Tomb Editor Manual

rickyturaz

August 17, 2025

Contents

I Introduction **5**

1 History of Tomb Raider Level Editors **7**
1.1 Back to 2000: *Tomb Raider Level Editor* 7
1.2 Paolone's *Next Generation Level Editor* 8
1.3 MontyTRC89's *Tomb Editor* 8

II Getting started **9**

2 Installing Tomb Editor **11**

3 Starting a new project **13**

4 What is a level? **17**
4.1 Level map 17
4.2 Playable level 17
4.3 Crucial attachments 18
4.3.1 Item files 18
4.3.2 Texture files 19
4.3.3 Sound files and their catalog files 20
4.3.4 Level script 20
4.4 Conclusion 21

III Principles of game development **23**

5 Beta testing **25**
5.1 In General 25
5.2 Alpha Testing 26

Part I

Introduction

Chapter 1

History of Tomb Raider Level Editors

1.1 Back to 2000: *Tomb Raider Level Editor*

Tomb Raider marked a sensational new approach to 3rd person gaming. Fans not only fell in love with Lara and her moves, but with the imaginative and intriguing worlds of her adventures. It all started with Lara's visit to some Egyptian ruins back in 1996, and now with the release of the Tomb Raider Level Editor has come full circle, offering a different sort of adventure in another Egyptian setting. *Tomb Raider Chronicles* marks the end of the line of Tomb Raider games made with these development tools; but rather than viewing this as an end, the release of the editor makes it seem more like a beginning...

The **Tomb Raider Level Editor (TRLE)** includes a tutorial that will walk you through the basics needed to create your own stand alone Tomb Raider levels (but please read the legal disclaimer about commercial use of this product). Even though you will not be able to edit objects or animations (that means Lara's outfits), you have a wonderful variety of object sets from which to choose. You can sculpt and design on many different 'levels' – trigger events, create awe-inspiring spaces, simple to complex. . . and as you experiment you will learn more about what can be done, and quite possibly discover new methods of applying the knowledge you have acquired.

We sincerely hope you will enjoy inventing, designing, and building with

and for Lara as much as we have over the past 4 years. We thank all those who have held the enthusiasm for the Tomb Raider series, thereby contributing to its success. We wish you happy adventuring with Lara and the tools used to create her worlds. [2]

1.2 Paolone's *Next Generation Level Editor*

The **Next Generation Level Editor**, often abbreviated **NGLE**, is a modified version of the Tomb Raider Level Editor, created by Paolone, and released in January 2007. [3]

Tomb Raider Next Generation (TRNG) tools, improve the TRLE tools used to build custom levels with the engine, supplied by Eidos, of Tomb Raider - The Last Revelation.

Many objects have beed added, some imported by other TR adventures, like boat or frog-man, other builded ex-novo like Detector or Elevator.

There is a new scripter program named NG_Center. This program other to build your script.dat supplies other little tools. [5]

1.3 MontyTRC89's *Tomb Editor*

Tomb Editor (TE) is a level editor designed for the full range of classic Tomb Raider game series (1-5), as well as for contemporary engine reimplementation projects and game engines designed for community modding and level building. [4]

Part II

Getting started

Chapter 2

Installing Tomb Editor

First of all, you need to download and install the Tomb Editor pack on your computer. It is available eg. here.

The default route of Tomb Editor installed is `C:\Tomb Editor`. The contents of this main folder are:

- Tomb Editor program.
- Side programs dedicated to Tomb Editor: SoundTool, TombIDE, WadTool.
- Most of the files which are necessary to start a basic project and level for Tomb Engine. (But texture files for room faces must be find somewhere else. But this is still not necessary now, when you start reading this tutorial.)
- Other important files for Tomb Editor pack.

So when you have the Tomb Editor pack installed on your computer, then you are just ready to start building levels for Tomb Engine. [1]

Chapter 3

Starting a new project

After the installation of Tomb Editor pack, you are ready to make your very first Tomb Engine project. (Level map file extensions are well-known as "PRJ" files, but do not misunderstand: what we call "project" now is not a level, but a whole level set - i.e. your current Tomb Engine game itself, which will be released when you fully made it.)

But where do you need to place your projects? Well, NOT in Tomb Editor main folder - that is a place you usually never modify while editing. I suggest placing all of your projects nicely collected in a so-called general project folder. This could be called eg. `"My_Tomb_Raider_projects"`, created manually. (I created it in Documents folder.)

