all changed Faces, but it requires that TCs for these Faces were the same prior to change. Produces more seamless Texturing.
    - **Individual** method calculates new TCs for every changed Face individually thus introducing some discrepancy which could lead to noticeable seams. Should be used when **Uniform** method produce weird texturing.

Maze Creation Dialog

The dialog is used to specify Maze Creation parameters. The dialog consists of several parts:

1. ***Maze Structure***:
   a. ***Maze Size*** – specify the size of the Maze in cells.
   b. ***Max Rooms*** – controls the maximum amount of the Rooms in the Maze.
   c. ***Size of the Rooms*** – control the sizes of the Rooms (generated randomly in the specified range).
   d. ***Max Doorways*** – controls the maximum amount of the Doorways in the Maze.
   e. ***Max Exits*** – controls the maximum amount of the Exits (passages in the Maze's outer wall) in the Maze.
   f. ***Use Specific Seed*** – lets you 'freezing' a certain random seed to get the same Maze layout whenever you Re-Generate it.
   g. ***Re-Generate*** – this button generates a new Maze layout based on the specified parameters.
   h. ***Preview*** control shows you the Maze in 2D.
      - You can change the Maze layout by clicking on the walls to cycle through the Wall, Passage & Doorway repeatedly. This allows you tailoring the Maze as you want it.
      - Pressing CTRL button while clicking the cells will place/remove the preferred positions for some Maze elements (content such as Items, Loots, Monsters, Lights, etc.). Without preferred positions the content is placed randomly.
   i. ***Passage Pixels*** and ***Wall Pixels*** – control the amount of pixels used to draw the 2D representation of the Maze.

2. ***Maze Elements Tabs*** allow to controlling the placement of interactive and passive content of the Maze, including elements necessary for W8 to perform correctly (such as Physical Walls and Way Points):
   a. ***Physical Walls*** – without those the party will fall through the floor and will walk through the walls. Also allows optional creation of Physical Walls for the ceiling (usually not required) and control over the Footsteps Sound.
   b. ***Way Points*** – controls whether or not to generate Way Points – a necessary element for Monster Generators.
   c. ***Lights*** – controls the generation of the Light Sources.
   d. ***Monsters*** – controls the generation of the Monster Groups and Monster Generators.
   e. ***Doors*** – controls the generation of the Doors in the Doorways.
   f. ***Items*** – controls the generation of the Item Bundles.
   g. ***Loots*** – controls the generation of the Treasure Chests.

3. ***3D Parameters*** control the Look of the Maze:

   a. ***Dimensions*** for various parts of the Maze control its size.

   b. ***Create Ceiling*** – specifies whether or not to generate ceiling (option for open mazes such as hedge mazes.

   c. ***Create Floor*** – specifies whether or not to generate floor (probably always 'yes').

   d. ***Create Sunproof Roof*** – specifies whether or not to generate special geometry which will 'protect' the Maze itself from the Sun rays and thus the Maze won't look like lit by the sun when it is actually a dungeon.

   e. ***Create Walls' Tops*** – specifies whether or not to generate geometry for the Top parts of the walls. Can be useful for Automap. Also essential if the Maze can be overlooked by player from above.

   f. ***Create Gaps***– this depends on *Create Walls' Tops* option and when checked the gaps will be generated in Walls' Tops above all doorways. Useful for Automap but not suitable if the Maze can be overlooked by player from above.

   g. ***Generate Maze at Coordinates (0, 0, 0)*** or ***Generate Maze at the Current Camera Position*** – allow you controlling (more or less) the placement of the Maze in your Location.

   h. ***Texturing Parameters*** – control the texturing of various parts of your Maze, including Texture and few texturing parameters.

4. ***Other Controls***:

a. ***Save Maze*** – allows saving all information about your Maze (including preferred positions which are not preserved otherwise) at any time.

b. ***Load Maze*** – allows loading your saved Mazes.

c. ***Create*** – creates a Maze with the specified parameters.

d. ***Cancel*** – closes the dialog and does nothing.

Tutorials

New NPC Creation

Here are the basic steps to create a new NPC (not RPC).

1. Open NPC Editor.
2. Click "New", select "Anna" and click OK.
3. Choose your NPC script name. To avoid possible problems make that name without spaces. Make it simple. For example: "Cunny_the_Trader".
4. OK, now your NPC's name is "NewBornNPCName". Change it to anything you want - here you may use spaces: "Cunny the Trader".
5. Now you may assign a Drop Loot for your NPC, so it would be a unique item set or a common set of items - your choice.
6. We'll skip "RPC parameters" tab and open "Trader Parameters" tab.
7. Here we can specify the set of items NPC trades, what item types (s)he buys and how cunning (s)he is (I mean buy/sell rate).
8. Ok, now go to NPC Script page. Here comes the special part, Right now your NPC will behave and talk just like "Anna". If you want to change it you have to rewrite/delete/add quotes and accompany them with sound files (it is optional).
8a. To change a quote, select it and use menu or buttons to the right.
8b. To assign a sound to a quote, select a quote and use "Assign sound" option. The editor will then copy the designated sound file into the proper location, so the game will find it there.
9. Click "Apply" button to save your NPC in the memory.
10. Now it is time to create a monster for your NPC. Create a new monster in the monster editor (or pick an existing one). Make sure that NPC check-box is checked and select an NPC for that monster: "Cunny the Trader". Click "Apply changes" to save changes in the memory".
11. The final step is to place the monster on some level.
12. Open the Location Editor and select the Location you want your new NPC to be in. To be able to comfortable place the NPC you will probably want to see it, so check "Show Monsters", "Show visual Walls" and "Show 3D World for Selected location" checkboxes. You may check other checkboxes as well.
12a. In our case let's select "Lower Monastery" location, go to "Dynamic objects" tab and select objects named "lid05" from the list. This is the lid of the "Starting Chest" and we are going to place our NPC nearby. Check and uncheck "Synchronise 3D World with the selected entity" checkbox, so the editor will place the **Camera** nearby our chest.
12b. Remember the coordinates of the "lid05"- (132, 101, 1)
12c. Go to "monsters" tab and add new monster group entry. Select "Burz" from the "Lower Monastery" and click OK. Now modify monster group entry and change its coordinates and the monster. Now you may see your NPC and change its coordinates until they satisfy you.
12d. Click "Apply" button.
13. Push "Ctrl+S" to save everything you've just changes onto the hard drive.
14. Now you are ready to go in game, start a new game and see your NPC in action.

