

Particle Tool

by Atlas and ImBee

Documentation + manual

A bit of theory:

Particles are saved in **.msh** and **.prt** formats. The **.msh** format is used in the following games:

- Earth 2150: Escape from the Blue Planet,
- Earth 2150: The Moon Project,
- Earth 2150: Lost Souls,
- World War III: Black Gold,
- Heli Heroes,
- Frontline Attack: War over Europe/World War II: Panzer Claws II,
- Polanie II/KnightShift/Once Upon a Knight,
- Polanie III/KnightShift II Curse of Souls,

However, **.prt** occurs in:

- Earth 2160,
- 3D ParticleGen Visual FX (steam),
- Two Worlds,

The titles after Two Worlds were not included here because I no longer considered them.

Each game/title has the **.msh** or **.prt** format, but this does not mean that the **.prt** format from **Earth 2160** is the same as the format from **Two Worlds**, even though both have the **.prt** extension.

Below are groups of titles that have the same format. Each group begins with a new arrow:

.msh format:

- Earth 2150: Escape from the Blue Planet, Earth 2150: The Moon Project, Earth 2150: Lost Souls;
- World War III: Black Gold, Heli Heroes;
- Frontline Attack: War over Europe/World War II: Panzer Claws II;
- Polanie II/KnightShift/Once Upon a Knight, Polanie III/KnightShift II Curse of Souls;

.prt format:

- Earth 2160;
- 3D ParticleGen Visual FX (steam);
- Two Worlds;

The general structure of particle files can be divided into two types:

1. Dynamic:

- can only occur in **.msh**,
- appears in games:
 - * Earth 2150: Escape from the Blue Planet, Earth 2150: The Moon Project, Earth 2150: Lost Souls;
 - * World War III: Black Gold, Heli Heroes;
 - * Frontline Attack: War over Europe/World War II: Panzer Claws II;
 - * Polanie II/KnightShift/Once Upon a Knight, Polanie III/KnightShift II Curse of Souls;
- It is based on the structure of a single object, which can have nested children, and those children can also have their own children (**tree structure**).

2. ParticlesEmitter

- may occur in **.msh** and **.prt**,
- appears in games/titles:
 - * Polanie II/KnightShift/Once Upon a Knight, Polanie III/KnightShift II Curse of Souls **(this type appeared in KnightShift for the first time.);**
 - * Earth 2160;
 - * 3D ParticleGen Visual FX;
 - * Two Worlds;
- based on a structure composed of objects such as: Effect, Emitter, Particle, and PairParticleEmitter (also known as Group),

Operations can be performed on all these formats using the appropriate software. The list is presented below:

I. Dynamic:

1. Dynamic particle (msh) from Earth 2150: Escape from the Blue Planet, Earth 2150: The Moon Project, Earth 2150: Lost Souls;

- + You can convert it to a **.myaod** text file and edit it using a text editor such as Notepad++,
- + You can preview in the game,
- + You can convert to Dynamic particle from another game,
- It is not possible to edit in an external program with a particles preview because such a program does not yet exist.

2. Dynamic particle (msh) from World War III: Black Gold and Heli Heroes;

- + You can convert it to a **.myaod** text file and edit it using a text editor such as Notepad++,
- + You can preview in the game,
- + You can convert to Dynamic particle from another game,
- It is not possible to edit in an external program with a particles preview because such a program does not yet exist.

3. Dynamic particle (msh) from Frontline Attack: War over Europe/World War II: Panzer Claws II;

- + You can convert it to a **.myaod** text file and edit it using a text editor such as Notepad++,
- + You can preview in the game,
- + You can convert to Dynamic particle from another game,
- It is not possible to edit in an external program with a particles preview because such a program does not yet exist.

4. Dynamic particle (msh) from Polanie II/KnightShift/Once Upon a Knight, Polanie III/KnightShift II Curse of Souls;

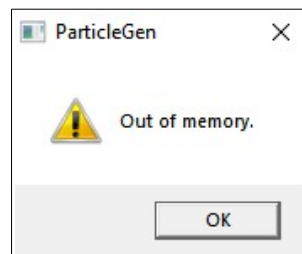
- + You can convert it to a **.myaod** text file and edit it using a text editor such as Notepad++,
- + You can preview in the game,
- + You can convert to Dynamic particle from another game,
- It is not possible to edit in an external program with a particles preview because such a program does not yet exist.

II. ParticlesEmitter:

1. ParticlesEmitter (msh) from Polanie II/KnightShift/Once Upon a Knight, Polanie III/KnightShift II Curse of Souls;

- + You can convert it to a **.myaod** text file and edit it using a text editor such as Notepad++,
- + It can be viewed in the game,
- + You can convert to ParticleEmitter from another game, e.g. **Earth 2160**.
- + The ability to convert to **.prt** format from **Earth 2160** and edit it in an external program with particle preview. This program is called **ParticleEdit** from the **Earth 2160 Digital Deluxe Content** package (**available for purchase on Steam**) (**requires Earth 2160**), specifically from the **Earth2160_SDK** directory.

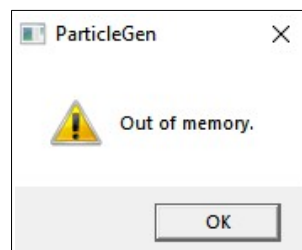
- You can convert to **.prt** format from **Two Worlds** and **3D ParticleGen Visual FX (Steam)**, but when testing the converted file in **ParticleGen** for **Two Worlds** and **Particle Gen (Steam)**, an “**Out of memory**” error pops up.



2. ParticlesEmitter (prt) from Earth 2160;

- + You can convert it to a **.myaod** text file and edit it using a text editor such as Notepad++,
- + You can preview in the game,
- + You can convert to ParticleEmitter from another game, e.g. **KnightShift**,
- + It can be edited in an external program with a particle preview. The program is called **ParticleEdit** from the **Earth 2160 Digital Deluxe Content** package (**available for purchase on Steam**) (**requires Earth 2160**), specifically from the **Earth2160_SDK** directory.
- + After conversion to **KnightShift** format, it is compatible with **KnightShift**.

- You can convert to **.prt** format from **Two Worlds** and **3D ParticleGen Visual FX (Steam)**, but when testing the converted file in **ParticleGen** for **Two Worlds** and **Particle Gen (Steam)**, an “**Out of memory**” error pops up.



3. ParticleEmitter (prt) from Two Worlds;

- + You can convert it to a **.myaod** text file and edit it using a text editor such as Notepad++.
- + You can preview it in the game,
- + You can convert to ParticleEmitter from **3D ParticleGen Visual FX (steam)**.
- + After conversion to **3D ParticleGen Visual FX (steam)** format, it is compatible with **3D ParticleGen Visual FX (steam)**.
- + It can be edited in an external program with a particle preview. The program is called **ParticleEdit** from the **Two Worlds SDK** package.

<https://www.moddb.com/downloads/two-worlds-software-development-kit-tools-13-installer>

<https://steamcommunity.com/app/1930/discussions/0/1651043958625749369/>

https://www.dropbox.com/scl/fo/wp2y2ocm0qtsl0dwqknmg/AMQmiaJytYC3bS4h9I9Qcb0/two_worlds_editor?rlkey=6wxvavfom21nl1mj42zu4llpq&e=1&dl=0

(requires Two Worlds to be installed)

- You can convert to **.prt** format from **Earth 2160**. Sometimes it works, sometimes it doesn't. Sometimes it just displays a rectangle, sometimes it displays an **“Out of memory”** message, and sometimes the program simply crashes,
- You can convert it to **KnightShift** format, but when you try to view this particle in the game, the game crashes,

4. ParticleEmitter (prt) from 3D ParticleGen Visual FX;

- + You can convert it to a **.myaod** text file and edit it using a text editor such as Notepad++.
 - + You can view it in the program,
 - + You can convert to ParticleEmitter from **Two Worlds**,
 - + After conversion to the **Two Worlds** format, it is compatible with **Two Worlds**,
 - + You can edit it in a program with a particle preview. The program is called **3D ParticleGen Visual FX (available for purchase on Steam)**.
- You can convert to **.prt** format from **Earth 2160**. Sometimes it works, sometimes it doesn't. Sometimes it just displays a rectangle, sometimes it displays an **“Out of memory”** message, and sometimes the program simply crashes.
 - You can convert it to **KnightShift** format, but when you try to view this particle in the game, the game crashes.

Therefore, based on the above information about formats, we can create a graph for Dynamic particles and a graph for ParticleEmitters.

I. Dynamic:

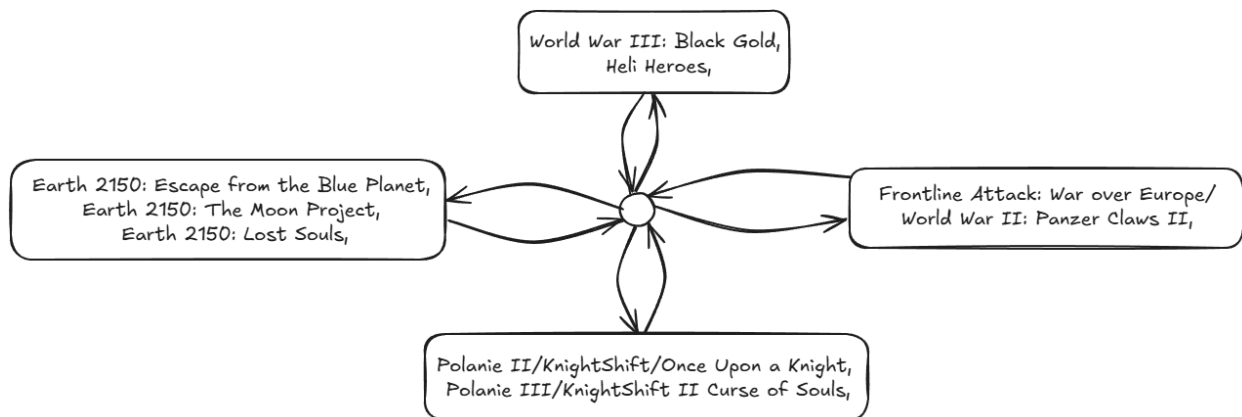


Figure 1: Conversion between Dynamic Particles formats is possible in every case.

II. ParticleEmitter:

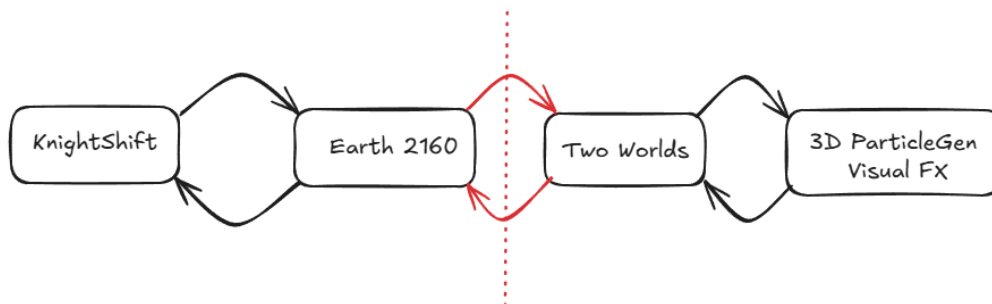


Figure 2: There is a format barrier between Earth 2160 and Two Worlds that prevents, for example, the KnightShift format from being launched in the particle editor for Two Worlds.

What is Particle Tool By Atlas and ImBee and what is it used for?

Particle Tool By Atlas and ImBee is a package of two programs used to convert particle files from **Reality Pump** games in order to enable easier editing and running of the same particle in different games from this studio. This package of programs is a “bridge” between different particle formats. These programs support the following games/titles:

- Earth 2150: Escape from the Blue Planet,
- Earth 2150: The Moon Project,
- Earth 2150: Lost Souls,
- World War III: Black Gold,
- Heli Heroes,
- Frontline Attack: War over Europe/World War II: Panzer Claws II,
- Polanie II/KnightShift/Once Upon a Knight,
- Polanie III/KnightShift II Curse of Souls
- Earth 2160,
- 3D ParticleGen Visual FX (steam),
- Two Worlds,

The package consists of two programs:

- Particle2MyAod.exe
- MyAod2Particle.exe

Particle2MyAod:

Particle2MyAod.exe is a program used to export data from particles in **.msh** and **.prt** formats from the titles listed above to **.myaod** format, which is very similar to **.aod**. Simply enter the name of the particle file along with the format.

File with ParticleEmitter:

If ParticleEmitter is detected, the program will ask you to specify the final format. You can choose **y/n**. If you choose **n**, the program will extract the particle file to **.myaod** without specifying a specific format, i.e., the program will simply extract the particle. If you enter **y**, you will be able to force the format to which **.myaod** will later be compiled (e.g., if you want to run particles from **KnightShift** in **Earth 2160**, enter **y**, and then force the **e2160** format. The **MyAod2Particle** program will then know how to compile **.myaod** to the **Earth 2160** format). The program will then extract the particle file.

The exported particles folder contains a **.myaod** file and an **_extra_data.cpp** file. The **_extra_data** file contains the entire beginning of the particles file in the form of specified variable values. You can modify this, but you don't have to. The most important file is **.myaod**.

The program also works in **argc&argv** mode, so you can call it from **cmd** or **powershell** and enter the argument next to the program name, which will export the particles without forcing the format:

Particle2MyAod.exe <input_file_name>

You can also enforce the correct format in the following way:

Particle2MyAod.exe <input_file_name> -force <ks/tw/pg/e2160>

or

Particle2MyAod.exe <input_file_name> --force <ks/tw/pg/e2160>

File with Dynamic particle:

Unfortunately, Dynamic particles are handled by editing the **DynamicParticle.cfg** configuration file. Depending on the title from which the particles originate, this configuration file must be modified – preferably immediately. The following options are set by default, which mean:

dynamic_particle_input_format = ks;

(The input Dynamic particle will come from KnightShift.)

force_specific_export_format = false;

(Should a specific format be enforced?)

forced_export_format = ks;

(The final output format for Dynamic particles will be the format used in the game KnightShift.)