Each project you make will be placed in its own main folder. Does it mean now you should also create manually a project main folder in the general folder? No, there is a TombIDE wizard which will do the whole project-creating procedure for you.

Projects are handled in **TombIDE (Tomb Integrated Development Environment or TIDE)** program, that is why the whole project-creating procedure is also being done there. So start `TombIDE.exe`, and the panel of TIDE Start page opens up. Click on "Create a new project" button now. The first page of a new panel opens up (General Information, 3.1):

- Let's suppose the project you start now has the name of "Lara's Newest Adventures". So type it now here.
- Click on "Browse" button, find and select the general project folder.
- After that, the little window in the middle of this first page shows that

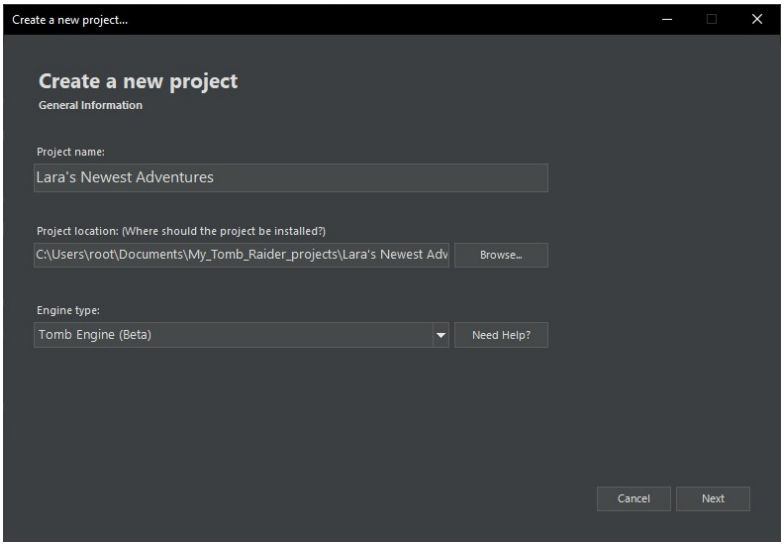

Figure 3.1: General Information

a subfolder in the general project folder will be created as the main folder of this project, having the project name.

- The engine type you choose now is naturally Tomb Engine.

Now click on “Next” button to continue the procedure on the next page of the panel (Extra Options, 3.2).

I suggest changing nothing here. Which means level map files will be handled in a folder called ”Levels”, which is a subfolder in the main folder of the project. (I mean, this is the default place for level map files, and you, the beginner probably should keep it like this.)

Now click on ”Create” button here, then look at the increasing bar at the bottom of the panel. When the bar is at 100 %, then you get a message that the project has been successfully created (3.3).

And this project main folder has been also created on the selected route, with the basic contents a TEN project should have. (Including Levels folder - still being empty - in that default position., 3.4)

Double-click on that row (or click on ”Open selected button” below), so the project opens in TIDE, you will be able to work on it. Each project opened in TIDE has more pages, now you can see its Level Manager page. (See the panel header which names the current project.)