New RPC Creation

Creating an RPC is somewhat similar to creating an NPC. You just have to pick an RPC as a base for your new RPC.

Making Dynamic Objects from Location Terrain

This tutorial will briefly show you how to create new Dynamic Objects that will almost look like a piece of Terrain.

It is possible to create additional ‘Terrain’ at any Location.
This ‘Terrain’ won't be the real Terrain with shadows as it usually is. But nevertheless it would resemble the real Terrain.
What I mean under the term 'Terrain': houses, huts, cliffs, boulders, trees, pergolas and so on - everything that is named Terrain in the Editor and could be found on the Terrain Tab.

On the Terrain tab on the Location Editor Dialog you can see 'Faces' list.
Invoking a context menu on one or several faces will reveal several menu items:
- ‘Create an Object from selected Mesh Faces’
- ‘Add the selected Mesh Faces to Face Buffer’
- ‘Create an Object using Face Buffer’
- ‘Clear Face Buffer’

The first one creates new Dynamic Object based on the selected Terrain Faces.
The second one adds selected Terrain Faces to the Face Buffer (be sure not using this menu item twice for the same Face(s) so not to create excess data). This menu item is designed specifically for the case when necessary faces belongs not to one but to two or more Terrain Meshes (which happens very often) and thus the first menu item cannot be used to create one object.
The third one creates new Dynamic Object based on the faces currently in the Face buffer.
The fourth one simply clears the Face buffer when you erroneously added some unneeded Faces to it.

Here step-by-step example in which we create a hut with a door at the beach at Lower Monastery.
1. Select 'Bayjin' location in the Location Editor and go to the 'Meshes' tab.
2. Note a hut in the 3D window inside which a coordinate {156.8323, -19.3224, -0.3952} is located. This is the most simple-to-copy hut. Determine that all hut's Faces are scattered in four Meshes (193, 334, 333, 192).
3. Repeatedly select all necessary Faces in each of these Meshes and add them to the Face Buffer (it pays off to note Face's Textures when you select necessary Faces). Ideally, you have to use menu item ‘Add the selected Mesh Faces to Face Buffer’ four times – once for each Mesh, but this is not a requirement – you can add Faces to the Face Buffer even one by one, but this is much slower than to add a bunch of them at a time.
4. Then create a Dynamic Object using Face Buffer via menu ‘Create an Object using Face Buffer’ and save the Location.
5. Select 'Lower Monastery' location in the Location Editor.
6. Go to the 'Dynamic Objects' tab, select menu item 'Add New Dynamic Object...' and pick the newly created hut from the ‘Bayjin’ location.
7. Place hut using dialog controls (green and orange arrows). Then select 'Modify Dynamic Object...' to set 'sunlit' flag. Make sure that ‘Blocks Passage (collidable)’ flag is set because you don’t want to walk through the hut’s walls, do you?
8. Create the hut's door in the same manner. Save the Location.
9. Optionally you can now select ‘Bayjin’ location and delete the new hut if you don't want it there. The hut at ‘Lower Monastery’ will remain intact.

Thus you can copy ANY piece of terrain to any location you want as a Dynamic Object.

Making a Teleporter

This tutorial will briefly show you how to create a Teleporter from one Location to another.

1. We need to create a place of arrival. Select a Location you want to be a destination Location. Let it be 'Bayjin' in our example. Let's get straight to the good Bayjin chest, shall we?
2. Find the chest and place the camera before it (approximate coordinates are 84.4867, -71.2785, -1.0324).
3. Now go to 'Triggers' tab and select the second Trigger named 'BA110'. We select this trigger as a template because it is exactly the one we want to make – it is of type Teleporter (place of arrival).
4. Right-click the selected Trigger and choose ***Add New Trigger→Add New Common Trigger*** menu item. Trigger 'BA110' is already selected as a template Trigger so click OK.
5. Now change the name of the new Trigger. Since it is a Teleporter you are only allowed to supply two number characters as a name. Type whatever you like but make sure the resulting name will be unique (the editor will check it for you). Let's type '07' and click OK.
6. Change 'Radius (for Spherical …)' to 3 instead of 10. We don't need that large radius. Click OK.
7. Your new Trigger will have the name 'BA107'. You can modify only two last characters of Teleporter's names.
8. The Trigger is placed right where your camera is. Now place your camera so you can see the Trigger. Note the arrow sticking out of its sphere – this is where your party will be facing when arriving to this 'place of arrival'. Using Active Entity Rotation controls rotate the Trigger so its arrow (facing) will point towards the chest.
9. Now if turn on the option to show Location Monsters in 3D Window (or if that option is already on) you can see that there is a Rinjin Monster is in the hut with the chest. This monster will fight with you for the chest, so you can move it out of the hut if you like.
10. Save your changes using ***File→Apply Changes*** menu.
11. Now it's time to set up the place of departure. Select a Location you want to be a departure Location. Let it be 'Lower Monastery' in our example.
12. Move your camera to the beach where the adventure usually starts. Note the remains of the space ship scattered all over the place. One large piece has neatly leaned to the cliff wall. Move the camera closely to it. Let's make our Teleporter here.
13. Now go to 'Triggers' tab and select the first Trigger named 'MO102'. We select this trigger as a template because it is exactly the one we want to make – it is of type Teleporter that teleports to some place (to 'Upper Monastery' in this case).
14. Right-click the selected Trigger and choose ***Add New Trigger→Add New Common Trigger*** menu item. Trigger 'MO102' is already selected as a template Trigger so click OK.
15. Now change the name of the new Trigger. Since it is a Teleporter you are only allowed to supply two number characters as a name. Type whatever you like but make sure the resulting name will be unique (the editor will check it for you). Let's type '07' and click OK.
16. Now we have to select the destination. We want our chest after all. So click the magnifier button right next to the controls describing where this Teleporter leads to.
17. In the dialog select 'Bayjin', expand it and select 'BA107' (the destination Teleporter we created two minutes ago). Click OK.
18. You can check or uncheck 'Ask Before Teleporting' flag. I think it is pretty clear what it does. Click OK.
19. Save your changes using ***File→Apply Changes*** menu.
20. Now you can go in game and check how your teleporter system works.
21. After you have tried that you might want to create the back way teleporter…

Creating a New Location

This tutorial will briefly show you how to create a New Location. The New Location initially will be the exact twin of the one it was created from. Than you can customise it with whatever that the Editor allows.

