The NGLE Encyclopedia

by AkyV

Have you been editing your Tomb Raider levels with the original/official TRLE or TREP or anything else? Are you planning to use TRNG? Would you like to know what the new features will be (compared to TRLE!) if you use TRNG? Perhaps you've just installed TRNG, and you'd like to know what your new possibilities are here? Then this is a tutorial for you. In this tutorial I will show you (put in a nutshell) the TRNG features.

Download TRNG from Skribblerz Tools page.

This is what you count on in the tutorial:

- The tutorial is based on the installation of the original Mk4 installer that contains the 1.2.2.6 version of TRNG and the 1.03.039 version of NG Center.
- All the features in this tutorial are "touchable" features (new buttons, new triggers etc). So the tutorial doesn't contain the kind of information such as "TRNG has bigger memory than TRLE if you do this and this" etc.
We also don't mention now the bugs that are presented in TRLE but have been fixed in TRNG. (For example, the petrol will disappear in TRLE from the scale of ELEMENT_PUZZLE object if you save the game and load that. This is fixed in TRNG.)
And the new type Room Editor error messages are also not included.
- If you encounter a feature (when editing with TRNG) that is new for you but you don't find that feature listed here that probably means the feature can also be used in the original TRLE.
- This tutorial won't discuss Paolone's TR Level Manager.

1. The installation of TRNG

Back to Top

The TRNG (Tomb Raider Next Generation) is not an independent program pack. I.e. when you're running the Mk4 installer then Mk4 will install TRNG into the main folder of the original TRLE.

2. The main parts of TRNG

Back to Top

When TRNG has been installed then you will see the contents of the TRLE main folder have been changed:

Programs:

- Tomb4.exe has been updated. (You can change the icon of Tomb4.exe in NG Center/Tools tab, using Icon Replacer function.)
- The winroomedit.exe has been removed. The Room Editor for the TRNG is the so-called NGLE (Next Generation Level Editor): ngle.exe.
- Tom2pc.exe has been removed. The Level Converter for the TRNG is the NG_Tom2PC.exe.
- Tools folder has been created: there are some nice little tools here to support building levels.
- When you were running Mk4 then you also installed the so-called NG Center (not in TRLE main folder everyway). This is the tool for scripting with TRNG. I.e. you can't use Script.bat to make the Script any more.

Other files:

- New DLL files have been created.
- Though you can't see any of them now, a lot of TXT files can be created when you're using TRNG. These files will be automatically saved in a logical place. For example, if a TXT has been created when you've been using the NG Center, that will be saved automatically in NG Center main folder. (A MEM file will also be created with each game crash TXT report.)

3. Updating TRNG

Back to Top

The TRNG can be updated.
The newest update patch for TRNG will pop up on Skribblerz or Paolone's official TRNG site – if Paolone has made that patch. (Room Editor Drop Down Menu Bar/Help/Check for Updates – another method to download Paolone's site.)
Or you can download the patch directly via NG Center (on Updates tab, with Check for available Updates button).

If you want to use an older patch (instead of the freshest one that is installed) then click on Change Version button in NG Center, on Settings tab.

4. Compatibility with TRNG

Back to Top

- With the original TRLE project files: complete. (Almost. Sometimes you can find small bugs.) Project files from NGLE to TRLE: limited. (Naturally. I mean the new TRNG features of a project edited by NGLE won't show up in TRLE or will show up in an improper way.)
- With TREP: you can use an older version (1.0.9.13.) of NGLE Room Editor (downloaded from Skribblerz Tools page) for TREP. (Moreover, you can use Drop Down Menu Bar of Room Editor, choosing Help\Object IDs for Trep – updated at each Output WAD operation.)
- With WADMerger: when installing/updating TRNG, the objects.h file will be updated both in TRLE folder and WADMerger folder. (Use 1.98.0.103 exe of WADMerger if you want to use v130 WAD files.)
- With TRViewer: replace the original TRViewer.exe by the TRViewer.exe of the Tools folder.
- With StrPix: replace the original StrPix3.exe by the StrPix3_v130.exe of the Tools folder, if you want to use v130 WAD files.
- With other programs in Tools folder: these are created by the Mk4 installer, so they are compatible, of course.
- If you want to know more about the so-called NG Header (that may cause compatibility problems) then see NG_Stripper.exe and its Readme in Tools folder.

The compatibilities with other programs are not controlled by TRNG.

5. NG Center

Back to Top

The NG Center is more than the scripting tool for TRNG, i.e. it has five main tasks:

- You can make the script for your game on Script tab.
- You can edit the strings for your game on Strings tab.
- Reference tab contains much information you need: the commands in Script, the OCB values of Moveable objects etc.
- On Tool or Tool2 tabs you can open some nice little tools to help your level building. (It has nothing to do with Tools folder.)
- On Updates tab you can start an update on TRNG.

Additional information:

- See the info about the actual NG Center version here: NG Center\Updates.
- NG Center has some additional functions for the tasks: printing the text in TXT files, finding texts, printing the fields of the actual Script command in a little yellow pop-up window, bookmarks in scripts etc.
- NG Center will be always on the uppermost program on the screen, checking “Keep always…” option on Settings tab.

6. Scripting on NG Center Script tab

Back to Top

Edit your script in the main window.

The buttons:

- Save button will save Script.txt (but won’t update Script.dat).
- Build button will save Script.txt and will update Script.dat and your language DAT.
- Reload button: this button will solve the problem if you edited the TXT’s of the NG Center directly in the TXT, and not here in NG Center.

Extra functions:

- Type the Script command with the equality sign and hit F1. If the first field of the command is the ID number of the command then the next free ID will be typed now automatically.
- Use > sign to divide the command into more parts if your Script command is too long so that won’t fit the screen. – So use it:

```
Command=   Field1, Field2, Field3, >
        Field4, Field5, Field6
```

instead of this:

```
Command=   Field1, Field2, Field3, Field4, Field5, Field6
```

- Some entries on the Reference tab have decimal and hexadecimal numeric ID. You can use (any of) them in the Script, instead of typing the text of the entry in the Script. For example, choose “MNEMONIC CONSTANTS” list, and see CUST_FLARE constant. As you see, it has 40 decimal and $0028 hexadecimal ID – you can use them instead of the “CUST_FLARE” text, when you want to use CUST_FLARE constant in the Script.
- Some constants can be attached to each other, with the + sign. For example, type FO_ENABLED+FO_LOOP, if you want to use both FO_ENABLED and FO_LOOP in an Organizer Script command. It will also work with their ID’s: 1+2 or $0001+$0002. Moreover, sum the values, if you don’t want to type all of them. (So, for example, you can type 3 instead of 1+2.)

- If you don't want to use a field of a Script command, then type IGNORE constant there. Or its -1 or $FFFF ID number. (Except: never type an ID number instead of "IGNORE" text if that field is a textual field. See for example Texts and NumTexts fields of the StandBy command. Now type * sign if you don't want to type IGNORE.) – A field with IGNORE means the feature of the field will be ignored or will use its default value.

7. Editing strings on NG Center Strings tab

Back to Top

Selecting your language:

- Select your language TXT file at Language box: the contents of this TXT will be shown now here.
- Select your language TXT file at Main Language box: the DAT file of this TXT will be used by the game now.

The main parts of the strings:

- The "classic" contents of the language txt file have been split into three sections in NG Center: Strings, PSXStrings, PCStrings. It's easiest to identify the "classic" strings now, because they have ID numbers in NG Center.
- ExtraNG is an empty folder for your new strings, used by new, TRNG features. Use Add button to fill the folder with newer and newer, empty string entries. And use Remove button if you want to remove an unused, unnecessary string.

Editing strings:

To edit a string, select it in the main window, then type its new text in the lower window. Click on Update button to update the text of that string and to save your language TXT.

Notes about editing ExtraNG:

- Sometimes you need to type an Extra NG entry, word for word, into a Script textual field. You can skip that, if you don't type the text, but type the ID number of the entry, with an exclamation mark. (For example, !132 means "the contents of ID132 Extra NG String".)
- You don't always need to type the text for an ExtraNG string into the string. Instead of that, you can type that text into a TXT file, then save the TXT in Script folder. Then type the TXT name with @ sign into that string. (For example: @Text.TXT.)
If you click on Open button when a TXT type string like that is selected, then the TXT will open.

Text conversion:

Convert texts to new format (on Tool tab): converts your old strings into ANSI character set, because that is what TRNG supports.

8. Technical functions in Room Editor

Back to Top

Functions from the keyboard:

- CTRL+M: closing/opening Room Editor to/from the Tray.
- CTRL+SHIFT+I: crashes intentionally the frozen editor.

Settings window:

- SHIFT+F2 or Settings button or Drop Down Menu Bar, choosing Shapes\Set Random Smooth Parameters: it will open "NGLE Settings" window to do some technical settings.

- Some settings in NGLE Settings window (see more about them below):

= Maximize NGLE window at start: check it and when you're opening Room Editor, the editor window will open as if you hit ALT+ENTER as well (to show the very bottom part of the editor window).
= Disable Memory Extender Patch: check it if you have problem with your computer configuration, which is too low for Room Editor.
= Disable New Stand-by System: if this option is not checked then your computer will be burdened less if you're just not using the running editor.

Enhanced panels:

Though I won't really mention it in this tutorial, you will encounter enhanced panels in Room Editor. See, for example, how the panel of "Find Object" command looks. – The general features of the enhanced panels:

- Use the Yes/No options or SPACE to change between the alphabetic and the numeric order of the list.
- Hit an alphabetic key to select the next list entry that starts with that alphabetic character.
- Hit F1 to open the actual contents of the panel in a TXT file.

Technical information:

- See the info about the actual TRNG patch here: Room Editor Drop Down Menu Bar/Help/About NGLE or here: NG Center\Updates.
- Changes (in Room Info Box): each step - when you're editing – is counted here. (Each start of the editor or each time when you load a project will turn it back to 0.)
- You can see some information about some operations (in the bottom left corner of Room Info Box) if that operation has just happened.
- Room Editor Drop Down Menu Bar/Help/Keyboard Commands to see the new keyboard commands for NGLE Room Editor.

9. Screenshots

Back to Top

In Room Editor:
See Drop Down Menu Bar, choosing an entry of Screenshots menu (or a key combination you see there) to take a screenshot of a part of the Room Editor (placed in TRLE main folder).

In the game:

- Hit F3 so the game will put a Shotxxx.bmp screenshot in TRLE main folder. (xxx is the ID number.)
- If you want to take not only a shot but a range of shots when hitting F3 then use a CUST_SCREENSHOT_CAPTURE constant in a Customize Script command. (See QSF constants for the resolution.)

10. Shortcuts for some main programs

Back to Top

In NG Center:
Click on the colored buttons on the bottom right part of the Script tab. (Some paths for the programs must be defined by you, on Settings tab.)

In Room Editor:
Click on the little arrow on the bottom of the editor, next to the text window, to open the list. Select the program, then click on Go button next to the arrow (or hit SHIFT plus the number key

you see next to the name of the program). (The paths of the programs must be defined by you, in NGLE Settings window, clicking on Settings for launch of external programs button. – See more details on the panel that opens when clicking on the Settings for… button.)

11. Project files and level conversion

Back to Top

Opening the actual projects in Room Editor:

- CTRL+L or Last Projects button (replacing "Load Objects" button of TRLE): to open a project you've used lately.
- Reload last project at start (in NGLE Settings window): check it and Room Editor (just has been launched) will open a project you opened last here.

Backup procedures in Room Editor:

- Remove 'autosave.prj' also with normal exit (in NGLE Settings window): check it and autosave.prj will be removed when you're closing Room Editor by clicking on the "X".
- Backup files for your project (saved in the same folder where the project file is):

= see Auto-Backup (in NGLE Settings window) to create backup files automatically, or
= click on Backup Project button to create backup files manually. (The editor also lets you to create a backup when the editor crashes.)

Level conversion:

- You don't need to accomplish the "usual" converting procedure (save project – output WAD – convert level), Room Editor will do that instead of you, if you click there on Play or Exit&Play buttons. (The difference: Play button will close the editor on the Tray, Exit&Play will exit the editor.)
- Save image of map after output wad command (in NGLE Settings window): check it and the actual horizontal 2d Map (as a Last_2dMap file) of the level will be put into TRLE main folder when outputting a WAD.
- New features of the level converter:

= Statistics button: opens a TXT log file about the output information.
= Crypt Tr4 files option: still not active.

Technical information:

- Open a project and the "Room Editor" title will be replaced by the actual TRNG patch version number and the path of the actual project.
- If you have a "MISSING" error message in the Select Level list of your game that means you have a [Level] block for that level in the Script but you don't have a TR4 level file for that level.

12. Exploring the map in Room Editor

Back to Top

Customizing old functions:

- Now you can use not only Drop Down Menu Bar of Room Editor (Room/Center), but SHIFT+C as well to put back the room vertical axis into the center.
- If the spin/zoom is too fast, then also hit SHIFT with the cursor keys/PAGE UP/PAGE DOWN, to slow it down. (See Settings for Spinning and Zoom in NGLE Settings window to customize the spin/zoom speeds.)
- You can customize the size of the window of the Preview mode, in NGLE Settings window (see: Preview Screen).

New functions:

- V2d Map button: click here to see the vertical 2d Map. Choose the map direction (north/south/east/west) next to that.
- Selecting a room from the list of “the rooms that have been edited lately”:

= G or Go back old room button: taking one step backwards in the list to select a new room.
= CTRL+G: taking one step forwards in the list to select a new room.
= M: the just selected room will be put into the list, though you won’t edit it now.
= U: removing the just selected room from the list. (The oldest room in the list will be removed automatically, if the list becomes longer than 10 rooms.)

13. Triggers

Back to Top

The new contents of Set Trigger Type (STT) panel:

New buttons in STT:

- Hide Constants List button switches on/off the trigger names, so you can see the triggers identified by their ID numbers. (For ACTION and CONDITION triggers in Window &, for FLIPEFFECT triggers in Window #. Other triggers don’t have ID’s). - You can also see the ID of the selected trigger on the top of the STT.
- Export AnimCommand button enables you to use the required trigger as a WADMerger AnimCommand attached to a frame of an animation of a Moveable object. (Only with ACTION or FLIPEFFECT triggers.)
- Export Script Trigger button enables you to use the required trigger in the Script. (Only with ACTION, FLIPEFFECT or CONDITION triggers.)
- Find buttons are trivial, but it’s worth mentioning P buttons: hitting them, the actual contents of some windows (#, &, E) next to the buttons will open in TXT files. (In the TXT’s you can find the ID’s of the triggers as well.)

New triggers in STT:

- ACTION triggers enable you to do special operations with (mostly) Moveable objects.
- CAMERA triggers now can be selected for the placed Cameras and Fixed Cameras.
- There are plenty of new FLIPEFFECT triggers to trigger Lara or something that is (mostly) not a Moveable object. (The FLIPEFFECT triggers known from TRLE have OldFlip name now. The ID is the same: for example, you used FLIPEFFECT 28 in TRLE to adjust fog bulb colors – and the ID of that trigger is 28 in NGLE as well.)
- FMV triggers naturally for triggering FMV’s.
- PARAMETER triggers (and some OBJECT triggers) to define the subject of CONDITION type triggers.
- SINK triggers now can be selected for Sink bulbs.
- TIMER_FIELD triggers will solve a problem: what if you have two triggers with overlapped each other, and one of them uses Window & for timing, but the other one uses Window & for something else. Naturally, the rules of overlapping triggers don’t allow that. The solution: use Window & for the “something else” trigger, type 0 in Window & of the timed trigger and then overlap a third trigger with them. This newest trigger is a TIMER_FIELD trigger to define the timer value for the timed trigger.
- The useless BODYBAG and CUTSCENE triggers have been removed.

New types of triggers in STT:

- CONDITION triggers will be overlapped with the executable triggers: the executable trigger will be executed only if the condition is true.

- MONKEY trigger has been removed – but don't worry, now "if Lara is monkey swinging" is one of the CONDITION triggers.

New box for triggers in STT:
Some triggers will also use Window E, because the other four boxes (Window Trigger, Window #, Window Type, Window &) are sometimes not enough for the exact definition of the trigger.

Some help to use the tutorial:

- Abbreviation of some triggers in the tutorial: A – Action, C – Condition, F – Flipeffect. (For example, F118 means "FLIPEFFECT trigger with ID 118 number".)
- You don't know what the exact meaning of F118 trigger is? Well, then switch on Hide Constants List button, select the trigger (ACTION, FLIPEFFECT, CONDITION), then type the ID of the searched trigger in the trigger ID window (A and C – Window &, F – Window #). After that, switch off Hide Constants List button – and the trigger with that ID will be selected, so you can identify that.
(Anyway, you can use these windows to choose some other feature with that method. For example, hide the names at F65 trigger with the Hide Constants List button and then choose another string ID in Window & to choose another string for the trigger – which will be realized when you switch off Hide Constants List button.)

Exportable triggers instead of non-exportable triggers:

- Maybe you'd like to export triggers (either as an AnimCommand or as a scripted trigger) even when you don't have the possibility for that: FLIPMAP, FINISH etc. triggers. Well, you HAVE the possibility for that. All you need to do is to find the proper ACTION or FLIPEFFECT trigger.
For example, A43 trigger does the same as when you just want to trigger an object in the usual way.
- And sometimes it's worth PLACING (so not exporting) those "substitute" triggers:
For example, see A44. That trigger will do the same as when you stop an object by an ANTITRIGGER. Special triggers can't be overlapped, so, eg. you can't overlap that ANTITRIGGER with a HEAVY to change that ANTRITRIGGER into a HEAVYANTITRIGGER. But if you place an A44 TRIGGER then you can overlap it with a HEAVY, so that TRIGGER will work as a HEAVYANTITRIGGER.

The TriggerGroups:

The TriggerGroups are triggers in the Script.
I mean, the TriggerGroup is a Script command that contains one or more triggers that are defined by the Export Script Trigger button. - We can have some purposes to use TriggerGroups:

- You can save the amount of the placed triggers. So, for example, if you have eight overlapped triggers then you can use only one trigger instead of them. That one trigger is an F118 (usually use that in "Single" mode) that activates a TriggerGroup. The F118 identifies the TriggerGroup by an ID number. Find the TriggerGroup with the same ID in the Script to know which TriggerGroup will be activated if that F118 will be activated. That TriggerGroup contains the eight triggers. (To understand the trigger order in the TriggerGroup, read this: the highest trigger at the overlapped triggers – this is the first one that will be selected if you click on the square of the triggers – is identical with the first trigger in the TriggerGroup.)
- TGROUP_SINGLE_SHOT constant of the TriggerGroup command makes the TriggerGroup work as a "One Shot" trigger – which is good because the "real" One Shot works (properly) only with activating objects in the usual way. (Activating F345 trigger will make possible to activate once again a TriggerGroup with TGROUP_SINGLE_SHOT flag.)
- Some special Script commands use one or more triggers. Those triggers are defined in a TriggerGroup. That Script command will apply the ID of that TriggerGroup to know which triggers will be used.
- Some triggers will work properly only exported in TriggerGroups. (See the description of

cameras.)
- You can define even complex conditions in TriggerGroups. (See more about it below.)
- If you have a trigger exported in a TriggerGroup Script command then the TGROUP…INDEX constants (attached to the triggers in the Script) will change the real subject of the trigger into some actual subject.
(For example see TGROUP_USE_EXECUTOR_ITEM_INDEX: use it if the F118 will be activated by a HEAVY. The trigger in the TriggerGroup for example is an ACTION trigger to kill a dog with ID23. But you want each enemy to be killed, stepping on that HEAVY trigger. That's why you use a TGROUP_USE_EXECUTOR_ITEM_INDEX constant, attached to the trigger in the TriggerGroup. This constant will overwrite the ID in the trigger. The new object ID of the trigger will always be the ID of the object that is just activating that HEAVY. – As you see, after all, that is not important now, if the subject of the trigger is the dog with ID23 or any other enemy.)

14. Rooms and room properties

Back to Top

New rooms in Room Editor:

- CTRL+ALT+SHIFT: defines the actual maximum number of room slots.
- New Up, New Down, New Tunnel Room buttons: creates a new room and connects that at once to the selected room.