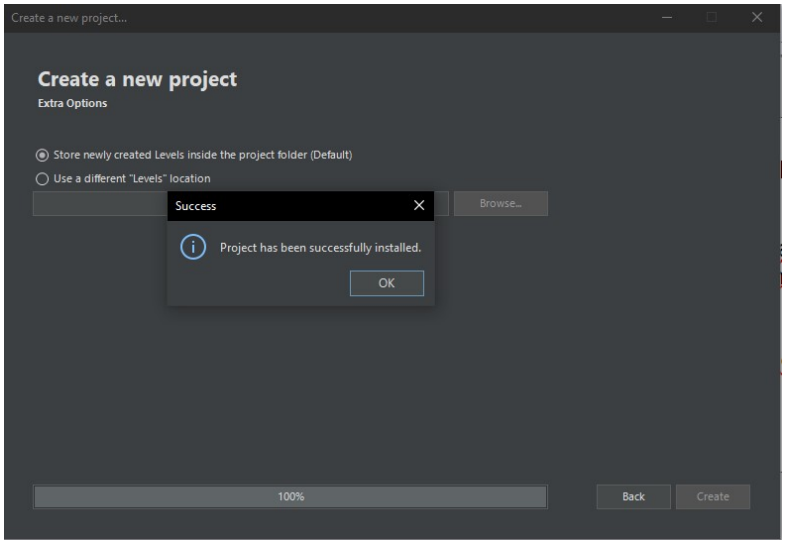

Figure 3.2: Extra Options

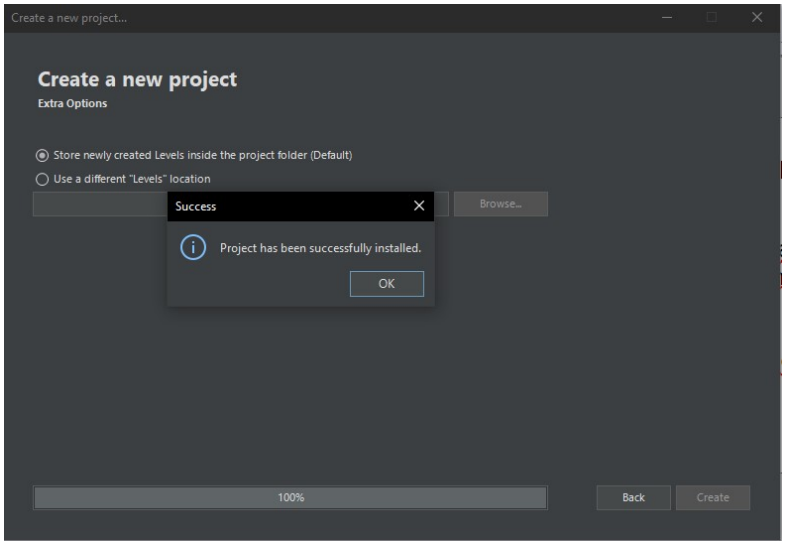

Figure 3.3: Project has been successfully installed

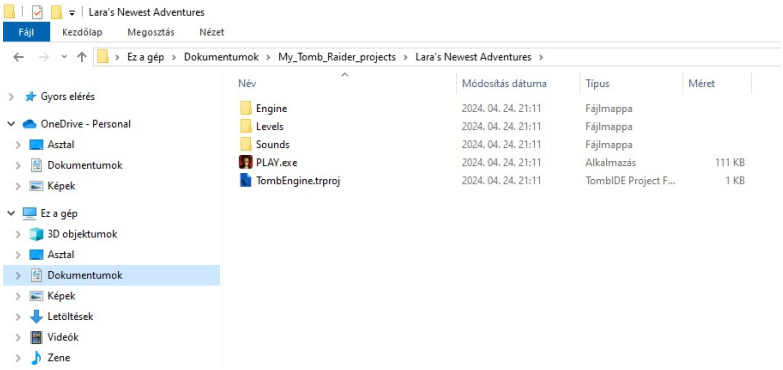

Figure 3.4: Project directories have been created (Since TEN 1.7., the project main folder has an Assets subfolder as well.)

Now click on the red arrow in the upper left corner of the page, to go back to TIDE Start page, closing this project now in TIDE. (Since TEN 1.7., you can see the blue TIDE icon in that corner, instead of the red arrow. If you click on that icon, then a menu opens. One of the menu options is an arrow, with "Back to Start Window..." name. Click on it to go back to TIDE Start page.) [1]

Chapter 4

What is a level?

Before we discuss how to start a new level, important to understand, that actually what a "level" is. *(Don't do anything now, just read and listen.)*

4.1 Level map

Level maps are the editable versions of levels. So when you create, modify a level in the level editor program, then the changes are saved and stored in level map files. TE has its own level map formula, whose extension is called `PRJ2`.[1]

At the moment you don't have PRJ2 files, but soon you will create them and save them in Levels folder of the project main folder. (In fact, in subfolders of Levels folder, and each subfolder is dedicated only to one level of the project, having a name which nicely refers to the level name.)

4.2 Playable level

The playable version of the level is a level file what the game will play as a level. This playable file is made by a conversion, converted from the PRJ2 level map file of this level. (The conversion will be a very simple task to you: only a simple click on a button.)

[1] Previously level map files was known as files with `PRJ` extension.

4.3 Crucial attachments

Literally or "only" technically, but levels are useless without their crucial attachments. (Which you will attach in TE to the level map. Then you need to save the map to keep it.)

- Item files
- Texture files
- Sound files and their catalog
- Level script

4.3.1 Item files

An item file (attached to a level) contains all the Moveable and Static objects (Lara, creatures, statues, furniture, effect emitters etc.) and sprites [2] which can be used in that level. In TE you can select more than one item file for a level. (And if an object is there in both, then you can set which one of them should be applied in the level.)