To handle a specific format from another game where there are dynamic particles, simply enter one of these formats: **e2150**, **ww3_or_hh**, **ww2_or_fa_or_pc2**, **ks**

in this place after the sign =

dynamic_particle_input_format = <enter the format here>;

Aby konwertować do innej gry należy odblokować przełącznik ***force_specific_export_format*** na ***true***.

The final format to which we are converting should be entered in ***forced_export_format*** after the = sign.

forced_export_format = <enter the format here>;

The compiler should already know how to compile a directory with exported particles.

I created my own special compiler, **MyAod2Particle.exe**, to compile the **.myaod** format.

To automatically extract all files without forcing a format, you can use **_EXPORT_ALL.ps1** or **_EXPORT_ALL.bat**.

MyAod2Particle:

MyAod2Particle.exe is a program used to compile **.myaod** particle files into a predefined format from a given **Reality Pump** game.

The program is launched by double-clicking and entering the name of the input directory. If the **extra_data** file in the directory does not exist, the program will create a classic particles header by generating the appropriate information, but if the **extra_data** file exists, it will recompile this file into a particles header.

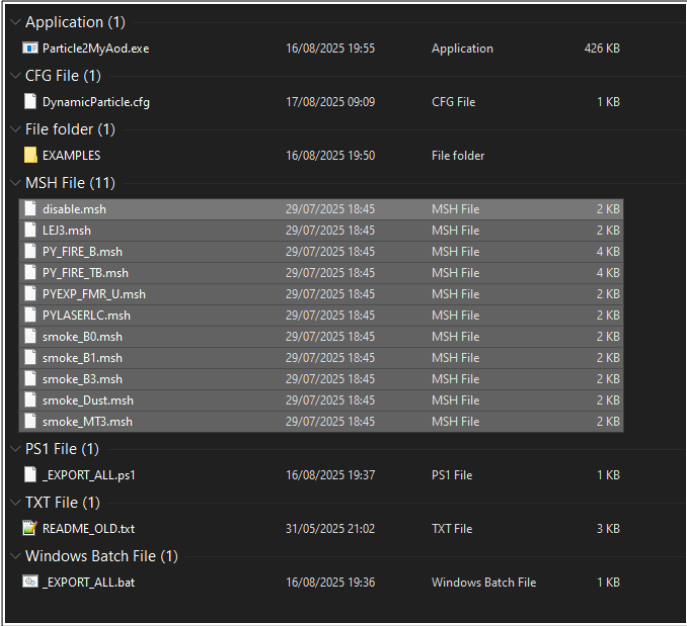
The program also works in **argc&argv** mode, so you can call it from cmd or powershell and enter the argument next to the program name, i.e.

MyAod2Particle.exe <input directory name>

To automatically compile/import all folders, you can use **_IMPORT_ALL.ps1** or **_IMPORT_ALL.bat**.

Typical and example scenarios for using ParticleTool:

a) I want to export Dynamic particles from Earth 2150:



Application (1)			
Particle2MyAod.exe	16/08/2025 19:55	Application	426 KB
CFG File (1)			
DynamicParticle.cfg	17/08/2025 09:09	CFG File	1 KB
File folder (1)			
EXAMPLES	16/08/2025 19:50	File folder	
MSH File (11)			
disable.msh	29/07/2025 18:45	MSH File	2 KB
LEJ3.msh	29/07/2025 18:45	MSH File	2 KB
PV_FIRE_B.msh	29/07/2025 18:45	MSH File	4 KB
PV_FIRE_TB.msh	29/07/2025 18:45	MSH File	4 KB
PVEXP_FMR_U.msh	29/07/2025 18:45	MSH File	2 KB
PYLASERLC.msh	29/07/2025 18:45	MSH File	2 KB
smoke_B0.msh	29/07/2025 18:45	MSH File	2 KB
smoke_B1.msh	29/07/2025 18:45	MSH File	2 KB
smoke_B3.msh	29/07/2025 18:45	MSH File	2 KB
smoke_Dust.msh	29/07/2025 18:45	MSH File	2 KB
smoke_MT3.msh	29/07/2025 18:45	MSH File	2 KB
PS1 File (1)			
_EXPORT_ALL.ps1	16/08/2025 19:37	PS1 File	1 KB
TXT File (1)			
README_OLD.txt	31/05/2025 21:02	TXT File	3 KB
Windows Batch File (1)			
_EXPORT_ALL.bat	16/08/2025 19:36	Windows Batch File	1 KB

Figure 3: For example, I have a pool of particles from E2150.

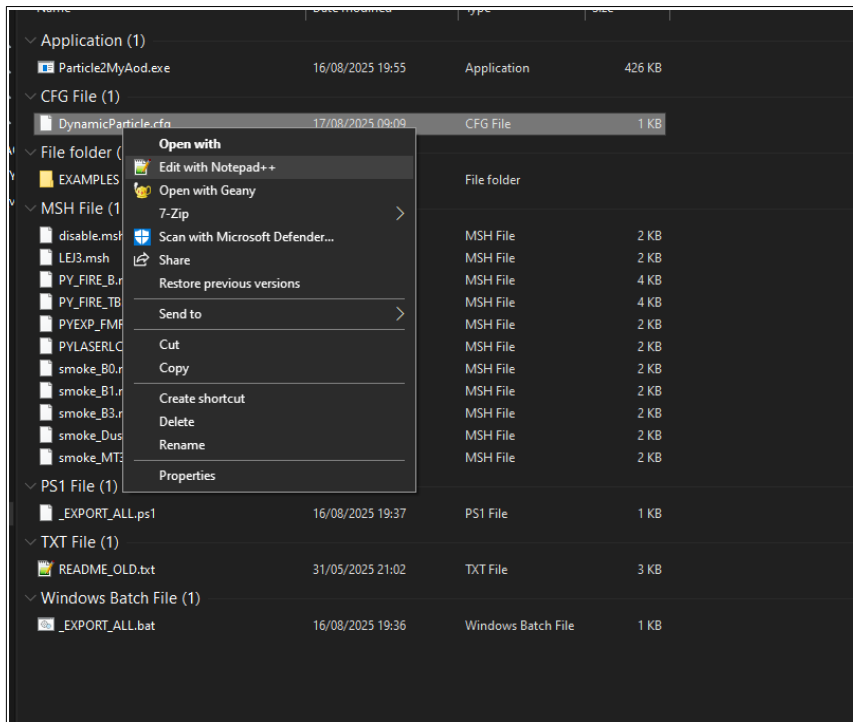


Figure 4: I am editing DynamicParticle.cfg.

```
// Dynamic Particle Config:
dynamic_particle_input_format = ks;
force_specific_export_format = false;
forced_export_format = ks;
// Available formats:
// e2150
// ww3_or_hh
// ww2_or_fa_or_pc2
// ks
```

Figure 5: I see options.

```
// Dynamic Particle Config:
dynamic_particle_input_format = e2150;
force_specific_export_format = false;
forced_export_format = e2150;
// Available formats:
// e2150
// ww3_or_hh
// ww2_or_fa_or_pc2
// ks
```

Figure 6: I change the format to the appropriate input format.

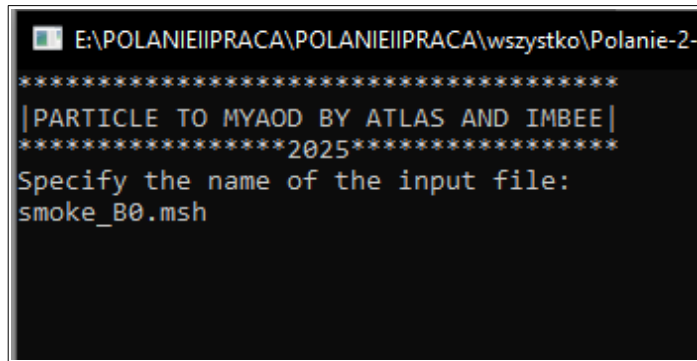


Figure 7: I open the program and enter the file name.
Then I press enter.

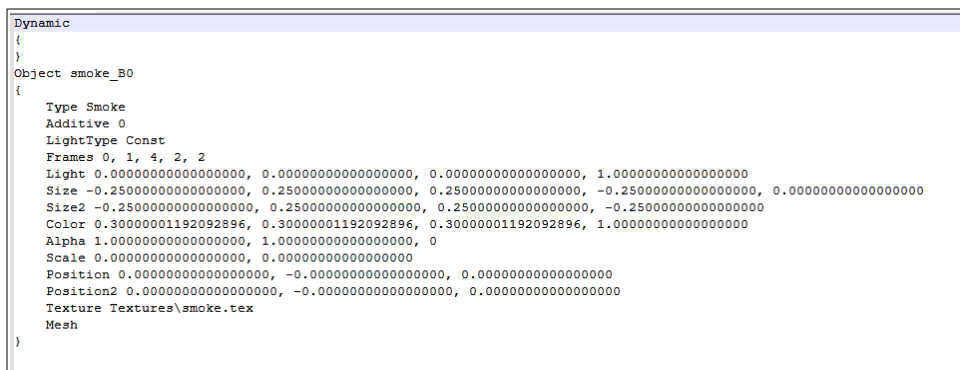


Figure 8: Done—I can edit Dynamic Particles data.

b) I want to convert Dynamic particles from Earth 2150 to World War III Black Gold:

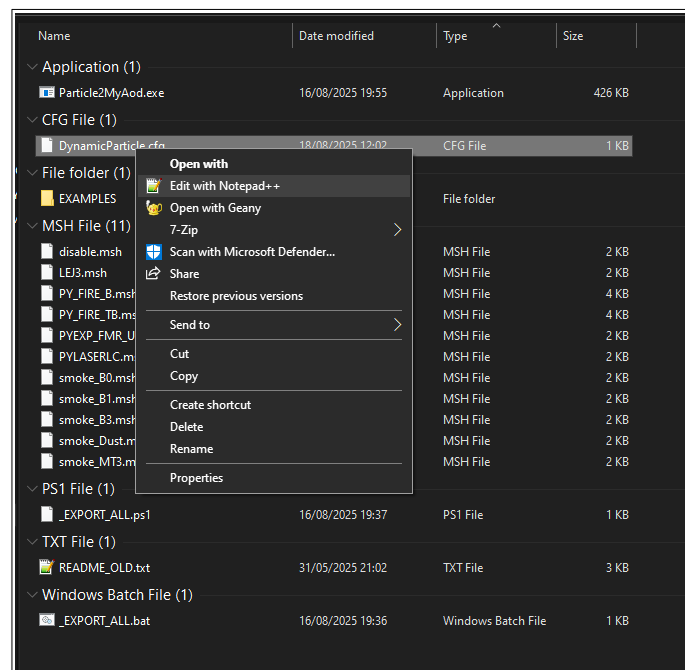


Figure 9: I am editing DynamicParticle.cfg.

```
// Dynamic Particle Config:
dynamic_particle_input_format = e2150;
force_specific_export_format = false;
forced_export_format = e2150;
// Available formats:
// e2150
// ww3_or_hh
// ww2_or_fa_or_pc2
// ks
```

Figure 10: I notice the options.

```
// Dynamic Particle Config:
dynamic_particle_input_format = e2150;
force_specific_export_format = true;
forced_export_format = ww3_or_hh;
// Available formats:
// e2150
// ww3_or_hh
// ww2_or_fa_or_pc2
// ks
```

Figure 11: I set the input format to e2150, unlock force_specific_export_format to true, and set forced_export_format to ww3_or_hh.

```
E:\POLANIE\PRACA\POLANIE\PRACA\wszystko\Polanie-2-Da
*****
|PARTICLE TO MYAOD BY ATLAS AND IMBEE|
*****2025*****
Specify the name of the input file:
LEJ3.msh
```

Figure 12: I open the program, enter the name of the particle, and press enter.

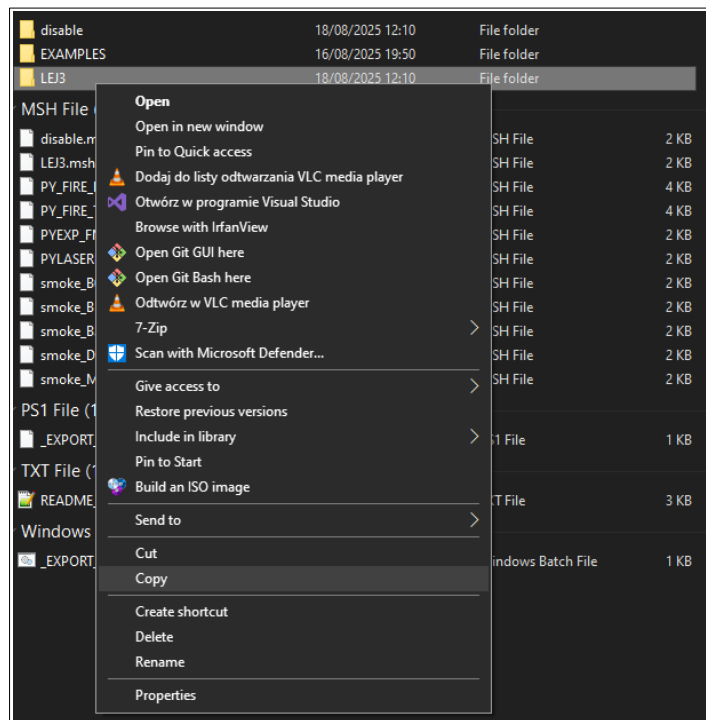


Figure 13: I copy the received directory with the exported particles to the directory from MyAod2Particle.

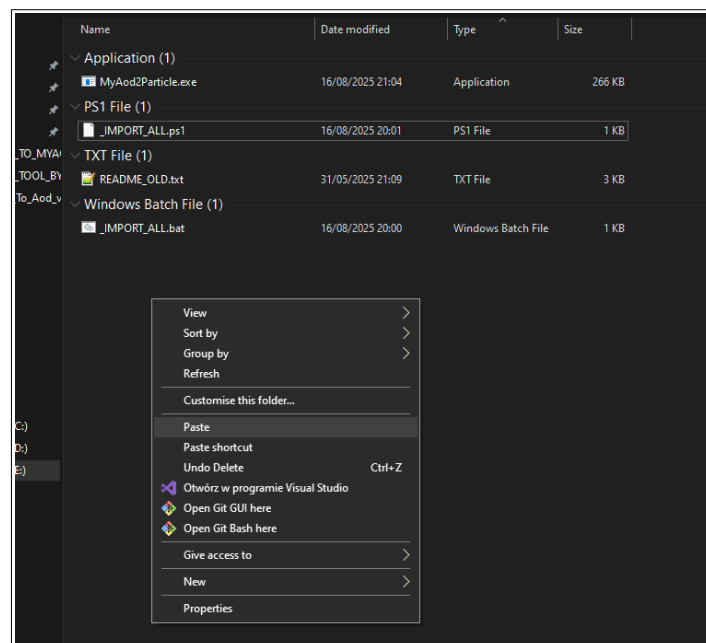


Figure 14: I'll paste it here.