Room properties adjusted by Room Editor buttons:

- Clicking on Normal button again and again, the text on the button will change: Water, Snow, Rain, Q-Sand. So this is the way to create water, snow, rain or quicksand (swamp) in the selected room. Snow, rain or quicksand also has the intensity, shown in the well-known water intensity box.
- Clicking on DMG button: now the selected room will be a so-called damage room, in which Lara's health will be permanently decreased (shown by a new bar, the damage bar) by itself.
- Clicking on Cold button: now the selected room will be a so-called cold room, in which you will see Lara's frosty breath. If the cold room is a water room, then it means the water is icy here, so Lara's health will be permanently decreased (shown by a new bar, the cold bar) by itself.

Room properties adjusted by Script:

- Rain or snow also needs the proper room texture, room geometry and script. The Script command for rain is Rain (see RAIN constants for more details), the Script command for snow is Snow (see SNOW constants for more details).
- Use a CUST_RAIN constant in a Customize Script command to customize the rain. (See FR constants for some possibilities.)
- MirrorEffect Script command is an enhanced version of the old Mirror command. Use the new command if you want to create mirror rooms. (See MIR constants to choose the mirror. See FMIR constants if your objects are mirrored in a bad way.)

Triggers for room properties:

- Activating F88 will switch on/off the mirror in mirror rooms.
- Activating F115 will set the room as outside/water/snow/rain/quicksand/damage/cold room. Activating F116 will remove that feature.
- Activating F117 will change the rain/snow intensity in the given room.
- Activating F157 will change the Rain command data in the level. F156 will do the same for snow.
- Activating F361 will fix a possible bug in the given room: if you see the wrong rain intensity, though you are next to that room.

15. Room structures

Back to Top

Selecting an area in Room Editor:

SHIFT+F5: now you can type the sizes of an area to select, instead of selecting that by drawing the mouse.

Forming floors/ceilings/walls in Room Editor:

New complex methods to form the floor/ceiling:

See these Drop Down Menu Bar commands (or a key combination you see there) in Shapes menu:

- Smooth Slope Floor, Smooth Slope Ceiling
- Stepped Slope Floor, Stepped Slope Ceiling
- Random Smooth Floor Up, Random Smooth Floor Down, Random Smooth Ceiling Up, Random Smooth Ceiling Down (see Setting for Random Smooth Floor/Ceiling in NGLE Settings window to customize them)
- Pyramid Floor, Pyramid Ceiling, Inverse Pyramid Floor, Inverse Pyramid Ceiling (check Peaked Pyramids in NGLE Settings window if you don't want "usual" pyramids)
- Bend Floor, Bend Ceiling, Inverse Bend Floor, Inverse Bend Ceiling (see Bend Type in NGLE Settings window to customize them)
- Dome Floor, Dome Ceiling, Inverse Dome Floor, Inverse Dome Ceiling

New methods to form wall sections in Room Editor:

See these Drop Down Menu Bar commands (or a key combination you see there) in Shapes menu:

- Grid selected wall, Grid all walls
- Remove all grids

Removing what you've edited:

Remove editing from the selection (also in Shapes menu) or CTRL+F11: use it both for floor/ceiling and wall editing operations.

Room Info Box information:

- CTRL+J: it switches on/off some Room Info Box information on the title bar of Room Editor.
- F, C, W, H – see them only if some squares are selected:

= F, C (only if one square is selected): F – the deepest point of the floor here, C – the highest point of the ceiling here.
= W, H (only if more squares are selected): W – the east-west length of the selected area, H – the north-south length of the selected area.

- Show extra infos in Room Info Panel (in NGLE Settings window) – check it and F/C values will be redefined: you will also see there the highest point of that floor square/the deepest point of that ceiling square.

Overlaps and Boxes:

They show the actual and the maximum amount of two numbers about an interesting thing you cannot exceed. – This is what I'm talking about:

The position of the enemies with artificial intelligence is controlled by the so-called boxes (that have nothing to do with the grey Box button). I mean, enemies are moving between “invisible” boxes when they are moving in the rooms. (The values will be refreshed only at the next Output WAD operation.) – See a simple example to understand:
Open a brand new project. Now you have only one room, with Size 18×18. This room has only one box, with Size 18×18, filling the whole room – so now your project has only that box. Now create a column in the middle of the room, using the green Wall button. That square-shaped column is naturally an obstacle for the enemies, so now the Room Editor splits the only one box of the room into four boxes:
a, one box (Box1) between the south and the north room wall and between the west room wall and the column west wall,
b, one box (Box2) between the south and the north room wall and between the east room wall and the column east wall,
c, one box (Box3) between the east and the west room wall and between the north room wall and the column north wall,
d, one box (Box4) between the east and the west room wall and between the south room wall and the column south wall.

“Overlap” means the “invisible” 3D area where a box is overlapped with another box. When you have only one box in your project then you naturally don’t have an overlap. But when you have four boxes in that room then you have eight overlaps:
a, For Box1: it is overlapped with Box3 (at north) and Box4 (at south).
b, For Box2: it is overlapped with Box3 (at north) and Box4 (at south).
c, For Box3: it is overlapped with Box1 (at west) and Box2 (at east).
d, For Box4: it is overlapped with Box1 (at west) and Box2 (at east).

“Fake” collision:

Activating FLIPEFFECT triggers with “Collision” name (from F310 to F329) will simulate collision. For example you have a square-based block raised by 2 clicks, but you want that block to act as if that were a block raised by 6 clicks. That’s why you activate a “Collision” trigger there that will simulate a further raising by 4 clicks. Now the block will be seen in 2 clicks high in the game, though the game engine will detect that as if that were 6 clicks high.

The description of the “fake” collision feature:

- This method has more advantages. For example, now you can create a “fake” collision box around an object, so Lara can climb on the top of the fake box as if she were climbing on the object itself. (So, after all, this method is an enhanced version of Edit Object TRLE function.)
- You can have maximum two Collision triggers on the same square: one for the floor and one for the ceiling.
- Collision triggers are fake triggers which means they create the collision in the Room Editor so they don’t need to (and they won’t) be put into TOM files when outputting the WAD. So it can be anything (TRIGGER, PAD etc.), i.e. nothing will happen if somebody/something gets into the trigger area. (So they don’t be overwritten by the trigger overlap rules.)
- Always place them in the rooms where the collision effect will happen. (I.e. the trigger for ceiling collision always must be in the room of the ceiling.)
- Don’t export these triggers.
- F330 is a special (and not a fake!) Collision trigger. Its purpose is to forbid Lara catching and hanging on the fake ledges created by the fake Collision triggers. (Just think about it: see that fake 6 clicks high block again. Lara can catch the 6 clicks high, invisible edge, if she jumps up. But naturally it is very ugly if she’s hanging on an invisible edge.
Of course, it is not a problem, if the object in that fake collision box fits nicely the whole box, so when Lara’s grabbing the fake ledge it will look as if she were grabbing the ledge of the object.)

16. Room textures

Back to Top

Texture panel in Room Editor:

- You can use the mouse wheel to roll the texture tiles on the texture panel. (Customize it at Settings for Mouse Wheel, in NGLE Settings window.)
- CTRL+T: changes the green triangular part on the selected tile on the texture panel.
- Textures (in Room Info Box): shows the actual and the maximum amount of tiles on the texture panel.
- Current Texture (in Room Info Box): shows the ID (defined by the position of the tile on the texture panel) of the selected tile on the texture panel.

TexInfos in Room Editor:

TexInfos (in Room Info Box): shows the actual and the maximum amount of a number about textures you cannot exceed.
What you should know about TexInfos number:

- The counter starts from 256, as if you'd placed at least one copy of all the texture tiles that have ID's from 0 to 255.
- All the tiles over ID 255 – if that is the first copy of the tile you're just placing on the map – is +1 in the counter.
- All of the partial tiles – if that is the first copy of the partial tile you're just placing on the map, whatever the ID of the whole tile – is +1 in the counter. Green triangular parts are not included.
- Each tile over ID 255 or each partial tile will be included in the counter even if you removed all the copies of that tile from the map.
- If TexInfos number is too big then use these commands from the Texture menu of Drop Down Menu Bar: Remove unused tail infos, then Check integrity textures. Then see the log file of the second command in TRLE main folder: TailInfo_log.txt.

New features with the placed textures in Room Editor:

- Turn Selected Textures in Texture menu of Drop Down Menu Bar: type F (floor), C (ceiling) or W (wall), then 1 (90°), 2 (180°), 3 (270°), 4 (random) in the windows popped up to rotate the tiles on the selected area of the map.
- Create Floor Triangular Tex in Texture menu of Drop Down Menu Bar (or SHIFT+F3): adjusts the square-shaped texture tile on broken floor squares.
- Create Ceiling Triangular Tex in Texture menu of Drop Down Menu Bar (or SHIFT+F4): adjusts the square-shaped texture tile on broken ceiling squares.
- Clear Floor Selected Textures in Texture menu of Drop Down Menu Bar (or ALT+F9) or Clear Ceiling Selected Textures in Texture menu of Drop Down Menu Bar (or ALT+F10): removes only some textures of the room.
(It's a bit complicated, the basic tutorial of NGLE could help. See: Drop Down Menu Bar of Room Editor, Help menu, NGLE Help command.)
- Substitute Textures in Texture menu of Drop Down Menu Bar: replace all the copies (placed on the map) of the selected tile of the texture panel, by the tile whose ID you'll type in the window popped up.

Big textures in Room Editor:

Big textures are 128×128 pixel sized texture tiles. – You have two possibilities to use the big texture function:

- Click on Big Texture button or hit CTRL+B to switch on/off the big texture function. If it is on, then four 64×64 tiles will be selected (as if they were a 128×128 tile) if you click on the texture panel. That big tile will placed if you click on the editor window now. (But the big tile is not a "real"

tile: you will see it, eg. if you open Animation Ranges panel: you can find the “little” tiles there.)
- If you want the big tiles to be the real tiles then transform your project type into a so-called v50 type. (A basic project – i.e. when the little tiles are the real tiles – is a so-called v49 project type.) If you have a v50 project then the color of the Big Texture button switched on is red which means you can’t switch off the button at that project.
To transform a v49 project into v50, first of all, create a TGA file for your level in which the size of the tiles is 128×128. Attach it to the project (in the usual way) then open MapConverter2.exe in Tools folder. Check the transformation data (from 64×64 to 128×128) then click on Start Conversion button. (It’s recommended to check “Modify ONLY…” option before converting.) – A backup v49 project file will also be created, use it if something went wrong.

Animation Ranges panel in Room Editor:

New buttons:

- If you did anything in this panel (except deleting a range or selecting the range texture tiles) then click on Assign for refreshing.
- Test button is trivial, but it’s worth mentioning R button which will remove all the ranges.
- Only with v50 projects: if the size of the big tiles is disturbing here, then click on Reduce button. (It really doesn’t change anything.)

New boxes:

- Anim-Type box: to choose the type of the animation range. (The “classic” animation range must be Frames.)
- You can change the frame-rate value of some ranges (fps=frame per second or spf= second per frame) in the box next to the Assign button.
- You can change the UVRotate value of some ranges, just above the frame-rate box. (If you choose “DEFAULT (from Script)” value then type an UVRotate command in the Script.)

New types of animation ranges:

- Select P-Frames in the Anim-Type box (there can be maximum two P-Frames range per level). If you place a texture tile of a P-Frames range in the map, then that tile won’t be animated. – See the description of animation ranges triggers to animate P-Frames.
- It’s not a problem if you use only one tile in a range that is Full-Rotate in the Anim-Type box. (But you can use more tiles in the range, naturally.) Because the tiles of the range won’t be swapped for each other. I mean, all the tiles of the range will be animated on the squares where the tiles are placed, but the tiles won’t be swapped there for other tiles.
Imagine that as a picture strip: all the frames of the strip are the tile that is placed on that square, so that tile will be rolled on that square again and again.
(Note: the animated railroad textures on the ground for train levels must be parts of a Full-Rotate range in TRNG.)
- I think if you choose Half-Rotate type range in the Anim-Type box then you will experience the range will be buggy. (This technique is made to use the TR3 type animated tiles when each tile contains four “micro” tiles. – See the basic animating textures tutorial on Skribblerz if you’d like to know what you should do with TR3 animated tiles in TRNG.)
- If you choose River-Rotate type range in the Anim-Type box then the range will actually work as a combination of a Frames and a Full-Rotate range: the tile will be scrolled but will be swapped for another tile as well.

Script and triggers for animation ranges:

- Use F56 to stop the animation of a non-static (non-P-Frames) range.
- Use F57 to restart the animation of a stopped range.
- Use F58 for the first P-Frames range or F59 for the second P-Frames range. The triggers will swap the given static tile of the range for another static tile of the range (Window &). (Note: the

tile in Window E is the tile that is placed there originally, so not the tile that is just seen there.)
- Use F60 to invert the direction of the scrolling at Full-Rotate or River-Rotate ranges. (I recommend using each F60 only once.)
- You can create a “program” for a given P-Frames range (to swap the tiles with a given order and time). Now the range is not controlled by F58 or F59.
The program can be defined by a TextureSequence Script command. (See SEQ constants for some features in this “program”.) This program is controlled by F94 (to start the TextureSequence with the given ID on a given range) and F95 (to stop the TextureSequence with the given ID). (Note: it’s not recommended to use a TextureSequence for more than one P-Frames range in a level.)

Texture Sounds panel in Room Editor:

New buttons:

- You need to click on Assign Bump button for refreshing the bumped textures.
- You need to click on Assign Sound button for refreshing the sound of the textures.
- Test button is trivial, but it’s worth mentioning R button which will remove all the bumped textures and all the special texture sounds.
- Only with v50 projects: if the size of the big tiles is disturbing, then click on Reduce button.

Additional properties:

- In TRLE, you had to use Load Texture Snds and Save Texture Snds (in Room Editor) to prevent a bug about bump map/texture sounds feature. Now that bug is fixed. (Moreover, if you use those buttons that is what seems buggy now.)
- Maybe you don’t know if the player has checked Bump Mapping in the setup or not but it is important for your game. So use ForceBumpMapping Script command to force checked or unchecked bump map feature for your game.

17. Flipmaps
Back to Top

- Use an F125 instead of a FLIPMAP or a FLIPON trigger to switch on a flipmap. Or use an F126 instead of a FLIPOFF trigger to switch off a flipmap. (They are useful if you want export the flipmap triggers into the Script.)
- The usual flipmap triggers are special triggers if we’re talking about the STT codebit buttons. Because the “two or more triggers activate a thing” (achieved by switched on/off STT codebit buttons) usually works only with objects – but there is an exception: it also works with flipmaps. That’s why if you want to export this thing into a TriggerGroup, with FLIPEFFECT triggers (“two or more FLIPEFFECT triggers activate a flipmap”) then you can use F124 triggers. – It’s recommended to use it with TGROUP_SINGLE_SHOT TriggerGroups.
- Use an F333, and the flipmap will switch on/off continuously, in a given rhythm. F334 will stop that.

18. Light, fog, sink and special visual effects
Back to Top

Room Editor features:

- Effects (in Room Info Box): shows the actual and the maximum amount of the placed light and fog bulbs.
- Click on any Red/Green/Blue box (Object Tint, Ambience, Colour) between – and + signs or the X, Y, Len, Cut, Int, Out or In box, between < and > signs to type the light value.

New method for activation of sink bulbs:

You can use an A46 instead of the “usual way” to activate a sink bulb. – Notes:

- Don’t place an A46. Only use it exported in a TriggerGroup.
- The trigger to start the TriggerGroup (F118) must be in “Multiple” mode now. (So the sink will draw Lara continuously after her activating it, the trigger will be launched again and again by itself after its activation.) It means she doesn’t need to be over the trigger – after activating it – to be drawn to the sink. And it means you need to switch off the sink in some other way – for example, removing the sink bulb with switching on a flipmap, or activating an F192 trigger whose purpose is to stop TriggerGroups working in “Multiple” mode.

Defining colors for triggers or Script commands:

Define a color with RGB values in a ColorRGB Script command. Some features will be linked to the ID number of that command to use the color that those RGB values define.
ColorRGB command is used

= for these Script commands: Customize (with CUST_BAR or CUST_DARTS constants), Parameters (with PARAM_COLOR_ITEM or PARAM_SHOW_SPRITE constants), WindowsFont or
= for these triggers: F152, F154, F155, F291.

Note: always place a ColorRGB in the Script above the command of the feature that will use this ColorRGB command. – For example: if you use a ColorRGB for a WindowsFont command, then always use this order:

ColorRGB= ….
WindowsFont= …

Visual range:

Behind the visual range, the room textures will disappear far away. WorldFarView Script command can define a new visual range for your whole game. LevelFarView Script command will define a new visual range only for the actual level.
Activating F159 you will able to change the visual range during the game.

Fog features:

- Maybe you don’t know if the player has checked Volumetric FX in the setup or not but it is important for your game. So use ForceVolumetricFX Script command to force checked or unchecked fog bulb feature for your game.
- Activating F61 will do the same in the game as if you’d checked Volumetric FX in the setup. On the other hand, activating F62 will do the same in the game as if you’d unchecked Volumetric FX in the setup.
- FogRange Script command will redefine the start point and the end point of the distance fog. Activate F194 or F227 if you want to change them (with a sudden move) during the game. – Special triggers to adjust the start/end point:

= Activate F195 so the start point will change to a new distance in a given time, by degrees.
= Activate F228 or F229 so the start/end point will change back and forth, continuously, in a given distance interval. Stop it activating F230.
= Activate F196 so the start point will change back and forth, continuously, between the old and the new position. Stop it activating F198.

- As you know, some old TRLE features are used to define the fog color: Fog Script command for fog bulbs or the distance fog, or F28 OldFlip trigger for fog bulbs. F224 is a new method for fog colors, but only for the distance fog.
- F197 is a special trigger for enabling/disabling the fog types. It is not easy to understand how it

works. For example, it is able to switch off both the fog bulbs and the distance fog at the same time.
- Activating F226 will adjust the visibility of fog bulbs. (So the fog bulbs, even though they are enabled, won't be seen from a given distance.)
- Use a CUST_FIX_WATER_FOG_BUG constant in a Customize Script command. It will fix a bug (when the fog shines on textures where it shouldn't be shone).
- F225 fog-trigger can't be useable any more.

Layers on the sky:

- Activating F150 will apply/remove layer1 or layer2.
- Activating F152 will change the layer1 or layer2 color, immediately. (Use a ColorRGB Script command to define the color.)
- Activating F153 will change the speed/direction of clouds on the layer1 or layer2 color, immediately.
- Activating F154 will change the layer1 color, step by step. F155 will do the same with layer2. (Use a ColorRGB Script command to define the color.)

Lightning and thunder:

- Activating F151 will switch on/off the lightning/thundering feature.
- F359 is probably a "work in progress" trigger for the lightning, so you can't use it for the time being.

A special light effect:

Activating F355 will create a colorful "burst of light" on the screen for a given time. (If the burst is infinite then remove it by activating F356.)

19. Texts, images, sprites and bars

Back to Top

Texts on the screen:

New text features:

- If you want to redefine your font (except for the diary or the new type savegame panel) then don't reform font.pc but place a FONT_GRAPHICS object in your WAD.
FONT_GRAPHICS can be edited by NG Font Editor function in NG Center\Tools.
- Do this if you want to customize the texts you could find before as well, in TRLE (inventory texts, the text that can be written by Legend Script command etc.) (see more about it below):

= for the color: use a CUST_SET_TEXT_COLOR constant in a Customize Script command,
= for the size: use a TextFormat Script command.

- Some TRNG features have texts, though they are not about texts (for example, standby camera effect). To customize the text color, size and position (for some of them), use a TextFormat Script command or F66, F75, F76, F81 triggers. (See more about it below.)
- If you want to print the ammo amount of the weapon in Lara's hands on the screen (when you are directly in the game, so if the inventory is not open) then use a CUST_SHOW_AMMO_COUNTER in a Customize Script command. (See more about it below.)
You also need a FONT_GRAPHICS object in the WAD if you also want to show ammo/weapon pictograms with the ammo amount.
- Activating F223 will put the Statistics menu on the screen (as if you'd opened the menu by manually, as usual).
- See more about the diary or the new type savegame panel below.
- Though in TRLE your only possibility to print any text you want on the screen is Legend Script

command (when the level starts), in TRNG you have more possibilities for that (and any time you want).
It's your decision if you use a complex forming for these TRNG texts or not.
If you don't use then these are your possibilities:

= Activating F65 will print a string from the [Strings] section of your language file, for a given time.
= Activating F149 will print a string from the [Strings] section of your language file, till hitting ESC.
= Activating F64 will print a string from the [ExtraNG] section of your language file, for a given time. (It will probably not work above String ID 255.)
= Activating F360 will print a string from the [ExtraNG] section of your language file, always "forever". (It will also work above String ID 255.)
= Activating F148 will print a string from the [ExtraNG] section of your language file, till hitting ESC.
= Activating F201 will scroll a string from the [ExtraNG] section of your language file, vertically (from down to up).