1. To create a New Location in the Location Editor select any Location that has word '(unused)' in its name or a 'Footstep Test Level'. You will be asked whether you want to create it. Click 'Yes'.
2. Select the Location you want to base your New Location on. Click OK.
3. The new location has just been created.
4. Now you can do with the New Location whatever you want including deleting anything without being in fear of breaking something (in the original Locations it is unwise to delete some entities because their usage can be hard-coded and thus it will prevent the game from loading the Location).

Note, that Sky data is not copied from the source Location during the New Location creation. If a custom Sky is required for the new Location you must create it separately (if possible). For further information see Sky Tab section in this document.

The Name for your New Location is really a problem. What is the Name for a Location? This is a string that you see when the Location is being loaded in game.
For the 'Footstep Test Level' you can specify any name and that's all right. For others (those initially having '(unused)' in their names) you cannot specify separate names. Instead it is only possible to specify a general name for all of the New Locations (except for 'Footstep Test Level'). This would likely be some kind of 'Mystery Area' or 'Forgotten Lands' or something.

To change a Location Name (including the original ones) you would need the Interface Strings Editor. It could be accessed from the Main Dialog using ***String Editors→Interface Strings Editor…*** menu.

1. Launch Interface Strings Editor Dialog.
2. Click the small button with the 'i' icon and read the information carefully.
3. Now find the string that serves as a start for the 'Location Names' role.
4. This string and several strings below serves as Location Names. You can freely change them.

Note, that you should not assign string roles unless you have the Editor showing wrong strings. Strings Roles do not affect the game, they are used to tell the Editor which string it should use and where.

Dealing with Dynamic Object Tutorial 1

In this Tutorial I'm going to show how to use Dynamic Objects in practice to do something. I'm going to implement the following: in the Lower Monastery you open the door on the beach and immediately get hit by a boulder falling from above and take small damage.

1. Create a boulder object from Terrain (see the 'Making Dynamic Objects from Location Terrain' Tutorial). Call it 'objBoulder' (for instance).
2. Modify our boulder and on the '3D Object' tab add single new Frame using context menu in 'Show Frames:' list control (see 'Adding More Frames to 3D Object' tutorial for details). Click OK.
3. Now move the boulder to the some place above the door.
4. Now in the Dynamic Object Tab select the second Frame using the 'Show and apply Positioning… the following Frame(s) only:' control.
5. Move the boulder so it lies before the door (like it has already fallen). Moving the boulder with the second Frame selected will move the boulder's second Frame only thus effectively positioning the second Frame while the first Frame remains where it is.
6. Modify our boulder and on the 'Dynamic Object' tab check 'Start Moving on Activation Only' and 'Stop Moving after the Whole Frame Cycle is Done' check-boxes. Click OK. Thus you'll specify certain rules of boulder's movement in the 3D World. It will be moved only when activated and it will stop once moved using all its frames (won't continue moving back-forth).
7. Modify our boulder and check 'Has Trigger' check-box just below the tabs. The Trigger Tab will appear. Set Trigger name to 'trigBoulder' (for instance). Check 'Works only once' check-box – we don't want it to fall more than one time. Set 'Type' to 'Unknown 3'. Click OK.
8. Select the entrance door object (either in 3D Window or in the list – this is object 'Box09'). Modify it and select Trigger Tab. Check 'Affect Following Entities' check-box and using menu ***Add New Link→Add Link to a Dynamic Object*** select our boulder object. Click OK.
9. Save your changes. And go in game to see what we have done. In game if you are hit by the boulder it may kill you outright – this is because your party is considered 'squashed' by it.
10. To avoid it Modify the boulder object and on the Dynamic Object Tab uncheck 'Blocks Passage (collidable)' check box. To add the flavour to the boulder fall go to Trigger Tab, check 'Open/Activation Sound' check-box and select a sound (you can use 'ambients\rock thud large.wav' for instance). Click OK.
11. Save your changes. And go in game to see what we have done. Now it won't kill you but you can pass through it which is not very convincing. Ok. You can always do something about it. For example: create a third Frame for our boulder which moves it out of the view below the ground and create several smaller rocks with 2 Frames each, place the first Frame of each of them under the ground and the second frame on the area just before the door, also make the door affect them all. Doing that you'll get your boulder split over the sturdy heads of the adventurers into several smaller pieces.
12. Now, what about a little damage to cheer the player up?
13. Go to the Triggers Tab. Add new Common Trigger using 'dummy01' as an example.
    a. Change name to 'trigBoulderPain' (for instance).
    b. Set 'Type' to 'Pain Trigger'.
    c. Check 'Works only Once' check-box.
    d. Check 'Position and Shape are inherited … or aren't required' check-box (we don't need any position for that type of Triggers).
    e. Set 'Usage Count' to 1.
    f. Uncheck 'Infinite' check-box.
    g. Set 'Power' to 1.
    h. Delete a variable from all variable lists and set variable combo-box to 'Do not modify any Location Variables'.
    i. Click OK.
14. Modify the entrance door and select Trigger Tab. Using menu ***Add New Link→Add Link to a Common Trigger*** select 'trigBoulderPain' Trigger. Click OK.
15. Save your changes. And go in game to see what we have done. Now your party will sustain some amount of damage when the boulder falls on them.

Using Location Variables

In this Tutorial Location Variables are introduces in the example. The base of this tutorial is the Dealing with Dynamic Object Tutorial 1.

1. – 12. See Dealing with Dynamic Object Tutorial 1.

There is another way to add the damage part.