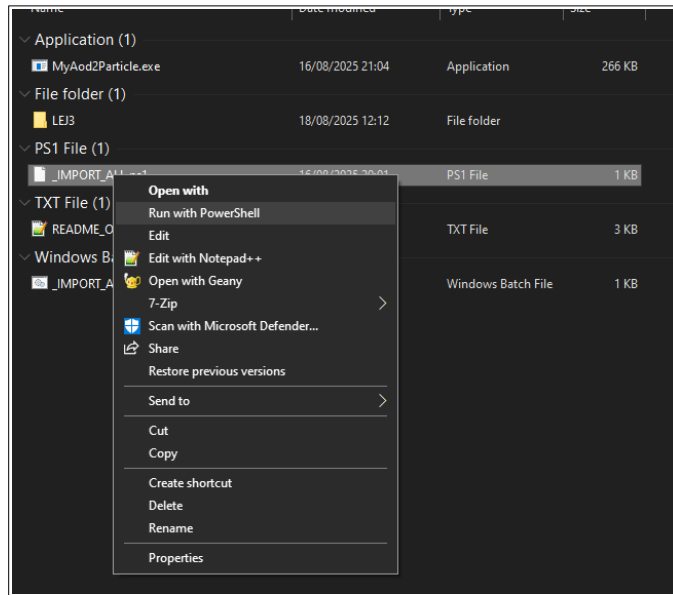


Figure 15: We can launch the program by double-clicking and entering the name of the directory to be compiled, or we can launch the `_IMPORT_ALL.ps1` script via PowerShell to compile all directories with particle data.

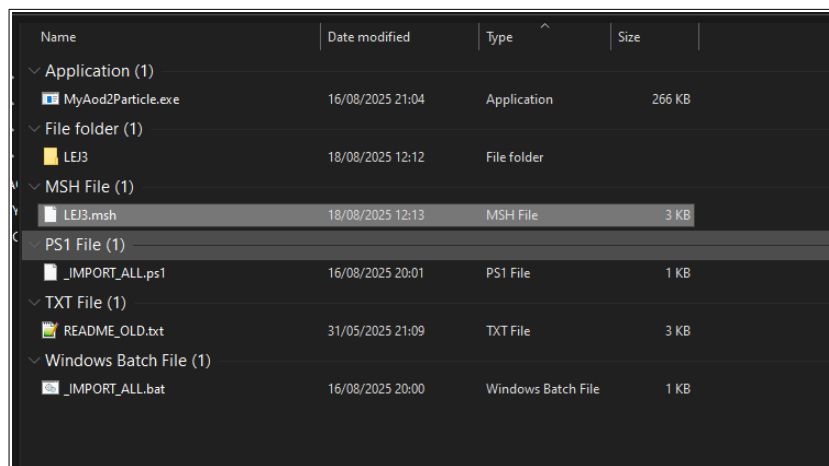


Figure 16: Done - Particles is ready to be added to the game.

c) I want to convert ParticleEmitter from KnightShift to Earth 2160:

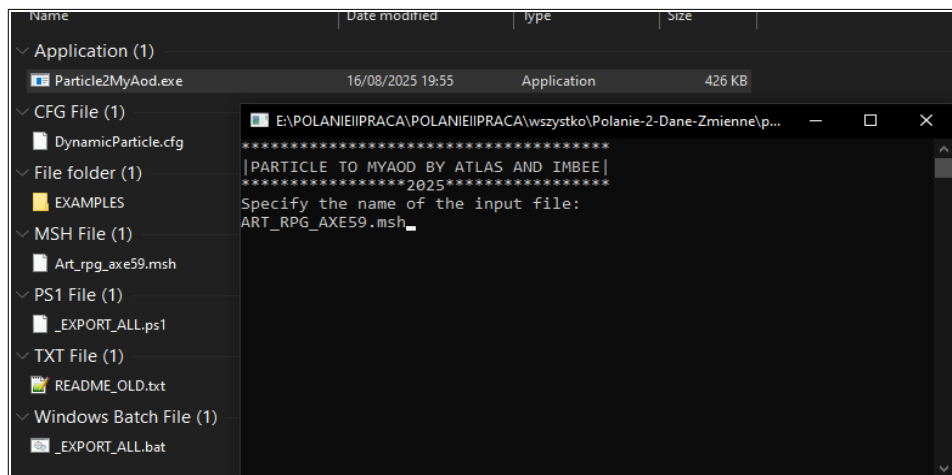


Figure 17: I double-click Particle2MyAod.exe and enter the name of the .msh file.

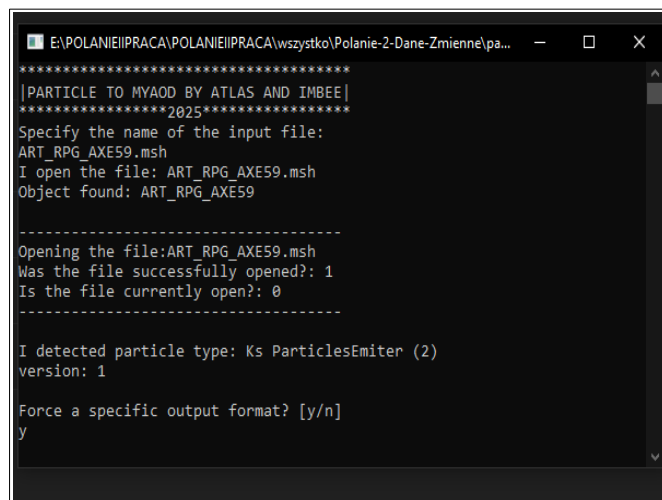


Figure 18: I am introducing y.

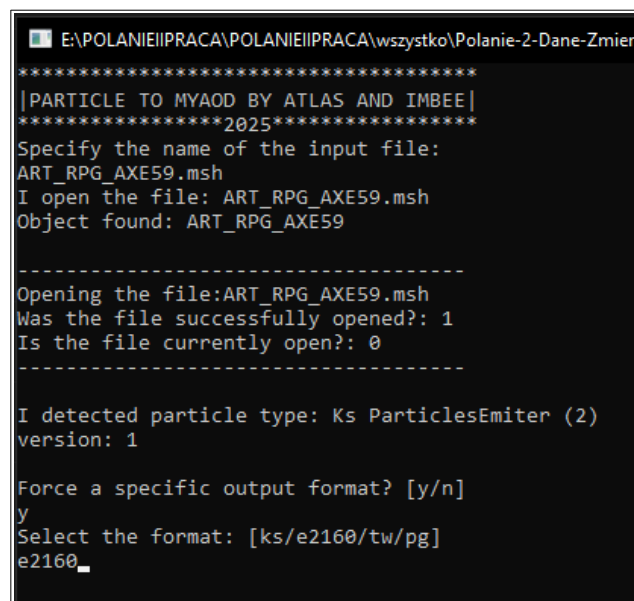


Figure 19: I am forcing the e2160 format.

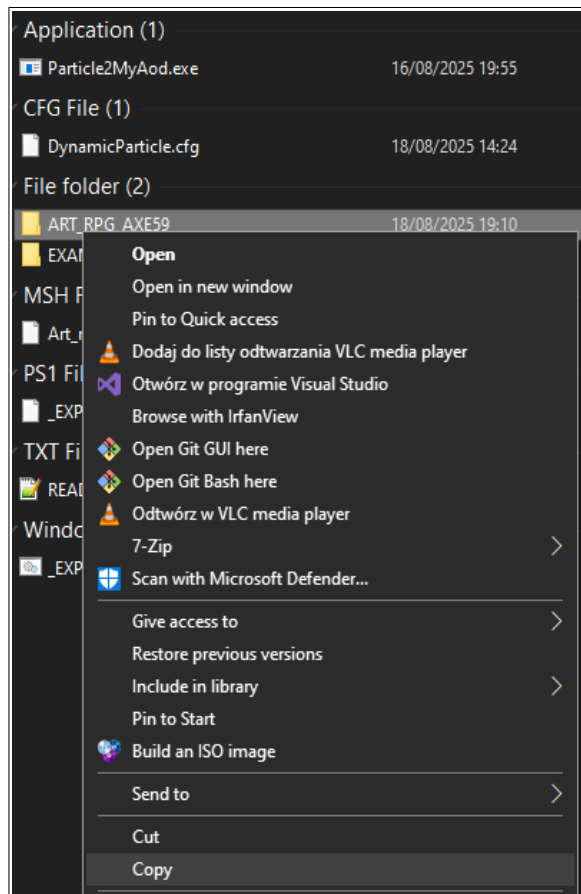


Figure 20: I copy the directory with particlesa data to the directory with MyAod2Particle.

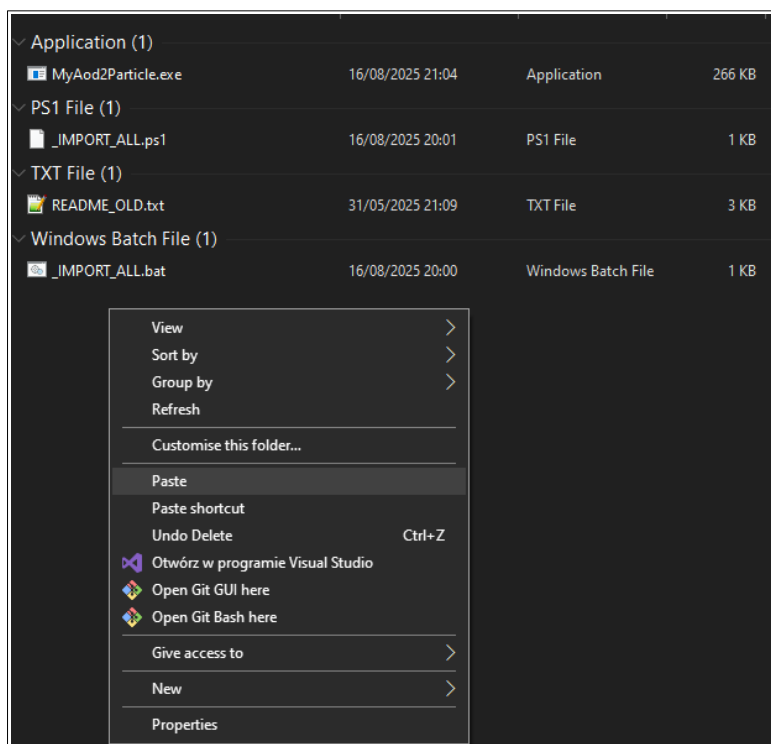


Figure 21: I'm pasting it.

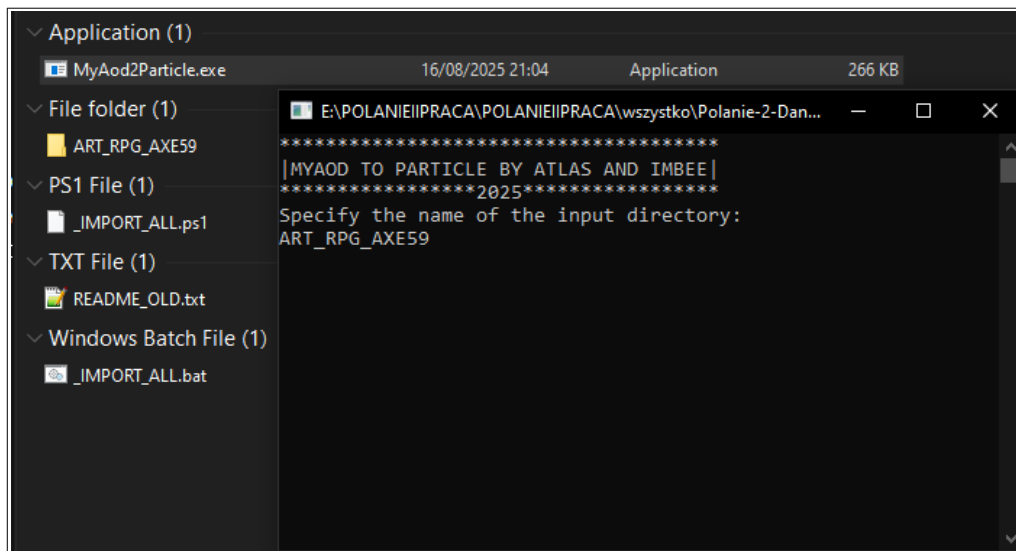


Figure 22: I double-click to open the program and enter the name of the directory containing the particles data.

