

Particle Tool

by Atlas and ImBee

Dokumentacja + instrukcja

Trochę teorii :

Particlesy są zapisane w formacie **.msh** oraz **.prt**. Format **.msh** występuje w następujących grach:

- 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,

Natomiast **.prt** występuje w:

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

Tytuły po Two Worlds nie zostały tutaj wpisane ponieważ nie brałem ich już pod uwagę.

Każda z gier/tytułów ma format **.msh** albo **.prt** jednak nie oznacza to, że format **.prt** z **Earth 2160** jest taki sam jak format z **Two Worlds** mimo, że obydwa posiadają końcówkę **.prt**.

Poniżej wymieniam grupy tytułów, które posiadają taki sam format. Każda grupa zaczyna się od nowej strzałki:

Format **.msh**:

- 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;

Format **.prt**:

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

Ogólną strukturę plików particlesów można podzielić na dwa rodzaje:

1. Dynamic:

- może wystąpić jedynie w **.msh**,
- występuje w grach:
 - * 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;
- oparty jest na strukturze pojedynczego obiektu, który może mieć zagnieżdżone dzieci, a te dzieci tak samo mogą mieć swoje dzieci (**struktura drzewa**).

2. ParticlesEmitter

→ może wystąpić w **.msh** oraz w **.prt**,

→ występuje w grach:

* Polanie II/KnightShift/Once Upon a Knight, Polanie III/KnightShift II Curse of Souls
(ten rodzaj pojawił się w grze KnightShift po raz pierwszy);

* Earth 2160;

* 3D ParticleGen Visual FX;

* Two Worlds;

→ oparty na strukturze złożonej z obiektów takich jak: Effect, Emitter, Particle oraz PairParticleEmitter (inaczej Group),

Na tych wszystkich formatach można wykonywać operacje za pomocą odpowiednich programów. Zestawienie przedstawiam poniżej:

I. Dynamic:

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

- + Można przekonwertować do tekstowego pliku **.myaod** i edytować z poziomu edytora tekstu np. Notepad++,
- + Można podejrzeć w grze,
- + Można dokonać konwersji na Dynamic particle z innej gry.

- Brak możliwości edytowania w zewnętrznym programie z podglądem particlesa ponieważ taki program jeszcze nie istnieje.

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

- + Można przekonwertować do tekstowego pliku **.myaod** i edytować z poziomu edytora tekstu np. Notepad++,
- + Można podejrzeć w grze,
- + Można dokonać konwersji na Dynamic particle z innej gry.

- Brak możliwości edytowania w zewnętrznym programie z podglądem particlesa ponieważ taki program jeszcze nie istnieje.

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

- + Można przekonwertować do tekstowego pliku **.myaod** i edytować z poziomu edytora tekstu np. Notepad++,
- + Można podejrzeć w grze,
- + Można dokonać konwersji na Dynamic particle z innej gry.

- Brak możliwości edytowania w zewnętrznym programie z podglądem particlesa ponieważ taki program jeszcze nie istnieje.

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

- + Można przekonwertować do tekstowego pliku **.myaod** i edytować z poziomu edytora tekstu np. Notepad++,
- + Można podejrzeć w grze,
- + Można dokonać konwersji na Dynamic particle z innej gry.

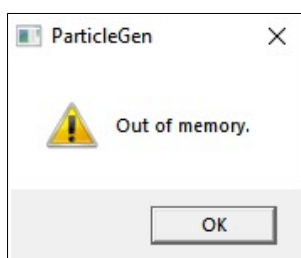
- Brak możliwości edytowania w zewnętrznym programie z podglądem particlesa ponieważ taki program jeszcze nie istnieje.

II. ParticlesEmitter:

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

- + Można przekonwertować do tekstowego pliku **.myaod** i edytować z poziomu edytora tekstu np. Notepad++,
- + Można podejrzeć w grze,
- + Można dokonać konwersji na ParticleEmitter z innej gry np. **Earth 2160**.
- + Możliwość przekonwertowania do formatu **.prt** z **Earth 2160** i edytowanie go w zewnętrznym programie z podglądem particlesa. Program ten nazywa się **ParticleEdit** z pakietu **Earth 2160 Digital Deluxe Content (dostępne do kupienia na steam)(wymaga posiadania Earth 2160)**, a dokładnie z katalogu **Earth2160_SDK**.

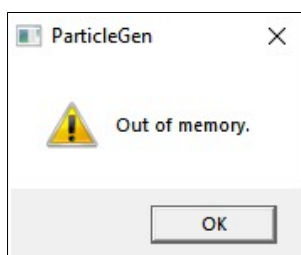
- Można dokonać konwersji na format **.prt** z **Two Worlds** oraz z **3D ParticleGen Visual FX (steam)** ale podczas testów przekonwertowanego pliku do **ParticleGen** dla **Two Worlds** oraz **Particle Gen (steam)** wyskakuje błąd “**Out of memory**”.



2. ParticlesEmitter (prt) z Earth 2160;

- + Można przekonwertować do tekstowego pliku **.myaod** i edytować z poziomu edytora tekstu np. Notepad++,
- + Można podejrzeć w grze,
- + Można dokonać konwersji na ParticleEmitter z innej gry np. **KnightShift**,
- + Możliwość edytowania go w zewnętrznym programie z podglądem particlesa. Program ten nazywa się **ParticleEdit** z pakietu **Earth 2160 Digital Deluxe Content (dostępne do kupienia na steam)(wymaga posiadania Earth 2160)**, a dokładnie z katalogu **Earth2160_SDK**.
- + Po przekonwertowaniu na format **KnightShift** jest on w kompatybilny z **KnightShift**.

- Można dokonać konwersji na format **.prt** z **Two Worlds** oraz z **3D ParticleGen Visual FX (steam)** ale podczas testów przekonwertowanego pliku do **ParticleGen** dla **Two Worlds** oraz **Particle Gen (steam)** wyskakuje błąd “**Out of memory**”.



3. ParticleEmitter (prt) z Two Worlds;

- + Można przekonwertować do tekstowego pliku **.myaod** i edytować z poziomu edytora tekstu np. Notepad++,
- + Można podejrzeć w grze,
- + Można dokonać konwersji na ParticleEmitter z **3D ParticleGen Visual FX (steam)**.
- + Po przekonwertowaniu na format **3D ParticleGen Visual FX (steam)** jest on w kompatybilny z **3D ParticleGen Visual FX (steam)**.
- + Możliwość edytowania go w zewnętrznym programie z podglądem particlesa. Program ten nazywa się **ParticleEdit** z pakietu **Two Worlds SDK**

<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

(wymaga posiadania zainstalowanego Two Worlds)

- Można dokonać konwersji na format **.prt** z **Earth 2160**. Czasem to działa albo nie działa. Raz się po prostu wyświetli sam prostokąt albo wyświetli się komunikat **“Out of memory”**, albo po prostu program się crashuje.
- Można dokonać konwersji do formatu **KnightShift** ale przy próbie zobaczenia tego particlesa w grze - gra crashuje.

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

- + Można przekonwertować do tekstowego pliku **.myaod** i edytować z poziomu edytora tekstu np. Notepad++,
- + Można podejrzeć w programie,
- + Można dokonać konwersji na ParticleEmitter z **TwoWorlds**,
- + Po przekonwertowaniu na format **TwoWorlds** jest on w kompatybilny z **TwoWorlds**,
- + Możliwość edytowania go w programie z podglądem particlesa. Program ten nazywa się **3D ParticleGen Visual FX (dostępny do kupienia na steam)**.

- Można dokonać konwersji na format **.prt** z **Earth 2160**. Czasem to działa albo nie działa. Raz się po prostu wyświetli sam prostokąt albo wyświetli się komunikat **“Out of memory”**, albo po prostu program się crashuje.
- Można dokonać konwersji do formatu **KnightShift** ale przy próbie zobaczenia tego particlesa w grze - gra crashuje.

Zatem na podstawie powyższych informacji o formatach możemy sporządzić graf dotyczący Dynamic particlesów, oraz graf dotyczący ParticleEmitterów.

I. Dynamic:

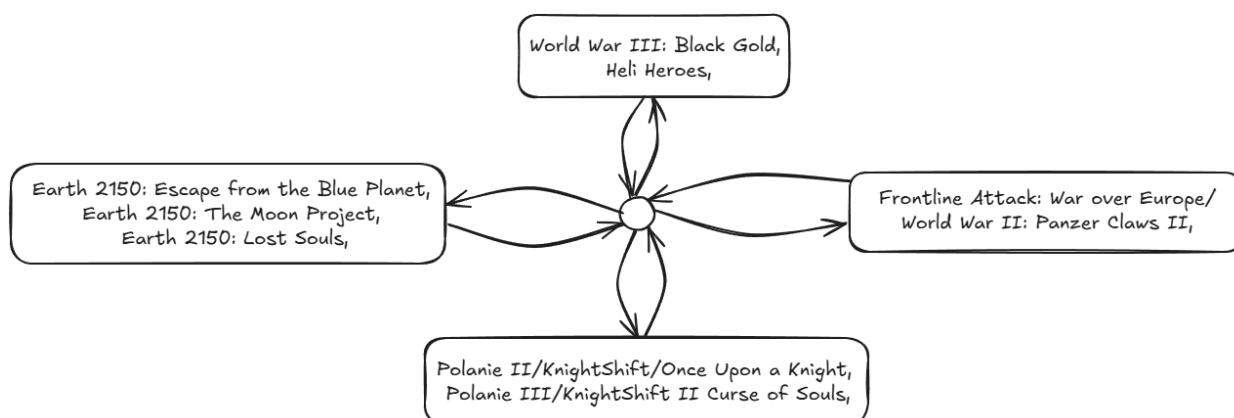


Figure 1: Konwersja pomiędzy formatami Dynamic particlesa jest możliwa w każdym przypadku.

II. ParticleEmitter:

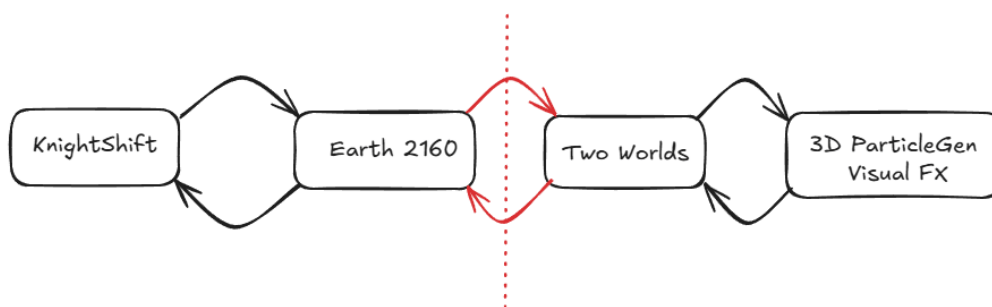


Figure 2: Między Earth 2160, a Two Worlds istnieje pewna bariera formatów, która uniemożliwia uruchomienie np. formatu KnightShift w edytorze particlesów do Two Worlds.