13. Go to the Triggers Tab. Add new Common Trigger using 'dummy01' as an example.
    a. Change name to 'trigBoulderPain' (for instance).
    b. Set 'Type' to 'Pain Trigger'.
    c. Check 'Works only Once' check-box.
    d. Check 'Palpable' check-box so the Trigger will be activated when you are in its area.
    e. Set the radius to 5.
    f. Set 'Usage Count' to 1.
    g. Uncheck 'Infinite' check-box.
    h. Set 'Power' to 1.
    i. Delete a variable from all variable lists and set variable combo-box to 'Do not modify any Location Variables'.
    j. Click OK.
    k. Position the Trigger sphere before the door.
14. Modify the entrance door and select Trigger Tab. Set Location Variables combo to 'Set the Following Location Variables'. In variables list using menu 'Add Link to a New Variable' add a New Location Variable. Let it be 'varActivateBoulderPain'. Do not add the link to 'trigBoulderPain' Trigger in the Affected Entities List. Click OK.
15. Modify 'trigBoulderPain' Trigger. In the list 'Can be Activated/Deactivated only…' list use menu 'Add Link to a Variable' and select 'varActivateBoulderPain' and set 'Expected Value' to 'Is TRUE (Is Set)'. Click OK.
16. Save your changes. And go in game to see what we have done. Now your party will sustain some amount of damage when the boulder falls on them.

Thus Location Variables can be used by Common Triggers and Triggers within Dynamic Objects to implement depending actions.

Location Variables don't need to be declared somewhere. As soon as some Trigger sets or resets a Variable it is supposed to be declared and can be used by another Trigger to depend on.

Dependence on a Variable can be of two types:

1. Expecting it to be TRUE (to be set).
2. Expecting it to take on a value (0, 1, 2, etc.). Certain objects having models that have so-called Segments set their Variables to integer values rather than to TRUE/FALSE values.

Right now there is no way to make the object a Segmented one. But in the nearest future it would be possible. The Segment is a one or more Frames of the Object's 3D Model. In other words Segmented Objects have their Frames divided into one or more segments. Each activation of such an Object forces it to perform moving using only those Frames that comprise the current Segment.

Let's look at the example. Load the 'Upper Monastery' and find the 'microdial' Object. Modify it and select the 'Trigger' tab. You can see that it sets the 'Ding' Variable. Now select the 'microbutton' object and check out its 'Trigger' tab. You can see that it affects 'mico_blow' Trigger. Go to 'Triggers' tab and check out 'mico_blow' Trigger's properties. Note that it only can be activated when 'Ding' Variable is set to value 3. That means that only when 'microdial' object is at its rightmost position pushing the red 'microbutton' object will lead to 'mico_blow' Trigger's activation which in turn will activate 'glowball', 'lightning_03' and 'lightning_04' Dynamic Objects.

Let's return to the 'Dynamic Objects' tab and find these three Objects. Both lightnings do nothing special and play decorative role, but the 'glowball' object activates four more objects: 'BLASTRING' which plays decorative role, 'MonaCamShakeBox' which shakes the camera to add the flavour to the explosion, 'micor_door1' – the whole glass door to remove it from the player's view and 'micor_door02' – the broken glass door to place it before the player's view.

That's how all that small incident with the microwave oven is implemented using Segmented Object 'microdial' and several more Objects.

The only thing that remains unsolved is where the item 'Microwave Chip' is taken from when the microwave explodes? Unfortunately, this behaviour is hard-coded and we cannot reproduce it modifying the game data. To be more specific: Trigger 'micro_door2' within the Dynamic Object

‘micor_door02’ is expected to be in the game data files. If you delete this Dynamic Object than the Location won’t be loaded in game and you’ll see an error. That’s why the Golden Rule 2 is placed in this document.

But still there is a possibility to give party an item in this case. You just create one more Dynamic Object or Trigger with Loot that has the single item you want player to get and make a certain Object Activate that Object with the Loot.

Creating a Local (Separate) Sky for a Location

In this Tutorial we will learn how to create a separate Sky for a Location for a purpose to change the Sun texture, for instance.

1. Launch Location Editor, select a Location (let it be Lower Monastery) and select Sky Tab.
2. In the 'Local Sky File Name' edit box you see 'DefaultSky' – the file containing the Sky data should have this name and be placed into the 'Monastery' folder to be used by Lower Monastery as a Local Sky.
3. In the 'Current Sky File' edit box you can see what Sky is being used by the Location at the moment. It is 'DefaultSky.LVL' in the 'Test' folder. If you change the Sun texture right now all Locations (except for the Cosmic Circle) will get the new Sun. But what if we want this Sun to be in the Lower Monastery only?
4. Click 'Create Local Sky' button and select 'test\DefaultSky' as a template. Slick OK once Message Box appears claiming that Sky cloning succeeded. Note that if you want to create Sky just like in Cosmic Circle for your Location you should select 'Circle\CircleSky'.
5. Now you can see in the 'Current Sky File' edit box that the Location is using the Sky data file from the 'Monastery' folder. Now if you change the Sun texture it will affect only those Locations that have their data files in the 'Monastery' folder and their 'Local Sky File Name' equals to 'DefaultSky'. There is two such Locations: Lower and Upper Monastery.
6. Yes, you cannot create different Skies for LM and UM – this is the limitation caused by the fact that some info is hard-coded in the game.
7. Now, change the Sun's texture to something else and go check how it looks in game.
8. Now there comes another limitation. If you leave UM and get to Arnika Road you'll be surprised to see that the Monastery's new Sun shines over the Road as well. And this is regardless the fact that Arnika Road uses the DefaultSky from the 'Test' folder and we have not changed the Sun there.
9. What is wrong? The whole issue is in Local Sky File Names. If your party is being teleported from the Location A to Location B the Sky is reloaded when and only when the Local Sky File Names for these locations are different. It does not matter if the Skies used by these Locations have their data in different folders and are actually different – if the Local Sky File Names for the Locations A and B are the same the Sky would not be reloaded.
10. You can partially overcome this problem. But this is a bit awkward way (because the player will see two loading screens). The idea is to teleport to the interim level and than immediately to teleport to the place you intended to get initially. The interim location may be the Camp without Rapax (Wilderness Clearing), since it uses another Sky Name (CampSky). Thus the Sky will be reloaded.

Adding More Frames to 3D Object

In this small Tutorial we will learn how to quickly add more Frames to 3D Object. This tutorial can be applied while modifying a Dynamic Object or a Monster, Spell or Missile Model.

All our attention would be given to ‘Show Frames’ list. Apart from the determining which Frame(s) to show in the 3D Window it allows adding/deleting Frames.

The 'Insert New Frames' command does the following thing:
1. If a frame X is highlighted it inserts one or more frames (you select the number) AFTER Frame X and before the following Frame X+1 (if such exists).
2. At the same time it makes all those NEW frames so that the movement between frames X and X+1 is gradually divided among new frames.