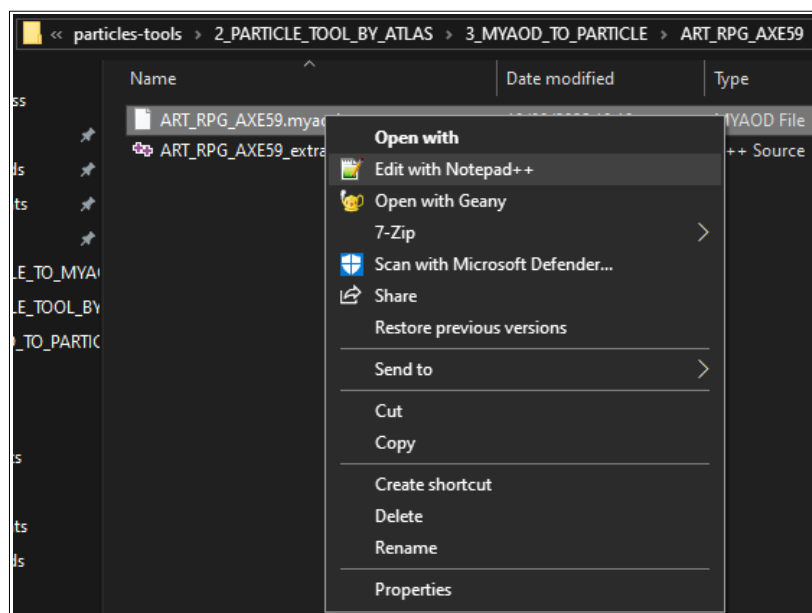


Figure 23: I go to the directory with the extracted particles data and check the contents of the **.myaod** file.


```

3418 {
3419     Emitter E4
3420     EmitterIndex 3
3421     Particle P4
3422     ParticleIndex 3
3423     annotation
3424     time 0.0000000000000000
3425     loopedEmission 2, 1, 0, 0.0000000000000000
3426     drawParticleEmitter 1
3427     simpleOneParticleSwitch 0
3428     2Dmask 0
3429     hardwareCursor 0
3430     stopInPartialPause 0
3431     finishMissile 0
3432     emitsGroupsSwitch 0
3433     emitsGroups
3434     onlyEmittedByOtherEmitterSwitch 0
3435 }
3436 gameRate 20
3437 endValue_0 0.0000000000000000
3438 endValue_1 0.0000000000000000

```

Figure 24: I check and remember the gamerate.

Name	Date modified	Type	Size
ParticleEdit_NO_SSE.exe	17/08/2025 09:55	Shortcut	3 KB
ParticleEdit_SSE.exe	17/08/2025 09:55	Shortcut	3 KB

Figure 25: I launch ParticleEdit.exe from E2160.

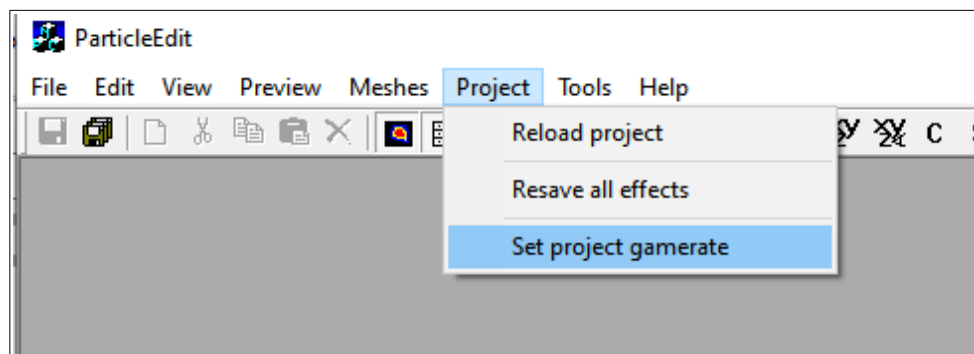


Figure 26: I locate and click on the **Set project gamerate** option.

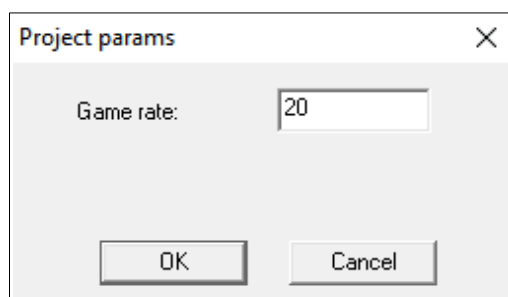


Figure 27: I set the value to the one we read from the file earlier.

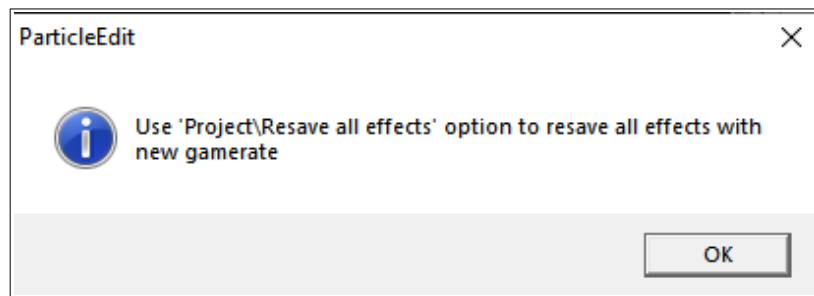


Figure 28: This message is displayed.

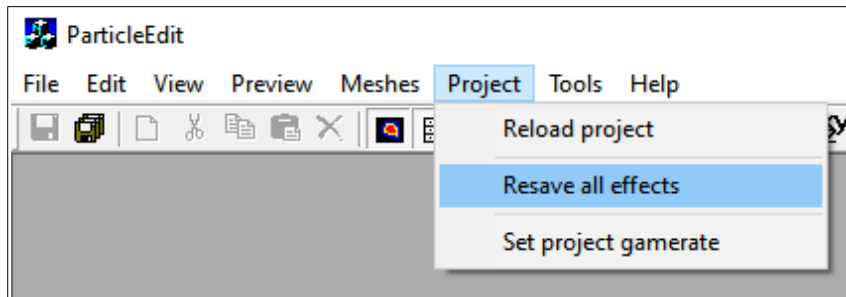


Figure 29: We locate and click **Resave all effects**. To ensure that the operation is performed correctly, please wait a moment.



Figure 30: Once the operation is complete, we shut down the program.

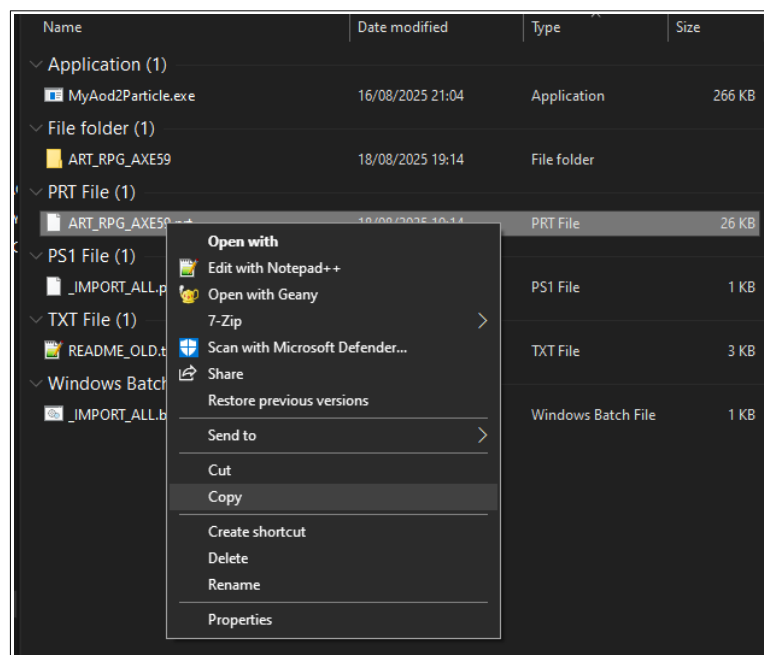


Figure 31: Copies the output .prt file.

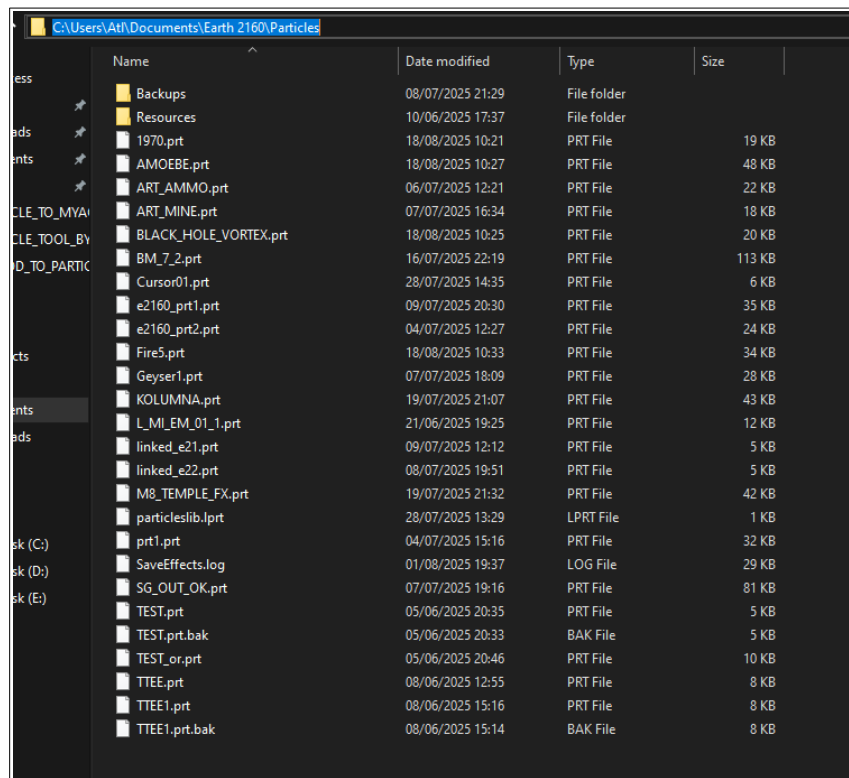


Figure 32: I am locating the directory with .prt files for the ParticleEdit program from Earth 2160.

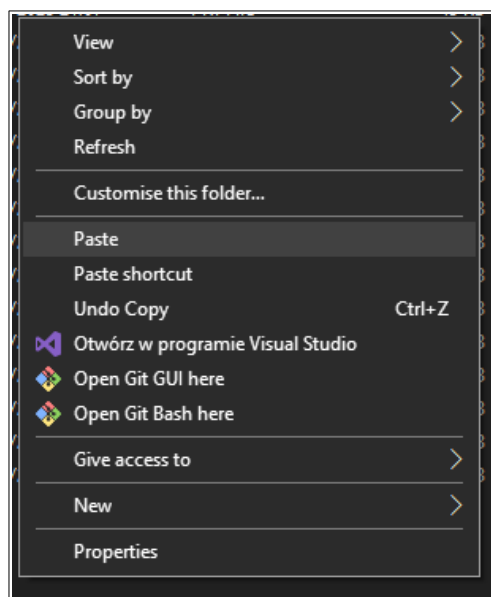


Figure 33: I'm pasting it.



	ParticleEdit_NO_SSE.exe	17/08/2025 09:55	Shortcut	3 KB
	ParticleEdit_SSE.exe	17/08/2025 09:55	Shortcut	3 KB

Figure 34: I am restarting ParticleEdit from Earth 2160.

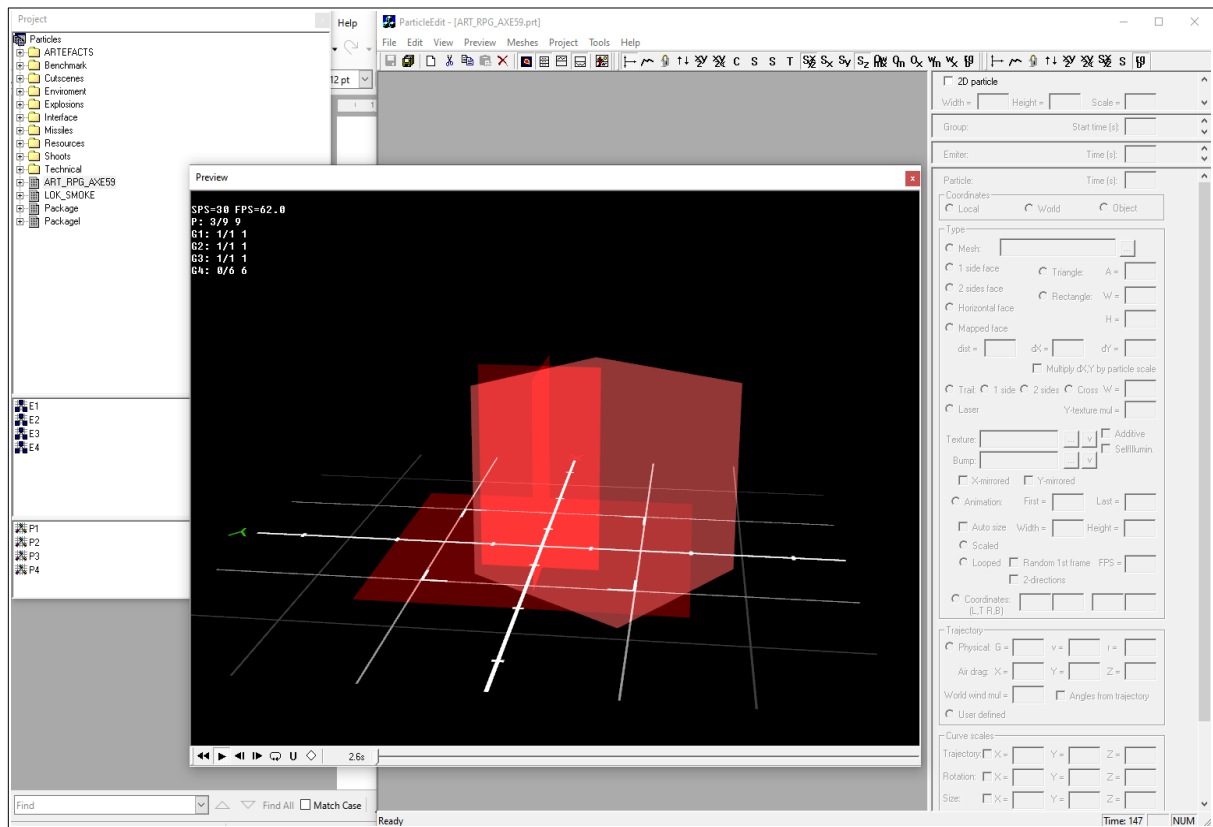


Figure 35: As we can see, the particle has started, but to get rid of these red textures, you need to convert .tex to .tga and put the textures in the Textures directory in the E2160 game directory.