If you use then these are your possibilities:

= Activating F207 will print a string from the [PSXStrings] section of your language file, according to the complex forming.
= Activating F208 will print a string from the [PCStrings] section of your language file, according to the complex forming.
= Activating F209 will do the same as F208 will do, but hitting ESC will remove the text off the screen.
= Activating F203 will print a string from the [ExtraNG] section of your language file, according to the complex forming.
= Activating F210 will do the same as F203 will do, but hitting ESC will remove the text off the screen.
= Activating F206 will scroll a string from the [ExtraNG] section of your language file, horizontally (from right to left), according to the complex forming.
= Activating F205 will scroll a string from the [ExtraNG] section of your language file, vertically (from down to up), according to the complex forming.

"Complex forming" means the forming values are defined in a Parameters Script command that uses a PARAM_PRINT_TEXT constant. (See CL constants for colors. See FT constants for positions and sizes.) The constant has an ID. The trigger of the text is linked to this ID, so if the trigger has been activated then the text will be printed on the screen, using the forming values in that command.
But if you don't use the complex forming then the texts will be printed according to default values (size, color, position) – unless you use a "non-complex forming" for them, using TextFormat Script command or F66, F75, F76, F81 triggers. (See more about it below.)
(Anyway, each text in TRNG has the default values, except PARAM_PRINT_TEXT texts, diary and the new type savegame panel.)

Special triggers:

= Activating F202 will remove all the texts of the screen that are just being rolled there vertically.
= Sometimes you timed the text to be on the screen "forever". There are two methods to remove them:

< Activating F67 will remove all the texts that is printed by any of the triggers listed just above. (Except the scrolled ones.)
< Activating F204 will remove the given [ExtraNG] string off the screen.

Let's see now everything about forming the texts:

- First of all, let's define the type of the texts used in TRNG:

= Old Text: texts you can also find in Last Revelation or TRLE: texts in the inventory, in the Statistics Screen etc.
= TRNG Text: texts (except Ammo and Feature Text) that the new TRNG (first of all, non-textual) features use: the amount of done rotations (at swing bars), the text for the standby camera effect, the timer of A52 trigger etc.
= Ammo Text: text that is used for showing ammo amount on the screen by CUST_SHOW_AMMO_COUNTER.
= Feature Text: texts of text/picture type TRNG features (diary, new type savegame panel).
= Independent Text: you can print them on the screen "with or without complex forming". (See more about them above.) – "Independent" means this text is about only the text itself. (Because, for example the text of the standby effect is not independent, that is linked to the standby feature.)
"A" type Independent Texts are the texts that will be written on the screen by a FLIPEFFECT that doesn't use a PARAM_PRINT_TEXT Script constant.
"B" type Independent Texts are the texts that will be written on the screen by a FLIPEFFECT that uses a PARAM_PRINT_TEXT Script constant.

- To form the texts, these are the tools:

= Font.pc in graphics\wads folder: the basic font of the game (except Feature Text) is defined here.
= FONT_GRAPHICS object: to customize the font for the level/title having the WAD of the object (except Feature Text), using NG Font Editor.
= CUST_SET_TEXT_COLOR constant (see TT constants to define the type of the text): to customize the color of Old Text. – Notes:

a, Customizing of the level texts, the customization command must be placed in a [Level] block usually.
b, Customizing of the level texts – if you want the validity of the customization for the whole game – or the title texts, the customization command must be placed in [Title] block.

= TextFormat Script command:

a, To customize all the Old Text sizes of the game (except: the text of Legend Script command), place it in [Title] block, using only the SizeCharacterMenu field of the command. (See SC constants for the sizes.)
b, To customize color, position and size of TRNG Texts and "A" type Independent Texts of the level, place it in the [Level] block, using any fields of the command but SizeCharacterMenu field. (See CL constants for color, see FT constants for position and size.) – Notes:

< The command in the [Title] block will work only for title texts.
< I think this operation doesn't work on all the TRNG Texts. It works, for example, for the standby effect text or the A52 timer etc. (So some texts default forming values may not be edited.)

= F66, F75 or F81 triggers: they can do the same what TextFormat command can do.
Activating any of them, that will overwrite (for that level or title level) the value defined previously by TextFormat or another trigger like that. – Notes:

a, It works for the TRNG Texts that are just written or will be written.
b, It works for the "A" type Independent Texts that will be written.
c, It is valid only on that level/title level, until the next trigger like that.

= F76 trigger annuls (on the actual level/title level) every effect of TextFormat command or all the previous F66/F75/F81 triggers

a, for the TRNG Texts that are just written or will be written, and

b, for the “A” type Independent Texts that will be written.

= See PARAM_PRINT_TEXT constant to customize “B” type Independent Texts.
= See CUST_SHOW_AMMO_COUNTER constant to customize Ammo Text: (See CL constants for colors, see FT constants for position. You can’t use the FT constants now for the sizes: use SC constants for the sizes. See SHOWC constants for more features.) Place it in a [Level] block usually. Place it in [Title] block, if you want it to be valid for the whole game.
= WindowsFont Script command (see WFF constants for some features): you can define here all the parameters of Feature Text – for the level if the command is in that [Level] block or for the title if that is in [Title] block. (You also need one or two ColorRGB Script commands to define the font colors.)
Note: the font type you chose in this command must also be typed in a [Strings] or an [ExtraNG] string.

Diary:

The ingredients:

- A pickable object (pickup, quest, puzzle or key) is needed as your diary object. (Never use the DIARY_ITEM object.) If that is in the inventory, then select it there, and the diary will open on the screen.
- You always need one or more WindowsFont Script commands to define the font(s) in the diary and its/their properties.
- A Diary Script command is needed to define the properties of the diary. (See LDF constants for some features. See PL constants to define the structure of the diary.)
Note: the diary won’t work, if you don’t use a proper order for its Script commands. – A recommended order:

ColorRGB= ….
Pickup/Puzzle/Key= ….
WindowsFont= ….
Diary= ….

- The image of the diary on the screen is an “Image” BMP (see more about it below). Type the Image ID in the Diary command.
- The default contents of each diary (as a TXT file) must be placed in the Script folder. The Diary command will be linked to this TXT with @ sign, such as, for example @Diary01_default.TXT, which means this TXT is the default contents for the first diary. (The @Diary01_default.TXT must also be typed in an [ExtraNG] string.)
- If you want to add new contents to a diary (attaching it to the actual contents) then do the same: type it in a TXT - for example, @Diary01_new01.TXT, which means this is the first new contents for the first diary -, save it in the Script folder, then type @Diary01_new01.TXT in an [ExtraNG] string.

Key commands for the diary in the game:

- right arrow or SPACE: turn a page forward,
- left arrow: turn a page back,
- SPACE: jumps to the first page from the last page,
- ESC: exit.

The structure of the diary TXT files:

#END_PAGE# commands in the TXT will split the contents of the TXT into pages. For example, if the TXT has three pages, then it will look like this:

top of TXT (Each new contents of the diary added to a previous TXT must be started with an

#END_PAGE# command at the top of the TXT. But never use it in the TXT for the default diary contents.)
contents on the first page
#END_PAGE#
<FORMAT>
forming commands
<END_FORMAT>
contents on the second page
#END_PAGE#
contents on the third page
bottom of TXT (Never close the last page of the TXT with an #END_PAGE# command.)

Each page can have a forming section – if you don't want to apply the default forming values (defined in the Diary Script command) on the given page – started by a <FORMAT> sign and ended by an <END_FORMAT> sign. The forming section on the second page means now there are one or more forming value on the second page that will overwrite the default forming value on that page.
The forming commands are used in a "command, equality sign, forming value" way.
See more about forming commands in the description of Diary Script Command in NG Center\Reference.
(Note: if you want an image on a diary page then you need a forming command everyway, because you will identify the "Image" BMP ID for that image in a command like that.)

Triggers for the diary:

- Activating F219 will add the contents of the given string (i.e. the new contents of the diary, see example @Diary01_new01.TXT above) to the end of the actual diary contents.
- Activating F220 will remove all the actual contents of the given diary. (Never let Lara to open a totally empty diary, or else the game will crash! So open some new contents after F220, before Lara will do that.)
- Activating F221 will remove the contents of TXT you added last to the given diary.
- Activating F222 will force the game to open a diary at the given page.

The diary contents on the following levels:

- I highly recommend if you use a pickable item as a diary, then use that as a diary in the following levels as well. It also means the same inventory name and the Diary command with the same ID.
- The contents of the diary when the level ends will be dragged to the next level automatically. (In the new level, type IGNORE in the Diary Script command in the field of the default contents.)

Savegame features:

Customization on the old type savegame panel:

The characters may move on each other on the savegame panel with some FONT_GRAPHICS and/or TextFormat values. To solve the problem, remove the day/hour/minute/second values from the savegame entries, using a CUST_NO_TIME_IN_SAVELIST constant in a Customize Script command.

The new type savegame panel:

The new type savegame panel will replaces the old type savegame panel on the given level/title. The new savegame panel is a background image, having not only the usual savegame entries, but a screenshot of the saved position and some additional info (small/big medipacks Lara just has, the played time etc.) about the saved position as well.
Use a SavegamePanel Script command to replace the old panel and define the parameters of

the new panel. (See SPF constants for some features. See SPL constants to define the structure of the panel.)

Furthermore:

- You always need WindowsFont Script commands to define the font(s) in the panel and its/their properties. (I recommend placing SavegamePanel command after – i.e. under - the WindowsFont command.)
- The image of the panel on the screen is an “Image” BMP (see more about it below). Type the Image ID in the SavegamePanel command.
- You need a CUST_INNER_SCREENSHOT constant in a Customize Script command to define the resolution of the screenshot. (See QSF constants for the resolution values.) – Without this command, there won’t be screenshots on the new type panels.
- To define the additional info description of the new savegame panel of the level/title, type a TXT file, placed in the Script folder. The SavegamePanel command will be linked to this TXT with @ sign, such as, for example @panel_info.TXT, which means this is the additional info description for the new savegame panel. (The @panel_info.TXT must also be typed in an [ExtraNG] string.) See more about additional info commands in the description of SavegamePanel Script Command in NG Center\Reference.
See an example – this is what you type in the TXT:

Actual number of large medipacks (in inventory):
(L-PACKS)
Found Secrets in game:
(SECRETS) of 70

The texts in brackets are commands. (L-PACKS) for the large medipacks Lara just has, (SECRETS) for the secrets Lara has found so far. So, for example, if she has 16 large medipacks and found 19 secrets, then this info will show up in the savegame panel when you select that savegame entry:

Actual number of large medipacks (in inventory):
16
Found secrets in game:
19 of 70

Images on the screen:

- Use BMP files for images.
- The image name you must use is “Image” and the ID number, so, for example, Image60.BMP.
- Place the image files into the TRLE main folder, or create a subfolder there (with Pix name) for them.
- Image Script commands will define the way the images will be placed on the screen. (See IF constants for some features.)
- If you think it’s hard to define the coordinates of the image on the screen for Image command, then Get Screen Frames function in NG Center\Tools can help you.
- Activating F217 will place the given image on the screen, according to the given Image Script command, for a given time. If you placed that “forever” (only with pop-up images), then remove the image, using F218.
(Note: don’t misunderstand, the images with Image commands are independent of the images of diary or savegame images, though you can use an image for those purposes.)
- See converter.exe in Tools folder. Place the program wherever you want and the program will convert the JPG files in the same folder into BMP files. – Notes:

= The EXE is not useful with the pictures the TRNG doesn’t use. (For example, ImageX.bmp and Load.bmp are TRNG files so you can convert them from ImageX.jpg or Load.jpg.)
= I don’t recommend using the EXE in Tools folder.

Sprites on the screen:

- Sprite Editor function in NG Center/Tools to attach the required texture to sprite-holder objects.
- Format Sprite Slot function in NG Center/Tools to arrange the amount and size of the sprites.
- As you know, some procedures use sprites (flame, rope, underwater bubbles etc.), but you can also use any sprite to place that directly on the screen (on the surface of the screen as if that were an image). You can use any classic sprite-taking object for that. However, Paolone made an object directly for that feature: see CUSTOM_SPRITES.
(If you don't know what you should place on the screen – an image or a sprite? -, then please note there are advantages and disadvantages on the both sides. For example, the amount of the images on the screen is limited.)
- Use a Parameters Script command with a PARAM_SHOW_SPRITE constant to define the way the sprites will be placed on the surface of the screen. (See FSS constants for some features.)
Note: a ColorRGB Script command is necessary to define the transparency color if you use the transparency feature.
- Activating F357 will place the given sprite on the screen, according to the given Parameters/PARAM_SHOW_SPRITE Script command, for a given time. If you placed that "forever", then remove it, using F358.

Bars on the screen:

- Use a CUST_BAR constant to create new bars or to customize the good old TRLE bars (air, dash, health, loading level):

= See BAR constants to choose the type of the bar.
= See FBAR constants for some features.
= You need ColorRGB commands to define the colors of the bar.
= Activating F331 will put the given new bar on the screen. F332 will remove that.

- Use a Damage Script command to customize the TRNG bars created by Paolone (cold water, damage). (See DMG constants for some features.)

20. Cameras

Back to Top

Customization for the original cameras:

- You can customize the basic (chase, combat, look) cameras, if you use a CUST_CAMERA constant in a Customize Script command. (See FCAM constants for some features.)
You can also do some customization on chase, combat, look cameras, if you use PARAM_SET_CAMERA constant in a Parameters Script command. (See FSCAM constants for some features.) But this time the customization is valid only a given section of the level: activating F214, the customization will be applied for a given time. If you activated the customization "forever", then disable that by F215.
- Use a CUST_ESCAPE_FLY_CAMERA constant in a Customize Script command, and the given flyby sequence will be able to be interrupted by the required key.
- Use a CUST_PAUSE_FLY_CAMERA constant in a Customize Script command, and the given flyby sequence will be paused when the required key is pushed down.
- Use a CUST_TEXT_ON_FLY_SCREEN constant in a Customize Script command, and the texts will be seen on the screen even if a flyby sequence is just working.

Applying the original cameras in new ways:

- Activating F121 will disable the combat or the look camera. Activating F122 will restore that.
- If you want, use an A41 instead of a trigger to activate a Camera/Fixed Camera in the usual way. – Two things:

= Don’t place an A41. Only use it exported in a TriggerGroup.
= The (Fixed) Camera won’t switch off if Lara leaves the trigger squares of this TriggerGroup. So you need to time this camera or (only with Cameras) hit the look key or extract weapons to switch off the camera.

- A42 defines the camera target instead of CAMERA_TARGET object – so now it could be any Moveable object, even a moving enemy. (It works properly only as an exported trigger, placed before - ! – the A41 trigger of its camera in the TriggerGroup.)
- A45 is useable to start/stop a flyby camera sequence, instead of the “usual way”. (The sequence won’t start again automatically if Lara is standing on the trigger when the sequence ends.)

New camera effects:

- The so-called “follow” cameras are Cameras or Fixed Cameras that follow Lara, though one of the camera position (east-west, north-south, up-down) is fixed. Never start those cameras in the usual way now, always use an A41 in a TriggerGroup. F119 will define the type of the follow camera. F119 must be overlapped with the TriggerGroup-starter F118 trigger.
If you set the follow camera to work “forever” then use F120 to stop that. (You can also try the “good old” features to stop these cameras. So eg. if it is not a Fixed Camera, then it may stop if Lara draws her weapons.) – Nevertheless, this trigger is able to switch off any (Fixed) Camera.
If the target of the follow camera is not Lara, then define that using A42.
- Cameras or Fixed Cameras can also have some special effects to show their targets. Never start those cameras in the usual way now, always use an A41 (always timed!) in a TriggerGroup. F123 will define the required effect. F123 should be in the TriggerGroup that contains A41. Always place F123 after A41 in that TriggerGroup.
To stop these camera effects, just use the methods I wrote at the description of A41 trigger.
If the target of the camera is not Lara, then define that using A42.
- The standby camera effect is a special effect that will be accomplished by the chase camera. Use a StandBy Script command to define that effect. (See TSB constants to define the type of the effect. See FSB constants to define some other properties.) - The StandBy that has ID 1 is special one, because it will be started automatically, if the player won’t hit a key for a given time. The standby effects with other ID numbers will be started by F346, for a given time.

21. Sound and audio

Back to Top

New audio features:

TRNG has the so-called new sound engine for some new features. The new engine can play the background and the foreground audio at the same time, on two audio channels. F68 is used to activate audio files on channel1 (usually for the background audio) and F129 is used to activate audio files on channel2 (usually for the foreground audio). – So you don’t need CD triggers any more to play audio. (But you can use CD triggers, if you want.)

Other features of the new sound engine:

- Now you can play not only WAV formats. (However, see cm.exe or start_me.exe in Tools folder to convert MP3 files into WAV.)
- F68/F129 can activate the audio files any times.
- Now you can play audio files with bigger ID numbers than 111 (to 255). (And it doesn’t matter which audio slot you will use for background or foreground audio.)
- Use F69 or F130 to stop immediately the audio file just playing.
- Use F133 to adjust the volume for the audio file that is just playing or just starts playing.
- Customize the fade-out of the audio files in a Customize Script command, using a CUST_NEW_SOUND_ENGINE constant. (The SEXT constants – made for

CUST_NEW_SOUND_ENGINE – are obsolete now, don't use them any more.)
- The old sound engine won't continue the audio files if you saved the game when the audio was playing and then load that savegame. Now you can customize that, with a CUST_CD_SINGLE_PLAYBACK constant in a Customize Script command. (See CDM constants for the possibilities.)
- If you use CD triggers, then you will see that some features of the new engine will also be valid with CD triggers. But using a CUST_SET_OLD_CD_TRIGGER constant in a Customize Script command (and not using customization for the fade-out and the continuation), the game will "solve the problem", and the CD triggers will work exactly as in TRLE. (It naturally disturbs if you also want to use F68/F129 triggers to play audio files in the same level.)
- The default value for the new sound engine is "enabled". But if you use NewSoundEngine Script command with "disabled" value, then the new engine will be disabled for the whole game. (Maybe it is necessary. For example, Von Croy's classic "Angkor Wat" cutscenes cannot be reproduced if you use the new engine.)
- Maybe you want to use the old method with the new engine: "interrupt the old (background) audio temporarily when the new (foreground) audio plays". If you want that, then play the new audio with using F193.
- The CD txt file in wads folder won't contain audio files to play by not CD triggers.

Imported audio files:

Use F131 or F132 to play the so-called imported audio files. The imported files are files that will be saved in Script.dat when you're building your Script. So you don't need the audio file any more if you built the Script. (Some advantages of this method: it is useful if you don't want to attach some audio files directly to your game. For example, because it's you who made that so you don't want to release that. Or it is useful if you have no more empty space in the 000-255 audio slot intervals for a newer audio file. Etc.)
Place the file somewhere in TRLE main folder, and use an ImportFile Script command that will place that file into Script.dat when you're building the Script next time. (Use FTYPE_SOUND constant in the command. It means this file is imported as an audio file.)
Note: you have two IMPORT constants for the command, but I think only IMPORT_MEMORY works with imported audio files. The aggregated size of IMPORT_MEMORY type imported files is limited, but there is an advantage with that constant: sometimes the audio files start with a little slip – i.e. not exactly when they have been activated - (mostly if you play the game from a CD), but this constant will prevent that problem.

New sound features:

- We have some OldFlip Flipeffects ("inherited" from TRLE) if we want to play some sounds in TRNG in a direct way. ("Direct" = not by Sound AnimCommands, as usual.) But this function has also been enhanced: use F70 or F71 to play any sound from sound\Samples folder. (The single mode is used for single sounds, the forever mode is used for loop sounds. Select 1-30 seconds if you want to play a loop only for a given time.) Then activate F72, F73 or F74 to interrupt playing them.
- If the sound slot is active in your WAD, but doesn't have a sound file, then (the faint and short) back_jm1.wav of the sound\Samples folder will be attached to that sound slot when converting the level. (See another thing if you have problem with missing sound files for existing sound slots: a CUST_DISABLE_MISSING_SOUND constant in a Customize Script command.)
- Use a CUST_SFX constant in a Customize Script command, to redefine some hardcoded sounds. (See TS constants to select the type of the sound.)
- The so-called v130 type WADs (made by WADMerger) are able to play sound slots that are not original TR4 sound slots but other TR or special slots. (Be careful: v130 function is buggy!) Activating F168 is the direct way to play a sound file of a non-TR4 sound slot.