For example: Frame X rotates object by 10 degrees, Frame X+1 rotates it by 90 (this rotation is not from frame X, but from the initial position), and thus we have quite a jerky movement for 80 degrees while going from Frame X to Frame X+1. Let's make it smoother. Select Frame X and add 7 new Frames. You will automatically get the following layout (no need to manually adjust each new Frame’s rotation):
Frame X rotates by 10 degress (we did not change it).
Frame X+1 (new frame) rotates by 20 degress.
Frame X+2 (new frame) rotates by 30 degress .
Frame X+3 (new frame) rotates by 40 degress .
Frame X+4 (new frame) rotates by 50 degress .
Frame X+5 (new frame) rotates by 60 degress .
Frame X+6 (new frame) rotates by 70 degress .
Frame X+7 (new frame) rotates by 80 degress .
Frame X+8 (the former frame X+1) rotates by 90 degress (we did not change it).

You get pretty smooth movement (rotation in this case). So when you create a rotating, moving, falling, etc. object you might want to make it having two Frames, adjust the last Frame so it takes the final desired position and then add as many interim Frames as you like. For complex movements you might need to apply this procedure more than once.

If you want add one or more frames AFTER the all existing ones select the last frame and use 'Insert New Frames' command.
If you want add one or more frames BEFORE the all existing ones select the 'All Frames' item and use 'Insert New Frames' command.
'Insert a Frame's duplicates' behaves similarly (as to where to insert new frame(s)) but it allows the selection of the frame which will be used to clone new frame(s) (by default it is the one you have selected) thus making all new Frames to be exactly as the cloned Frame.

Creating ‘NPC’ Dynamic Objects and Triggers

In this small Tutorial we will learn how to create Dynamic Objects and Triggers based on an NPC. The bright examples of such objects are: famous Trynton Fountain that bestows 5 Intelligence Points to the Party upon correct answer to the riddle and Mook Callisto Ship.

1. Open Location Editor, select ‘Trynton Upper Branches’ Location, select ‘Dynamic Objects’ Tab. Locate ‘Fount_Riddle’ object. This is the object that simply activates a Trigger named ‘_TRYNFOUN’.
2. You may select ‘Triggers’ Tab and find this Trigger there. Both this Trigger and the Dynamic Object do not have anything special that allows them to invoke ‘TRYNFOUN’ NPC upon activation.
3. Yes! You might have already noticed that the name of the NPC is ‘TRYNFOUN’ and the name of the Trigger is ‘_TRYNFOUN’! This is the KEY feature.
4. The underscore symbol at the begging of the Trigger’s name serves as a sign that this Trigger could trigger an NPC. Thus the Trigger (or Dynamic Object’s Trigger) should have the NPC’s name prefixed by underscore symbol.
5. Additionally (but not necessarily) you may specify the type of the NPC’s Script to use. There are 3 types of NPC Script: ‘NPC’, ‘RPC’ and ‘VOC’. By opening the NPC Editor Dialog you can see 3 Script Tabs – one for each Script Type. To specify the type of the NPC’s Script to use you should add ‘NPC_’, ‘RPC_’ or ‘VOC_’ just between the first underscore symbol and the NPC’s name (a good example of it is the Dynamic Object ‘_VOC_EWAXXLIFT1’ at ‘Umpany Base Camp’.
6. Do not mix the Dynamic Object’s name and the name of the Dynamic Object’s Trigger – it is the Trigger’s name that should meet those specific conditions.
7. Also do not mix the NPC’s name and the name of the NPC’s Script – it is the NPC’s name you should use as a part of the Trigger’s name.
8. Note: it seems that all other Trigger’s activities does not work when it invokes an NPC.

The meaning of this ‘discovery’ is hard to underestimate. For example now we can set/reset Facts by simply visiting certain places and then other ‘NPC’ objects or NPCs would ‘know’ about it. We can implement more ‘riddle pools’ with rewards (see the Intelligence 5 tutorial below). Unfortunately non-‘NPC’ objects cannot ‘see’ Facts and that’s quite a limitation, otherwise we would have full interoperability between all Objects, Triggers and NPCs in all Locations.

How to Increase Party's Intelligence by 5

In this small Tutorial we will understand how the +5 Intelligence can be gained not only via famous Trynton Fountain.

1. Study 'TRYNFOUN' NPC and check the command where the Fact 'FACT_TRYNNIE_FOUNTAIN_ANSWERED_CORRECT' is set to TRUE.
2. This command simultaneously increases Party's Intelligence by 5. This happens because it is hard-coded that when this Fact is set to TRUE your Party will receive the bonus. It does not matter if the Fact is already TRUE or not, so you can use that trick as many times and in many places as you want.
3. Now if you going to use this command somewhere you have to modify 'TRYNFOUN' NPC's VOC Script so it won't get hurt by your actions. What do I mean? It's simple: if player manages to get to your bonus place before the Trynton Fountain and you set the Fact to TRUE than when visiting the Trynton Fountain later in game player will get nothing because the Fact would be set to TRUE.
4. So you have to create a new Fact and change all Fact-dependent commands in the 'TRYNFOUN' VOC Script to use that new Fact. After that you must add a command that sets the Fact 'FACT_TRYNNIE_FOUNTAIN_ANSWERED_CORRECT' to TRUE in order to keep the Fountain's ability to increase Intelligence.

Unfortunately this will work with Intelligence only.
But it is better to have anything than nothing, isn't it?

Voice for New NPCs

<This Tutorial was written by Lexigon_5 and is presented here without any modifications.
Thanks go to Lexigon_5.>

Yes i have found a way for those of us interested in modding to get our newly created NPC,s voice activated.

The first point to mention is that this is not done through the MG Editor although the editor will pick up the changes just like with everything else.
Instead the work is done through that marvelous little tool "SLF-Explore" that Lana so kindly introduced me to.

Lets start with the actual RPC. You will create your RPC in the usual way using the MG Editor and we will take "Qwerty" from the Deathstalker 3.0 as our example.
If you open up the MG Editor and go into the NPC editor, then select Qwerty and go to NPC scripts tag.

When you look at any of the script blocks the first thing you notice is that the dialog is exactly the same as the NPC that she was modelled from. In this case Myles.
That dialog can be changed, but more about that later.

You will notice that there are two sets of dialog scripts. One in NPC scripts which relates to dialog that you have with an NPC outside of your party, and the other set in RPC scripts which relates to all the comments made by the NPC as a party member while on your travels.