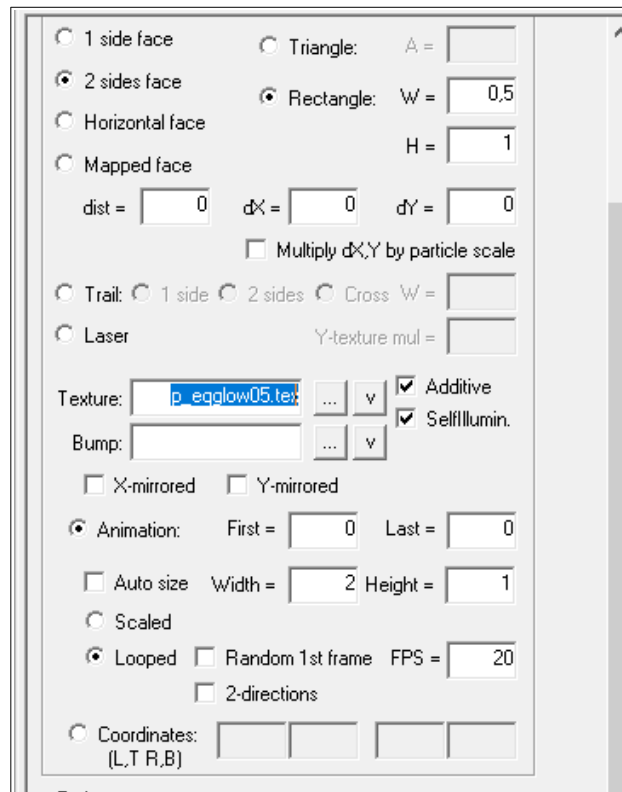


Figure 36: You can jump between ParticleEmitter objects and change the names of textures and meshes.

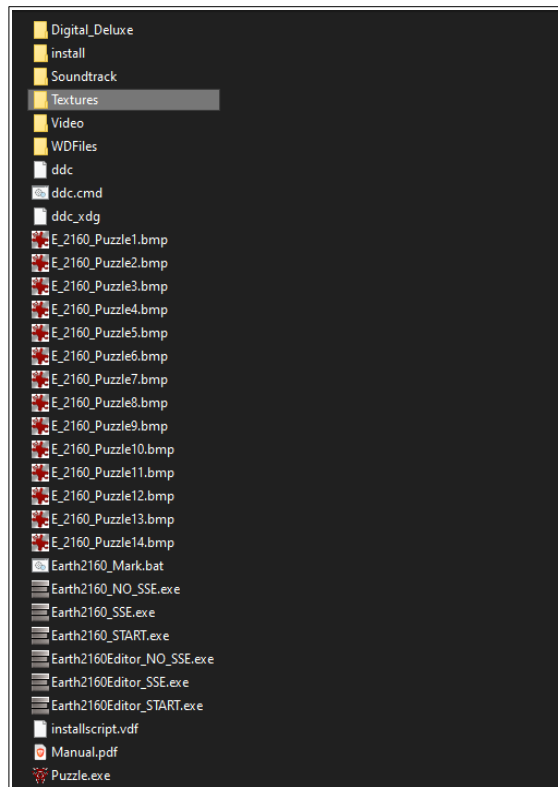


Figure 37: We create the Textures folder in the Earth 2160 game folder and place our textures in .tga format there. This way, you don't need to pack these files into WD.

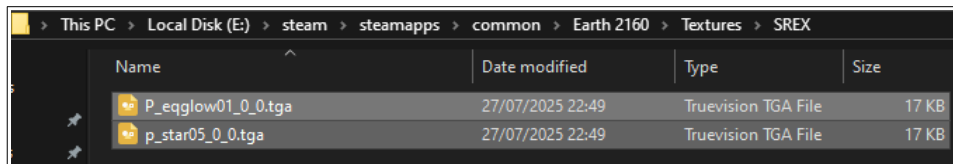


Figure 38: I added these textures myself.

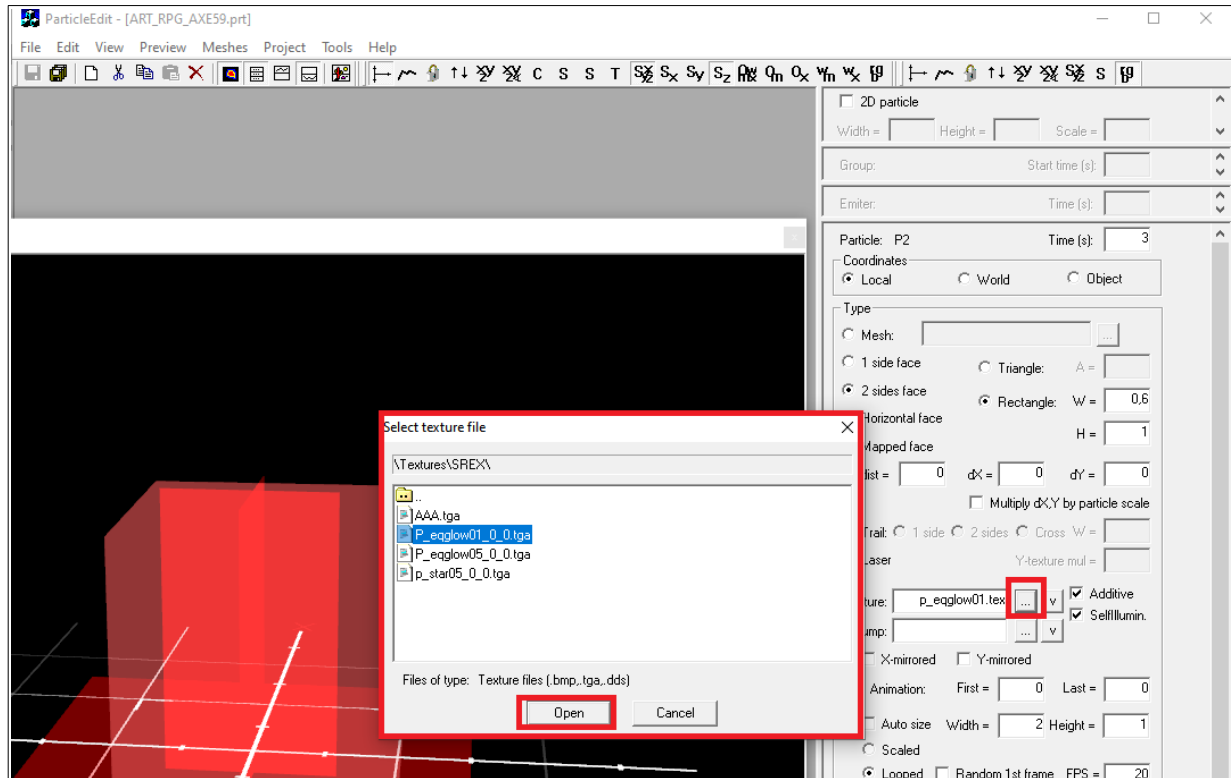


Figure 39: We select tga textures from the texture catalog. We do this for each .tex texture. At the same time, we can also change the mesh to, for example, Sphere.

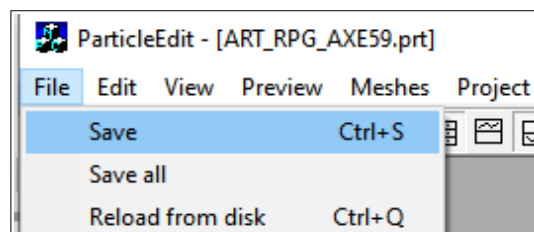


Figure 40: We save the particles.

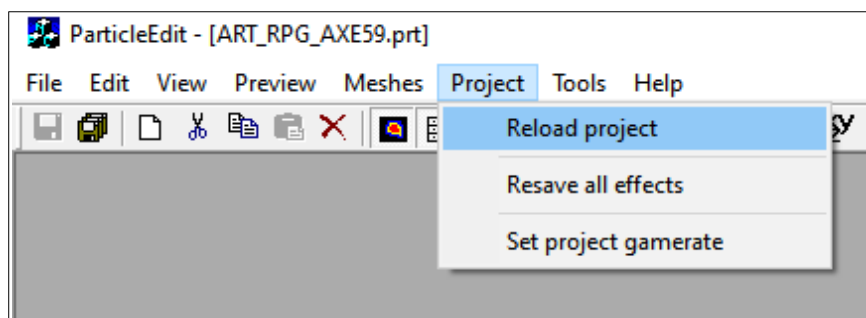


Figure 41: To refresh the project in order to see the changes, we can click on the **Reload project** option.

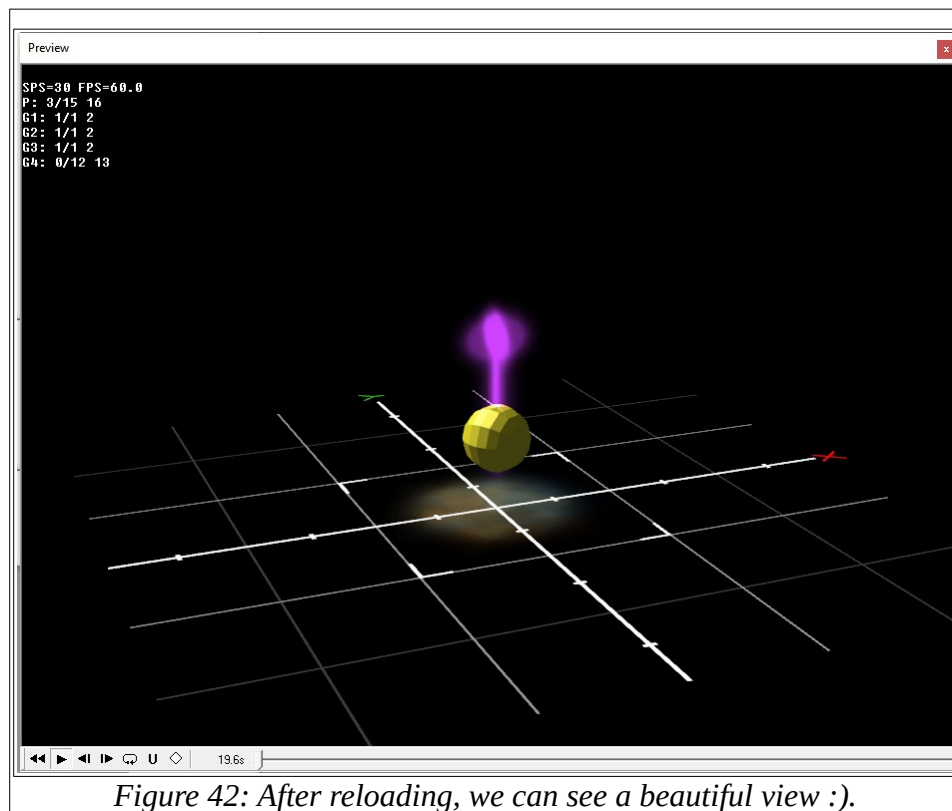


Figure 42: After reloading, we can see a beautiful view :).

d) I want to make changes to the ParticleEmitter from KnightShift.
We export the file to E2160 format as in point c). We edit it as we see fit.

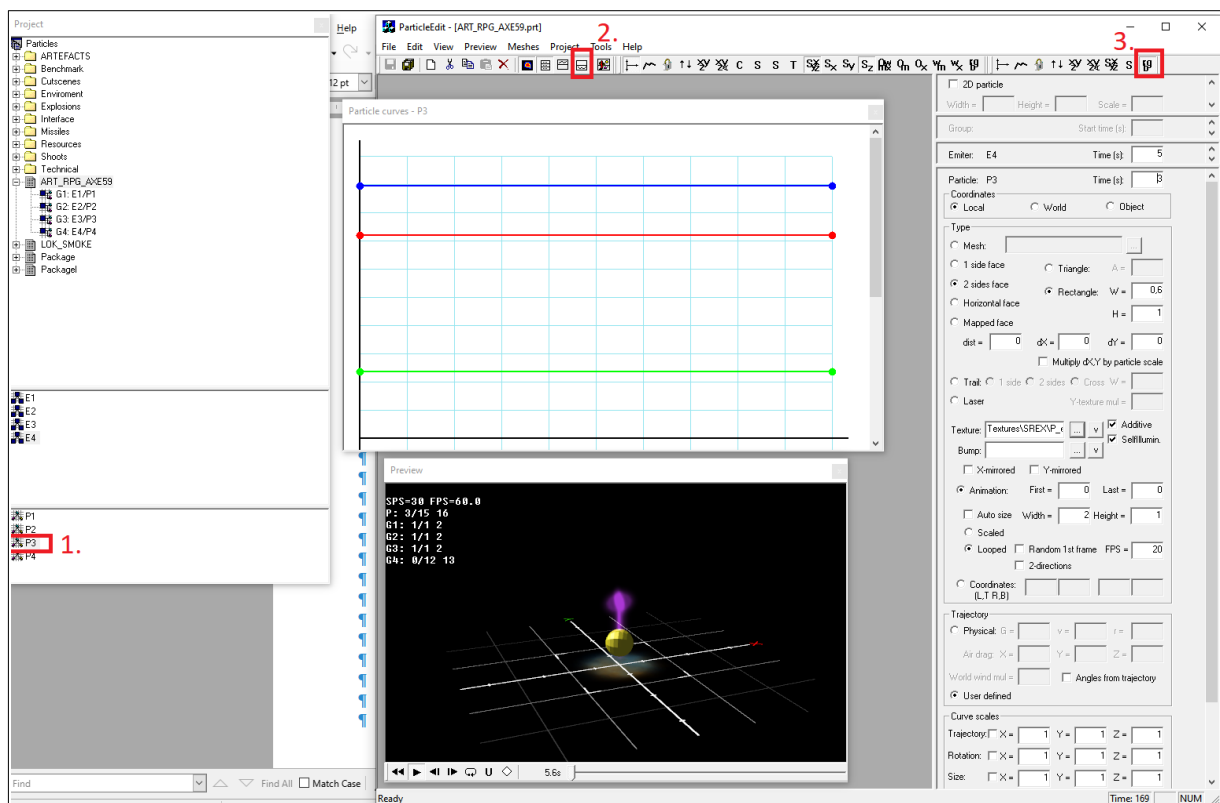


Figure 43: Now, for example, we identify what we want to change and make the changes. For example, I will change the color of this weapon border.