Additional tools:

- Use SoundSettings Script command to adjust preset values for music volume, SFX volume,

sound quality (seen the Options menu) when the game starts. (See SQ constants for the quality possibilities.)
- If you copied the original Last Revelation or Chronicles audio files into your audio folder then you can change their names into ID numbers, using Change Track Names function in the Tools tab of NG Center.

22. Special properties

Back to Top

- When you're running the program in windowed mode, then you will see this title at the top of the window: Tomb Raider – The Last Revelation. Use a WindowTitle Script command if you want to see another title there at the given level.
- Activating F97 will save the game, in files with backgame name, placing them in TRLE main folder. (They have nothing to do with the usual savegame files.) Activating F98 will load the game saved in backgame files.
- Activating F51 will disable a given keyboard command for a given time. (Keyboard commands are the usual commands of the game. For example "Action" command means Key CTRL. If the Action is disabled that means if the player hits CTRL – or any other key that is attached to the Action command – then nothing will happen.) If the disability is infinite then activating F52 will restore the key command.
- Activating F53 the game will think the player has just hit a key to activate a given key command – though the player hasn't hit that key.
- Use a CUST_SET_SECRET_NUMBER constant in a Customize Script command (for a new maximum secret number for your level/game) if you don't want to see the original 70 maximum secret number in the Statistics.
- A lot of feature should use big numbers, but the triggers for these features are unable to define directly the big numbers. So you should define the big numbers in a Parameters Script command that will use a PARAM_BIG_NUMBERS constant, and those triggers will be linked to the big numbers in that Script command.

23. Moveable and Static objects

Back to Top

Common properties:

Items (in Room Info Box): shows the actual and the maximum amount of the placed effects (light, camera, fog, sink, sound), Moveable or Static objects, triggers.

Moving/rotating the objects:

Probably a lot of them will work with Lara as well:

- A29, A36: to move a Moveable object linearly, continuously back and forth,
- A30, A31, A32, A33, A34, A35: to move a Moveable object linearly,
- A63, A64, A65, A66, A67, A68: to jerk a Moveable object linearly,
- A5: to rotate a Moveable object endlessly around its vertical axis,
- A18, A19, A20, A21: to rotate a Moveable object around its vertical axis, till reaching a given direction,
- A1, A2, A3, A4: to rotate a Moveable object around its vertical axis, till it turning a given degrees,
- A11, A12: to jerk a Moveable object around its vertical axis,
- A80: to rotate the Moveable object around a vertical axis that is in the given distance from the object,
- A10: to rotate a Moveable object endlessly around its horizontal axis,
- A22, A23, A24, A25: to rotate a Moveable object around its horizontal axis, till reaching a given direction,
- A6, A7, A8, A9: to rotate a Moveable object around its horizontal axis, till it turning a given

degrees,
- F167 (for Moveables) or F166 (for Statics): to move an object linearly, according to a given “program”. The “program” is defined in a Parameters Script command that uses a PARAM_MOVE_ITEM constant. (See FMOV constants for special features. See DIR constants for the direction of the move.)
- F173 (for Moveables) or F172 (for Statics): to rotate an object around its horizontal/vertical axis, according to a given “program”. The “program” is defined in a Parameters Script command that uses a PARAM_ROTATE_ITEM constant. (See FROT constants for special features. See ROTH and ROTV constants for the direction of the rotation.)

Additional information:

- The jerking movements are used only to correct the improper positions, so the player mustn’t see those sudden movements.
- When you move Moveables linearly by ACTION triggers then there are two methods to define the speed:

= if it is an ANIMATING then type the speed value in the OCB window,
= if it is another Moveable object then use a CUST_SPEED_MOVING constant in a Customize Script command.

But when using A80 trigger then type a number into the OCB window of the object that is bigger than 127 – this is the way to adjust the rotation speed now.
- To stop the linear/rotative movements of Moveables, activate an A47 trigger. – Other possibilities to stop the movements:

= If you use A80 then it’s better if you stop the rotation by activating A81 because now the mode of the stop is customizable.
(If you use A81 in “Pause” mode, then the object will stop immediately. It is useful, if you want to restart the rotation again later with the same axis, activating A85. Or else the axis will be recalculated in the stopped position before the restart - that is activated by only A80 now.)
= F178 will stop all the Moveables in a given room that have been moved by PARAM_MOVE_ITEM.
= F179 will stop all the Statics in a given room that have been moved by PARAM_MOVE_ITEM.
= F177 will stop a given Static that has been moved by PARAM_MOVE_ITEM.
= F176 will stop all the Moveables in a given room that have been rotated by PARAM_ROTATE_ITEM.
= F175 will stop all the Statics in a given room that have been rotated by PARAM_ROTATE_ITEM.
= F174 will stop a given Static that has been rotated by PARAM_ROTATE_ITEM.

The ItemGroups:

ItemGroup Script commands define a bunch of Moveable or Static objects. These bunches can be triggered at the same time, using only one trigger for each activation. (So these triggers are linked to the ID number of the given ItemGroup command. The bunch – defined by that ItemGroup – will be triggered if that trigger is activated.)

- Activating F145 will activate all the objects of the given Moveable ItemGroup,
- Activating F146 will deactivate all the objects of the given Moveable ItemGroup,
- Activating F137 or F144 will move all the objects of the given Moveable ItemGroup linearly, continuously back and forth,
- Activating F138, F139, F140, F141, F142 or F143 will move all the objects of the given Moveable ItemGroup linearly.
- Activating F147 will stop the linear/rotative movements of all the objects of the given Moveable ItemGroup.
- Activating F354 in “Statics. Move. Stop the motion” mode will stop the linear movements of all

the objects of the given Static ItemGroup.
- Activating F354 in “Statics. Move. Stop all rotations” mode will stop the rotative movements of all the objects of the given Static ItemGroup. (See more about Static ItemGroups at the description of Static objects.)

Special effects on objects:

AddEffect Script command will define a visual effect you can attach to a given object (even Lara). (See ADD constants to define the type of the effect. See FADD constants for some special parameters of the effect. See JOINT constants to define the place of the effect. See MIST_COL constants to define the color of some effects.) – These triggers will place/remove the effect of the AddEffect with the given ID on/off the object:

- Activating A48 will place the effect on the given Moveable object.
- Activating A49 will remove the effect off the given Moveable object.
- Activating A83 will place the effect on the given Static object.
- Activating A84 will remove the effect off the given Static object.

Other tools in NG Center:

- Fast 3D function in Tools2 tab for these things of the given mesh: information about it, rotating/moving it, resizing it, mirroring it, making its textures shining/transparent, merging it with another mesh.
- Texturize DXF File function in Tools2 tab will merge meshes or restore the texture of the mesh exported from Metasequoia.
- Object Manager function in Tools2 tab will create an “encyclopedia” for your objects.

Other tools in Room Editor:

- Load Objects button is missing (is replaced by Last Projects button). But you can use SHIFT+O for the same purpose now.
- CTRL+F or Rotate Object button: new methods to rotate the selected Moveable/Static object around the vertical axis.
- Advanced Search button: a new searching method for placed effects, Moveable/Static objects, triggers or floor types (such as climb, monkey, toggle opacity etc.).
- Drop Down Menu Bar/Help/Info about Project or SHIFT+F1 to see the names of the WAD and TGA files (and their paths) attached to the actual project.
- CTRL+O: updates (reloads) the actual WAD of the project.
- Hide Objects button: if you are texturing your room then the placed objects sometimes bother you to do that. Switching on this button for hiding placed effects (light, camera, fog, sink, sound), Moveable or Static objects temporarily so they won’t bother you. (It is important to switch off this button with some other operations. If you don’t want to see the warning panel for that, then check Don’t show tips for Hide Objects button in NGLE Settings window.)
- Open any object list (Find Object, Edit Object etc.) then select the object, and click on Rename button to rename the object slot in your WAD. (Please note the new name will be shown only in Room Editor and the TXT file in wads folder. So, eg. you will see the original name in WADMerger.)

Only for Moveable objects:

Moveables (in Room Info Box), instead of the Objects info from TRLE: shows the actual and the maximum amount of the placed Moveable objects.

New object slots are created:

The objects themselves and the animations for some of them can be found in Paolone’s demo project files:

- the motorboat from TR2 Venice, with Lara's animations for driving it (MOTOR_BOAT, MOTOR_BOAT_LARA)
- the shoal of small fishes from TR3 jungles (FISH_EMITTER). Activate F343 to kill them
- the kayak from TR3 South Pacific, with Lara's animations for driving it (KAYAK, KAYAK_LARA)
- the diver (and his harpoon missile) from TR3 London (FROG_MAN, FROG_MAN_HARPOON)
- the rubber boat from TR3 Antarctica, with Lara's animations for driving it (RUBBER_BOAT, RUBBER_BOAT_LARA)
- the tightrope from TR5 Rome and VCI building (TIGHT_ROPE). Activate A77 to disable a tightrope, then A78 to enable that again
- the electric-eyed big Roman floating head (and his parts) from TR5 Rome (LASER_HEAD, LASER_HEAD_BASE, LASER_HEAD_TENTACLE)
- the dragon snake (and his missile) from TR5 Rome (HYDRA, HYDRA_MISSILE)
- the enemy submarine (and its missile) from TR5 Russia (ENEMY_SUB_MARINE, ENEMY_SUB_MARINE_MIP, SUB_MARINE_MISSILE)
- the swing bar from TR5 VCI building (PARALLEL_BARS)
- new BRIDGE objects tilted 3, 4 or customized clicks (BRIDGE_TILT3, BRIDGE_TILT4, BRIDGE_CUSTOM). See OCB values for the customization
- PANEL objects are simple, transparent objects with the only purpose to create collisions.
- FISH object slot is renamed into LOCUST_EMITTER.

Old object slots with new tasks:

- You can use the keypad of TR5 VCI building now, if you use the proper OCB values in a SWITCH_TYPE object slot. (Customize it using KeyPad Script command.)
- You can use TWOBLOCK_PLATFORM objects as elevators, if you type an Elevator command into the Script. (See EF constants for some customization of the elevator.)
Some elevator types won't move by itself, you need to activate A27 or (with a keypad) A28 to move the elevator.
- You can use a detector (as a QUEST_ITEM1 object) – as a tool of Lara to find the actual positions of the required Moveable objects – if you type a Detector command into the Script. (See DTF constants for some customization of the detector.)
Note: see Paolone's demo project about detectors in which you can also find MISC_SPRITES sprite-taking object, which is not used in TRLE, but now is useful for taking the sprites of the detector.
- Use CombineItems Script command if you want to create your own "merging the combo items" techniques. (So, this way eg. you can create a shotgun that is a combination of PICKUP_ITEM1 and PICKUP_ITEM2.)

New effects on objects:

a, effects about vitality:

- Activating A38 will hurt the given enemy.
- Activating A14 will kill/remove the enemy in some peculiar ways. (An enhanced version of F46 OldFlip trigger.)
- Activating A13 (in "ghost trap" mode) all the living WRAITH3 creatures will start towards the given LARA_START_POS, and will die if they slam into the floor during the time of the "travel". (Note: several modes of A13 trigger are not useable or not useful.)

b, effects about the look of the object:

- Activating A13 (in "hide object" mode) will turn the object into invisible. Activating A13 (in "show object" mode) will restore the visibility. (Note: if an object or a part is invisible that also means the object/part doesn't have collision.)
- Activating A53 will change the transparency of an object.
- Activating A82 will apply a preset effect of the game engine (for example: "the harpy getting

energy") on the given object.

c, effects about position:

- Activating A73, A74, A75, A76 (by HEAVY with a NEF_EASY_HEAVY_ENABLING constant for the enemy in an Enemy Script command) will move the enemy if he's standing on the trigger (as if he were standing on a conveyor). Activating A69, A70, A71, A72 will do almost the same, but only if the enemy is above the trigger (it is useful to drag flying/swimming enemies).
- Activating A40 will transport the enemy to a LARA_START_POS object with a given OCB value.
- Activating A58 will freeze the given enemy for the given seconds. Or, if the freezing is infinite, then unfreeze the object by A59.
But if you want to interrupt the actual animation of all the Moveable objects of the level, then activate F347. If the freezing is infinite, then unfreeze them by F348.

d, special effects:

- Activating A79 will change the OCB value of the given object. (Be careful: savegames sometimes are not able to contain these kinds of changes.)
- Activating F344 will change the horizon in the level.

Customization for objects in Customize Script commands:

- Use a CUST_KEEP_DEAD_ENEMIES constant so the dead enemies won't disappear (as before TR4).
- Use a CUST_WEAPON constant to customize the given weapon. (See WEAP constants for some new properties.)
- Use a CUST_HARPOON constant to transform your harpoon gun into a classic TR2/TR3 underwater weapon. (See HRP constants for some new properties.)
- Use a CUST_AMMO constant to customize the given ammo. (See AMMO constants for some new properties.)
- Use a CUST_FLARE constant to customize flares. (See FFL constants for some new properties.)
- Use a CUST_DARTS constant (with a given ID number) to customize dart emitters (having the same number in their OCB windows). (See DRT constants for some new properties.) – You also need a ColorRGB Script command if you want to change the dart color.
- Use a CUST_PARALLEL_BARS constant to customize the swing bar. (See PB constants for some new properties.)
- Use a CUST_WATERFALL_SPEED constant to adjust the speed and the direction of the waterfall.
- Use a CUST_SAVE_LOCUST constant so the locusts won't disappear when you save the game and then you load that.
- Use a CUST_LIGHT_OBJECT constant to customize AMBER_LIGHT, WHITE_LIGHT or BLINKING_LIGHT objects.
- Both jeep (ignition key) and motorbike (nitro) uses PUZZLE_ITEM1. If you want to use both the jeep and the bike in the same level then you can solve this problem by using a CUST_SET_JEEP_KEY_SLOT constant.
(Note: if you have both the vehicles in your level then use VEHICLE_EXTRA for the jeep. I.e. rename the VEHICLE_EXTRA of the motorbike to the new MOTORBIKE_LARA object slot.)
- Now the vehicles will collide with objects if Lara drives the jeep or the motorbike. But use a CUST_BIKE_VS_ENEMIES constant to customize the collision of the motorbike with the enemies. (Use the required HIT constant to define the type of the enemy collisions.)
- Use a CUST_ROLLING_BOAT constant to customize the pitch/swing mostly for boats, but even for anything else, if you want. (See FRB constants for some new properties.)

Other customization for objects:

- Enemy Script command will customize some properties of (not only) the enemies. (See NEF, TCF or EXTRA constants for some new properties.)
(Note: use a CUST_ADD_DEATH_ANIMATION constant in a Customize Script command to define the dying animation of a new mortal enemy, if you transformed an immortal enemy into mortal by Enemy Script command - see NEF_SET_AS_MORTAL constant.)
- Use AssignSlot Script command if you want to use “cloned” objects. For example, if you’d like to CROCODILE1 and CROCODILE2, though you know we have only one CROCODILE slot. - Notes:

= The clone objects will use the animations of the original object. (So it’s highly recommended the clone object will have the least important differences – mesh tree etc. – compared to the original one.)
= AssignSlot was originally made to make TRNG be able to use boats in TRNG, using OBJ constants. But it is not necessary any more in the newer TRNG’s.

New values to type in the OCB window:

If you want to use more than one OCB value at an object, then sum them up:

- SCORPION: 33 – now the scorpion attacks Lara even if another creature is living as well.
- JEEP: 1 – to switch on the headlights when Lara is in.
- MOTORBIKE: 1- to switch off the headlight when Lara is on.
- ROLLINGBALL: 1 – silent ball, 2 – hurts enemies, 4 – Lara can push it once, 8 – Lara can push it any times, 16 – smashes SHATTERs, 32 – naturally moving in water rooms, 64 – activates Lara’s triggers.
- FLAME_EMITTER3: 888 – the electric arch kills/ignites Lara, 889 – the electric arch hurts Lara till a burning death, 890 – the electric arch doesn’t hurt Lara.
- WATERFALLMIST: you need to use a complex formula that defines some features of the mist balls: the amount, the density, the size, the intensity of the pulsing, the color, the place on the square.
- PUSHABLE_OBJECT (any): 0-31 - the square-based object is so many clicks high, 32 – can be pushed off ledges, 64 – you need to use this number everyway for the new OCB features, 128 – cannot be pulled, 256 – cannot be pushed, 512 – cannot be moved in west-east direction (Room Editor facing), 1024 - cannot be moved in north-south direction, 2048 – the west side is climbable (if it is a minimum 8 clicks high, square-based object), 4096 - the north side is climbable, 8192 - the east side is climbable, 16384 - the south side is climbable.
- SWITCH_TYPE (1/2/3): (Lara’s switching-on animation ID)+8192 - Lara will use this animation for this switch. - Notes:

= The next animation ID must have the switching-off animation.
= Add 4096 to the value if the switch has State ID 1 in off position and State ID 0 in on position.

- SWITCH_TYPE (1/2/3/4): 16384+X – now you can use the keypad in this slot, in which X is the code. But if X is used by the 100X formula, then X=01-10 means the level to which the keypad will send the elevator.
- MOTOR_BOAT/RUBBER_BOAT: 1 – to switch on the headlight when Lara is in, 2- HEAVY type triggers won’t be activated by the boat, 4 – Lara in boat won’t activate her triggers, 8 – Lara can look around in the boat.
- FISH_EMITTER: 0 – default shoal, 1-127 - the amount of fishes, 128 - slow fishes, 256 - friendly fishes, 512 - single fishes (not in a shoal), 1024, 2048, 4096, 8192 – define the look of the fishes, 16384 - jumping out of water, 32768 – timid fishes.
- KAYAK: 1 – presents some wake, 2 – if the kayak has an animation for Lara jumping in from the bank, 4 - if the kayak has an animation for Lara getting in when the kayak is lying on the ground, 8 - Lara in kayak usually can activate her triggers, but can’t activate HEAVY triggers, but this OCB reverts that, 16 – performs rapid mist when the kayak comes down a slope that must be a rapid, 32 – Lara can look around in the kayak.
- LASER_HEAD: (see the description in NG Center. I think this OCB value doesn’t work

properly).
- HYDRA: 1 – for the first hydra, if there are two of them on a square, 2 – for the second hydra, if there are two of them on a square, 256, 512, 768, 1024, 1280, 1536, 1792 – to customize missiles, 2048 – the hydra won't produce a little flame before shooting, 4096 – the hydra won't shoot, will perform only direct attacks.
- ENEMY_SUB_MARINE: 0-4095 - customizes the frame value between two shots, 4096 – to attack Lara even when she's not in water, 8192 – disables bubbling from the submarine.
- PARALLEL_BARS: 100×A, in which A is the length of the jump (about in squares). (The value will be multiplied in the case of PB_PROGRESSIVE_CHARGE type customization.)
- TIGHT_ROPE: -1, 0, 1, 2, 3 - the difficulty of the tightrope walking.
- BRIDGE (any): 0-63 – CUSTOM_BRIDGE will be tilted so many clicks, 64 – now Lara can hang on a BRIDGE only with this OCB value, 128 – Lara won't slide on this BRIDGE tilted too much, 256×A – to define the depth of the BRIDGE, in which A is 1/16 square.

OCB Calculator function in NG Center/Tools2 can help you to calculate the complex OCB values.

Meshes:

- Activating A50 will turn a given mesh of a Moveable object into invisible.
- Activating A51 will turn a given invisible mesh of a Moveable object into visible again.
- Activating F341 will "transform the object slot into another one". (I.e. swapping all the meshes of two objects slots for each other, each mesh for the mesh with the same ID. The similar collision and pivots seem important.)
- Activating A37 will do the same that F341 does, but transforming a placed object, not a slot.

Collision:

- Activating A61 will remove the collision box of the object.
- Activating A62 will restore the removed collision box of the object.

Animation:

- Activating A15, A16 or A17 will force a new animation on the Moveable object (usually an enemy, of course).
- Activating A39 will force a new State ID for the next animation of the Moveable object (usually an enemy, of course).