In this respect you are a lot more limited with female NPC's as opposed to Male NPC's because we only have Vi Domina and Private Sparkle as female voices with both sets of NPC and RPC script blocks, but at least having one other than VI is something to work with.
When you highlight any of the comments made by a created NPC (lets say script block 13) and and press the "play quote sound" icon you will get an error message saying.

Sound for this file is missing!

File name is:
'npcs\NPC_QWERTY\NPC_QWERTY_13.mp3'

If you do the same in RPC scripts you get much the same but just substitute the last line

for
'pcs\RPC_QWERTY\RPC_QWERTY_13.mp3'

This is where the big clue is. We now know the names of the folders that our sound files will go into.

Now to the nitty gritty.
First of all we are going to go into our data\sound folder in the installed game, and then we add two new folders called "npcs" and "pcs".
Inside the npcs folder we put a new folder called "NPC_QWERTY" or whatever his\her name is.
Inside the pcs folder we put a new folder called "RPC_QWERTY" or whatever his\her name is.

We are now ready to go to SLF-Explore.
In the left hand box we navigate to our data\sound files and when we open up sound we see six folders including the NPCS and PCS folders.

We will do just the NPCS folder and then duplicate the process with the PCS folder.
You just go into NPCS and select whomever you want. We will say Private Sparkle for example, and just drag the "Gap" Folder along with all the mp3 files and drop them into our Wizardry 8\Data\Sound\npcs\NPC_QWERTY folder over on the left of the screen.
(no idea at this stage what the "gap" folder is for, the sounds work in the MG editor

without them, but since they belong in the NPC folder i just put them in anyway)

Here comes the painstaking part. You will get no sound in the MG Editor or the game until you rename each NPC_SPARKLE_0?? mp3 file to the name of your NPC. These files are all in MP3 format and work just fine with windows media player. (I am going to rename all the "Gap" folder files also, even if i don't know what they do).

The good news is that in my trial so far the spoken quotes match up to the relevent script quotes once you go back to the MG Editor to play the sound for the quotes and so you don't have to re-shuffle them all to get them in the right order.
Example: When i go to script block 7 for blinded Qwerty's blindness statement i get the text:
"DAMMIT!!! How can i aim without my sight??"

When i click on the play quote sound i get Private Sparkles voice saying:
"Oh my God,i can't see!!!"

without having to do any more work, and its like that all the way down the scripts as far as i can tell so far.
Of course we could use any of the many female voices in the pcs folder for the RPC scripts, but then we would have a silent NPC outside of the party, which is fine if you want your female NPC to be just a joiner.
Then again if you wanted an NPC that gave out a quest then they would have dialog that is not contained in the sound files and male or female they would be silent.
So this whole process is not without restrictions.

That is of course unless you have some serious sound recording software\Hardware that allows you to make high quality voice recordings which you can save into MP3 format. In which case you can make entirely new and unique dialog for any NPC you create as long as you name the file after your NPC and put it in the correct folder.
And yes it will work because i have tried it and had Qwerty wondering around speaking in my voice. The sound quality was awful but at least i proved it can be done.
So all you would be modders please keep your eyes peeled for some good quality, not too expensive soundware and get back to me in this thread if you stumble across any.
Forget about text to speech programmes though. there is just no character or emphasis in the voices of even the best programmes out there. I tried severel demo's and they are just terrible.
Its a case of getting together with a friend or two with some good soundware and some well thought out lines.

one last thing i should mention is that this whole process while not too complex and time consuming will take up quite a bit of space for your mod should you decide to go with it.
Roughly 5mb per NPC if you use both the NPC and RPC scripts, which makes just one new voice activated NPC bigger than all of the mods already out there bar Lana's latest 1.1 version.

Well i hope all this gives those of you wanting to mod some inspiration. It certainly has got me going now.

In case you all did not know there is now a new update for MG editor from July 10th. It is version 1.8.0 So it is only 2 weeks old.
This new update slightly changes my tutorial in the sense that you can now assign sounds to your NPCs from within the editor, once you know where to locate them from, and the editor will now create those folders i mentioned and put them in the data\sound folder just the way i described.
You will still need SLF-Explore to locate and extract the speech files into a temp folder so that you can access them via the editor though as the editor won't enable you to navigate into the Sound.slf data file to assign the sounds.
Funnily enough i have actually found that it is faster to just copy them all into the correct sound folder and rename them one by one, the way i was doing it allready, but for just the odd sound selection as you come to it this fix is useful.

Try using ***Miscellaneous→Rummage W8 File Collection...*** menu!
Maybe it is not as handy as 'SLF Explorer', but it does its work pretty good.

Using Planar Water Triggers

In this small Tutorial I draw your attention to Water Triggers.

Load ‘Mt. Gigas Upper Caves’ Location. Go to ‘Triggers’ Tab and find ‘waterTriggerPlane’ Trigger. Note its position and its Shape Type. Those Triggers are used to specify the air-water borders. Whenever the Party crosses such Trigger the ‘Underwater’ flag is reversed.

You can use such Trigger in Lower Monastery to make additional Underwater area where I bet you always wanted to get to. You’ve got first to delete a couple of Physical Faces ‘guarding’ the area though, but this is now not a problem at all.

What is really a problem is how to make water double-side in LM. I thought I know how to do it but I don’t. Making a corresponding material double-sided does not work for some reason. I checked other Locations and even in LM they have two equal Terrain Faces for such things: one is visible from one side and the second is visible from the other side.

So for now you have to create a Dynamic Object based on the Terrain Meshes and this would be the ‘water surface’ you’ll see from under the water.

MLS (Monster Model Script) File Information

Note that missiles and spells using the similar MSL scripts describing the behaviour of their 3D models though they use different Keywords. This Tutorial describes only Monster Model Scripts.

Some notations:

<> are used to denote mandatory elements.
[] are used to denote optional elements.
Words without brackets are Keywords.

Symbol # serves as a comment in the MLS file and effectively comments the whole line.
It is a good idea to start each Keyword/Variable from the new line.

The table below displays some variables that can be specified for the all models that are used for various actions of a single monster and are united in the single MLS file.
The meanings of some variables are not specified in the table (though some of them appear pretty obvious) because I'm not sure of what they really do and do not have time to thoroughly test it. If you are sure you know what this or that variable (keyword) does just let me know and I update this document. Thanks.