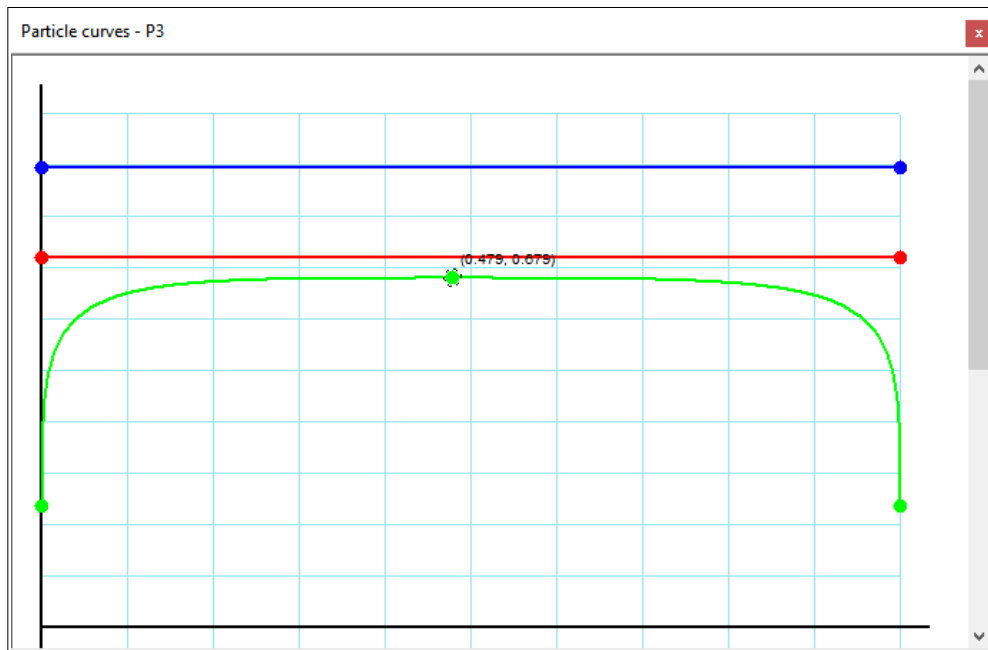


Figure 44: I manipulate points.

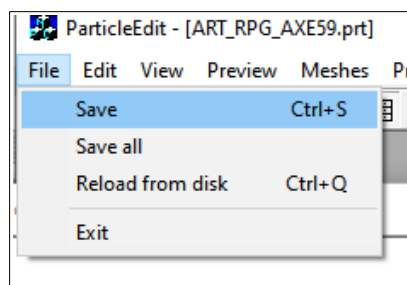


Figure 45: I am saving the file.

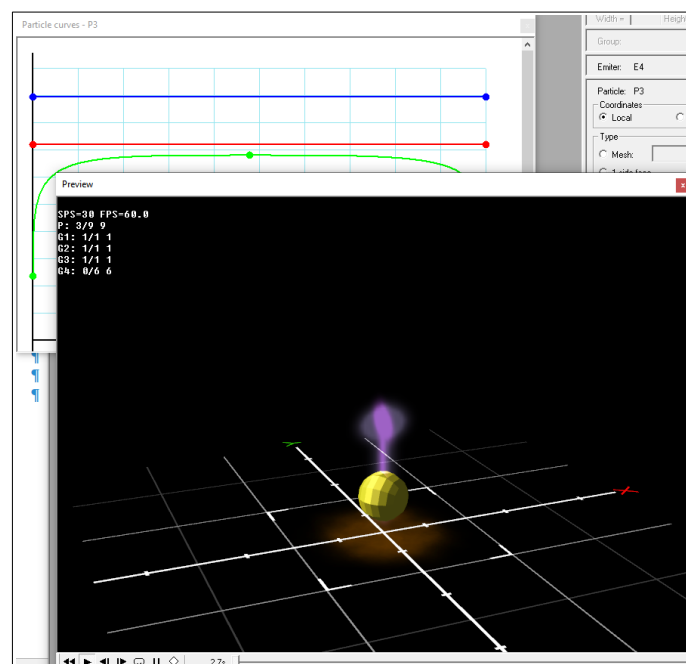


Figure 46: The border has changed to white.

If we want to put this file back into KnightShift, we need to restore the entries for the previous .tex textures and meshes. This can be done with ParticleEdit at this point or during the conversion back to .myaod format using a text editor such as Notepad++.

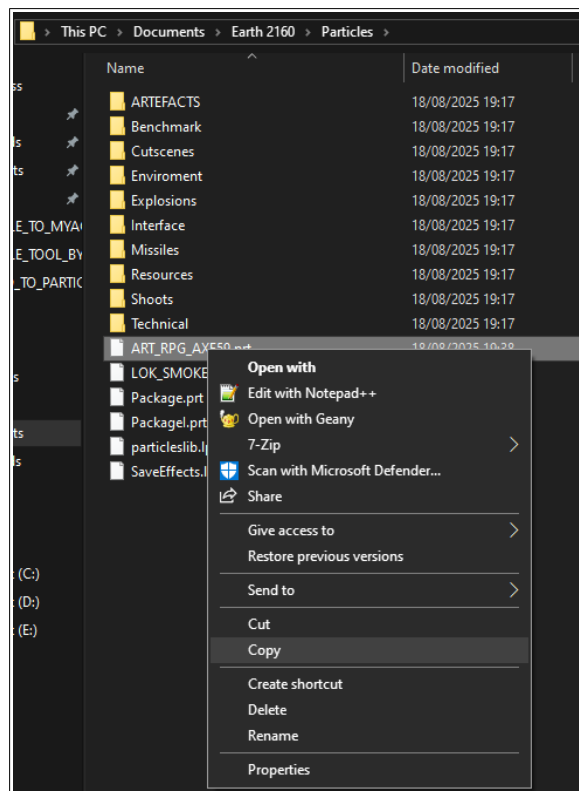


Figure 47: Copy the .prt file from the ParticleEdit particles directory.

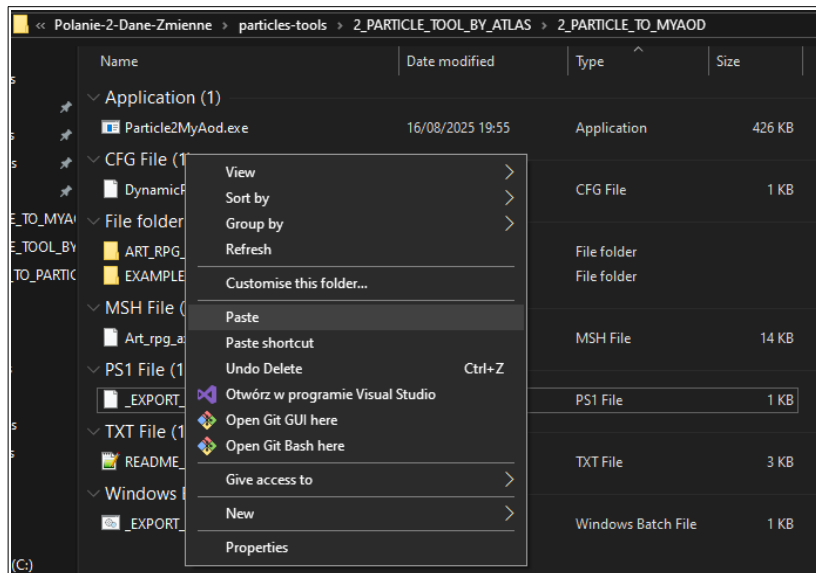


Figure 48: Paste into the Particle2MyAod directory.

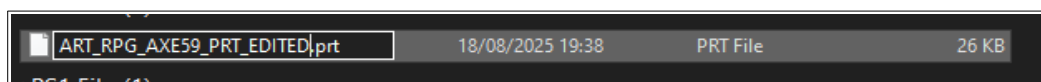


Figure 49: We are changing the file name as a precautionary measure so that the output directories do not get mixed up.

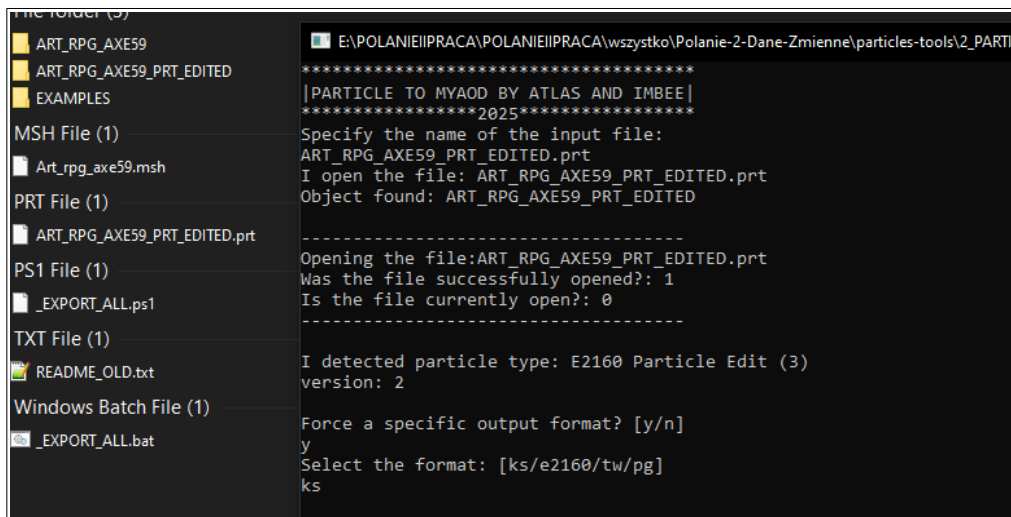


Figure 50: We export by enforcing the ks format.

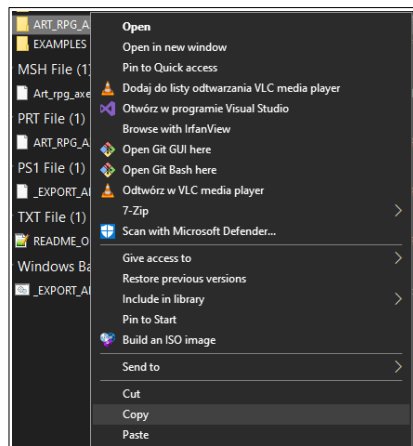


Figure 51: We copy the directory that was created.

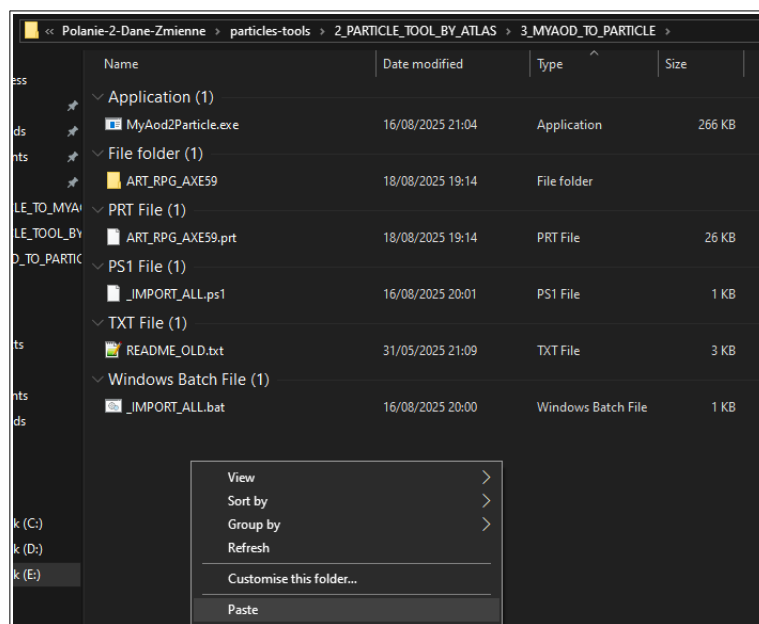


Figure 52: Paste it into the MyAod2Particle directory.

```
E:\POLANIEIIPRACA\POLANIEIIPRACA\wszystko\Polanie-2-D
*****
|MYAOD TO PARTICLE BY ATLAS AND IMBEE|
*****2025*****
Specify the name of the input directory:
ART_RPG_AXE59_PRT_EDITED_
```

Figure 53: We import.

MSH File (1)			
ART_RPG_AXE59_PRT_EDITED.msh	18/08/2025 19:53	MSH File	14 KB
PRT File (1)			

Figure 54: Done—you can upload the file to KnightShift.

e) I want to run a .prt file from 3D ParticleGen Visual FX (Steam) in ParticleGenie from Two Worlds:

Name	Date modified	type	Size
Application (1)			
Particle2MyAod.exe	16/08/2025 10:55	Application	176 KB
CFG File (1)			
DynamicParticle.cfg			
File folder (2)			
1970			
EXAMPLES			
PRT File (1)			
1970.prt			
PS1 File (1)			
_EXPORT_ALL.ps1			
TXT File (1)			
README_OLD.txt			
Windows Batch File (1)			
_EXPORT_ALL.bat			

Figure 55: I launch the **Particle2MyAod.exe** program. I enter the **file name**, then I enter **y** to force a different format, and I enter **tw**.

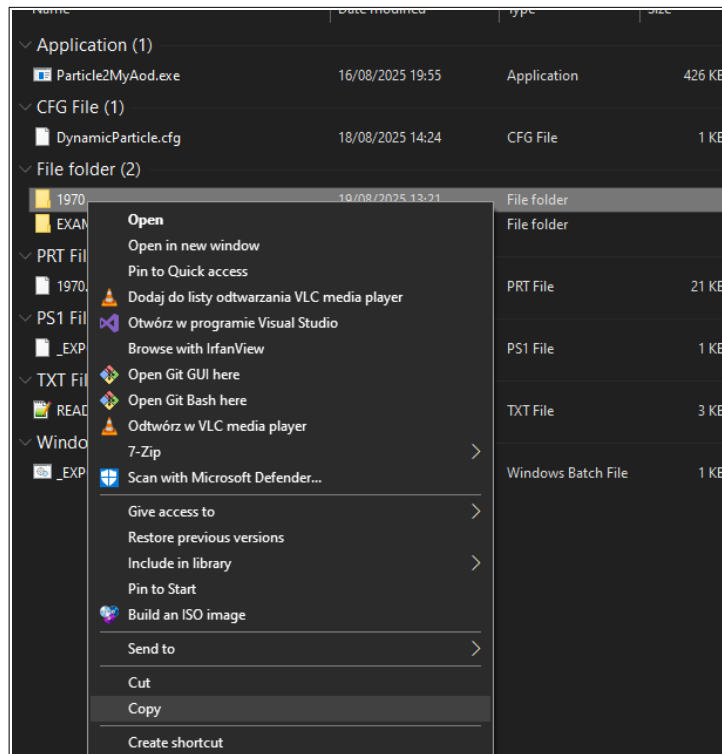


Figure 56: I copy the obtained directory.