AnimCommands:

There are some Flipeffect triggers (with "AnimCommand" name) that are not intended to place in the map but use always as an AnimCommand in WADMerger. - Triggers with AnimCommand name for not Lara are:

- Activating F87 will put the effect of the AddEffect Script command with the given ID on the object at that frame, for a given time. (Or, in infinite case, aborted by A49.)
- Activating F216, the turn counter will be reset directly at the given frame, using PB_PROGRESSIVE_CHARGE customization with swing bars. (The indirect method is when Lara slips aside on the bar.)

Inventory effects:

- Use an Equipment Script command (only at the level, which is the first level of your game!) to define the amount of an inventory item (weapon, ammo, medipack, key etc.) that Lara will have in the inventory when the game starts. (See LOAD constants if you want a weapon now in the inventory not having its basic state. For example, if you want the shotgun to have the wideshot ammo as chosen ammo now.) Value -1 means "unlimited".
- Activating F48 will give another piece of the selected item to Lara. Activating F49 (only at

multipliable items) will take a piece of that item from Lara. (Never use it to remove a weapon!)
- Activating F47 will take all the pieces of that item from Lara. (Never use it to remove a weapon!)
- Activating F50 (only at multipliable items) will set the amount of the pieces of that item, in the inventory.
- Activating F306 will pop up the inventory at the selected item.
- According to TRLE, the KEY_ITEM12 will be in the inventory "forever", after Lara picking that up. So she can use it again and again, that will remain in the inventory. You can do the same trick with other key and puzzle items (not COMBOs) in TRNG, using a CUST_SET_INV_ITEM constant in a Customize Script command.

Exportable triggers instead of non-exportable triggers:

If you want, use an A43 instead of a trigger to activate an object in the usual way or an A44 instead of an ANTI trigger to deactivate an object in the usual way. – Three things:

- As you know, you can't overlap two TRIGGERs if one of them activates an object with A timer value, but the other activates another object with B timer value. But it is possible, if two A43 are overlapped with different timer values.
- You can do some tricks with triggering the objects of the usual way if you switch on/off the codebit buttons in STT. It is not possible with ACTION triggers. (However, with clever triggering/scripting, using some TRNG features, there are solutions for that.)
- If you want to open/close doors you can also use A26.

Troubleshooting:

If an object is being moved then its position won't be saved sometimes. If you encounter this problem (mostly with ANIMATING objects) then activate A60 before the first save with that moving object.

Only for Static objects:

Statics (in Room Info Box): shows the actual amount of the placed Static objects.

About Static slots:

- One hundred new slots (with EXTRA name) are possible for more Static objects in your WAD. (Made by WADMerger.)
- Use a CUST_SHATTER_RANGE constant in a Customize Script command (making some other Static slots shatterable) if you think the ten shatter slots for your level is too low.

Values to type in the OCB window:

If you want to use more than one OCB value at an object, then sum them up:

- 4 – the collision box will be removed,
- 8 – the object has a so-called "glass transparency" (50% level of seeing its textures),
- 16 – the object has a so-called "ice transparency" (81% level of seeing its textures),
- 32 – Lara will be hurt when touching the object,
- 64 – Lara will catch fire when touching the object,
- 128 – Lara (and the object) will explode when she's touching the object,
- 256 – Lara will be poisoned when touching the object,
- 512 – the collision of the huger objects will be valid for the whole object,
- 1024 – this SHATTER can be smashed only in a "hard" way, i.e. only by exploding ammo, SPHINX, motorbike, jeep, boulder, falling PUSHABLE_OBJECT,
- 2048 – the HEAVY under the object will be activated when Lara's touching the object.

Customizing the values of the OCB features in Customize Script commands:

- Use a CUST_STATIC_TRANSPARENCY constant to customize the glass or ice transparency level value,
- Use a CUST_SET_STATIC_DAMAGE constant to customize the hurt and the poison value.

Some OCB features can be removed/restored by triggers (or can be added even if the object didn't have originally that feature):

- To remove: for collision box (F161), for glass/ice transparency (F165), fort hurt (F186), for burning (F188), for explosion (F182), for poison (F184).
- To add/restore: for collision box (F162), for glass transparency (F164), for ice transparency (F163), for hurt (F185), for burning (F187), for explosion (F181), for poison (F183).
- Activating F354 (in several modes of Window &) will also add/restore/remove the OCB features, but for a whole ItemGroup.

Other effects on the objects:

- Activating F160 the object will shatter. (Even if it is not a SHATTER.)
- Activating F180 the object will explode.
- Activating F189 the object will become invisible.
- Activating F190 the invisible object will become visible again.
- Activating F354 (in some modes of Window &) will also do what F160, F180, F189, F190 does, but for a whole ItemGroup.
- The object (only if it has the "classic", inner light attribute!) can have a special light/color effect, if you use a PARAM_COLOR_ITEM constant in a Parameters Script command. (See COLTYPE constants to define the effect type.) Activating F191 will start the effect that is defined in the PARAM_COLOR_ITEM with the given ID.
(Note: you also need one or two ColorRGB Script commands for PARAM_COLOR_ITEM to define colors for the effect.)
- The object (or all the objects of an ItemGroup) can be resized if you use a PARAM_SCALE_ITEM constant in a Parameters Script command. (See FSCA constants for some attributes of the effect.) Activating F352 will start the effect that is defined in the PARAM_SCALE_ITEM with the given ID. Activating F353 can stop the effect if that is continuous.

24. Timing techniques

Back to Top

- Organizer Script commands are used to activate TriggerGroups in a given order when there are more or less pause between two activations. (So Organizers are fabulous tools to define the sequence of happenings or to time things that can't be timed in other way. For example, switch on a flipmap with an F125 trigger in a TriggerGroup of the Organizer, then, after the required time, an F126 trigger – in the next TriggerGroup of the Organizer – will switch off the flipmap.) – See FO constants for some features of the Organizer.
The triggers for Organizers:

= Activating F127 will start the Organizer with the given ID number.
= Activating F128 will interrupt the Organizer.
= Activating F290 will start again the interrupted Organizer.

- Activating A52 will type the actual timer value of the given object on the screen.
- As you know there was a timer (the so-called "Screen Timer") in the "Race for the Iris" TRLR level on the screen when the young Lara and Von Croy were racing with each other. If you have a Timer command in the Script, then you can show that timer in your level as well (for any purposes). TRNG lets you to control that timer with F86 trigger.

25. Conditions

Back to Top

CONDITION triggers:

CONDITION triggers must be placed overlapped with executable triggers. The executable triggers will be executed only if the condition (defined in the CONDITION trigger overlapped with them) is true. – The CONDITION triggers are:

1. If the value of the “Trigger” window is OBJECT:

- C14: a given Moveable object is just active/inactive.
- C21, C23 or C24: a given Moveable object is just performing a given animation.
- C22: a given Moveable object is just performing an animation with a given State ID.
- C26: Lara is just touching a given Moveable object.
- C37: the given Moveable object has just a given transparency level.
- C54 or C55: Lara has just a given distance from a given Moveable object.

2. If the value of the “Trigger” window is PARAMETER:

a, inventory conditions:

- C1: the given item is just not in the inventory.
- C2: the given item (at least one copy) is just in the inventory.
- C3: there is just at least the given amount of the given item in the inventory.
- C4: there is just less than a given amount of the given item in the inventory.

b, conditions for Lara’s action:

- C5: Lara is just performing a given action (walking, monkey swinging etc.) or not.
- C30: Lara’s just performing a given animation.
- C31: Lara’s just performing an animation with the given State ID.

c, conditions for Lara’s position:

- “Fragmented” triggers: Lara is just in a given position of the square below her:

= C6, C7 or C8: the position is a strip, two strips crossed each other or a “mini square”.
= C60: the position is a circle, having the same center as the square.
= C61, C62, C63 or C64: the position is a quarter-circle (or a part of it), having the center on a square corner.
= C65, C66, C67 or C68: the position is a half-circle (or a part of it), having the center on the middle of a square side.
= C69, C70, C71 or C72: the position is a triangle (having one corner as one corner of the square).
= C73, C74, C75 or C76: the position is a triangle (having one side as one side of the square).
= C77: the position is a rhombus.
= C78: the position is a customized triangle defined in a Parameters Script command having a PARAM_TRIANGLE constant.
= C79: the position is a customized quadrilateral defined in a Parameters Script command having a PARAM_QUADRILATERAL constant.
= C80: the position is a circle (or a part of it), having the customized center defined in a Parameters Script command having a PARAM_CIRCLE constant.

(See more about fragmented conditions in “Miscellaneous2” demo project file of Paolone, the usage of them for BRIDGE objects, to create special collision on them.)

- C9: Lara is just in a given position above the trigger square.
- C10 or C11 (the two triggers do the same): Lara is just not in a given position above the trigger square.
- C81: the room where Lara just is is a snow/rain/water etc. room.

d, conditions with Lara and other objects:

- C27: Lara is just touching a Moveable object from a given object slot.
- C28: the creature Lara is just touching is friendly, a mortal enemy or an immortal enemy.
- C33: Lara is just touching a Static object from a given object slot.
- C34: Lara is just touching a given Static object.
- C35: Lara is just driving/holding the given object.
- C59: Lara has just selected an item (key, puzzle etc. but medipack/ammo/weapon) in the inventory.

e, other conditions for Lara:

- C17: Lara's found at least the given amount of secrets so far.
- C18: Lara's found exactly the given amount of secrets so far.
- C25: Lara just has a given status (poisoned, invulnerable etc.) or not.
- C29: Lara just has the given vitality.
- C36: the random value just generated when Lara is on the trigger area is a number from an interval (between 1 and a given value).

f, conditions for the player:

- C12: the player is just (not) hitting the given key.
- C13: the player is just (not) hitting the key that is attached the given key command.
- C19: the code typed last by the player in any keypad of the level is the given value.

g, other conditions:

- C15: the conditions defined in a given TriggerGroup Script command (see below) are just true.
- C16: the conditions defined in a given MultEnvCondition Script command (see below) are just true.
- C20: the so-called "Screen Timer" just has the given value. (Attention: "higher" and "lower" values are inverted in the trigger.)
- C32: the given animation range (except P-Frames) is just active or not.

g, conditions for variables:

- C38-C53, C56-C58: see more about them at the description of the variables/Memory Zones.

"Multi" conditions:

Using MultEnvCondition Script command is a method to define special conditions for (C16) CONDITION triggers:

- because you can define complex conditions in MultEnvConditions, or
- because some peculiar (but simple) condition cannot be defined directly in CONDITION triggers, but only in MultEnvConditions.

(See ENV constants to see the type of the conditions in this Script command. Other – additional - constants that are used by MultEnvConditions: ROOM, DENV, HOLD.)

Conditions in TriggerGroups:

As I said above (see C15 trigger) we can have one or more condition triggers in the TriggerGroups. It is useful, because (due to the rules of the overlaps) we are not allowed to place more than one CONDITION trigger on a square.
So, for example, if we want to place one and another one condition in one and another one CONDITION trigger, then we'll do something else: place only one C15 trigger, that is linked to a TriggerGroup. In the Script, that TriggerGroup will have that two CONDITION triggers.

"Mixed" TriggerGroups – the basics:

But we can also have "mixed" TriggerGroups, in which we have both executable and CONDITION triggers. – See this very easy example:

TriggerGroup= TriggerGroup ID, CONDITION trigger, executable trigger

Instead of placing a C15 CONDITION trigger and an executable trigger on the same square, place only the F118 trigger to start that TriggerGroup. The same thing will happen: if F118 is triggered then the executable trigger in the TriggerGroup will be executed only if the CONDITION trigger in the TriggerGroup is true. (Note: don't forget to set F118 triggers as "multiple" triggers in trigger Window E, if the TriggerGroup linked to them have one or more CONDITION triggers.)
Or a special example:

TriggerGroup= TriggerGroup ID, executable trigger1, CONDITION trigger, executable trigger2

Now that's what will happen: activating F118, the first executable trigger will be activated everyway, but the second executable trigger only if the CONDITION is true.
But naturally we can have TriggerGroups with more than one CONDITION triggers. – Let's see this easy example:

TriggerGroup= ID, ConditionTrigger1 (CT1), ConditionTrigger2 (CT2), ExecutableTrigger1 (ET1), ExecutableTrigger2 (ET2)

So, ET1 and ET2 will be executed only if both CT1 AND CT2 are true. (Note: you don't need TGROUP_AND constant to define "AND" connection between the conditions. Forget about that constant.)
But the trigger order is very important now. – You will understand that if we change the trigger orders in the TriggerGroup just above:

TriggerGroup= ID, CT1, ET1, CT2, ET2

This TriggerGroup says this: if CT1 is true, then ET1 will be executed AND CT2 will be studied. If CT2 is also true, then ET2 will be executed.

"OR" connections in "mixed" TriggerGroups:

But we can have not only AND connections in mixed TriggerGroups. If you want "OR" connection, then use a TGROUP_OR constant. – For example:

TriggerGroup: ID, CT1, CT2+TGROUP_OR, ET1

This TriggerGroup says this: if CT1 is true OR if CT2 is true then ET1 will be executed.
Or let's see a bit more complex case:

TriggerGroup= ID, CT1, CT2, CT3+TGROUP_OR, ET1

This TriggerGroup says this: if CT1 is true and if (CT2 or CT3) is true then ET1 will be executed.

"NOT" connections in "mixed" TriggerGroups:

See for example this for a “NOT TRUE” situation, using a TGROUP_NOT constant:

TriggerGroup= ID, CT1, CT2+TGROUP_NOT, ET1, ET2, ET3

This TriggerGroup says this: if CT1 is true AND if CT2 is NOT true, then ET1 and ET2 and ET3 will be executed. – But let’s change it a bit:

TriggerGroup= ID, CT1+TGROUP_OR, CT2+TGROUP_NOT, ET1, ET2, ET3

This TriggerGroup says this: if CT1 is true OR if CT2 is NOT true, then ET1 and ET2 and ET3 will be executed.
(As you see, the TGROUP_OR can also be connected to the first condition, if that is followed by a TGROUP_NOT.)
And now let’s see a “double negation”:

TriggerGroup= ID, CT1+TGROUP_NOT, CT2+TGROUP_NOT, ET1, ET2

This TriggerGroup says this: if CT1 is NOT true AND/OR if CT2 is NOT true, then ET1 and ET2 will be executed.
(AND/OR means sometimes it’s not easy to judge, if you have two TGROUP_NOT constants after each other, then that is a “not OR not” or a “not AND not” situation. So be careful with adjusting your setup here, test the situation cautiously.)

“ELSE” connections in “mixed” TriggerGroups:

There is one more constant to control the mixed TriggerGroups. It is TGROUP_ELSE. – Let’s see a simple example:

TriggerGroup= ID, CT1, ET1, CT2+TGROUP_ELSE, ET2

This TriggerGroup says this: first of all, the game will study CT1 condition. If that is true, then ET1 will be executed. But if CT1 is not true, then the game will skip ET1 and will study CT2 condition. If CT2 is true then ET2 will be executed.
At last, let’s see a complex example:

TriggerGroup= ID, CT1, CT2+TGROUP_OR, ET1, ET2, CT3+TGROUP_ELSE, ET3, >
CT4+TGROUP_ELSE, CT5+TGROUP_NOT, ET4, ET5, ET6

This TriggerGroup says this: the game studies CT1 and CT2. If any of them is true, then ET1 and ET2 will be executed. If none of them is true then CT3 will be studied. If that is true than ET3 will be executed. But if it is not true, then the game will study CT4 and CT5. If CT4 is true and CT5 is not true then ET4, ET5 and ET6 will be executed.

“OR” or “NOT” connections in TriggerGroups with only condition triggers:

You can use OR and NOT connections even in C15 triggers. – Let’s see an example:
If there’s an executable trigger placed, overlapped with a C15, and this is in the TriggerGroup linked to C15:

TriggerGroup= ID, CT1, CT2+TGROUP_OR

then the executable trigger will be activated if CT1 or CT2 is true.

GlobalTriggers:

GlobalTriggers are triggers that use conditions. If the condition is true then an/more executable

trigger(s) (the “True Trigger”) will happen. If it is not true, then nothing will happen or (an)other trigger(s) (the “False Trigger”) will happen.
GlobalTriggers are “global” because they will study the condition during the whole level. I.e. the GlobalTriggers will be activated automatically at each frame of your level. – See this to understand:

1. The level starts.
2. Frame#1: the GlobalTrigger study its condition. If it is true then the executable trigger(s), in a TriggerGroup linked to the GlobalTrigger will happen. If it is not, then nothing will happen – or (an)other executable trigger(s) in a TriggerGroup (if it exists), also linked to the GlobalTrigger, will happen.
3. Frame#2: the GlobalTrigger study its condition. If it is true then the executable trigger(s), in a TriggerGroup linked to the GlobalTrigger will happen. If it is not, then nothing will happen – or (an)other executable trigger(s) in a TriggerGroup (if it exists), also linked to the GlobalTrigger, will happen.
Etc.

GlobalTriggers are triggers that CANNOT be placed. All you need to do to define a GlobalTrigger is to type a GlobalTrigger Script command. (See FGT constants for some features. See GT constants to define the condition for the GlobalTrigger. See TSCR constants for GT_TITLE_SCREEN condition. See GTD constants for GT_DISTANCE… conditions. See HOLD constants for GT_LARA_HOLDS_ITEM condition. See KEY1 or KEY2 constants for GT_GAME_KEY… conditions. See Keyboard Scancodes in NG Center/Reference for GT_KEYBOARD_CODE condition.)
The disabled GlobalTriggers won’t do anything, won’t study their conditions. If you want to disable a GlobalTrigger permanently, then these can be your reasons:

- in a given section of the level you don’t want the GlobalTrigger to activate any executable trigger, or
- you are sure that nothing will happen in a given section of the level that the GlobalTrigger should study, so you disable the GlobalTrigger to save some memory for the engine. (Other trick for that is the “one shot mode” by an FGT constant: the GlobalTrigger did what it had to do, so you don’t need that any more, a “one shot” mode won’t let that be activated any more.)

Use FGT_DISABLED constant so the GlobalTrigger will be disabled when the level starts. Use F109 to disable/enable the given GlobalTrigger any time during your level.

“Tricks” with GlobalTriggers:

GlobalTriggers can use several types of conditions, including a TriggerGroup “full of conditions” (as if that were a TriggerGroup for a C15 trigger) – but you can use a trick “against” that “full of conditions” thing.
I mean, GT_ALWAYS constant means the condition is always true, the GlobalTrigger will activate the True Trigger at each frame of the level. But you can place conditions in the TriggerGroup of the True Trigger. – Let’s see an example:

GlobalTrigger= 1, IGNORE, GT_ALWAYS, IGNORE, IGNORE, 1, IGNORE
TriggerGroup= 1, CT1, ET1, CT2+TGROUP_ELSE, ET2

So, GT_ALWAYS lets TriggerGroup=1 be activated at each frame of the level, but ET1 will happen only if CT1 is true, and ET2 will only happen if CT1 is not true and CT2 is true.

After that, it’s easy to understand, that both the condition and executable TriggerGroups of the GlobalTrigger can use both condition and executable triggers. – For example, with GT_CONDITION_GROUP (so if the condition of the GlobalTrigger is “only” a condition TriggerGroup):

GlobalTrigger= 1, IGNORE, GT_CONDITION_GROUP, IGNORE, 1, 2, IGNORE
TriggerGroup= 1, ET1, ET2, CT1
TriggerGroup= 2, CT2, CT3+TGROUP_OR, ET3

So, ET1 and ET2 will be executed at each frame of the level. (It is important now, because ET1 and ET2 will define the needful parameters of CT1). If CT1 is true (using the actual values from the parameters defined by ET1 and ET2) then the game will study CT2 and CT3. If any of them is true, then ET3 will be executed.
(This "condition TriggerGroup for GlobalTriggers, also having executable triggers" makes sense if the subject of the condition is not constant, so if the condition can have various parameter values. – You will understand it at the description of the variables.)

26. Lara

Back to Top

CTRL+H or Move Lara here button (in Room Editor): the object of Lara will be transferred to the selected square.

Effects on Lara:

Effects about vitality:

- Activating F89 will decrease Lara's health. Activating F90 will increase that.
- Activating F113 will poison Lara. Activating F114 will remove the poisoning.
- Activating F92 will extinguish the flames on Lara.
- Activating F63 will kill Lara in the required way.
- Activating F91 Lara will be invulnerable for a given time. (Abort the invulnerability by F93 if that is infinite.)
- Activating F110 will give Lara some extra air under water.
- Activating F104 will (not) let Lara to have infinite air under water.
- Activating F111 will decrease the damage when Lara is in a so-called damage room.
- Activating F112 will decrease the damage of the cold when Lara is in a cold water pool.