Czym jest Particle Tool By Atlas and ImBee i do czego służy?

Particle Tool By Atlas and ImBee to pakiet dwóch programów służących do konwersji plików particlesów z gier **Reality Pump** w celu umożliwienia ich prostszej edycji i uruchamiania tego samego particlesa w różnych grach tego studia. Pakiet tych programów jest “mostem” pomiędzy różnymi formatami particlesów. Programy te obsługują następujące gry/tytuły:

- 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,

Pakiet jest złożony z dwóch programów:

- Particle2MyAod.exe
- MyAod2Particle.exe

Particle2MyAod:

Particle2MyAod.exe to program, który służy do eksportu danych z particlesów w formacie **.msh** i **.prt** z tytułów przedstawionych wyżej do formatu **.myaod**, który jest bardzo podobny do **.aod**. Wystarczy wprowadzić nazwę pliku particlesa wraz z formatem.

Plik z ParticleEmitemerem:

W przypadku wykrycia ParticleEmitera program zapyta o wymuszenie końcowego formatu. Możemy wybrać **y/n**. Jeśli wybierzemy **n** to program wypakuje plik particlesa do **.myaod** nie wymuszając jakiegось konkretnego formatu czyli program po prostu wypakuje particlesa. W przypadku gdy wprowadzimy **y** to będziemy mogli wymusić format do którego później ma zostać skompilowany **.myaod** (np. jeśli chcę uruchomić particlesa z **KnightShift** w **Earth 2160** to wpisuje **y**, a potem wymuszam format **e2160**. Program **MyAod2Particle** już sam będzie wtedy wiedział jak skompilować **.myaod** do formatu **Earth 2160**). Następnie program wypakuje plik particlesa.

W folderze wyeksportowanego particlesa znajduje się plik **.myaod** i plik **_extra_data.cpp**. W pliku **extra_data** znajduje się wypakowany cały początek pliku particlesa w postaci podanych wartości zmiennych. Można to modyfikować ale nie trzeba. Najważniejszy jest plik **.myaod**.

Program działa również w trybie **argc&argv**, więc można go wywołać z cmd albo z powershella i wpisać argument obok nazwy programu co spowoduje samo wyeksportowanie particlesa bez wymuszenia formatu:

Particle2MyAod.exe <nazwa_pliku_input>

Można również wymusić odpowiedni format następującym sposobem:

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

lub

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

Plik z Dynamic particlesem:

Obsługa Dynamic particlesów niestety odbywa się przez edycję pliku konfiguracyjnego **DynamicParticle.cfg**. W zależności od tytułu, z którego pochodzą particlesy należy zmodyfikować ten plik konfiguracyjny – najlepiej natychmiastowo. Standardowo ustawione są następujące opcje, które znaczą:

dynamic_particle_input_format = ks;

(Wejściowy Dynamic particle będzie pochodzić z KnightShift.)

force_specific_export_format = false;

(Czy wymusić specyficzny format?)

forced_export_format = ks;

(Wyjściowy końcowy format Dynamic particlesa będzie formatem z gry KnightShift)

Aby obsłużyć specyficzny format z innej gry gdzie jest Dynamic particles wystarczy wpisać jeden z tych formatów: **e2150, ww3_or_hh, ww2_or_fa_or_pc2, ks**

w to miejsce po znaku =

dynamic_particle_input_format = <tutaj wpisujemy format>;

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

Końcowy format do którego konwertujemy należy wpisać w ***forced_export_format*** po znaku =
forced_export_format = <tutaj wpisujemy format>;

Kompilator już sam powinien wiedzieć jak skompilować katalog z wyeksportowanym particlesem.

Do kompilacji formatu **.myaod** stworzyłem swój specjalny kompilator **MyAod2Particle.exe**.

Aby automatycznie wypakować wszystkie pliki bez wymuszania formatu, można użyć **_EXPORT_ALL.ps1** lub **_EXPORT_ALL.bat**.

MyAod2Particle:

MyAod2Particle.exe to program, który służy do kompilowania plików **.myaod** particlesów do ustalonego wcześniej formatu z danej gry **Reality Pump**.

Program się włącza przez dwuklik i wpisuje się nazwę katalogu wejściowego. Jeżeli plik **extra_data** w katalogu, nie istnieje to program stworzy klasyczny nagłówek particlesa generując odpowiednie informacje, jednak jeżeli istnieje plik **extra_data** to przekompiluje ten plik na nagłówek particlesa.

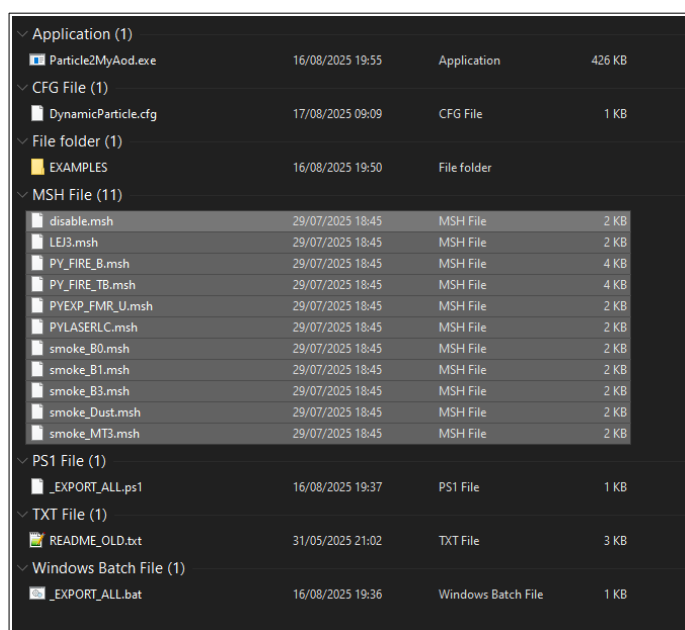
Program działa również w trybie **argc&argv**, więc można go wywołać z **cmd** albo z **powershella** i wpisać argument obok nazwy programu czyli

MyAod2Particle.exe <nazwa katalogu wejściowego>

Aby automatycznie skompilować/zaimportować wszystkie foldery, można użyć **_IMPORT_ALL.ps1** lub **_IMPORT_ALL.bat**.

Typowe i przykładowe scenariusze używania ParticleToola:

a) Chcę wyeksportować Dynamic particlesy z **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
PVLASERLC.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: Przykładowo mam jakąś pulę particlesów z **E2150**.

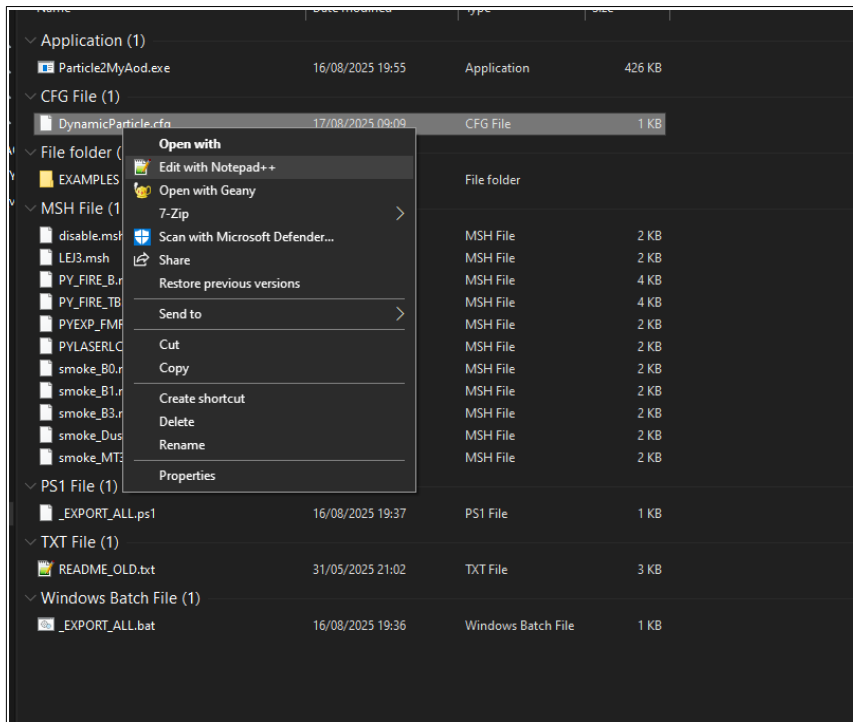


Figure 4: Edytuję *DynamicParticle.cfg*.

```
// 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: Zauważam opcje.

```
// 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: Zmieniam format na odpowiedni wejściowy format.

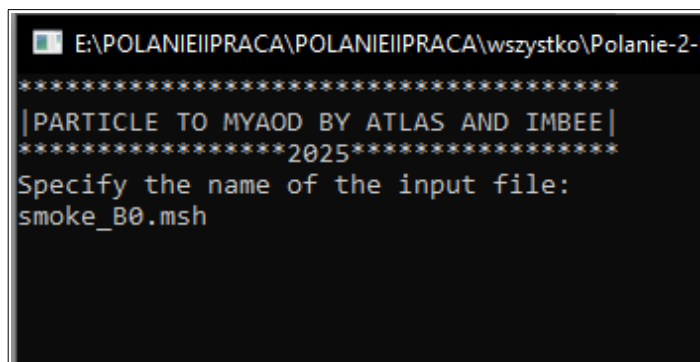


Figure 7: Włączam program i wpisuję nazwę pliku.
Następnie klikam enter.