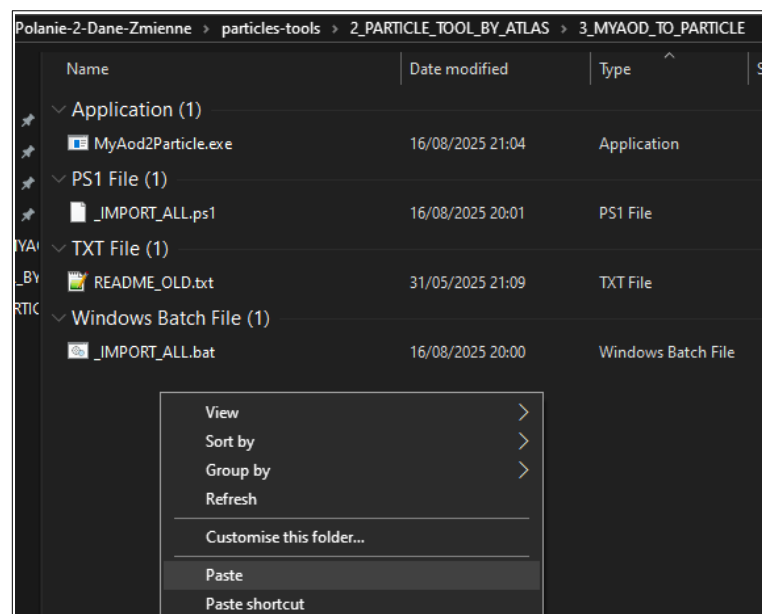


Figure 57: I paste it into the directory with **MyAod2Particle.exe**.

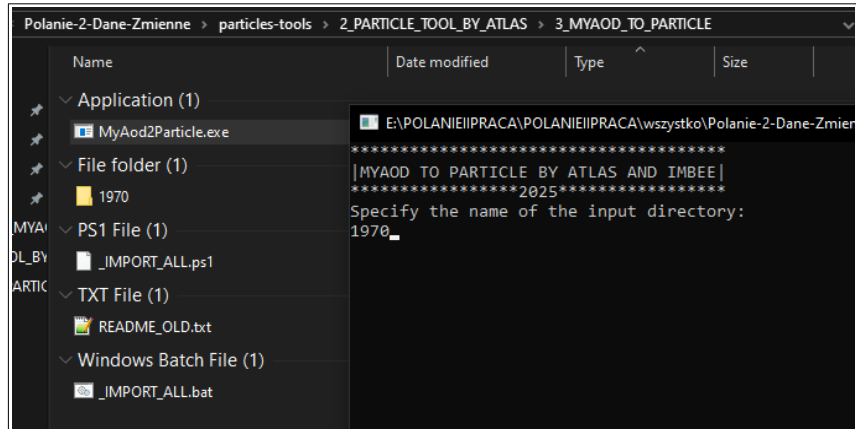


Figure 58: I turn on **MyAod2Particle** and enter the name of the pasted directory into it.

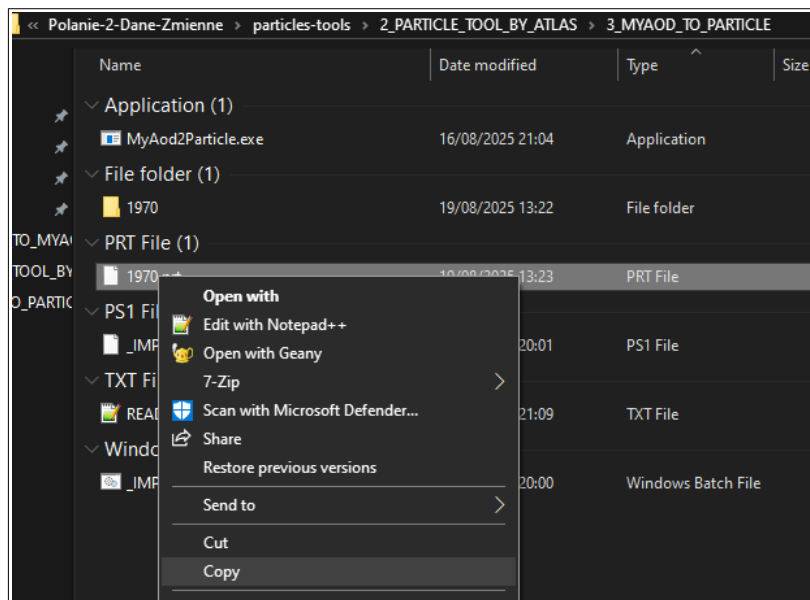


Figure 59: I copy the resulting **.prt** file.

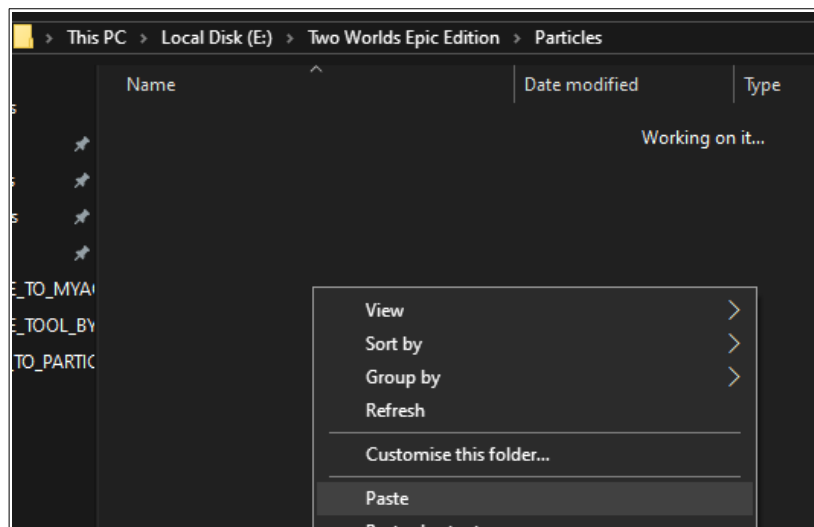


Figure 60: I paste it into the **Particles** directory in the **Two Worlds** game folder.

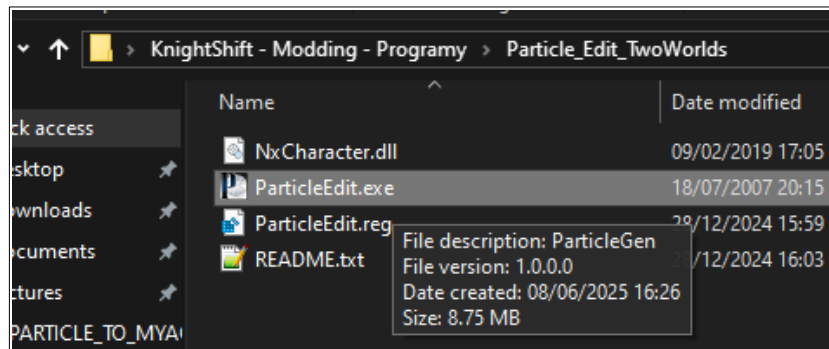


Figure 61: Uruchamiam **ParticleGena** z **Two Worlds SDK**.

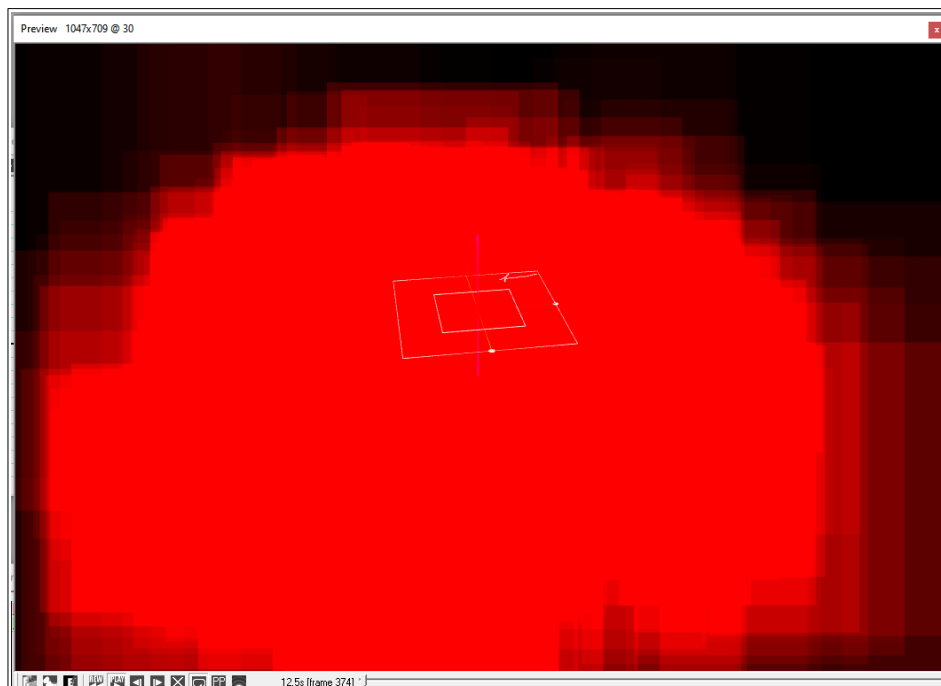


Figure 62: As we can see after launching particles in the program, the file works, but you still need to transfer the appropriate texture from 3D Particle Gena to Two Worlds.

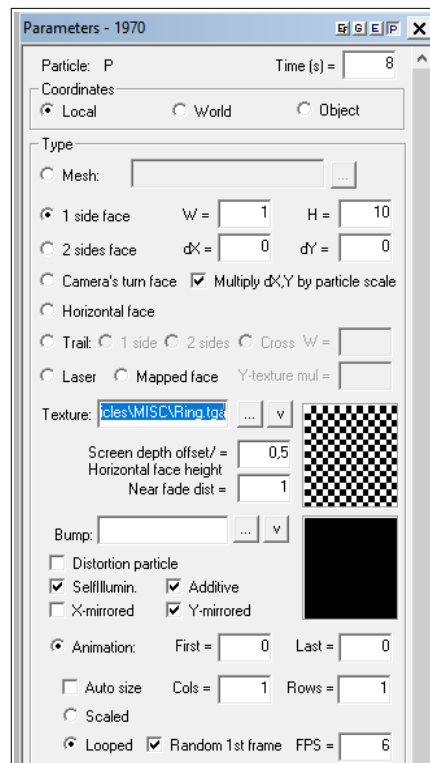
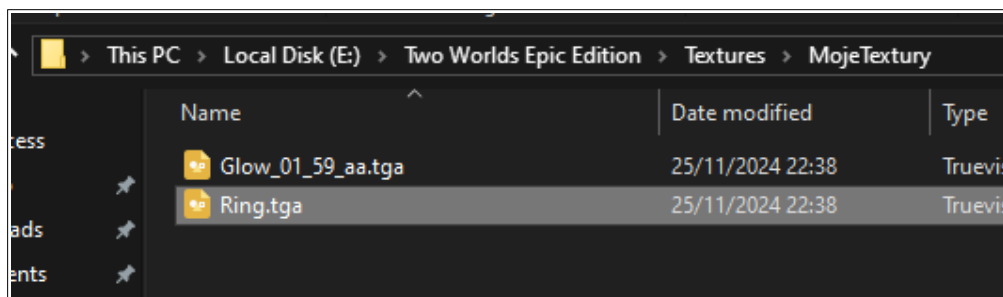


Figure 63: We can locate the activated texture in the object parameters.



*Figure 64: Place the required texture in the Textures directory in **Two Worlds** (I created an additional directory inside called MyTextures).*

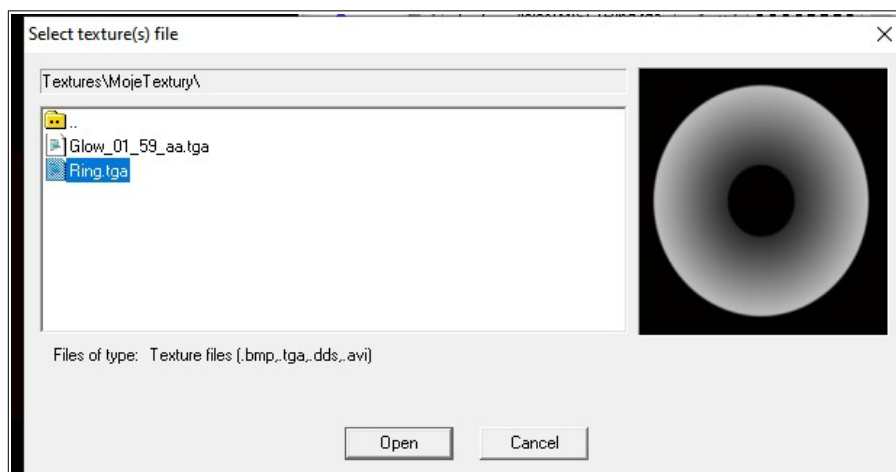


Figure 65: In the particle program, select the added texture.