Effects about the things in Lara's hands:

- Activating F108 will remove the holsters/weapons from her body (not preventing her from using the weapons), or restore them.
- Activating F107 will remove the weapons from her body and she can't use them, or restore them.
- Activating F96 will take the weapons/ammo from Lara.
- Activating F83 will make Lara holster weapons/drop flare or torch.
- Activating F199 will light or put out the torch in Lara's hand.
- Activating F200 will give the torch into Lara's hand or remove that from there.

Effects about position:

- Activating F134, F135 or F158 will move Lara forward (as if she were on and/or over a conveyor) while she's in the area of the trigger.
- Activating F136 will adjust the gravitation, i.e. push Lara upwards or pull her downwards.
- Activating F79 will transport Lara to a LARA_START_POS object with a given OCB value.
- Activating F349 or F350 will simulate a whirl in the water (positioned at a LARA_START_POS object with a given OCB value) that will drag Lara to itself.

General customization for Lara in Customize Script commands:

- Lara's health will be 100 % if she jumps a level. If you want her to keep the actual vitality when jumping a level, then use a CUST_KEEP_LARA_HP constant. (With KLH_ALL_LEVELS

constant, that will be true for all the levels of the game.)
- In the Tomb Raider games before TR4 Lara wasn't transparent when looking around. Use CUST_LOOK_TRASPARENT constant if you want to restore that old feature.
- If you'd like to make Lara collide with objects as if she were colliding with room walls then use a CUST_SET_STILL_COLLISION constant. (See COLL constants for some new properties.)
- Use a CUST_TR5_UNDERWATER_COLLISIONS constant if you want Lara to collide underwater with the walls as in TR5. (When she collides in TR4 she continues the swimming movements, but naturally won't move further. In TR5 she will be diverted when colliding.)

Animation:

See Animation Watcher tool in NG Center/Tools to study the animations.

Forcing animation:

- Activating F78 will force a new State ID for Lara's actual and next animation.
- Activating F77, F80, F169, F170 or F171 will force a new animation on Lara.

Making customized animations:

Animation Script command helps you to define the circumstances of a custom animation of Lara.
– Notes:

- If you want to start the custom animation by a key command then see KEY1 or KEY2 constants. If you want to start that by a key then see Keyboard Scancodes in NG Center/Reference.
- See FAN constants (and PLACE constants for FAN_SET_LARA_PLACE) for some features of the command.
- Use ENV constants (just as MultEnvCondition) or STATE constants to define some conditions for the animation.
- Other – additional - constants that are used by Animation (just as MultEnvCondition): ROOM, DENV, HOLD.

Customization for the animations in Customize Script commands:

- The #96 hanging animation is sometimes forced by the game engine when Lara is hanging. If it disturbs the setup of your custom hanging animation then disable the force, using a CUST_DISABLE_FORCING_ANIM_96 constant.
- The animation when Lara is pushing a rollingball can be customized if you use a CUST_ROLLINGBALL_PUSHING constant.
- If you don't want Lara to be snarling when she's shooting then use a CUST_DISABLE_SCREAMING_HEAD constant.

The perfect position as a base for Lara's animation:

TestPosition Script command defines a position for Lara, compared to a position of a Moveable object. This Lara position will be used by an ENV_ITEM_TEST_POSITION constant, in MultEnvTestPosition or Animation Script commands. (See TPOS constants for some TestPosition features.)

AnimCommands:

Triggers with AnimCommand name for Lara are:

- F101: if Lara is in the area of a HEAVY type trigger when the frame of this AnimCommand is being performed then that HEAVY will be activated.
- F102: this AnimCommand will do the same as Play Effect – Change Direction AnimCommand

does. But this time you can turn not only 180 degrees but 45, 90 or 135 degrees as well.
- F103: if the position of the hanging Lara is problematic after the animation (though you used Play Effect – Change Direction or F102 to define the proper position), then add F103 AnimCommand to a (any) frame of the animation, to solve the problem.
- F211: this AnimCommand will do the same as Command3 AnimCommand does. But this time the free hand effect will also work with the torch.
- F212: this AnimCommand will do the same as Command3 AnimCommand does. But this time the free hand effect will be aborted in the given time (so before the animation ends). (If you use F212 in a "forever" way, then activating F213 will abort the effect.)

Outfit:

- Remap Lara skin function in NG Center\Tools2 will help you to create a new outfit for Lara.
- F99 will swap Lara given meshes for the meshes placed in other Moveables object slots – this way:
For example, the selected slot in SKELETON_MIP and the selected method is "Lara Skin + Lara Joints (Slot+1) + Hairs (Slot+2) + ShootingHead (Slot+3)". It means, if you activate the trigger, then the skin will be swapped for with SKELETON_MIP (slot ID 36), the joints will be swapped for GUIDE (36+1=Slot ID 37), the hair will be swapped for GUIDE_MIP (36+2=Slot ID 38), and the screaming head will be swapped for VON_CROY (36+3=Slot ID 39).
The undermentioned triggers will work in a similar way:

= F105 will work as F99 in "One Shot" mode.
= F106 will, after all, disable any further swaps for the given elements.

- If you want some different hair types (so, if you want, for example, the young Lara's hair types for the adult Lara), then "inform the game about your idea", using a CUST_HAIR_TYPE constant in a Customize Script command. (See HAIR constants for the hair types.)
- F100 will swap a single mesh of Lara.
- F340 will swap the meshswap objects of Lara (if you want a new meshswap object for that function).

Note: at some points, TRNG outfit features for Lara seem buggy.

27. Level jumps
Back to Top

- F82 is useful instead of FINISH trigger, if you'd like to export the FINISH. (Moreover, F82 can use a timer for some delay, so now the trigger can be activated only when some time elapsed after the activation.)
- ResetHUB Script command will remove all the Key, Puzzle, Pickup and Examine items from the inventory. You can prevent that (except at Examine items), using a PreserveInventory Script command.

28. Cutscenes and FMV's
Back to Top

FMV's:

- Use WMV, MPG or AVI, skipping the original BIK format of TR4. (Don't forget to type the required format, into the PCExtensions block of the Script, replacing BIK.)
- The FMV name you must use is "FMV" and the ID, so, for example, FMV60.AVI.
- Place the FMV files into the TRLE main folder, or create a subfolder there (with FMVs or Store name) for them.
- An FMV Script command in your [Level] block is needed for that FMV file.
- Use an FMV trigger to play an FMV file.
- I don't recommend trying to play an FMV in more than one level. And you should also avoid to

try to play an FMV in a level, if you have played in that level once before.
- Launch setup_wmv_encoder.exe in Tools folder and now you can use WMV Encoder in NG Center/Tools 2 to convert AVI to WMV.
- Use a CUST_FMV_CUTSCENE constant in a Customize Script command to customize the FMV's. (See FMV constants for the possibilities.)
- Activating F54 will present a short "black/actual frame frozen" screen, for a given time. (It is useful – overlapped in "forever" mode with the FMV trigger - if you don't want an FMV_FADE_OUT customization for all the FMV's but only for the given one.) If this effect is infinite (except with that FMV case) then use F55 to abort it.

Cutscenes:

- If you want your whole level to be a cutscene level, then type a CutScene command into the Script.
- Activate F84 or F85 to present a fade-in or a fade-out on the screen, for a given time. (Usually it is naturally useful at the start/end of the cutscenes.)

29. Title
Back to Top

- Use ShowLaraInTitle Script command so Lara will be there in the title screen. (Cannot be controlled by the player, but can be controlled by cutscene methods.)
- Use a CUST_TITLE_FMV constant in a Customize Script command for an FMV (defined by an FMV Script command, as usual) that will start before the title flyby itself will start.
- If the amount of your levels is X then a FINISH trigger, that will load Lara to X+2 level, will load her to the title. But if that loads her to X+1 level, then it will load her to the end credits sequence – that can be familiar for you from the Last Revelation.
But that X+1 will work only if you use a CUST_SET_CREDITS_LEVEL constant in a Customize Script command.

30. Technical tools
Back to Top

Technical settings for the game:

- Turbo Script command will let you to prevent the technical issues caused by the slow frame rate. (See TRB constants for the type of solutions. – Be careful: some of them seems buggy.)
- Settings Script command will let you to achieve special technical settings, such as the player can edit savegames or not etc. (See SET constants for the setting types.)

Log tools for programs:

Log for Room Editor:
Start Tomb5_Log.exe in Tools folder, then start the Room Editor. See the log entries in the window of Tomb5_Log.exe.
When you exit the log.exe, then you can see the tomb5_log.txt (in Tools folder), containing all the log entries created after the last launch of the log.exe, but before this exit.
Note: if you hit CTRL+S (when the log.exe is running) then this bookmark will be placed in the log:

================ USER HIT CTRL+S KEYS ==================

Log for the game:
Start Tomb4_Log.exe in Tools folder, then start the game. See the log entries in the window of Tomb4_Log.exe.
When you exit the log.exe, then you can see the tomb4_log.txt (in Tools folder), containing all the

log entries created after the last launch of the log.exe, but before this exit.
Note: F308 will print the value of the given [ExtraNG] string into the log. (It is useful to create bookmarks in the log.) – Temporary trigger, you should remove it before releasing your level.

Diagnostics:

Diagnostic command:

The diagnostic messages (can be formed by only FONT_GRAPHICS) are messages on the game screen, about some actual values in the game (the ID of the audio file that is just playing, the ID of the animation of Lara she is just performing etc).
These messages are necessary only if you want to know a value of a parameter of the game, when you're editing your level. (So always remove diagnostic messages everyway before releasing your level.) The game will put the messages automatically on the screen, if you use Diagnostic Script command.

DiagnosticType command:

But there are a lot of diagnostic messages, so the bottom part of the message list won't fit the screen. You also need a DiagnosticType Script command which will separate the message list into one or more messages, and only the given part of the list will be printed on the screen.

- See DGX constants that will define the parts of the list. (There are some special constants. For example, DGX_ANIMATION lets you test the circumstances of your custom animation when the Animation command has been typed yet but the animation itself hasn't been created yet. Or DGX_CHEATS lets you to apply other test cheats, not only DOZY. Or DGX_LOG_SCRIPT_COMMANDS won't place anything on the screen, but "only" put TriggerGroup, Organizer, GlobalTrigger values as well into the log file. Etc.)
- See EDGX constants for some specialties. (For example, use EDGX_CONCISE_SCRIPT_LOG to move the unimportant data from the long log file. Or EDGX_SLOW_MOTION – only for this test! – will slow down the frame rate of your level. Etc.)
- Hit Key H to remove Diagnostic messages from the screen (for a couple of seconds) if they just disturb you.
- Hit Key F9 that will probably disable the newer log entries in the long log, till the key is being pushed down. (Works probably only with some DGX constants.)

Note: sometimes the things happen in the game very fast. That's why the game log TXT is very important, if you want to study the values but you can't follow their fast changes either in log.exe window or in the Diagnostic messages.

LogItem command:

LogItem Script command is a helpful tool for Diagnostics, because if you use this command, typing the ID of the given object here, then the parameters of that object will also be typed on the screen (can be formed by only FONT_GRAPHICS). (See FLI constants for some special features.) – It is useful, for example, if you want to create a TestPosition Script command to define the position difference between Lara and an object, and you need to see the position values of the object.

Additional tools:

Project analysis:

See NG Doctor function in NG Center\Tools2 for a project analysis.

The probability of crash:

Maybe it's important to let your programs crash with more little problems, so you'll know if something goes wrong and you need to re-edit something:

- Disable Crash Resume System (in NGLE Settings window): check it and Room Editor will be more inclined to crash.
- The higher inclination to crash is the default status of the game. Type a CRS=ENABLED Script command for a lower inclination.

Automatic attachments:

You can use ImportFile Script command not only for audio files.
For example, you want a TXT file to be there everyway at the player, so you don't want the TXT to be "lost" somehow when the player download your game. – Now this TXT will be a part of Script.dat and will be put into the player's computer when he/she runs first the game (so when he/she uses first the Script.dat), on the same path wherefrom it was saved.
Place the file somewhere in TRLE main folder, and use an ImportFile Script command that will place that file into Script.dat when you're building the Script next time. (So you don't need to attach the file directly to your game.)
Use FTYPE_USERFILE constant in the command. It means this file is NOT imported as an audio file this time.
And use IMPORT_TEMPORARY this time. (Now you don't have an aggregated size limit for the files.)

31. Variables

Back to Top

The purpose of the variables is "only" to take different numeric (or sometimes textual) values. So the variables are independent of any concrete feature of the game engine. That's why if you have a value in a variable then you can apply that value to control various TR4/TRNG procedures.

Notes:

- The default value of the numeric variables is 0, of the textual variables is empty.
- The numeric variable values can't have digits after the decimal point. So, for example, Value 25,76 will become 25 in the variables.

The possible variables are:

A, Variables for numeric purposes:

Current Value
Global Byte Alfa1-4, Beta1-4, Delta1-4
Global Long Alfa, Beta, Delta
Global Long Timer
Global Short Alfa1-2, Beta1-2, Delta1-2
Last Input Number (for the numbers the player typed in)
Local Byte Alfa1-4, Beta1-4, Delta1-4
Local Long Alfa, Beta, Delta
Local Long Timer
Local Short Alfa1-2, Beta1-2, Delta1-2
Store Long A-P
Store Short A1-P1, Store Short A2-P2
Store Byte A1-P1, Store Byte A2-P2, Store Byte A3-P3, Store Byte A4-P4

B, Variables for textual purposes:

Text1-4 (maximum for 79 characters)

Last Input Text (maximum for 79 characters the player typed in)
Big Text (maximum for 319 characters)

Rules to select a variable:

It doesn't matter which variable you'll use for the given purpose – if you don't forget about these things:

1. Globality/locality:
- Variables with "global" name (including Current Value, Last Input Number, Store variables and the textual variables) will be changed only if you change them. (But be careful with ResetHUB Script command!)
- Variables with "local" name will keep their values only in the given level.
2. The size of the numeric variable:
- The value of a "byte" variable must be always between 0 and 255.
- The value of a "short" variable must be always between -32 768 and 32 767.
- The value of a "long" variable (including Current Value and Last Input Number) must be always between -2 147 483 648 and 2 147 483 647.

See Paolone's tutorial about variables to understand: variables with different sizes cannot always be used at the same time.
3. You should use a given variable only for a given purpose at the same time. Or else the things may mess up.
4. Special variables:
- Sometimes Current Value is the only variable you can use for a special purpose. So if you can use another variable for a non-special purpose then it's better if you don't "waste" Current Value for the non-special purpose.
- Last Input Number works only with keypad switch at the moment. (But you can also use it as a usual "Global Long" variable.)
- There is no function in the game now that can be used with Last Input Text. (But you can also use it as a usual Text1-4 variable.)

The usage of the variables:

The main usage of the variables:

- If you have a numeric value in a variable then you can use it as a condition. (For example, "if the variable is 1 then do this, or if the variable is 2 then do that".)
- If you have a numeric value (in a variable or defined directly by a trigger) then you can force it into a Memory Zone field. (To change the value of the property of that field.)
- If you have a numeric value in a Memory Zone field then you can force it into a variable. (For example, to study the field value with a condition.)
- Local/Global Long Timer variables are timers. The numeric values in these variables are the values of the timer. (1 second means Value 30 in the variable.)
- If you have a numeric or a textual value in a variable then you can print that value on the screen.
See the "Variable Placefolders" list in NG Center\Reference. You can see the ID's of the variables here. For example, the ID of Local Long Alfa variable is #0070. Type this #0070 ID number into a string of the English.txt, then use one of the Independent Text-printing triggers (see above) to print that string on the screen. (So the game will print a given string on the screen, but that will not always mean the same value, because always the actual value of the variable is what will be printed.)
Or see this example: #0020 is Text1 variable. The value of the variable could be: hawk, tiger, warrior. If the actual value of the variable is "tiger" then this is what will be printed on the screen: "A tiger is just attacking Lara!", if this is what is typed into that string: "A #0020 is just attacking Lara!"
(Note: you may try other methods to print a variable value from a string. See for example the

name of the customized bar.)
- If you have a CUST_BAR for a customized bar, then you can type a numeric variable ID (for example #0070 for Local Long Alfa) there. The actual value of the variable will be the actual value of the bar. (Note: the bar can show only 0, 1, 2… 99, 100 values. So if you have a value in the variable below/above that interval, then you need a mathematical operation to transform the value into another value so that the new value will fit the interval.)

The additional usage of the variables:

- If you have a numeric value in a variable then you can do mathematical operations, using this variable and another number (in another variable or defined directly by a trigger) to change the variable value. - Notes:

= If you execute an operation with a value of a variable, then the old value of the variable will be overwritten, becoming the new value. (For example, Variable A is 7. Adding 5 to Variable A, the Variable A becomes 12.) – See some exceptions below.
= There is an "adding mathematical operation" with textual variables as well: you can attach one text to another.

- If you have a value (in a variable or defined directly by a trigger) then you can change the value of another variable with that. (For example, Local Short Alfa1 is just 9000. You copy that value into Global Long Timer which will turn the timer value into 300 seconds, because Value 30 is 1 second, so 9000/30=300.)
- You can do mathematical operations, using a Memory Zone field value and another number (in another variable or defined directly by a trigger) to change the field value.
- The Store variables are used to store numbers for a later application or to record object positions.
- There are some special tasks for the variables.

The triggers used for variables:

Triggers to copy the numeric value of a variable into another numeric variable:

- F271 will copy the value of any numeric variable into Current Value.
- F272 will copy the value of Current Value into any numeric variable.

Triggers to force a number (defined by the trigger) into any numeric variable:

F232, F234, F252 or F263 will force the number into any numeric variable.

Note: before forcing the value, turn the value of the variable into 0, if you experience the value is not proper after the forcing.

Mathematical operations:

- Between two numeric variables:

= F285 will add the value of any numeric variable to Current Value. (Overwriting "Current Value".)
= F286 will subtract the value of any numeric variable from Current Value. (Overwriting "Current Value".)
= F288 will multiply the value of Current Value by any numeric variable. (Overwriting "Current Value".)
= F287 will divide the value of Current Value by any numeric variable. (Overwriting "Current Value".)

Note: dividing a variable value by 0 won't happen.

- Between a number (defined by the trigger) and a numeric variable:

= F231 will add the number to any numeric variable.
= F233 will subtract the number from any numeric variable.
= F251 will multiply any numeric variable by the number.
= F253 will divide any numeric variable by the number.

- Other mathematical operations with numeric variables:
F284 will invert the sign of any numeric variable.

- Turning the value of a numeric variable into 0:

= F235 will remove the value from any numeric variable (if the value is a bit, i.e. the exponent of Value 2).
= F241 will remove the values from all the given numeric variables.

Triggers for timers:

= F264 will start the timer.
= F265 will stop the timer.
= F266, F267 or F268 will define the time from which the decreasing timer starts/where the increasing timer stops.
= F269 will put the timer on the screen.
= F270 will take the timer off the screen.

Note: you can form the look of the timer only using FONT_GRAPHICS or F269. If you also want to form other things (for example font color) then skip the now mentioned triggers to control the timer. Instead of that, put the ID of the Local/Global Long Timer into a string, and print/remove that string on/off the screen as if that were an Independent Text. (With some method – eg. GlobalTrigger – for the continuity, so the time value will be refreshed continuously, as if it were a real timer.) Now you can form the timer as if that were an Independent Text.

Triggers for Store variables:

- The usual usage of Store variables:

= F236 will copy a value of Current Value into a Store variable.
= F237 will copy the value of a Store variable into Current Value.

- The special usage of Store variables:
A55 will record the actual position of a Moveable object in a Store variable. It is important if you want to move this/another object later/now to the same position, activating A57 or (also receiving the horizontal position on the square, from A55) A56.

Triggers for the textual variables:

- F238 will copy the contents of a textual variable into another textual variable.
- F239 will copy the contents of an [ExtraNG] string into a textual variable.
- F240 will attach the contents of an [ExtraNG] string to Big Text variable.
- F242 will attach the contents of a textual variable to Big Text variable.
- F243 will attach the value of a numeric variable to Big Text variable.

The triggers for the Memory Zones:

See them below.