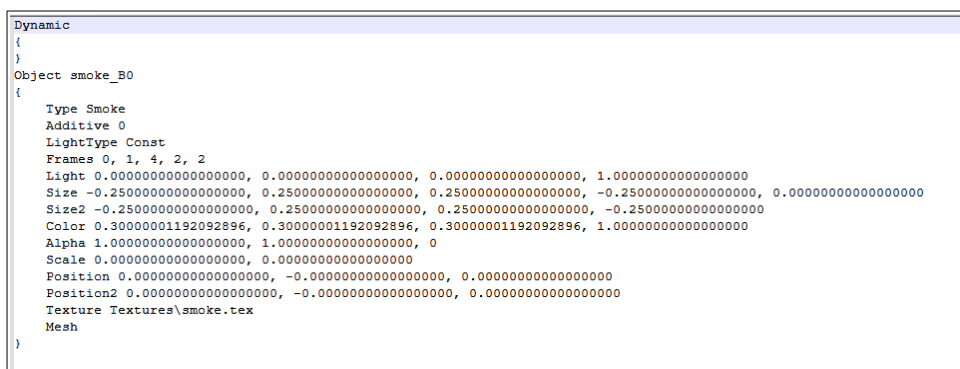


Figure 8: Gotowe - mogę edytować dane Dynamic particlesa.

b) Chcę przekonwertować Dynamic particlesy z Earth 2150 do World War III Black Gold:

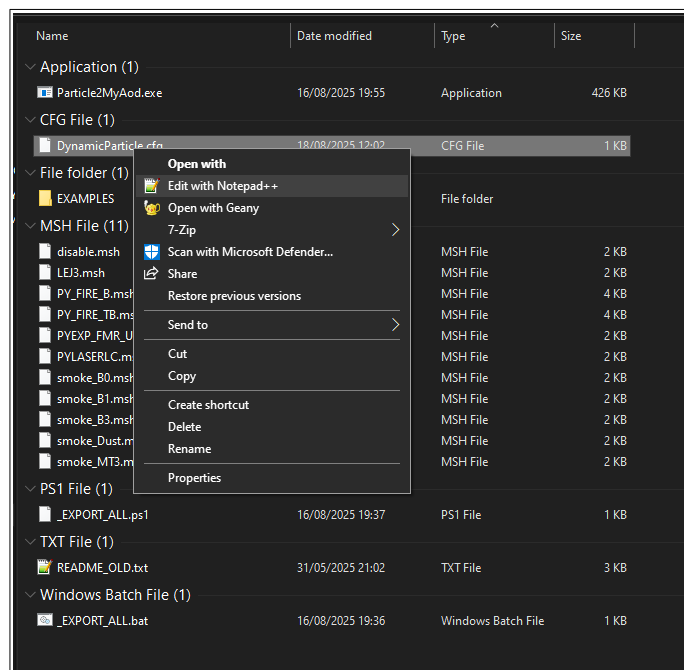


Figure 9: Edytuję DynamicParticle.cfg.

```
// 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: Zauważam opcje.

```
// 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: Ustawiam format wejściowy na e2150, odblokowuję force_specific_export_format na true i ustawiam forced_export_format na 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: Włączam program i wprowadzam nazwę particlesa i klikam enter.

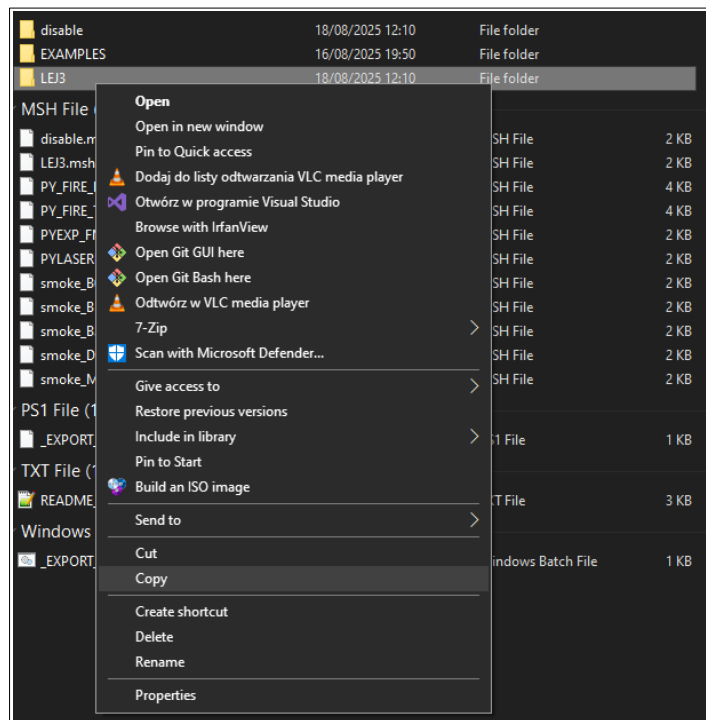


Figure 13: Kopiuję otrzymany katalog z wyeksportowanym particlesem do katalogu z MyAod2Particle.

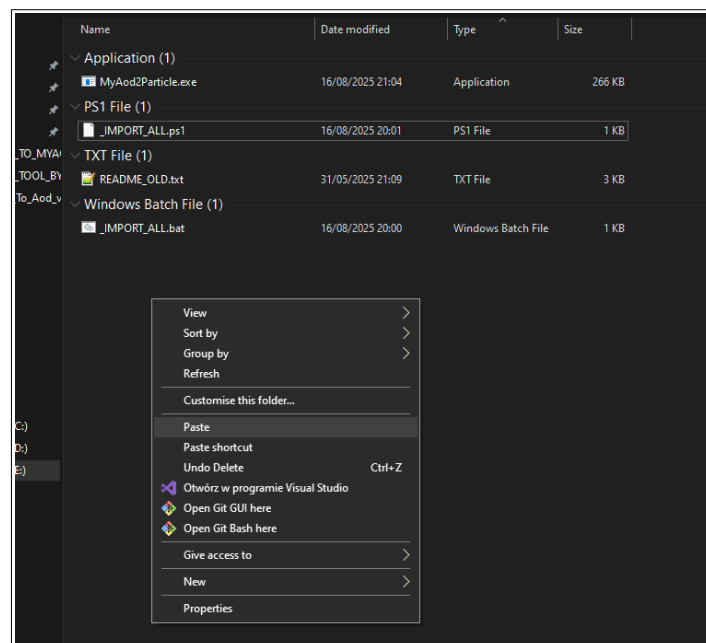


Figure 14: Wklejam.

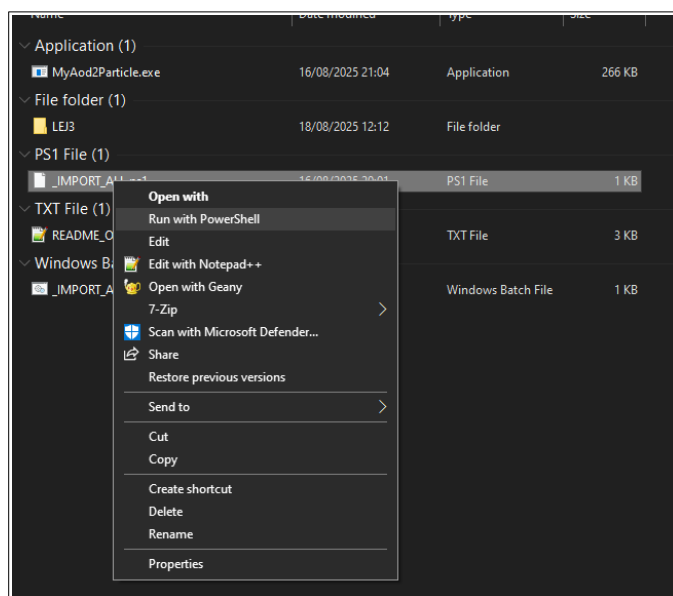


Figure 15: Możemy włączyć program przez dwuklik i wpisać nazwę katalogu do skompilowania, albo możemy włączyć skrypt `_IMPORT_ALL.ps1` przez powershella w celu skompilowania wszystkich katalogów z danymi particlesów.

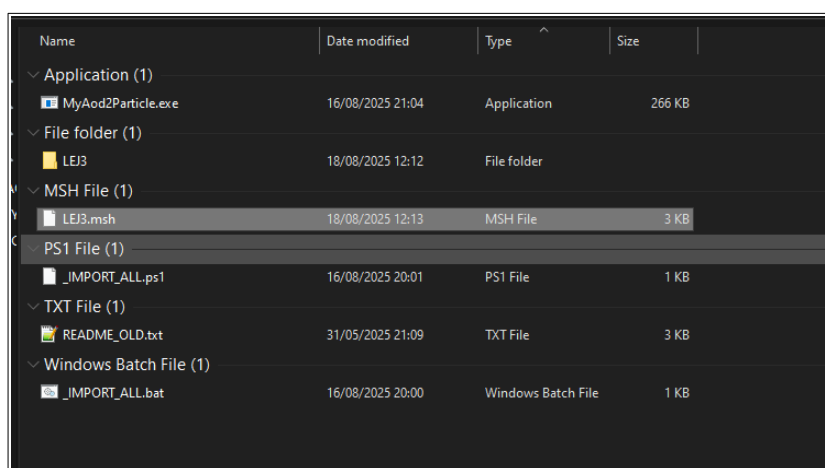


Figure 16: Gotowe - Particles jest gotowy do wrzucenia do gry.

c) Chcę przekonwertować ParticleEmitter z KnightShift do Earth 2160:

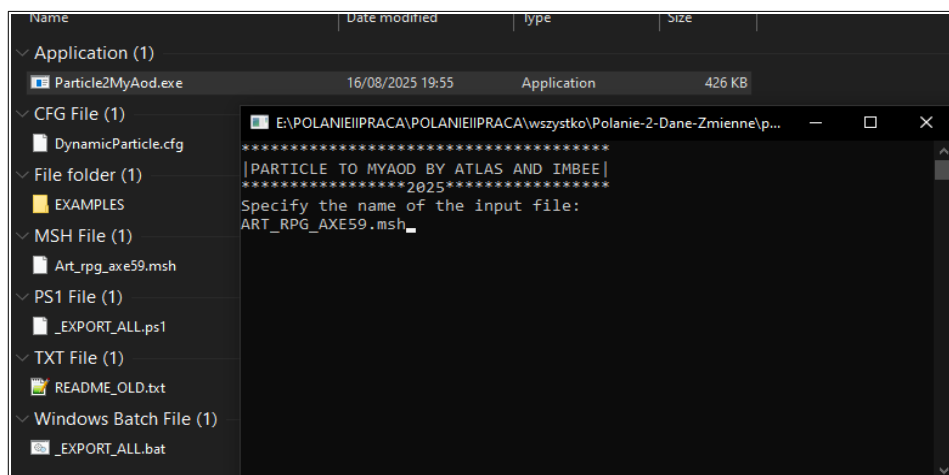


Figure 17: Włączam dwuklikiem Particle2MyAod.exe i wpisuję nazwę pliku .msh.

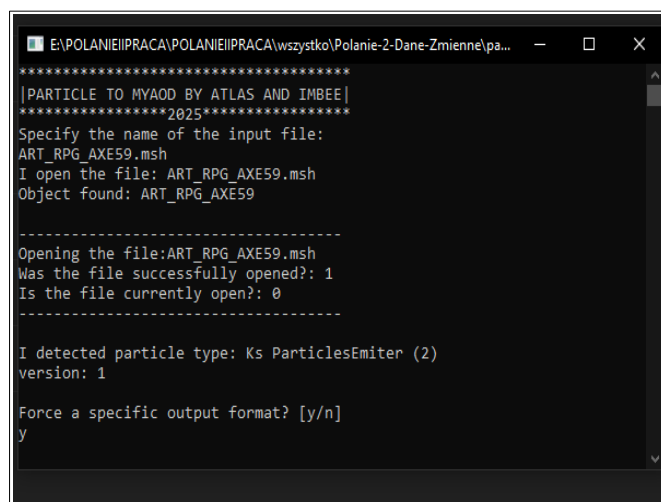


Figure 18: Wprowadzam y.