- Builders using TRLE/NGLE previously definitely know that the extension of an item file is `WAD`, acronym standing for *Where's All the Data?* [3], made/edited probably with WadMerger program. But it is not obvious any more, if your level editor program is Tomb Editor. In TE WAD extension is usually supported, but not preferred.
- `WAD2` extension means an enhanced item file (eg. with the feature of UV-mapping [4] which is not possible in WAD files). WAD2 files are made and useable only for TE levels, it is the preferred extension here. There is a tool in TE pack dedicated to edit item files, called WadTool. Just open a WAD in it, and save it. It will be saved automatically as

[2] In computer graphics, a **sprite** is a two-dimensional bitmap that is integrated into a larger scene, most often in a 2D video game. [7]

[3] WAD is generally a main architectural component of retro games - see e.g. https://en.wikipedia.org/wiki/Doom_modding for usage in *Doom* custom games.

[4] **UV mapping** is the 3D modeling process of projecting a 3D model's surface to a 2D image for texture mapping. The letters "U" and "V" denote the axes of the 2D texture because "X", "Y", and "Z" are already used to denote the axes of the 3D object in model space. [9]

a WAD2 item file. (Or you can naturally arrange even a brand new WAD2 in WadTool.)

- TEN engine is able to use only specific WAD2 files, which means WAD extension is not supported for TEN engine. (Except if WAD has only objects having non-TEN specific behaviors, like Statics.)
  You can find a menu option in WadTool, to convert your other item files into a TEN WAD2.
  "Specific" means eg. TEN Lara object is optimized for TEN, so a "casual" (non-TEN) Lara would not be animated properly there. (Later we discuss it in this tutorial - but not the details, because it is not a TEN WAD2 tutorial.)

WAD2 files for a TEN level should be placed in Levels folder, in the subfolder of that level. (It is recommended to use a sub-subfolder there for it.) - Though, technically it is not a must, they can be placed even anywhere in your computer.

4.3.2 Texture files

A texture [5] file (attached to a level) contains all the texture tiles which can be placed on room faces (floor sector, ceiling sector, wall section) in that level.

In TE:

- you can select more than one texture file for a level, placing tiles even from each in your level,
- not only TGA extension is supported for texture files, but even other ones: BMP, JPG, PNG etc.

Texture files for a TEN level should be placed in Levels folder, in the subfolder of that level. (It is recommended to use a sub-subfolder there for it.) - Though, technically it is not a must, they can be placed even anywhere in your computer.

[5] A *texture map* refers to a 2D image ("texture") that adds visual detail to a 3D model. The image can be stored as a raster graphic. A texture that stores a specific property—such as bumpiness, reflectivity, or transparency—is also referred to as a *color map* or *roughness map*. [8]

4.3.3 Sound files and their catalog files

Sounds saved in sound files can be emitted mostly by Moveable objects (mostly Lara or other creatures), but even by any effect (like a rumbling earthquake). Etc.

Sound files have WAV extensions - just like in the old times.

Sound files are not directly attached to a level map, for two reasons:

- Sound files for a TEN level should be placed in Sounds subfolder of project main folder. (Though, technically it is not a must, they can be placed even anywhere in your computer.) Initially you can find all the original sounds here of the legacy engines, in their own subfolders. - But naturally you are allowed to edit these contents, modifying, deleting, adding sound files here.
  These places are needed to be attached to that level.

- There must be one catalog file (or, unlike the older editors, even more catalog files), attached to the level, where the sounds you want to attach are named. The main reason is that sounds can have different properties for different levels, and the catalog will describe the properties that this sound will have on this level
  Sound catalogs can be even older types, familiar from TRLE/NGLE (Sounds.txt or SFX/SAM files of WAD files), or the ones having XML extensions, which are made for TE. Later you will be able to make custom XML files, but some of them are created when saving a WAD2 file. And there are even some preset XML catalogs in Catalogs/TEN Sound Catalogs subfolder of Tomb Editor main folder.
  There is a tool in TE pack dedicated to edit sound catalogs, called SoundTool. If you want, just open a non-XML catalog into it, and save it as an XML one.

4.3.4 Level script

Script means game or level data, described simply by typing some texts. There is a tool in TE pack dedicated to edit script. This is TIDE you have already used to create your project, and you definitely will use it later much to edit your script. [6]

[6] In TRLE or NGLE there is a "level block" in the script, where all the scripted data of that level are typed. In TE it works the same way - except if the engine for your TE

4.4 Conclusion

When you start building a new level for your project, then these are your tasks, recommended in this order:

1. Create a PRJ2 level map file.
2. Add this level to the project.
3. Check the needful contents of the level script. If your script has less contents than that, then the level is useless in the game.
4. Check the main settings (in TE) for the level. If they are wrong, then the level is useless in the game.
5. Check the crucial attachments.
6. You can try to edit something in the level map - then save it.
7. Convert the PRJ2 level map into a TEN playable level.
8. Start the level in the game, try what you have just edited in the map.