CONDITION triggers with variables:

- C38, C39 or C40: compares the actual value of the given numeric variable to a given number with PARAM_BIG_NUMBERS constant.
- C41, C42 or C43: compares the actual value of the given numeric variable to a given number.
- C44: the actual value of the variable should be bigger than (or equal with that) the “bit” value which is an exponent of Value 2.
- C45: the actual value of the variable should be smaller than the “bit” value which is an exponent of Value 2.
- C46, C47 or C48: compares the actual value of Current Value variable to a given number.
- C56, C57 or C58: compares the actual value of Current Value variable to a given another numeric variable.

Special triggers:

- Switch function:
F289 will activate a “program” that is defined by a given Switch Script command. (See SWT constants for some features.)
A #xxxx ID code of a numeric variable is typed in the Switch command. And some TriggerGroups are also identified in the command. Each TriggerGroup is linked to 1, 2, 3 etc. numbers. So, if the value of the variable is 1 then the TriggerGroup linked to 1 will be activated, if the value of the variable is 2 then the TriggerGroup linked to 2 will be activated etc.
(Note: thanks to Switch command feature, now eg. you have the possibility to place a keypad switch that has more than one good code.)

- Creating a random value:

= F303 will generate a random number for the given numeric variable, from the given interval.
= F304 will generate a random number for the given numeric variable, between 0 and the non-0 Current Value.

- Tomb (game) index/NGLE (Room Editor) index transformations:

= If you have the tomb (game) index of a room in Current Value, then F298 will transform that index into the NGLE (Room Editor) index of the room (in Current Value).
= If you have the NGLE (Room Editor) index of a room in Current Value, then F299 will transform that index into the tomb (game) index of the room (in Current Value).
= F297 will also do something with room NGLE/tomb index. But don’t use this trigger because it seems buggy.
= If you have the tomb (game) index of a Moveable object in Current Value, then F300 will transform that index into the NGLE (Room Editor) index of the object (in Current Value).
= If you have the NGLE (Room Editor) index of a Moveable object in Current Value, then F301 will transform that index into the tomb (game) index of the object (in Current Value).

- Other special triggers:

= F305 will set a new value for the given variable as the binary aggregate of the variable value and the given number. – For example:
The variable value is just 7. It is 111 as a binary number.
The given number is 99. It is 1100011 as a binary number.
The aggregate is 3.

0000111+
1100011=
0000011

Because you need to add the each upper digit to the digit below that, and these are the rules:

0+0=0
0+1=0
1+0=0
1+1=1

And 11 binary number is 3 as a decimal number.
= You need a ColorRGB Script command. If you activate F291 then the color code of the command will get into the given variable as a long number. (It is useful if you can force a long number color code like that into a feature to force that feature to apply that color code. But F291 won't do that.)
To understand the "long number color code", let's see an example:
RGB= 245/200/60. With hexadecimal numbers, it is RGB= $F5/$C8/$3C. "Forge" the values together: $F5C83C. Now turn that value back into decimal: 16107580 – and this is the long number I'm talking about.
= F302 will do the same as A60: if the subject of the trigger won't be saved, then it will be saved, if you activate this trigger. The difference: A60 works with a given Moveable object, F302 will work with the Moveable object having the given tomb (game) index in Current Value.
= Trigger only for the game log function (so you should remove this trigger before releasing your level):
F309 will print the actual value of the given variable into the log.

32. Memory Zones

Back to Top

The meaning of the Memory Zones:

Memory Zones contain the different properties of the game:

- Savegame Memory Zone: the amount/presence of given things are recorded here, so the game will know – if the player loads the game – the proper amount/presence of things when a savegame has just been loaded.
- Item Memory Zone: the most of the Moveable objects have some properties that can be represented by numbers. The numbers – representing the actual state of the object in that property – are recorded here.
- Code Memory Zone: this zone represents some special properties of the game, such as the actual intensity of earthquakes etc.
- Slot Memory Zone: each Moveable object slot in a WAD has the same values for each object that is placed from that slot. This zone represents the actual values for these general slot properties.
- Animation Memory Zone: some properties of the Moveable object animations are known for WADMerger, some of them are not. This field represents the actual values of these known/unknown animation properties.
- Inventory Memory Zone: this zone contains data about the inventory. (In fact, about the items in the inventory.)

Each memory zone has more fields. Each field controls a given property in its zone.
If you examine variable/memory zone triggers, you will see we can do several operations with memory zone fields. – The most important operations of them:

- We will force a value from a variable into a field. That field will take that value, so the property (controlled by that field) will be controlled by this new, forced value – i.e. the value of the property has been changed.
- We will force a value from a field into a variable. Then we will study that variable value in conditions, and the game will execute things according to the variable values: "if the value is X, then it means this and this are happening in that field, so the game will do Thing A. But if the value is Y, then it means that and that are happening in that field, so the game will do Thing B".

Let's see some examples to understand:

Each example means a different field:

- Savegame Memory Zone:

= 1800 is the value for the full underwater air bar of Lara. Forcing 900 for the bar, the bar will be only half-full.
= 3 is the value if it is the shotgun of the three weapons (shotgun, crossbow, grenade gun) which is on Lara's back. So study the value in a condition, and if that is 3, then we'll know that the shotgun is just seen on her back.

- Item Memory Zone:

= 16384 means Lara faces east exactly (according to Room Editor facing). So, forcing 16384 on her, she will face east, and she won't be able to turn from that way.
= -8192 means the flame, flown from the given emitter, just has the maximal length (2 squares long). So study the value in a condition, and if that is -8192, then we'll know that the given flame has just reached its maximal length.

- Code Memory Zone:

= The value is 2 if it is the look camera of the four cameras (chase, Camera/Fixed Camera, look, combat) which is just working. So, forcing Value 2 in the field, that's what will happen: move the look camera view with look button and cursor keys, and then release those buttons. Then the chase camera position will keep the last camera view you've adjusted by the buttons. (As if look button and cursor keys are just still being pushed down.)
= Some old Script command means a status for the level. (Lightning=ENABLED command means the lightning effect is working in the level, Layer1= X, Y, Z, A means there's a layer1 on your sky, Train=ENABLED means the level is a train level etc.) The field contains those statuses as flags. For example, Horizon is 4, Layer1 is 8, LensFlare is 2048, so if these statuses are active in the level, then the field value is 4+8+2048=2060. If you force 12 in the field that means 4+8 has remained enabled, but 2048 has been disabled – as if you'd created for a flipped room for all the rooms with the Sun, only to switch on NL button in the flipped rooms, and then activated a flipmap trigger for the rooms to switch off the Sun in all of those rooms.

- Slot Memory Zone:

= 160 is the size of the shadow for LARA slot. Force a smaller/bigger number here if you want to change the size of her shadow.
= There's a bit linked to each mesh to control the shatter function. Bit0 (1) for Mesh0, Bit1 (2) for Mesh1, Bit2 (4) for Mesh2 etc. So if you study the field at SKELETON slot, and the value is 2560=512+2048, then you will know the head (Mesh9, Bit9, 512) and the shield (Mesh11, Bit11, 2048) can shatter on skeletons.

- Animation Memory Zone:

= You want Lara to have a new next frame for a given animation from now on. So force a new next frame ID on the field that controls the next frame.
= If you forced a new value for acceleration before, then it's worth studying the acceleration value in a condition.

- Inventory Memory Zone:

= If you have a key, puzzle etc. item, then you have Key, Puzzle etc. Script commands to define X-Y-Z coordinates of the item in the inventory. But if you have another inventory item (small or big medipack, for example) then you need to force values into the fields of coordinates to change

the item X-Y-Z position in the inventory.
= F339 trigger (to study an inventory memory zone field value in a variable) will probably not work.

As you see, memory zone fields are pretty useful if you don't have a direct method to define an executable or a condition trigger.
For example, there are a couple of FLIPEFFECT triggers to change Lara's properties directly – listed above at Lara's description. But you must use a memory zone field if you want to change the size of her shadow.
Or: you want to study the amount of water in the big waterskins at Lara. The only way for that is if you put the actual value from "Inventory. Big Skin Bag (Byte)" field into a variable, and then use a CONDITION trigger for variables to study that value.

Triggers for Memory Zones:

Triggers to define the subject of Memory Zones:

- A54 will define the given Moveable object as the subject of Item Memory Zone.
- F292 will define the given Moveable object slot as the subject of Slot Memory Zone.
- F307 will define the given animation as a subject of Animation Memory Zone. – Notes:

= These animation ID's are so-called absolute animation ID's. (So all the animations of all the Moveable objects of the WAD are calculated now, one after the other, in the order of the object slot ID's.)
= I guess the memory zone works only with the animations of LARA object.

- F335 will define the given inventory item slot (pickable items or special items, such as MEMCARD_LOAD_INV_ITEM) as a subject of Inventory Memory Zone.

So you need to define the subject for Item, Slot, Animation or Inventory Memory Zone, before you use a field of that zone for anything. (This is logical. I mean, for example, if you want to study an object slot in Slot Memory Zone, before that, you need to define which that slot is.)
Savegame Memory Zone or Code Memory Zone doesn't have a subject.

Triggers to force values between Memory Zones and other numbers:

- Triggers to force memory zone field values into variables:

= F244 will force the value from Savegame Memory Zone.
= F256 will force the value from Item Memory Zone.
= F277 will force the value from Code Memory Zone. (Only into Current Value.)
= F293 will force the value from Slot Memory Zone.
= F295 will force the value from Animation Memory Zone.
= F339 will force the value from Inventory Memory Zone. (Probably won't work.)

- Triggers to force variable values into memory zone fields:

= F245 will force the value into Savegame Memory Zone.
= F257 will force the value into Item Memory Zone.
= F278 will force the value into Code Memory Zone. (Only from Current Value.)
= F294 will force the value into Slot Memory Zone.
= F296 will force the value into Animation Memory Zone.
= F336 will force the value into Inventory Memory Zone.

- Triggers to force a number (defined by the trigger) into memory zone fields:

= F246, F247, F254 or F262 will force the value into Savegame Memory Zone.

= F255, F259 or F261 will force the value into Item Memory Zone.
= F279, F281 or F342 will force the value into Code Memory Zone.
= F337 or F338 will force the value into Inventory Memory Zone.

Mathematical operations:

- Triggers to add Current Value variable values to memory zone field values:

= F273 will add the value to Savegame Memory Zone.
= F275 will add the value to Item Memory Zone.
= F283 will add the value to Code Memory Zone.

- Triggers to subtract Current Value variable values from memory zone field values:

= F274 will subtract the value from Savegame Memory Zone.
= F276 will subtract the value from Item Memory Zone. (The trigger name is wrong, mentioning "Savegame Memory".)

- Triggers to add a number (defined by the trigger) to memory zone field values:

= F249 will add the value to Savegame Memory Zone.
= F258 will add the value to Item Memory Zone.
= F280 will add the value to Code Memory Zone.

- Trigger to subtract a number (defined by the trigger) from memory zone field values:
F250 will subtract the value from Savegame Memory Zone.

- Turning the value of a Memory Zone field into 0:

= F248 will remove the value from SaveGame Memory Zone (if the value is a bit, i.e. the exponent of Value 2)
= F260 will remove the value from Item Memory Zone (if the value is a bit, i.e. the exponent of Value 2)
= F282 will remove the value from Code Memory Zone (if the value is a bit, i.e. the exponent of Value 2)

CONDITION triggers:

- CONDITION triggers directly for memory zones (to study the zone value, without using a variable):

= C52: the actual value of the Code Memory Zone field should be bigger than (or equal with that) the "bit" value which is an exponent of Value 2.
= C53: the actual value of the Code Memory Zone field should be smaller than the "bit" value which is an exponent of Value 2.

- Special CONDITION triggers (to compare directly a zone value to a variable):
C49, C50 or C51: compares the actual value of the Code Memory Zone field to the actual value of Current Value variable.

Special trigger:

If Lara is aiming at enemy, and you use "System. Item Memory address of enemy aimed by Lara" Savegame Memory Zone field (see below) to identify that enemy with a long number in Current Value variable, then F351 will transform that long number into a useful value (in Current Value). This "useful value" is the tomb (game) index of the enemy.

The list of memory zone fields:

Savegame Memory Zone:

- “Inventory. Ammo…” fields: the ammo amount.
- “Inventory…” fields with weapon names: the presence of the weapon, the type of the loaded ammo, lasersight is attached or not.
- “Inventory…” fields for medipacks or flare: the amount of them.
- “Inventory…” fields for binocular, crowbar, lasersight, examine: the presence of them.
- “Inventory…” fields for key, puzzle combo, pickup, quest: the presence of them.
- “Inventory. Puzzle…” fields: the amount of the given puzzle item.
- “Inventory…” fields for little/large waterskin: the presence of them or the water amount in them.
- “Inventory. Mechanical Scarab…” field: the presence of the scarab or its parts, or if they are combined or not.
- “Inventory. Remaining usage of Mechanical Scarab…” field: the remaining usage of the scarab.
- “Lara. Air for Lara…” field: the air amount in her lungs underwater.
- “Lara. Current Weapon (not necessarly in the hand)…” field: the given weapon is just in Lara’s hand or will be extracted when hitting SPACE.
- “Lara. Environment where lara is…” field: Lara is on dry land, swimming/floating on/in water, wading in water/quicksand.
- “Lara. Hands. Attached Lara Status…” field: some special flags for Lara’s actions. (For example, Flag 2 means she’s just drawing a weapon or igniting a flare.)
- “Lara. Hands. Item in the Hands of Lara (Current)…” field: see “Lara. Current Weapon” field but this time flare or torch status is also calculated.
- “Lara. Hands. Item in the Hands of Lara (Following)…” field: similar to (Current) field, but this time it’s worth forcing a weapon/flare here to extract.
- “Lara. Hands. Remaining time with lighted flare in the hand…’ field: the actual timer value of the lighted flare in Lara’s hand.
- “Lara. Hands. Weapon in the hand…” field: not a really useful field.
- “Lara. Item Index of Lara…” field: the tomb (game) index of LARA object.
- “Lara. Poison…” fields: two similar fields to calculate Lara’s degree of poisoning.
- “Lara. Rope. Speed sliding on the rope…” field: how far Lara is swung forwards/backwards with a rope.
- “Lara. Special Status of Lara…” field: some special flags for Lara’s statuses. (For example, the game adds 8 to the value if she’s burning.)
- “Lara. Special2 Status of Lara…” field: some special flags for Lara’s statuses. (For example, the game adds 16 to the value if she’s just looking through the binoculars or the lasersight.)
- “Lara. Test. Climb sector Test (=1 yes; =0 no)…” field: the walls are all climbable/non-climbable around the trigger square.
- “Lara. Test. Lara has a flare in the and (1 = yes)…” field: it defines if Lara has a working flare in her hand or not.
- “Lara. Test. Lara is on rope (different than -1 lara is on the rope)…” field: it defines if Lara is on a given rope or not on any rope.
- “Lara. Test. Lara is placing the weapon on the back (1 = yes)…” field: it defines if Lara is aiming at any enemy with any weapon or not.
- “Lara. Test. Throw out item from the Hands (1 = lara is throwing out item)…” field: this field contains flags for the torch. (On the floor, is just being ignited etc.)
- “Lara. Weapon on the back of Lara…” field: it indicates if a given weapon is just seen placed on Lara’s back.
- ”Statistics. Distance…” field: this field contains the distance Lara has traveled in the whole game so far.
- “Statistics. Killed Enemies…” field: this field contains the number of the enemies Lara has killed in the whole game so far.
- “Statistics. Secrets…” field: this field contains the number of the secrets Lara has revealed in the whole game so far.
- “Statistics. Used Ammos…” field: this field contains the number of the ammunition Lara has shot in the whole game so far.

- “Statistics. Used MediPacks…” field: this field contains the number of the small AND big medipacks Lara has used in the whole game so far.
- “System. Auto-Aiming for Enemy…” field: this field shows if the auto-aiming is just enabled or not.
- “System. Core Game Timer (one unit = 1/30 of second)…” field: this field contains the time that has passed in the whole game so far. (Containing the time when there are menus on the screen.) – Not always updated.
- “System. Disable special keys (15 disable inventory pause f5)…” field: studying the value of this field is a good possibility to activate something after Lara’s death.
- “System. Fog Bulb Color…” field: the actual color code of fog (if the color is defined by a trigger).
- “System. Item Memory address of enemy aimed by Lara…” field: see the description of F351 trigger above.
- “System. Number of current Level…” field: the ID of the level (used by FINISH triggers).
- “System. Screen Timer (Increased only when it is different than 0)…” field: the actual value of the so-called “Screen Timer”.
- “System. Unknown (Item chosen from Inventory?...” field: each inventory item has its own ID number here. The field will apply that value if the player selects that item in the inventory.
- “TRNG Index. Animation Index for Selected Animation Memory…” field: the subject of the Animation Memory Zone this time will be the animation number in this field.
- “TRNG Index. Index of last item found with testposition or condition…” field: for example, use a GT_COLLIDE_SLOT type GlobalTrigger, with BADDY_1 slot. If Lara collides with a BADDY_1 then a TriggerGroup will be activated. In this TriggerGroup, you will put an F244 trigger that puts the value of this “TRNG Index…” field into a variable. This value is the game index of the object that is defined by the latest condition. This condition is GT_COLLIDE_SLOT-BADDY_1 so this game index is the game index of the BADDY_1 Lara has just collided with now.
- “TRNG Index. Index of last item used by Lara…” field: the tomb (game) index of the object Lara has the interaction with last (pushing that button, getting on that motorbike etc.).
- “TRNG Index. Index of moveable performing last AnimComand…” field: for example, your BADDY_1 slot has an AnimCommand at a frame of AnimationX. This AnimCommand is a TriggerGroup that contains two triggers: one of the triggers (T1) is the one the baddy will execute at that frame. The other trigger (T2) is an F244 trigger that puts the actual value of this “TRNG Index…” field into a variable. It means T2 puts the tomb (game) index of the baddy just executing T1 into that variable.
- “TRNG Index. Item Index for Selected Item Memory…” field: the subject of the Item Memory Zone this time will be the object tomb (game) index in this field.
- “TRNG Index. Slot Index for Selected Slot Memory…” field: the subject of the Slot Memory Zone this time will be the object slot ID number (see NG Center\Reference) in this field.
- “TRNG Organizer Timer. (Increased always in game but not in inventory and pause)...” field: this field contains the time that has passed in the whole game so far. (Not containing the time when there are menus on the screen.)

Item Memory Zone:

- “Animation Now (Number of current animation)…” field: the actual animation of the subject.
- “Contact Flags ($2400 = damage lara on touching)…” field: it defines if Lara is just colliding with a Moveable object or not.
- “Custom Flags (Different flags in according with object type)…” field: it tells you about the activity state of the subject. (Eg. 191 is dead enemy etc.)
- “Custom_A, B, C, D…” fields: contain different information about different objects.

= ENEMY_JEEP:
A – the distance from the last AI_FOLLOW.
C – a counter is running here, so that the jeep will shoot a grenade when the counter reaches 0.
D – the ID of the actual AI_FOLLOW.
= GUIDE:
B – flags for the torch in his hand.