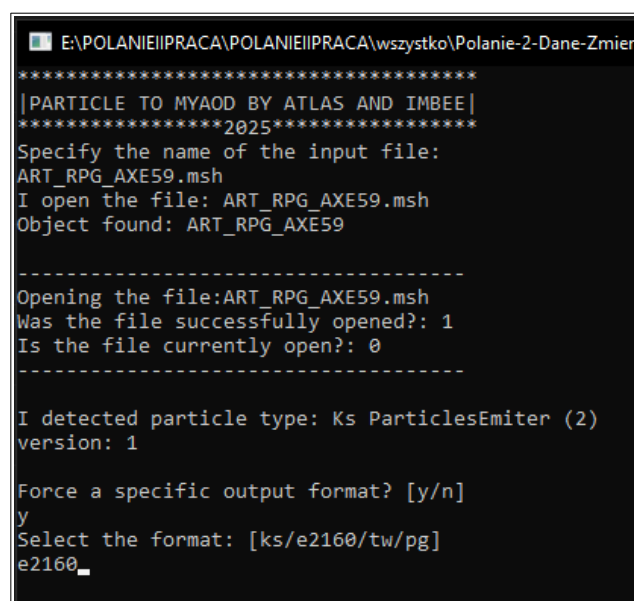


Figure 19: Wymuszam format e2160.

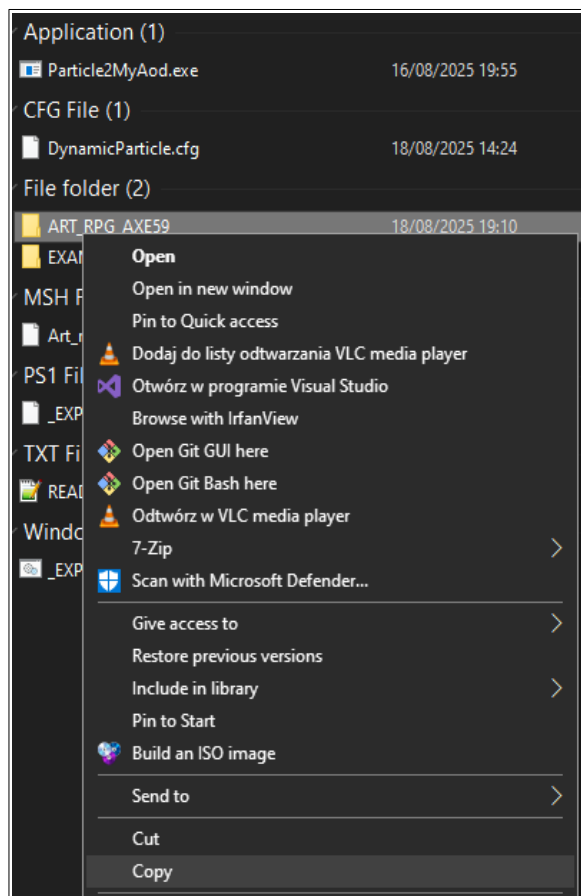


Figure 20: Kopiuję katalog z danymi particlesa do katalogu z MyAod2Particle.

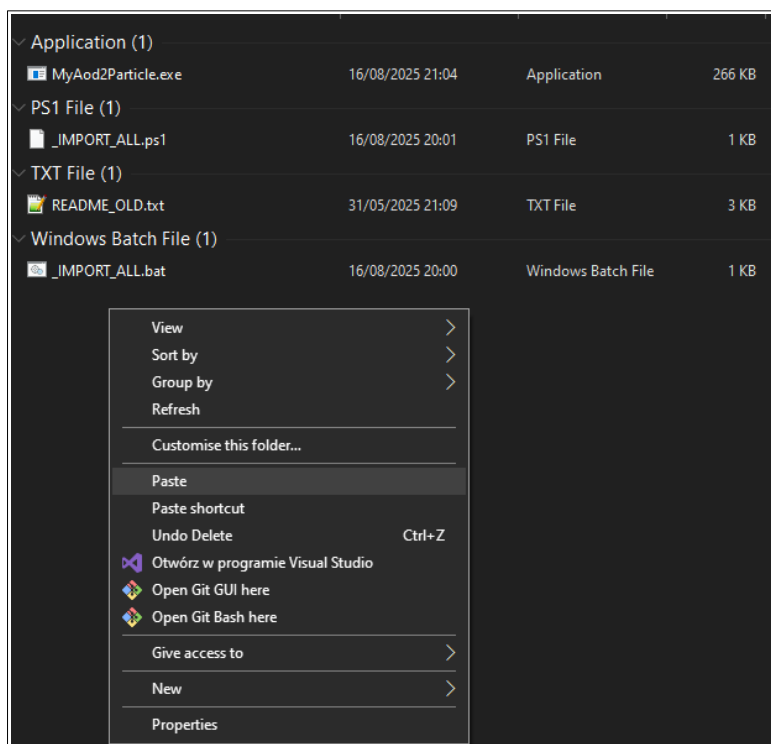


Figure 21: Wklejam.

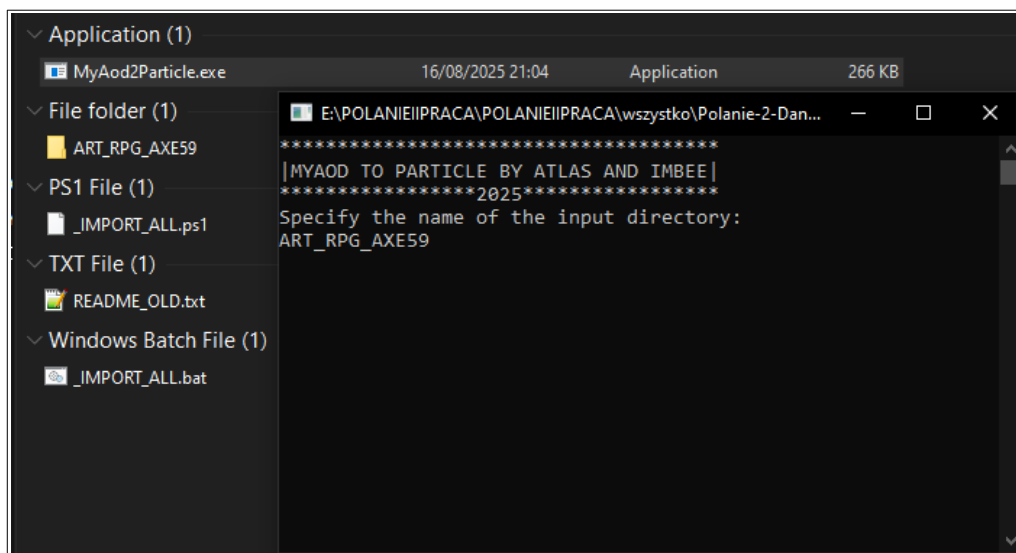


Figure 22: Włączam dwuklikiem program i wpisuję nazwę katalogu z danymi particlesa.