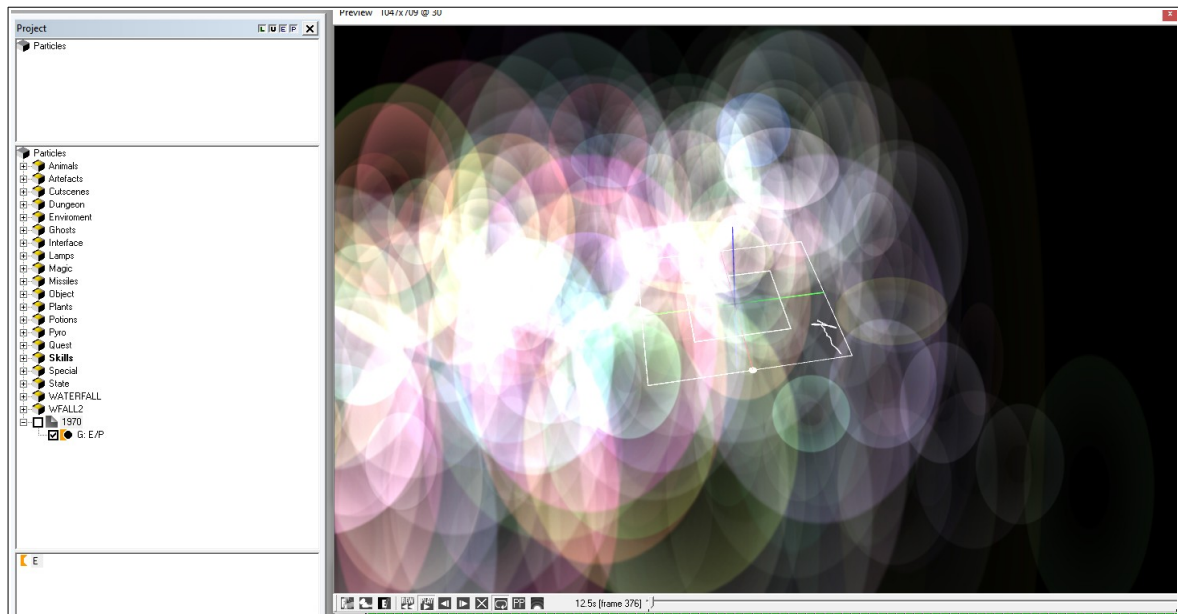


Figure 66: After saving the particles and restarting the program, we can see the final effect.

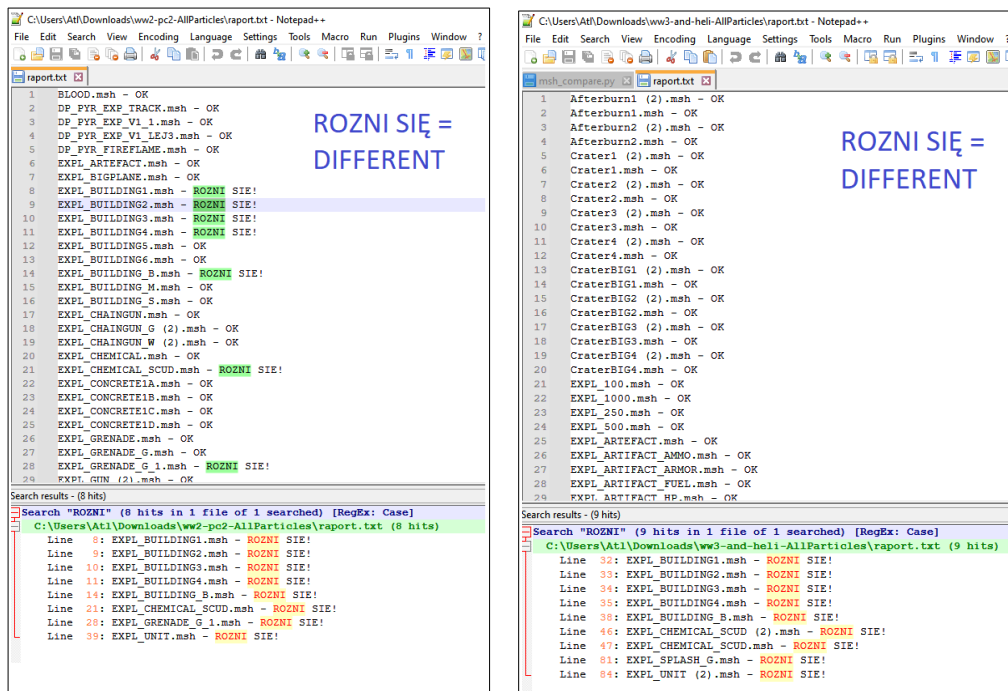
Additional information:

1. Test results:

The programs, specifically the export and re-import functions, were tested on:

- 1595 **.msh** files from KnightShift (i.e. all of them) – test result: **100%**,
- 3701 **.msh** files from KnightShift II Curse of Souls (i.e. all of them) – test result: **100%**,
- 584 **.msh** files from Earth 2150: Escape from the Blue Planet, Earth 2150: The Moon Project, Earth 2150: Lost Souls (i.e. all of them) – test result: **100%**,
- 1314 **.prt** files from Earth 2160 (i.e. all of them) – test result: **100%**,
- 1241 **.prt** files from Two Worlds (i.e. all of them) – test result: **100%**,
- 602 **.prt** files from 3D ParticleGen Visual FX (steam) (i.e. all of them) – test result: **100%**,
- 796 **.msh** files from Frontline Attack: War over Europe/
World War II: Panzer Claws II (i.e. all of them) – test result: **98.99%**,
- 263 **.msh** files from World War III: Black Gold, Heli Heroes (i.e. all of them) – test result **96.58%**,

Why aren't the last two results 100%?



This is because, by default, **Time** is stored in two float values in the **.aod** file, and the **Aod2Msh** compiler additionally performs certain mathematical calculations on these values and “casts” them, i.e., changes the type of the variable to **int64_t** in order to later store the value as **int32_t**. This results in a loss of information, which causes a difference in the field where the **Time** value is stored in **Dynamic particles**. The difference in values in the binary file is shown in the figure on the next page.

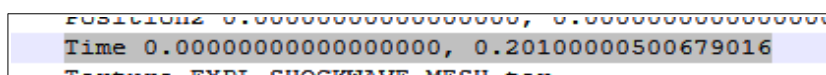


Figure 67: Time saved in the **.myaod** file.

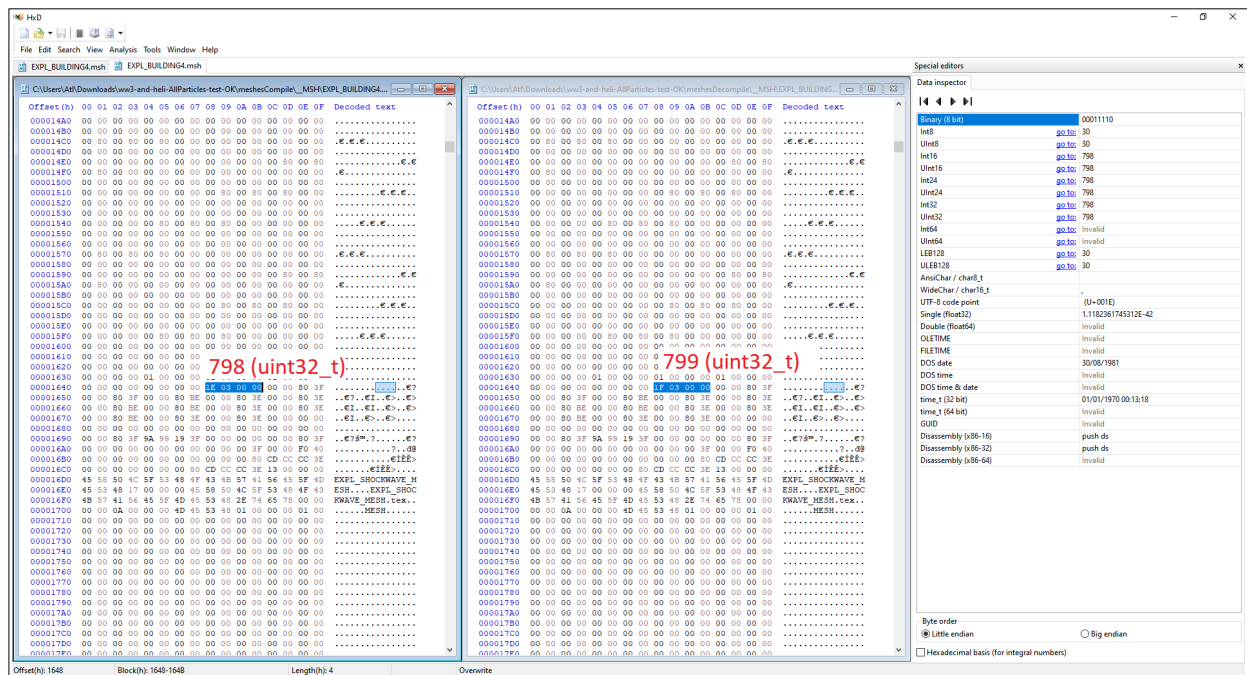


Figure 68: The visible difference in values is caused by strange mathematical calculations and changes in variable types.

This does not cause any major problems, but it is worth noting.
The images below show examples of such code causing these differences.

```
if ( _strnicmp(dword_456930, aTime, 4u) )
    break;
if ( sscanf(dword_456930 + 5, "%f,%f", &Time_1, &Time_2) != 2 )
    goto LABEL_287;
v33 = 1.0 - Time_2;
*( _DWORD * )(this + 4068) = ( _int64 )(Time_1 * 1000.0);
*( _DWORD * )(this + 4072) = ( _int64 )(v33 * 1000.0);
}
```

Figure 69: Screenshot showing the **Time** reading code after decompiling Aod2Msh in IDA.

```
sscanf_s(m_help_str.c_str(), "%f, %f", &help_time_1, &help_time_2);

DEBUG_PRINT("Time = ");
DEBUG_PRINT(to_string(help_time_1).c_str());
DEBUG_PRINT(", ");
DEBUG_PRINT(to_string(help_time_2).c_str());
DEBUG_PRINT("\n");

arg_dynamic_particle_data.time[0] = static_cast<int64_t>( help_time_1 * 1000.0 );
arg_dynamic_particle_data.time[1] = static_cast<int64_t>( (1.0 - help_time_2) * 1000.0 );
```

Figure 70: My version in the MyAod2Particle code.

```

int64_t help_time = static_cast<int64_t>(BinReader::GetBinVal<int32_t>(buffer, offset));

dynamic_particle_data.time[0] = static_cast<float>(static_cast<double>(help_time) / 1000.0);

help_time = static_cast<int64_t>(BinReader::GetBinVal<int32_t>(buffer, offset));

dynamic_particle_data.time[1] = static_cast<float>(1.0 - (static_cast<double>(help_time) / 1000.0));

```

Figure 71: Reading and calculating **Time** based on values in a binary file in the **Particle2MyAod.exe** program, where **dynamic_particle_data.time[]** is an array of type **float**.

2. Entries about textures in Dynamic particles from Earth 2150:

In the texture entries in the **.myaod** files with Dynamic particles from the game **Earth 2150**, you can see the name of the **Textures** directory added to the texture path.

```

Alpha 1.0000000000000000, 1.0000000000000000, 0
Scale 0.0000000000000000, 0.0000000000000000
Position 0.0000000000000000, 0.0000000000000000, 0.0000000000000000
Position2 0.0000000000000000, 0.0000000000000000, 0.0000000000000000
Texture Textures\KILWATER1.tex
Mesh

```

Figure 72: Additional name of the **Textures** directory.

In Dynamic particles from other games, you will notice that this entry is missing. This is because **Aod2Msh** from **Earth 2150** adds the name of the **Textures** directory before the texture name. Therefore, if you want to use a **.myaod** file with Dynamic particles from **Earth 2150** in other games, you need to correct this path yourself. The same applies to using **Aod2Msh** from **Earth 2150** on a **.myaod** file from **Earth 2150**.

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000400	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000410	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000420	15	00	00	00	54	65	78	74	75	72	65	73	5C	45	58	50	...Textures\EXP
00000430	5F	46	5F	55	32	2E	74	65	78	00	00	00	00	00	00	00	F U2.tex....

Figure 73: Proof that **Aod2Msh** from **Earth 2150** adds the text **"Textures\"** before the texture name.

3. Why should you change the gamerate in the ParticleEdit program from Earth 2160?:

By default, the **ParticleEdit** program from **Earth 2160** has a **gamerate** value set to 30. After uploading a particle with a different gamerate and saving the file, you may encounter a situation where animations simply behave strangely, e.g., a rotating object will start falling down in the last few seconds of the animation, which did not happen before. Due to problems with gamerate, it is recommended to change the gamerate before uploading any file and saving all effects. I have described how to change the gamerate correctly in one of the **scenarios** for using the **Particle Tool**.

4. No support for Boxes and Slots:

In some **.aod** files, you could see entries such as **Slots** or **Boxes** (see figure below). My programs **Particle2MyAod** and **MyAod2Particle** do not support these instructions. Why don't my programs support them? It's because behind all these **Boxes** and **Slots** there are some strange calculations that I don't fully understand yet. Any change to these values is done by changing the files with the **_extra_data.cpp** extension in the particles directory. If you want to make any changes, unfortunately you will have to discover/interpret this initial data block from the **.msh** file yourself for now.

```
Dynamic
{
}
Object CLUTTER1
{
    Type Track
    Texture CLUTTER01.tex
    Additive 0
    Position 0,0,0
    Light 0,0,0,0
    Frames 0,0,1,2,2
    Width 0.999
    Layer 2
}
Boxes 1
{
    Box 0.500, -0.500, 0.000, 0, 0, 0, 0
}
```

Figure 74: Sample **.aod** file with Boxes.

5. Why can curves in KnightShift particles only be partially manipulated in ParticleEdit in Earth 2160?

Unfortunately, due to a lack of 100% understanding of the relationship between **tessellations** (**teselate**) and **curves** (**curve**), in some ParticleEmitters from **KnightShift** running in **ParticleEdit** (**E2160**), you can see that the points of the curves extend beyond the graph, making it impossible to manipulate them. In the case of **KnightShift** and **.msh** files with ParticleEmitters, only tessellations are stored, because curves are removed during conversion from **.aod** to **.msh**. The **Particle2MyAod.exe** program performs interpolation on tessellations, creating curve points and curves, which it then writes to **.myaod**, but this is not 100% perfect, as I mentioned above, and some of the points on the graph extend beyond the graph, making it impossible to manipulate them. This problem can be corrected, but it is necessary to better understand the relationships between **curve points** and **tessellations**. If anyone manages to find this relationship, please contact me:

The Greatest Atlas: <https://steamcommunity.com/profiles/76561198107437731/>