D - the ID of the actual AI_FOLLOW.
= VON_CROY:
C – special properties (for example Value 1 means he's waving to Lara).
D - the ID of the actual AI_FOLLOW.
= BADDY_1, BADDY_2:
B - the room game (tomb) index where he just is (dead or alive).
C - his actual Uzi ammo.
= SETHA:
A – three counters is running here when any of the three shooting procedures is being performed: 0-151, 0-161, 0-199.
= SPHINX:
C - the actual distance from the north edge of the Room Editor window.
D - the actual distance from the west edge of the Room Editor window.
= CROCODILE:
B – moving his tail leftwards/rightwards at an AI_GUARD.
= HORSEMAN:
A - active or dead.
B - on the horse or not.
D - the ID of his AI_FOLLOW.
= MUTANT:
C – till the value here becomes a constant 999, the mutant won't notice Lara.
= BABOON_NORMAL, BABOON_INV, BABOON_SILENT:
A - the horizontal coordinate is contained here in which the baboon is placed in the Room Editor.
C - active or dead.
= HARPY:
A – a counter is running here when the shooting procedure is being performed.
= LITTLE_BEETLE:
A - the value is 1 if the beetles will be emitted from the floor.
B - the value is 2 if the beetles will be emitted from the ceiling.
C - the value is 4 if the beetles will be emitted from the wall. (Becomes a counter down when the beetles are being emitted.)
= WRAITH1, WRAITH2, WRAITH3:
B – a countdown counter that will be increased when the fire wraith collides Lara. Above about Value 10000, each collision will ignite Lara.
C – this is a counter from 0 to about 1000 or -1000. (Depending on the direction of the rotation of the wraith around Lara.)
D - an increasing positive value if the wraith is going farther from Lara or an increasing negative value if the wraith is coming closer to her.
= AHMET:
A - it contains the west-east row of squares in which the AHMET is placed.
C - it contains the north-south row of squares in which the AHMET is placed.
= WHEEL_OF_FORTUNE:
A, B, C, D – I suppose the value and its sign here depends on a plate being just black or white and in which angle being just rotated.
= SCALES:
B - filled by the wrong amount of water or not.
= FALLING_BLOCK:
A - when it starts shivering (due to Lara stepping on it) then a counter starts counting down, till the block will fall down.
= TRAPDOORs:
C – closed or not.
D – informs you about the tomb indices of the room just above the portal in which the trapdoor is placed and in which the trapdoor is placed.
= ROLLINGBALL:
A - how long the moving rollingball must "travel" in the north or south direction in the given moment, until that stops.
B - how long the moving rollingball must "travel" in the east or west direction in the given

moment, until that stops.
C - the ball is in water or not (with OCB=32).
= TEETH_SPIKES:
A, B - values are changing here at moving spikes. Crucial values are when the spikes are totally drawn back, and if they are totally extended.
C - when the spikes reach the topmost position, then a counter will start down. Reaching 0, the spikes will jump out of the ground.
= JOBY_SPIKES:
A – the degree of the actual speed of rotation.
B – the degree of the actual extension.
D - it is the length of the maximal extension.
= SLICER_DICER:
A - it contains the distance from the north edge of the Room Editor Window in which this trap is placed.
B - it contains the vertical coordinate in which this trap is placed.
C - it contains the distance from the west edge of the Room Editor Window in which this trap is placed.
= CHAIN, PLOUGH, STARGATE:
A – information about the chain/plough/stargate being active or not.
B - information about the plough/stargate being active or not.
D - information about the chain/plough/stargate having been activated before at least once or not.
= HAMMER:
A – not 0 if the hammer is just striking (or the first part of shivering after striking).
C – only 0 if the hammer is just in the starting (highest) position.
D - information about the hammer having been activated before at least once or not.
= BURNING_FLOOR:
A, B, C – three very similar counters run here. They count up when the fires are increasing, and count down when they are decreasing.
D - a counter starts running up when the floor starts firing, and stops when the fires start extinguishing.
= SPIKEBALL:
A, B - non-0 when the double doors of the spikeball are opening or if the knives of the spikeball are extended.
D – information about the position of the ball itself and the knives.
= FLAME_EMITTER (only blowing a horizontal or vertical flame):
A – a counter from a random value (when the flame starts stopping being blown) to 0 (when the flame will start being blown again).
B – the actual length of the flown flame.
C - when the emitter starts blowing, then a counter will count down to 0. When the emitter starts stopping blowing, then it counts back.
D – a random time interval is counted here: the time of the actual blowing procedure.
= FLAME_EMITTER2 (only with negative OCB number):
A - to trigger a flipmap (see Scales-puzzle from “Temple of Horus”): flipmap is triggered or not. To divert the lightning (see Lightning conductor-setup from the “Street Bazaar”): the pushable object is on the “diverting place” or not.
= FLAME_EMITTER3 (as flames):
A – a short counter runs down again and again. Each number of the counter means a burst of flame is just burst or not.
B - there are codebits here to indicate which bursts are burst in the given moment.
= FIREROPE:
B - two counters run here. The first one runs when the rope is burning, the second one runs when the rope is crumbling.
D - a counter starts here down when the rope catches fire. Reaching 0, the rope starts crumbling.
= TWOBLOCK_PLATFORM (if the platform will descend half a click if Lara steps on it):
A - you can force numbers here to modify the starting position/the degree of the descending for the platform.

B - the platform is in the base (ascended) or in the descended position.
= RAISING_BLOCK1, RAISING_BLOCK2:
B - the actual height of the ascending.
C - the block is totally descended or not.
= EXPANDING_PLATFORM:
B - the actual length of the expansion.
C - the platform is totally drawn back or not.
= SQUISHY_BLOCK2:
A – a counter runs here when the block has been triggered, till the shake stops and the block falls down.
= PUSHABLE_OBJECTs:
A, C – the block has been pushed once at least or not.
= SENTRY_GUN:
A - Value 1 and 2 will change with each other fast if the gun is shooting bullets.
B – the radar is turned exactly to the gun or to the other direction.
C – when the gun shooting bullets, a counter counts up here, never exceeding a maximal value.
= MINE (only to blow it up by triggers, i.e. if OCB=1):
A - when a trigger ignites the mine then a counter runs here till the mine explodes.
= OBELISK:
D - a counter runs here while the electric arch (see the obelisk-puzzle in “The Great Hypostyle Hall”) is coming from the obelisk.
= FLOOR_4BLADE, ROOF_4BLADE:
A – if the floor blades move upwards/roof blades move downwards or not.
D – the values are changing if the blades stop (for a moment or “forever).
= BIRD_BLADE, MOVING_BLADE:
A - bird blade is just closing or not.
D - information about the blade having been activated before at least once or not.
= CATWALK_BLADE:
D - the blades are just moving or not.
= PLINTH_BLADE:
D – the value is not 0 during the whole working of the blade.
= SETH_BLADE:
A – information about the position of the double blades.
B: - the double blades are moving downwards or not.
C – a counter counts the time down (according to the OCB number) to trigger the blades when reaching 0.
D - information about the blade having been activated before at least once or not.
= LIGHTNING_CONDUCTOR:
A – short counters run here again and again down. The lightning hits the ground when the counter reaches 0.
B - possibly the coordinate is here wherefrom the actual lightning comes.
D - only with OCB=2: if you’ve placed the pushable to the “diverting place”, a counter will start. When it stops the diverted lightning will blast.
= ELEMENT_PUZZLE:
A – the status of the scale (empty/filled/Lara’s lighting the petrol/the petrol is burning).
= (PICKABLE OBJECTS):
D - the item has just activated a PICKUP trigger or not.
= PUZZLE_HOLEs:
A - Lara is just placing the puzzle item here or not.
= TURN_SWITCH:
A – the last move (still no move/clockwise/anti-clockwise).
B – non-0 when the switch isn’t being moved (after the first usage).
= CROWBAR_SWITCH:
A - the switch is disabled (because a living BABOON_NORMAL near there) or not.
= PULLEY:
B - (see the obelisk-puzzle from “The Great Hypostyle Hall”): the pulley is forbidden (“Invisible”) or not.

C - the pulley has been used at least once or not.
D - the OCB number.
= DOOR_TYPEs (only with COG_SWITCH):
A - a short counter runs down here if Lara is moving the switch (and if the door is not in too high so that it can move).
= SEQUENCE_DOOR1:
A - the door has been opened or not.
= SEQUENCE_SWITCH1/2/3:
A - the switch is pushed in or not.
= GOD_HEAD:
A – 0 before reaching its maximum expansion.
B - the actual length of the expansion.
C – a counter runs here when the head has its maximal size. Reaching 0, the head starts drawing back.
= STATUE_PLINTH (with OCB=0):
A - the PUZZLE_ITEM5 is placed here or not.
B - the inventory has been opened automatically at PUZZLE_ITEM5 (and is still open) when you hit CTRL at this pedestal, or not.
= GRENADE (placed down):
B - a short counter runs here down when the grenade is exploding.
= STEAM_EMITTER:
A - as a horizontally blown steam: a counter starts down, stopping the blow by degrees. Reaching 0, the blow starts again.
As a (non-continuous) bubble-emitter under water: counts down the actual amount of bubbles that will be emitted.
B - as a horizontally blown steam: a random counter runs here, starting the blow. Reaching 0, the emitter will start stopping blowing.
C - as a horizontally blown steam: the actual length of the steam.
= EARTHQUAKE:
A - the actual intensity of the earthquake.
B - the actual intensity of the earthquake, but not as precise as in Custom_A. (I mean, there are only two values now.)
C – counters run here one after the other. Each sequence means an intensity range (“range for bigger quakes”, “range for smaller quakes” etc.).
= WATERFALLMIST (only if you customized it by OCB):
A - with standard, low or fast emitting: waterfallmist is active or not.
With random emitting: two counter runs here to count time for the emitting or the non-emitting phase.
B - the emitting just happens or not.
C - only with random emitting: random numbers, changing when the emitting phase shifts into non-emitting phase, or vice versa.
= SPRINKLER:
A - a counter runs up here when the sprinkler is activated. It stops when the last drip has come out of it.
= AMBER_LIGHT:
A - a loop counter runs continuously, again and again. (So each value means the actual degree of the pulsing intensity.)
= WHITE_LIGHT:
A - a counter runs here from the activation of the object till this light stops the hard blinking.
= PLANET_EFFECT:
A - the starting value is 50+OCB number-1. If the effect has been activated, the value turns into -1.
C, D - only the object that is directly linked to ANIMATING4 – see the globe-puzzle of “The Lost Library” – has a non-0 constant value here.
= LASER_HEAD:
A – head statuses (attacking, eyes have been shot etc.).
B – the value changes if the head has been ascended after triggering that.

C – a complex counter runs here. Sometimes stops, sometimes changes rate, according to what the head is doing.
D – more counters run here. The main ones: the one between two sudden head movements, and the one when the head is getting energy.
= HYDRA:
D - when the hydra is shattering ("dying") then a counter runs here.
= ENEMY_SUB_MARINE:
A – the degree of how much the submarine is just tilted left or right.
= FISH_EMITTER:
A - counts down the actual amount of fishes that will be emitted.
B – a short counter runs here again and again between a random value (a fish is just emitted) and 0 (when the next fish will be emitted).
C – the emitter has been triggered or not.
D – for, example, a sequence here, counting the emitted fishes, if you have 3 fishes in 3 fish types: 0, 1, 2, 3, 0, 1, 2, 3, 0, 1, 2, 3.

- "Facing…" fields: (not only for Lara). Each number in this field means the orientation of the subject, around the X, Y, Z axes.
- "Flags of Item…" field: contains flags such as "the object is just being hit", "is in AI_AMBUSH mode", "has not been killed" etc.
- "Frame Now (warning it's an abs value)…" field: the actual frame ID of the actual animation.
- "Height Floor below the item…" field: the actual vertical distance from the floor.
- "HP (Current life level. $C000 = unkillable)…" field: the actual degree of the vitality.
- "Object buttons. Five buttons + invisible button…" field: OCB codebits or Clear Body button are on or off.
- "OCB Code (The value you typed in OCB of this item)…" field: the value in the OCB window.
- "Position…" fields: the X, Y, Z coordinates.
- "Room (Room where is the item)…" field: the game (tomb) index of the room where the object is just located.
- "Slot Id (number of slot)…" field: the slot index (see: NG Center\Reference) of the object.
- "Speed…" fields: the actual horizontal/vertical speed.
- "State Id…" fields: the State values for the animation of the object.
- "Transparence level (0 = opaque / 126 transparent / over 127 removes item)…" field: the actual transparency level.
- "Unknown (Light…" fields: these fields show values about the ambience light of the room in which the subject item is just located.
- "Unknown (Pheraps accelleration on falling)…" field: show some animation State ID's with AI_FOLLOW type objects.
- "Unknown (Sprite…" fields: unknown.
- "Unknown (Used when enemy shot a granade = $c210)…" field: unknown.
- "Unknown Countdown (Some counter, not yet discovered)…" field: the usual object timer. (Useful field to study the timer value.)
- "Visible Mesh Flags (each mesh a different bit)…" field: define the meshes that are visible.

Code Memory Zone:

- "Audio Track Number…" fields: the ID of the audio file that is just playing in channel1 or 2.
- "Camera Mode…" fields: information about which camera type is just being used. ("Now" field for studying value, "Next" field for forcing value.)
- "Current Level number (more updated than savegame memory)…" field: see the similar Savegame Memory Zone field.
- "Dash Bar Value (0 - 120)…" field: Lara's actual power to dash.
- "Earthquake vertical movement (negative values)…" field: the intensity of earthquakes.
- "Frame 3d Counter. (It works in game and inventory but not in pause)…" field: this field contains the time that has passed in the whole game so far. (Not containing the time when there are F5/F6/Pause menus on the screen.)
- " Frame System Counter. (Works always, in game, inventory and pause screens)…" field: this

field contains the time that has passed in the whole game so far. (Containing the time when there are menus on the screen.)
- “Inventory Item just chosen from inventory (example a key to open a door)…” field: use it with “customized puzzle holes” to prevent the unnecessary “NO” sound of Lara.
- “Inventory Item required in game (example, crowbar for door)…” field: if the inventory opens when the player hits CTRL at an object to interact with that, then the value will be the slot ID of the object that needs for the interaction, till the inventory is open.
- “KeyBoard Game Command hit (Use bit operations with KEY1_ constants)… field: when the player moves Lara (or does anything) by the keys then you will see here the code of that key command.
- “Music volume (max = 100)…” field: this field contains the “general volume” of the audio – the one you see in Options menu.
- “Screen… “ fields: they inform you about your screen resolution.
- “Script Dat. Level Flags (Use bit operation to read or change)…” field: some old Script commands (YoungLara, Horizon, Train etc.) has a flag in this field to identify that property of the level.
- “Script Dat. Option Flags (Use bit operations to read or write)…” field: I couldn’t really interpret this field.
- “Sound SFX volume (max = 100)…” field: this field contains the “general volume” of the sound – the one you see in Options menu.
- “Speed Layer…” fields: defines Layer1 or 2 speed/direction values.
- “Test. Disable Fog Bulbs (1 = disable)…” field: fog bulbs are disabled or not.
- “Test. How entered in current game (New level=0; From savegame = 4)…” field: I found this field useless.
- “Test. There is a Flyby in progress (1 = true / 0 = false)…” field: a flyby sequence is just being performed or not.

Slot Memory Zone:

- “Explosion Mask. (Each bit a type of weapon able to do explode it…” field: define the meshes that can be shattered.
- “First Animation Index…” field: this field helps you to identify the first animation ID of an object slot.
- “First Mesh Index…” field: the value is “twice #AMI”. See #AMI values in the Select Object panels in Room Editor.
- “Flags. Main flags…” field: flags to define the object slot can use AI objects or he is a creature for water etc.
- “FootStep (Shadow below Lara or enemies)…” field: the size of the shadow.
- “HP. Max Vitality at start…” field: the maximum life points of Lara or other creatures.
- “Number of Meshes…” field: the amount of the meshes of the slot.
- “Pointer for Collision Procedure…” field: 0 in this field means the slot won’t have collision.
- “Pointer for Draw Extra Procedure (used by jeep, sidecar)…” field: I couldn’t really interpret this field.
- “Pointer for Emitter Procedure…” field: With 0 in this field, only the emitted effects (eg. the beam of the motorbike headlight) were seeable, the object itself was totally invisible.
- “Pointer for Initialization Procedure…” field: I couldn’t really interpret this field.
- “Pointer for Main Control Procedure…” field: force 0 here if you want to “freeze” the animation of the subject slot.
- “Pointer for Special Ceiling Procedure…” field: I couldn’t really interpret this field. (Works only with TWOBLOCK_PLATFORMs?)
- “Pointer for Special Floor Procedure…” field: forcing 0 “kills” the DUMMY trigger below the TWOBLOCK_PLATFORM.
- “Test Attack Lara (1=attack, 0 =ignore lara)…” field: I couldn’t really interpret this field. (Works only with BADDY_1 or 2?)
- “Unknown1 (Pheraps distance to enable the MIP version)…” field: defines the distance in which the object will change into its MIP version.
- “Unknown2 (Usually it has value 50)…” field: this field contains the horizontal distance in which

the enemies will detect Lara in front of them when attacking her.
- “Unknown3 (It will be copied in Custom_B field of Item structure)…” field: I couldn’t really interpret this field. (The values must be some kind of “creature flags”.)
- “Unknown4…” field: unknown.

Animation Memory Zone:

- “Absolute Index of first AnimCommand…” field: not a really useful field.
- “Absolute Index of first State Change...” field: not a really useful field.
- “First absolute Frame index…” field: defines the absolute index of Frame0 of the given animation.
- “Frame Rate…” field: the frame rate value of the animation.
- “Frame Size…” field: not a really useful field.
- “High Accelleration…” field: the “Accel” value of the animation.
- “Last absolute Frame index…” field: defines the absolute index of the last frame of the given animation.
- “Low Accelleration…” field: not a really useful field.
- “Next Animation index…” field: the “Next Animation” value of the animation.
- “Next Frame index…” field: the “Next Frame” value of the animation (as an absolute frame ID).
- “Next High Acceleration…” field: not a really useful field.
- “Next Low Acceleration…” field: not a really useful field.
- “Next Speed…” field: not a really useful field.
- “Number of Animation Commands…” field: not a really useful field.
- “Number of State Changes…” field: not a really useful field.
- “Speed…” field: the “Speed” value of the animation.
- “State Id…” field: the “StateID” value of the animation.
- “Unknown…” fields: unknown.

Inventory Memory Zone:

- “Distance of Cam. {Little values bigger items and vice-versa)…” field: the virtual distance in which the item is seen in the inventory.
- “Slot of Mesh to show in Inventory…” field: the slot ID of the given inventory item.
- “String index of Name…” field: the ID of the string that contains the name of the item.
- “Top Border in 2d plane. (Negative numbers move up the item)…” field: the vertical position of the item in the inventory.
- “Type Flags. (4 = USE, 32 = EXAMINE, $1000 = LOAD ect)…” field: the command attached to the given item in the inventory.
= “Unknown. (It may be only 1 or -1)…” field: unknown.
= “View Flags. (2 = turn endless, $4000 = usable?)…”: field: if the selected item is rotating or static in the inventory.
= “… Facing about the cam view on…” fields: how the item is rotated around X, Y, Z axis in the inventory.

Additional information:

- See Memory Zones tutorial on Skribblerz to know more about memory zone fields.
- You don’t need to use some fields because you have a direct method instead of them. However, I say, sometimes it’s worth using those fields with the variable method. Why? Because the subjects of the direct triggers are standard things, but if you use a variable value as a subject, then the subject of the trigger can be more than one things because the variable can have more than one values.
Or maybe you should use the variable method, because the sphere of action of the direct method are more little.
- The “byte”, “short” and “long” words in the names of the memory zone fields indicate the number the field may have.
For example, if you can see the “short” word in a field name then that means the smallest

expected value in that field is -32 768 and the biggest expected value in that field is 32 767.
So, if you have a field with “short” word then it’s highly recommended to use a “short” or a “long” variable for that field, because “byte” variables can’t handle the value of the field if that is between -32 768 and 0 or if that is between 255 and 32 767.
- It doesn’t matter if you use the values in decimal or hexadecimal format. (If you have a choice to choose between formats.)
- Some fields cannot take a forced value. (But can be studied in a condition.)
- Some value must be forced continuously for a permanent effect.
- Some fields lose the forced value at level jumps, or if you save the game after the force, and then load that savegame. (See this part of Paolone’s variable tutorial to understand the problem and the solution for that: “The wooden Door: how save and restore our changes”.)
- The “local/global” type of a variable and the local/global effect of it in the fields is not the same.
– See an example to understand:
The amount of the big medipacks is a global value, because, if you have X amount of big medipacks at the end of Level A, then you also have X amount of big medipacks of the start of Level B, if you jump at the end of Level A to the start of Level B.
So, if you use a local variable to define X medipacks at the end of Level A, and you jump to Level B, then the amount of the medipacks willl remain X – though that local variable will become 0 at the start of Level B.

33. TRNG in native languages

Back to Top

You can translate these texts:

- From TRLE main folder:
ng_constants.txt: the names for STT panel.

- From NG Center main folder:

= Help_Edit_Font.txt: the help file for NG Font Editor function.
= help_new_script_command.txt: the contents in NG Center\Reference\SCRIPT NEW commands.
= help_ocb_list.txt: the contents in NG Center\Reference\OCB LIST.
= help_old_script_command.txt: the contents in NG Center\Reference\SCRIPT OLD commands.
= help_remap_skin.txt: the help file for Remap Lara Skin function.
= help_wmv_encoder.txt: the help file for WMV Encoder function.
= scripter_constants.txt: the contents in NG Center\Reference_MNEMONIC CONSTANTS for new script commands. And the names of NG Center buttons and other functions.