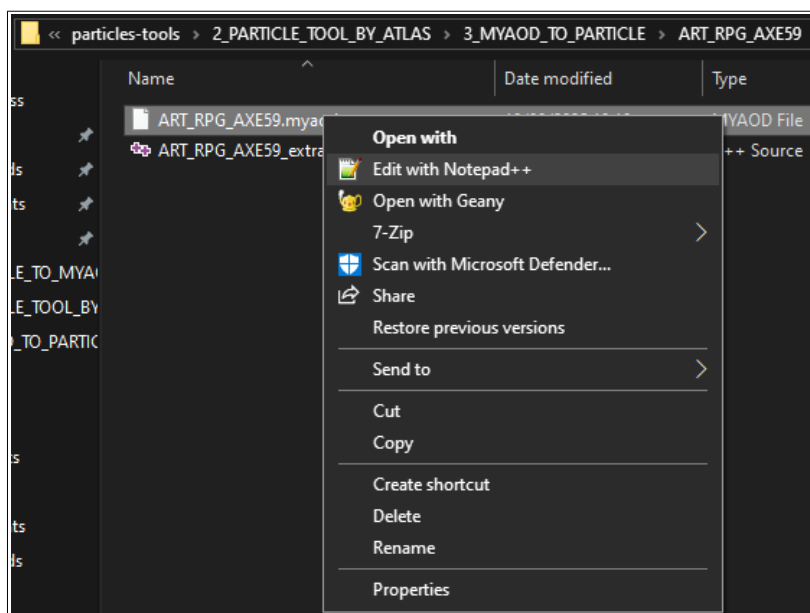


Figure 23: Wchodzę do katalogu z wypakowanymi danymi particlesa i sprawdzam zawartość pliku .myaod.


```

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: Sprawdzam i zapamiętuję 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: Włączam ParticleEdit.exe z E2160.

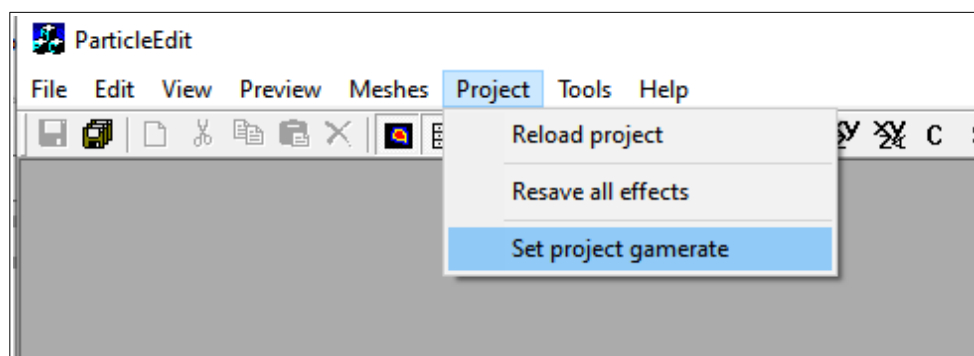


Figure 26: Namierzam i klikam opcję **Set project gamerate**.

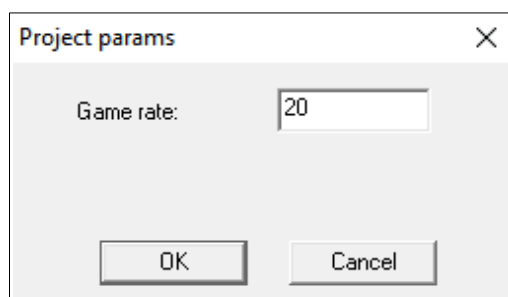


Figure 27: Ustawiam wartość na taką, którą wcześniej odczytaliśmy z pliku.

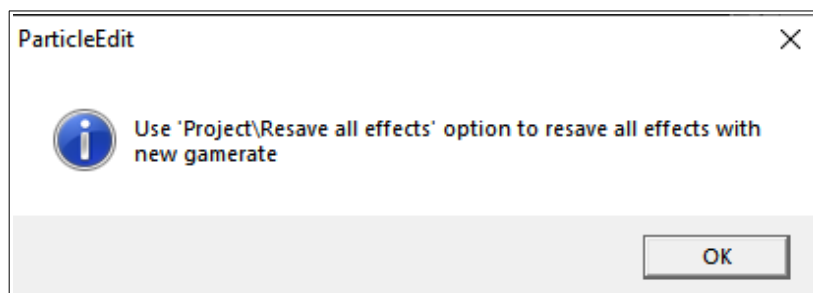


Figure 28: Wyświetla się taki komunikat.

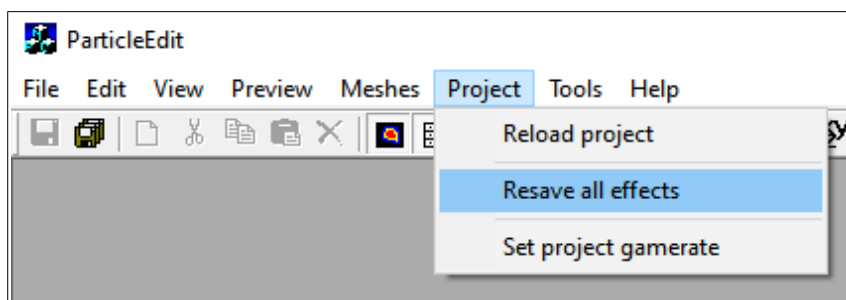


Figure 29: Namierzamy i klikamy **Resave all effects**. Aby operacja wykonała się poprawnie, należy chwilę poczekać.



Figure 30: Jak operacja się zakończy to wyłączamy program.

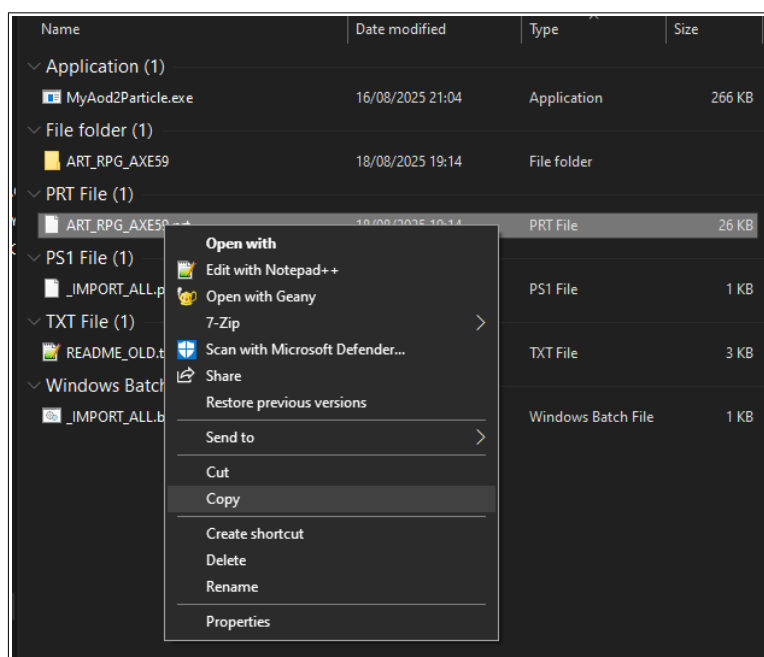


Figure 31: Kopiuje wyjściowy plik .prt.

Name	Date modified	Type	Size
Backups	08/07/2025 21:29	File folder	
Resources	10/06/2025 17:37	File folder	
1970.prt	18/08/2025 10:21	PRT File	19 KB
AMOEBE.prt	18/08/2025 10:27	PRT File	48 KB
ART_AMMO.prt	06/07/2025 12:21	PRT File	22 KB
ART_MINE.prt	07/07/2025 16:34	PRT File	18 KB
BLACK_HOLE_VORTEX.prt	18/08/2025 10:25	PRT File	20 KB
BM_7_2.prt	16/07/2025 22:19	PRT File	113 KB
Cursor01.prt	28/07/2025 14:35	PRT File	6 KB
e2160_prt1.prt	09/07/2025 20:30	PRT File	35 KB
e2160_prt2.prt	04/07/2025 12:27	PRT File	24 KB
Fire5.prt	18/08/2025 10:33	PRT File	34 KB
Geyser1.prt	07/07/2025 18:09	PRT File	28 KB
KOLUMNA.prt	19/07/2025 21:07	PRT File	43 KB
L_MI_EM_01_1.prt	21/06/2025 19:25	PRT File	12 KB
linked_e21.prt	09/07/2025 12:12	PRT File	5 KB