[1]

project is TEN. In that case script works a bit differently, as you will see. (Though, a level block also exists there.)

Part III

Principles of game development

Chapter 5

Beta testing

5.1 In General

Beta testing is perhaps the most important phase of any video game's development. Whether you've spent a month or a few years on your adventure, if you don't have it beta tested properly, all that work could be ruined by one simple little gameplay killer. If you understand the importance of beta testing, then read this tutorial carefully and take it to heart. *If you don't understand the importance of beta testing, then read this tutorial carefully and take it to heart.* I cannot stress enough how important the beta testing phase is.

When should beta testing begin? The short answer is after alpha testing is finished. When is alpha testing finished? That's easy! That's when you think your game is ready for release. Your game is the best you can get it and you're sure there is no more you can do and you feel happy about releasing it. Now it's time for beta testing. This seems to be a difficult concept for new Level Builders (and a few seasoned Level Builders) to get their heads around. Trust me, your game does not go for beta testing until you are convinced it's finished. If it's not finished, you're still in the alpha testing stages and if you employ your beta testers now they will be wasting their time and your game will not be beta tested properly.

When a game moves from alpha testing to beta testing, the only changes you should make would be to fix bugs, gameplay errors and textural errors. You should never build new areas or new gameplay or new puzzles during or after beta testing. Again I stress, do not begin beta testing until your

game is FINISHED. Beta testing is purely to iron out problems in finished games. If you ignore this advice the chances are you will release a buggy game. Beta testers take their job seriously and if games are released with bugs it reflects on them and the job they did. If those bugs are down to the Level Builder changing things during or after beta testing and the game is released with bugs you will never get beta testers to help you again. Beta testers will invest days and even weeks of their personal time to help you with your game. Respect that and don't let them down.

5.2 Alpha Testing

Don't cut corners and skip this step. Yes, I know, you're excited about your game and you want to show it off and that's marvellous and wonderful and admirable and quite understandable, but your beta testers are not there to do your job for you. If you upload an appallingly buggy game that hasn't been properly alpha tested, don't be surprised if your beta testers don't do a good job and don't volunteer later when you actually need beta testers. Alpha testing is your job, not your beta testers, and it's extremely important. I consider my alpha testing to be complete when I have a package I think is ready for release. In other words, I can find nothing to change or fix and in my heart I know my game is the best I can make it and I feel it's ready for release. Now I begin beta testing.

During alpha testing, it is easy to move Lara around and forget to put her back, place temporary triggers to open a door and then forget to remove them and a few other little things, so after I've packaged up my game for beta testing guess what? Yes, I then test the beta package from start to finish using only the files I will be uploading for my beta testers. This is my final Alpha test. If this goes well and all the level jumps work properly and I successfully hit a finish trigger, only then is the game uploaded and the download link sent out to the beta testers.

[6]

Index

Next Generation Level Editor, 8
NGLE, 8

Sprite, 18

Texture map, 19
Tomb Raider Level Editor, 7
Tomb Raider Next Generation, 8
TRLE, 7
TRNG, 8

UV mapping, 18

Bibliography

[1] AkyV. *TEN - How to start building a new project, a new level*. https://www.tombraiderforums.com/showthread.php?t=229293.

[2] Eidos Interactive Core Design Ltd. *Tomb Raider Level Editor manual*. https://core-design.com/goodies_tr5_downloads.html. 2000.

[3] *Definition of Next Generation Level Editor on WikiRaider*. https://www.wikiraider.com/index.php/Next_Generation_Level_Editor.

[4] MontyTRC89. *Tomb Editor page on GitHub*. https://github.com/MontyTRC89/Tomb-Editor.

[5] Paolone. *Official website of the Next Generation Tools*. http://www.trlevelmanager.eu/ng.htm.

[6] George Paolone and AkyV. *The NGLE Manual*. https://www.treditor.hu/7/gmac/nglemanual.html.

[7] Wikipedia. *Sprite (computer graphics)*. https://en.wikipedia.org/wiki/Sprite_(computer_graphics).

[8] Wikipedia. *Texture Mapping*. https://en.wikipedia.org/wiki/Texture_mapping.

[9] Wikipedia. *UV mapping*. https://en.wikipedia.org/wiki/UV_mapping.