Keyword	Function
crawls	Determine the creature's type of movement (if none is specified then it is a humanoid type of movement)
swims	
flies	
quadruped	
hoverrangestart <float> **hoverrangeend <float>**	Adjust Z coordinate(height) positioning according to values, positve and negative allowed. Mainly used for flyer/swimmer
bobrangestart <float> **bobrangeend <float>**	Determines range of height variation during anim play (note: negative values untested)
scalerangestart <float> **scalerangeend <float>**	Go together and determine the scaling of the original Model. Range allows the variable monster size which depend on the min & max monster hitpoints and based on the hit points with which the monster was generated
scalefactor <float>	Determines the scaling of the original Model.
movementrate <float>	Determines how fast and how far the engine moves a monsters [WALK] animation. movementrate + number of frames + framerate determine monsters number of steps for distinct distance on ground
rotationrate <float>	Determines animations rotation speed, e.g. when turning to attacks from other side, or 180 deg turn of patrols.
walkradius <float>	?
fightradius <float>	Determines minimum approach distance of party and monster. Adjust if monster is pentrated by cam, or appears too far away though already at short combat range.
targetheight <float>	Determines focus of attacks, respectively where dmg numbers/particles are generated.
cameraheight <float>	Point in 3D view from where monster is shown ?
deathscale ?	Adjusts scaling of death animation assigned in .mls script
sound_falloff ?	?
missilestart ?	Determines start frame for actions with missiles, required if any misssile is assigned to a monsters attacks/etc. May not exceed max frame number of all inflicted anims.
spellstart ?	Determines start of spell animation, required in case monster has spells assigned. ([SPECIAL] + [SPECIAL_ATTACK] are still unclear)

	May not exceed max frame number of all inflicted anims.
randomidlefps ?	?
shadow ?	As indicated adds shadow;)
opacity ?	?
addlight ?	Adds variable light, usually with parameter "pulsing" and accompaigned by keyword: GLOW. Example: slimes + elementals
glow ?	
lefthanded ?	Determines chance of monster to appear left-right mirrored
fulltransition ?	? Annotation: Only Bela uses [FULLTRANSITION]
spicemonster ?	?

Annotations:
Values for [walkradius], [fightradius], [cameraheight], [targetheight] [hoverrangestart/end] and [bobrangestart/end]
appear to be independent from changes on scalefactor, very probably refer only to X,Y,Z values in 3D editor.
[movementrate] appears connected 1:1 to framerate,
meaning if framerate of a smooth walking anim is raised also movementrate should be raised the same factor.

The following **case sensitive** keywords determine the model for the corresponding monster's action and optionally a FPS value (each 3D model file has its own FPS, but FPS in MLS files override model's value). We will call these 'Action Keywords'

ATTACK_BASH	**ATTACK_SPECIAL**	**DODGE3**	**SPICE2**
ATTACK_CLOSE	**ATTACK_SWING**	**GET_HIT**	**SPICE3**
ATTACK_CLOSE1	**ATTACK_THROW**	**GET_HIT1**	**TALK**
ATTACK_CLOSE2	**ATTACK_THRUST**	**GET_HIT2**	**TALK_SPICE**
ATTACK_CLOSE3	**BIRTH**	**GET_HIT3**	**TALK_SPICE1**
ATTACK_KICK	**DIE**	**GET_HIT4**	**TALK_SPICE2**
ATTACK_LASH	**DIE1**	**IDLE**	**TRANSITION**
ATTACK_MELEE	**DIE2**	**SPECIAL**	**WALK**
ATTACK_PUNCH	**DODGE**	**SPELL**	
ATTACK_RANGED	**DODGE1**	**SPICE**	
ATTACK_SHOOT	**DODGE2**	**SPICE1**	

The format is as follows:
<Action Keyword> <model name> [FPS]
The different ATTACK_XXX keywords correspond to the various actions chosen for a monsters attack types.
The **SOUND_FRAME** Keyword controls sounds during monster's action.
The format is as follows (the <number> probably means the Frame index (zero-based, i.e. starts from zero):
SOUND_FRAME <Action Keyword> <number> <monster sound file name>

The **SOUND_FOOTSTEP** Keyword controls footstep sounds during monster's action. The nature of the sound is probably determined by the physical floor material the monster is standing at.
The format is as follows (the <number> probably means the Frame index (zero-based, i.e. starts from zero):
SOUND_FOOTSTEP <Action Keyword> <number>

SKIN DEFAULT Keywords opens section where you can override Textures specified in the 3D Model file for usual monster state (not wounded). It goes without parameters – it just opens a section in the script file.

SKIN Keyword opens the next Texture override section (for the next monster state: wounded/heavily wounded). It goes without parameters – it just opens a section in the script file.

Note that all three SKIN sections must be present in the file as long as you swap at least one Texture.

The skinswap in the 2 SKIN sections refers to the override textures in SKIN DEFAULT section if any are specified.

Within each skin section you can specify any number of **SKINSWAP** keywords which tells the game which Texture to replace with which Texture. The format is as follows:
SKINSWAP <original texture name> <new texture name>

MSF (Monster Behaviour Script) File Information

Some notations:

<> are used to denote mandatory elements.
[] are used to denote optional elements.
Words without brackets are Keywords.

Symbol * serves as a comment in the MSF file and effectively comments the whole line.

The information below displays some commands that can be used for various actions of a Monster. The meanings of some commands is not specified (though some of them appear pretty obvious) because I'm not sure of what they really do and do not have time to thoroughly test it. If you are sure you know what this or that variable (keyword) does just let me know and I update this document. **"E: file_name.msf"** means that example of using a certain command can be found in that file.

Note that it is the **BEST PRACTICE** is to leave **single spaces** between commands and parameters otherwise commands might not work!

BEGINORDERS
ENDORDERS
I'm not sure why some of the scripts have those lines and other don't. Maybe for repetitious actions – those which can be finished and you want to start them over again (in other words – not one-shot).