linked_e22.prt	08/07/2025 19:51	PRT File	5 KB
M8_TEMPLE_FX.prt	19/07/2025 21:32	PRT File	42 KB
particleslib.lprt	28/07/2025 13:29	LPRT File	1 KB
prt1.prt	04/07/2025 15:16	PRT File	32 KB
SaveEffects.log	01/08/2025 19:37	LOG File	29 KB
SG_OUT_OK.prt	07/07/2025 19:16	PRT File	81 KB
TEST.prt	05/06/2025 20:35	PRT File	5 KB
TEST.prt.bak	05/06/2025 20:33	BAK File	5 KB
TEST_or.prt	05/06/2025 20:46	PRT File	10 KB
TTEE.prt	08/06/2025 12:55	PRT File	8 KB
TTEE1.prt	08/06/2025 15:16	PRT File	8 KB
TTEE1.prt.bak	08/06/2025 15:14	BAK File	8 KB

Figure 32: Namierzam katalog z plikami .prt programu ParticleEdit z Earth 2160.

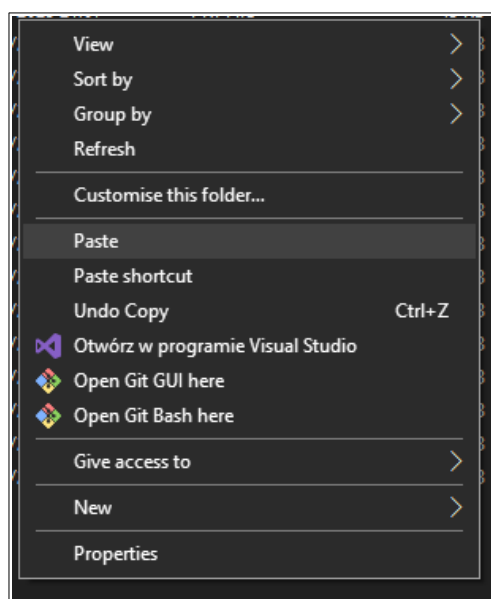


Figure 33: Wklejam.



	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: Ponownie uruchamiam ParticleEdit z Earth 2160.

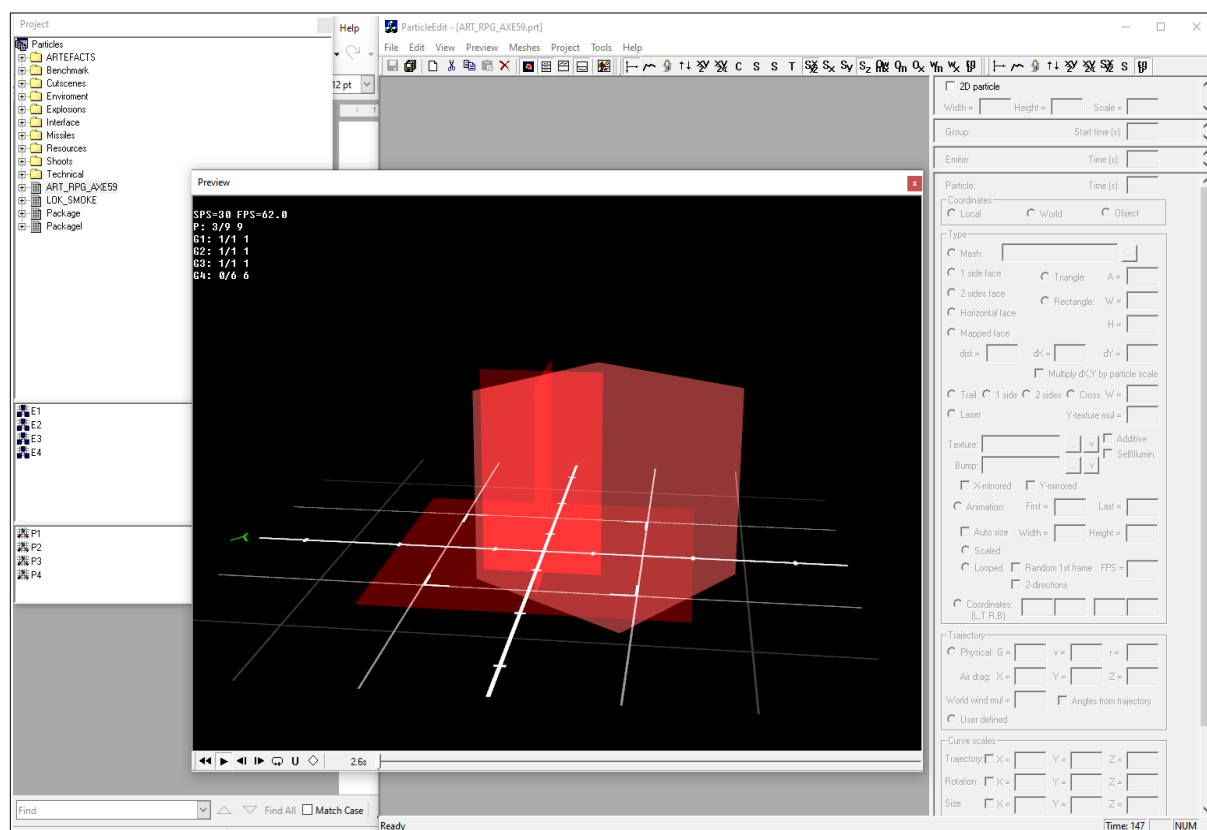


Figure 35: Jak widzimy particle się uruchomił ale, żeby pozbyć się tych czerwonych tekstur należy przekonwertować .tex na .tga i wrzucić tekstury do katalogu Textures w katalogu z grą E2160.

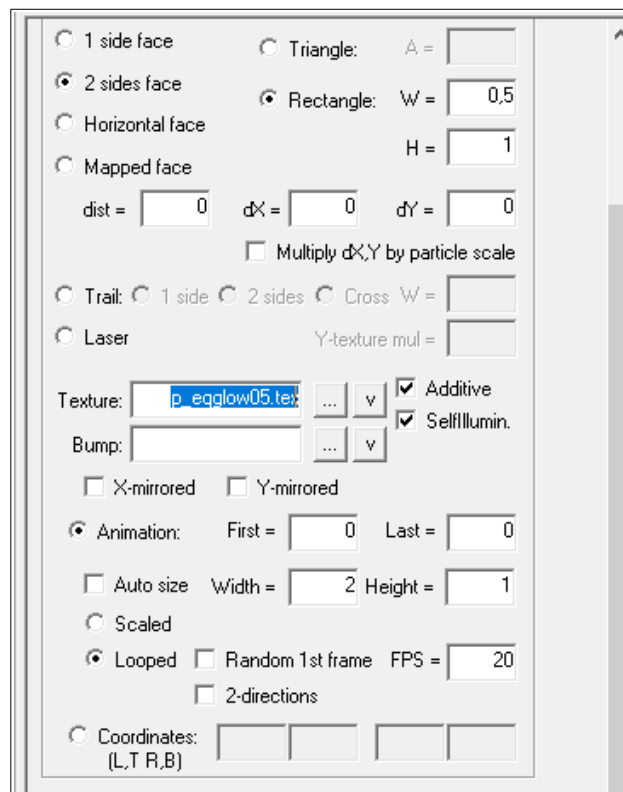


Figure 36: Można przeskakiwać po obiektach ParticleEmitera i zmieniać nazwy tekstur oraz meshy.

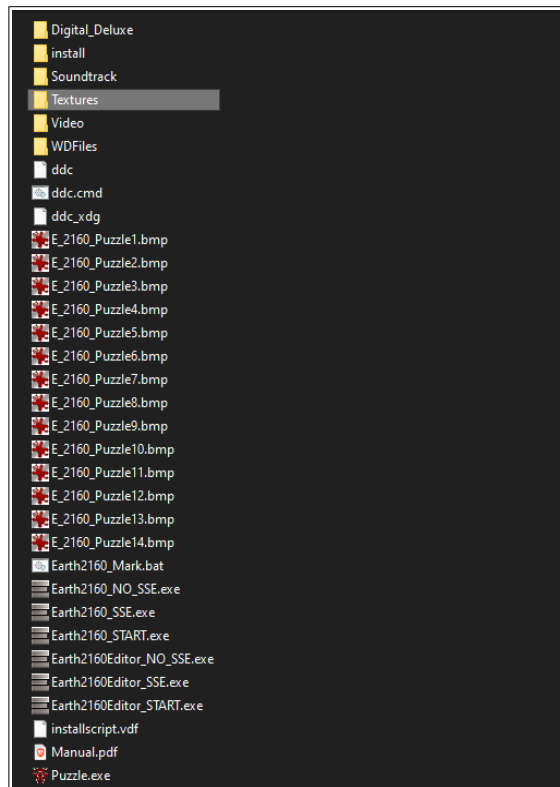


Figure 37: Katalog Textures tworzymy w folderze z grą Earth 2160 i wrzucamy tam nasze texture w formacie tga. W ten sposób nie trzeba pakować tych plików do WD.

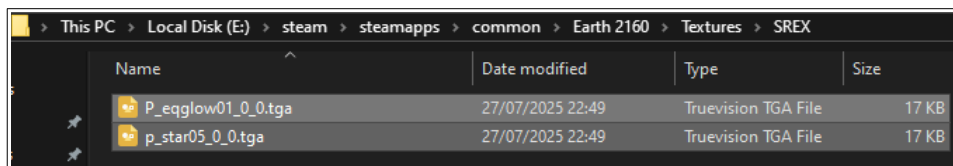


Figure 38: Ja sobie wrzuciłem takie texture.

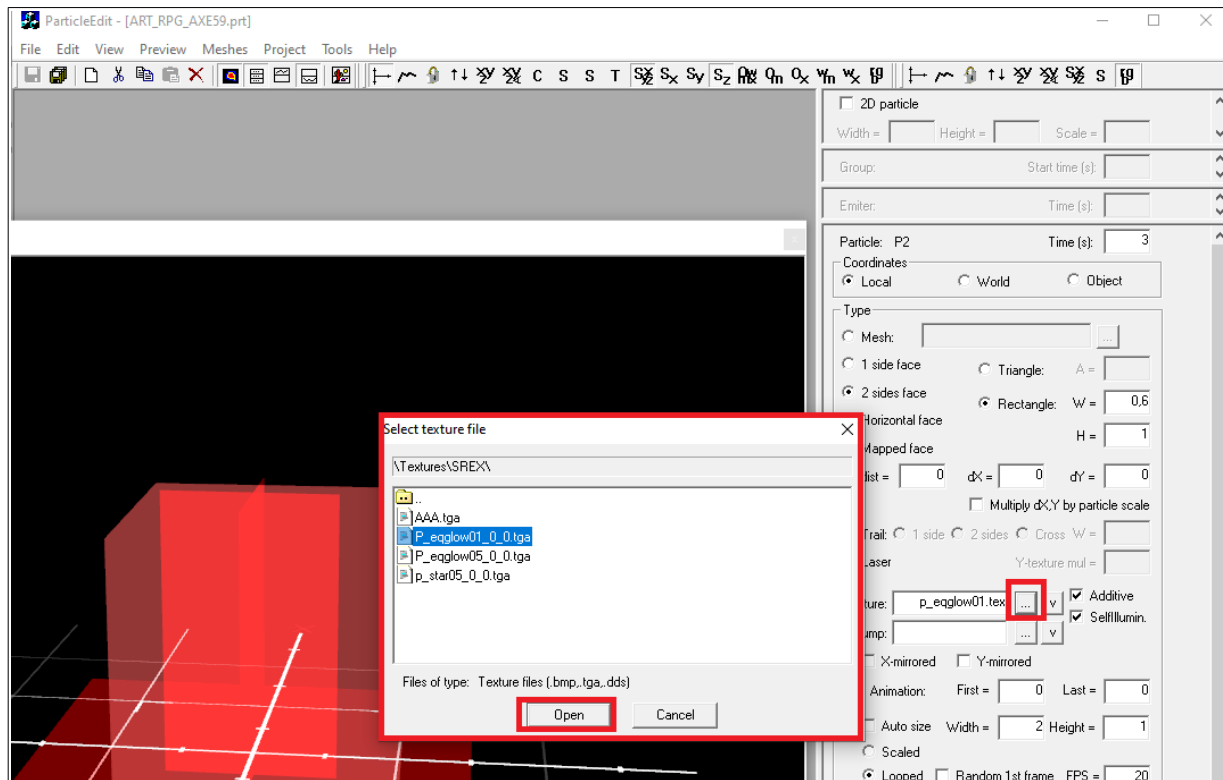


Figure 39: Wybieramy texture tga z katalogu z teksturami. Robimy tak dla każdej texture tex. Przy okazji możemy zmienić też mesha na np. Sphere.

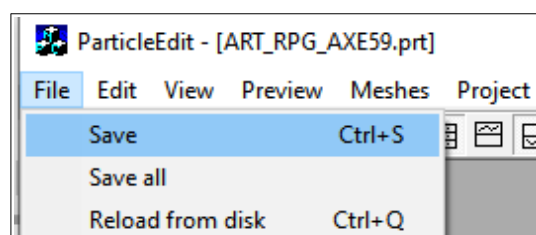


Figure 40: Zapisujemy particlesa.