POINTPATROL <spaces_delimited_list_of_targets>
* Cannot be used ***outside*** of BEGINORDERS ENDORDERS block!
This one forces Monster(s) walking through the list of specified targets. Targets could be **home**, **off_camera**, **<NPP_name>**.
E: arnikaguard.msf

RANDOMPOINTPATROL <spaces_delimited_list_of_ targets>
* Cannot be used ***outside*** of BEGINORDERS ENDORDERS block!
This one is the same as POINTPATROL, but probably the next NPP is chosen randomly each time.
E: bankguard.msf

WALKTO <target>
Orders the Monster to go to the specified target. **<target>** can be **home**, **off_camera**, **<NPP_name>**. If **PARTY** is specified instead of NPP then the Monster will walk to Party.
E: belapath1.msf, walktoparty.msf

DOACTION <action_name>
Those are hard-coded in the game and do various complex shit including assigning another behaviour script. Maybe one day *Cosmic Forge* will be able to allow using those actions.
List of currently hardcoded actions: **STARTGOLEMATTACK**, **ENDSAVANTWALK**, **UNLOCKUI**, **BELA_END_CC_WALK**, **ENDGARIWALK**, **ENDHOGARWALK**, **ENDHOGARWALKANDPUTTOSLEEP**, **ENDBELAWALK**.
E: belapath1.msf

PATROL < floating_point_number1> < floating_point_number2>
* Can be used ***outside*** or ***inside*** of BEGINORDERS ENDORDERS block!
This one forces Monster(s) random (is it) patrolling around the spot.
E: closepatrol.msf

DEAF
* Can be used ***outside*** or ***inside*** of BEGINORDERS ENDORDERS block!
Maybe tells Monster(s) to ignore the surroundings (battles).
E: closepatroldeaf.msf

TURNTOFACEPARTY
* Cannot be used ***outside*** of BEGINORDERS ENDORDERS block!
Tells Monster(s) to face the Party.
E: closepatrollook.msf

STAYHOME
* Cannot be used ***outside*** of BEGINORDERS ENDORDERS block!
Tells Monster(s) to not to move from their original spot.
E: faceeast.msf

GUARD <target>
* Cannot be used ***outside*** of BEGINORDERS ENDORDERS block!
?
<target> can be **home**, **off_camera**, **<npp_name>**.
E: guard.msf, npc_panrack.msf

FACE <direction> [NOBLOCK]
* Can be used ***outside*** or ***inside*** of BEGINORDERS ENDORDERS block!
Tells Monster(s) to face the certain direction.
When ***outside*** **<direction>** can be: **home**, **off_camera**, **<NPP_name>.**
When ***inside*** **<direction>** can be: **PARTY, EAST**, **NORTH**, **SOUTH**, **WEST**, **NORTHEAST**, **NORTHWEST**, **SOUTHWEST** or **SOUTHEAST**.
E: faceeast.msf

DISPOSITION <disposition>
Changes Monster(s) disposition. **<disposition>** can be **DISP_NEUTRAL**, **DISP_FRIENDLY** and **DISP_HOSTILE**.
E: harassguard.msf

SAY <number>
Tells Monster to say the Script Quote from the specified Script Block. Monster should obviously be NPC.
E: harassguard.msf

STOPPATROL
Looks like it tells Monster(s) to stop patrolling if they were on patrol.
E: harassguard.msf

STOP
? Stops any movement?
E: walktoparty.msf

LOOKABOUT <frequency> <duration>
* Cannot be used ***outside*** of BEGINORDERS ENDORDERS block!
? Maybe forces Monster(s) to look around (using some animations?).
E: lookabout.msf

FADEOUT
Effectively removes Monster from the Location by fading it out. Cool!
E: m1_trang_trans.msf

DISAPPEAR
Removes Monster from the Location.
E: movelavalord.msf

PLAY <number1> <floating_point_number2> <sound_file>
Plays specified sound with some parameters.
E: movelavalord.msf

NPCINTERACTION [<number>] [SUPPRESS]
Initiates NPC Interaction (just if you clicked the Monster to talk to it). **<number>** might mean the starting script block? **SUPPRESS** is optional (suppresses something, I guess).
E: npc_aletheides.msf

CYCLE <animation_sequence>
Tells Monster to play a certain Animation Sequence (MLS Action) (see MLS File Information about Actions)
E: npc_savant_henchman.msf

END
? Maybe finishes the script execution?
E: npc_aletheides.msf

Other Commands

SHOOT <missile_id> <target>
Supposedly shoots the specified Missile at the specified target. Sounds like fun stuff.
<target> can be **home**, **off_camera**, **<npp_name>**.

GIVE <item_ name>
The command gives the specified Item to Party. **<item_ name>** here is a bit dodgy: if an Item has 'custom model name' and it matches then this item will be given, otherwise it compares Items' actual names. Be careful if your game is not in English – a conversion of wide char Item's name into multibyte name is performed in the code and then compared to the given name.

TELEPORT <target>
The command teleports the Monster to the specified target.
<target> can be **home**, **off_camera**, **<NPP_name>**.

CAST
* Cannot be used ***outside*** of BEGINORDERS ENDORDERS block!
Some spell is cast? By ID or by Name?

DIE
It looks like there are no parameters. Monster just dies?

LOOKHERE <parameter>
This one is not clear. **<parameter>** can be **NOBLOCK**.

NPCNUMBER
* Cannot be used ***outside*** of BEGINORDERS ENDORDERS block!
This one is not clear.

FOLLOW
* Cannot be used ***outside*** of BEGINORDERS ENDORDERS block!
This one is not clear.

TRIGGER <trigger_name>
The command triggers the specified Trigger.

DELAY <floating_point_delay>
The command supposedly delays something – maybe further behaviour script execution?

Some coding practices for MSF files

<label>:
A label to where execution of the script might jump if certain condition is satisfied.
E: harassguard.msf

GOTO <label>
Moves code execution to the specified label.
E: harassguard.msf

IF <condition>
[commands]
ENDIF
A standard conditional expression. When **<condition>** becomes TRUE then **[commands]** are executed.
E: harassguard.msf

IF <condition>
[commands]
ELSE
[other_commands]
ENDIF
A standard conditional expression. When **<condition>** becomes TRUE then **[commands]** are executed otherwise **[other_commands]** are executed
E: harassguard.msf

PARTYNEAR(<distance>)
Expression returns TRUE if Party is within the specified distance.
E: neutraltalkingguard.msf

RANDOM(<distance>)
Expression returns TRUE if random returned even and false if odd?
E: neutraltalkingguard.msf

NOT
Can be put in front of an expression to reverse its value.