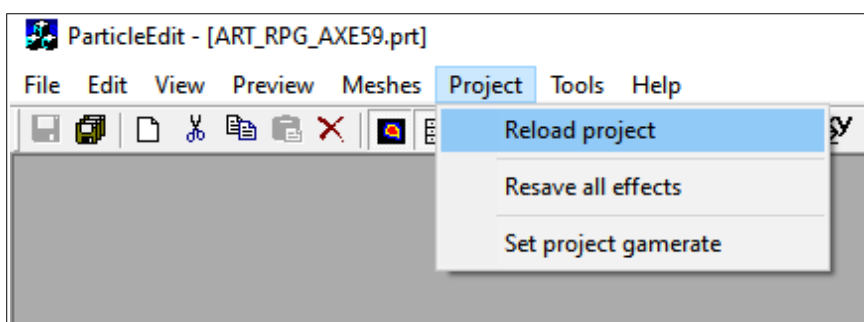


Figure 41: Aby odświeżyć projekt w celu zobaczenia zmian, możemy kliknąć opcję **Reload project**.

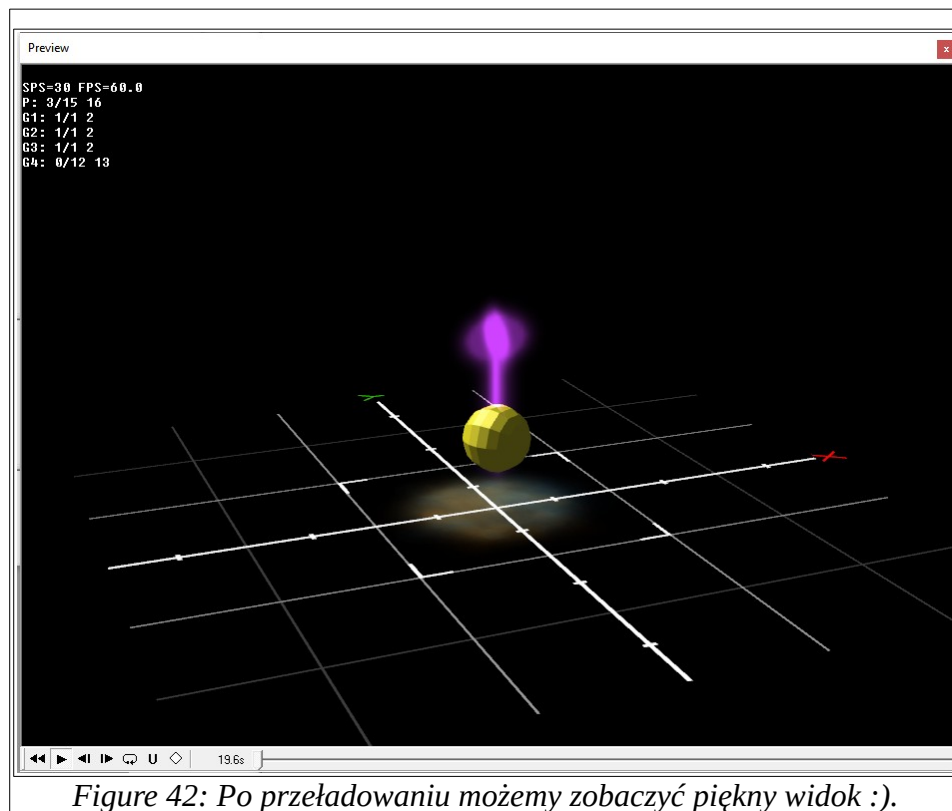


Figure 42: Po przeładowaniu możemy zobaczyć piękny widok :).

d) Chcę dokonać zmian w ParticleEmitterze z KnightShift. Eksportujemy plik do formatu E2160 jak w punkcie c). Edytujemy według uznania.

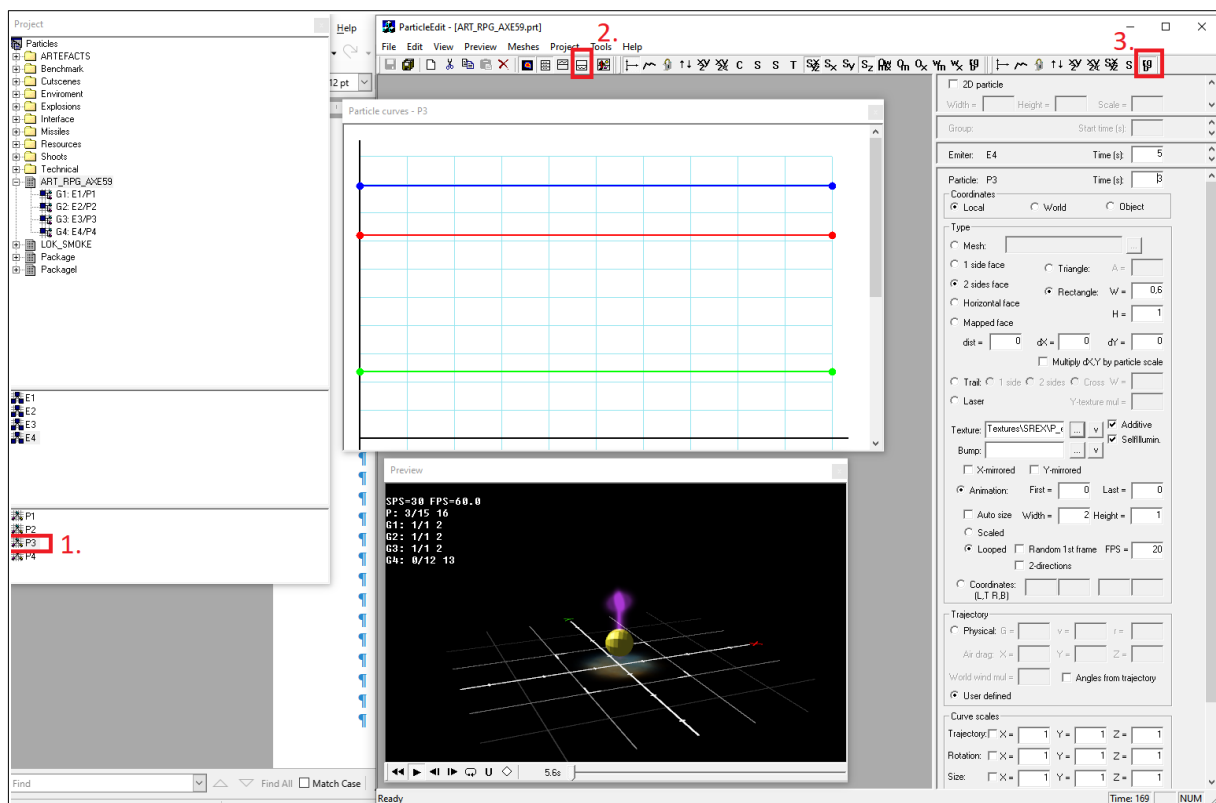


Figure 43: Teraz np. namierzamy to co chcemy zmienić i dokonujemy zmian. Ja przykładowo zmienię kolor tego obramowania broni.

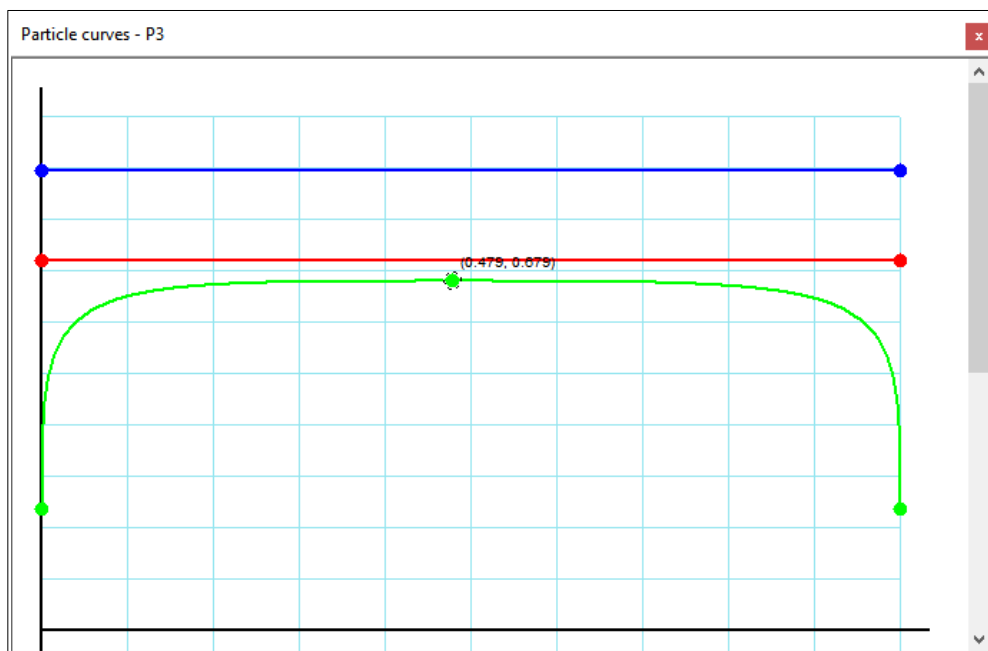


Figure 44: Manipuluję punktami.

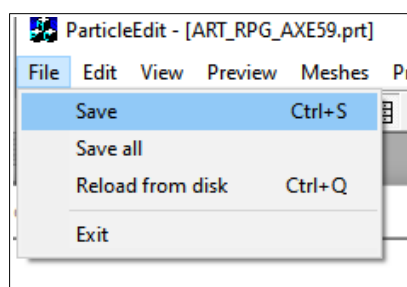


Figure 45: Zapisuję.

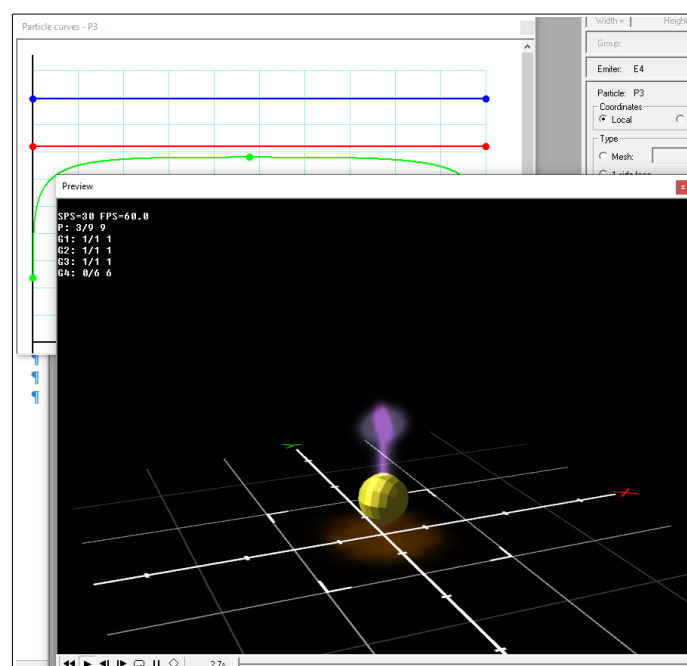


Figure 46: Obrazowanie się zmieniło na kolor biały.

Jeżeli chcemy z powrotem wrzucić ten plik do KnightShift to musimy przywrócić wpisy o poprzednich teksturach .tex i meshach. Można to zrobić ParticleEditem w tym momencie albo podczas powrotnej konwersji do formatu .myaod przez edytor tekstu np. Notepad++.

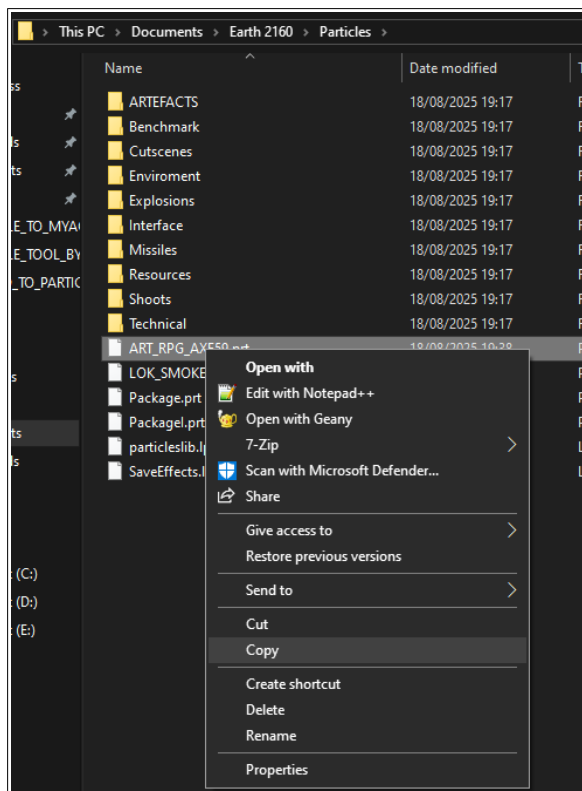


Figure 47: Kopiujemy plik .prt z katalogu particlesów programu ParticleEdit.

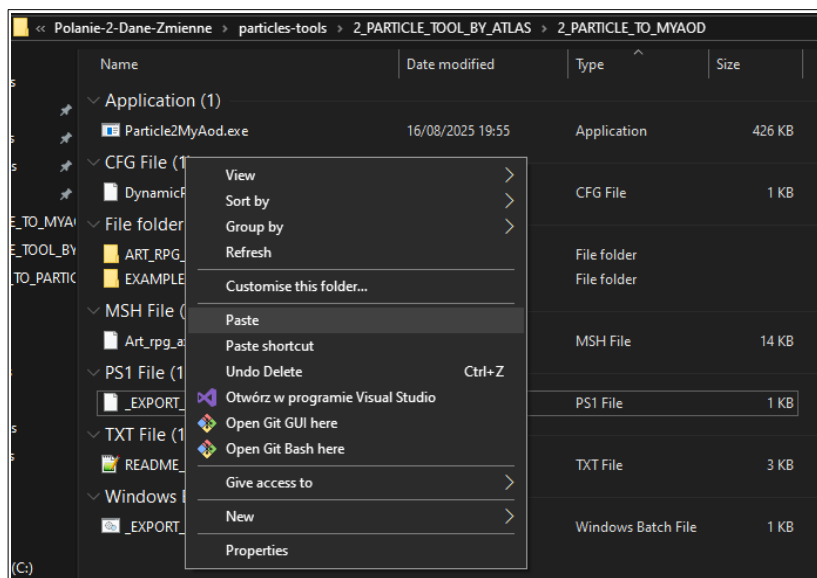


Figure 48: Wklejamy do katalogu Particle2MyAod.

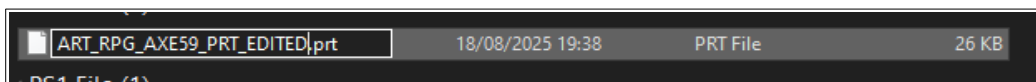


Figure 49: Zmieniamy profilaktycznie nazwę pliku, żeby katalogi wyjściowe się nie pomieszały.

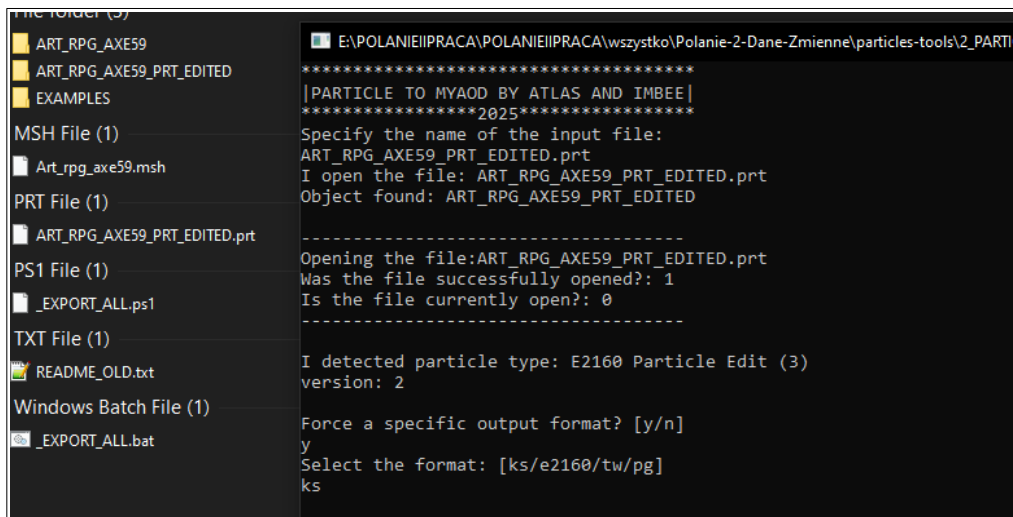


Figure 50: Eksportujemy wymuszając format ks.

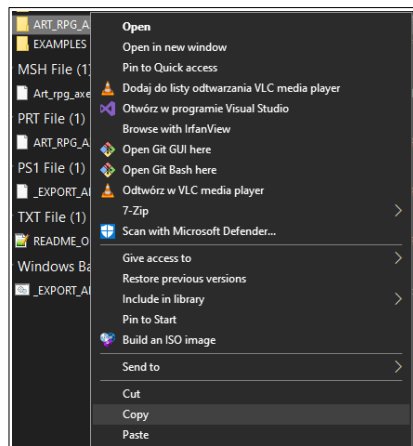


Figure 51: Kopiujemy katalog, który powstał.

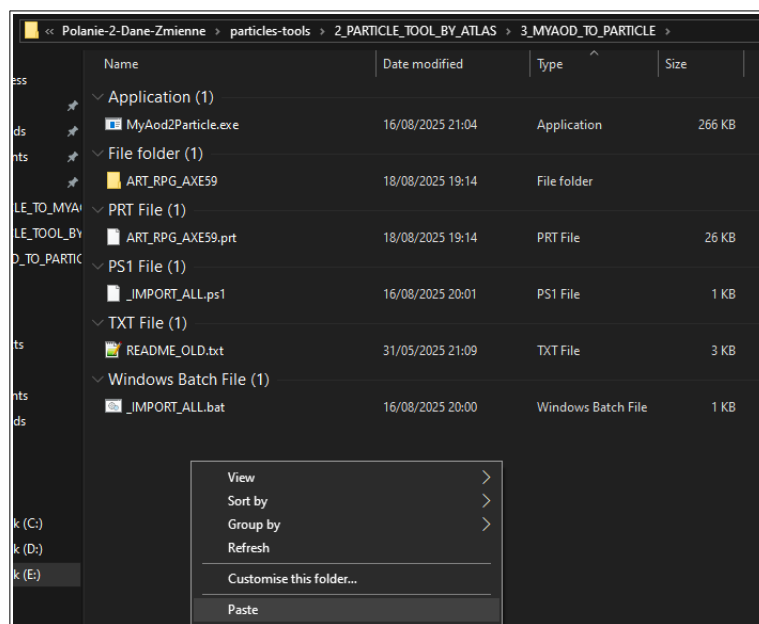


Figure 52: Wklejamy go do katalogu MyAod2Particle.

```
E:\POLANIEIIPRACA\POLANIEIIPRACA\wszystko\Polanie-2-Dane-Zmienne\particle2\Particle2MyAod.exe

*****
|MYAOD TO PARTICLE BY ATLAS AND IMBEE|
*****2025*****
Specify the name of the input directory:
ART_RPG_AXE59_PRT_EDITED_
```

Figure 53: Importujemy.

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

Figure 54: Gotowe - można wrzucić plik do KnightShift.

e) Chcę uruchomić plik .prt z 3D ParticleGen Visual FX (steam) w ParticleGenie z Two Worlds:

Name	Date modified	type	Size
Application (1)			
Particle2MyAod.exe	16/09/2025 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: Włączam program **Particle2MyAod.exe**.
Wprowadzam **nazwę pliku**, potem wprowadzam **y** wymuszając
inny format i wprowadzam **tw**.

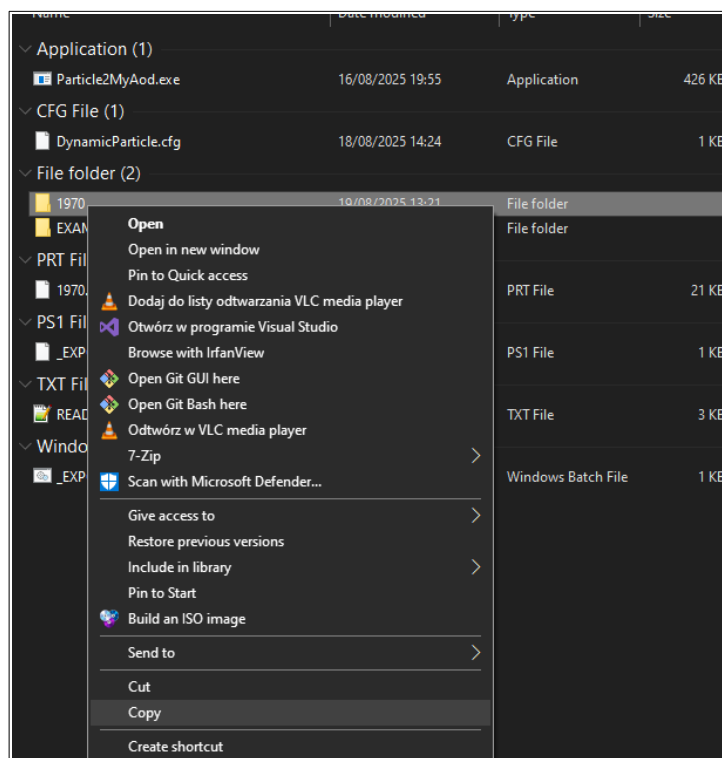


Figure 56: Kopiuję uzyskany katalog.

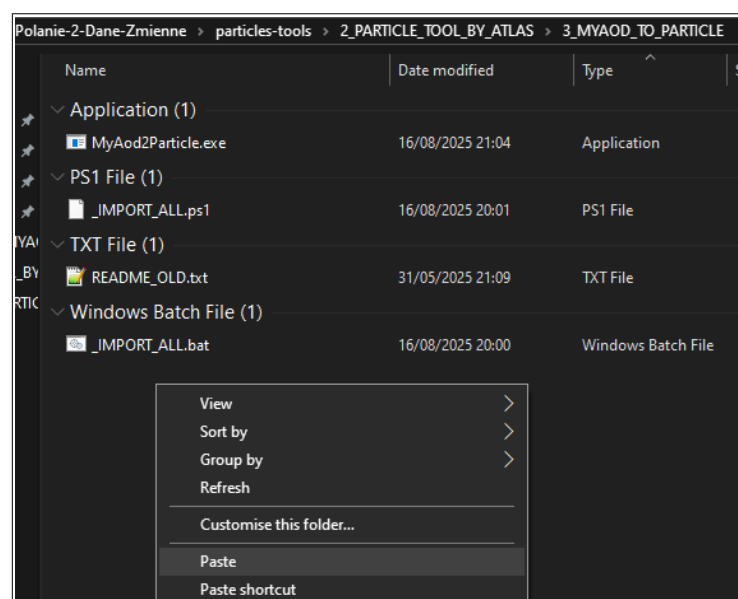


Figure 57: Wklejam do katalogu z **MyAod2Particle.exe**.

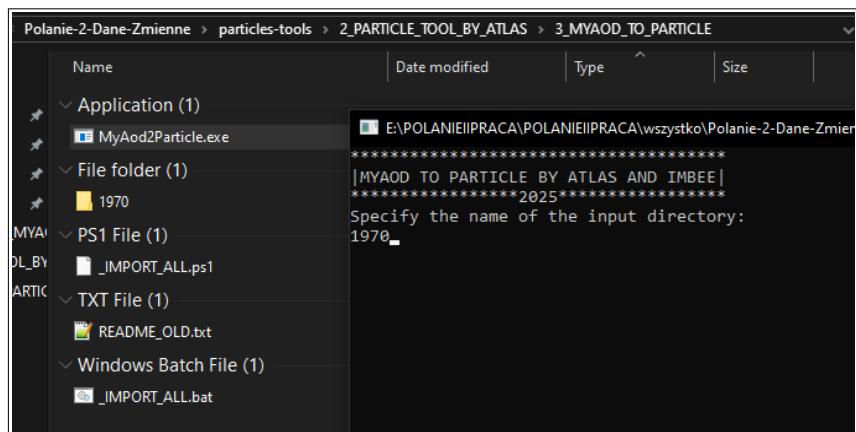


Figure 58: Włączam **MyAod2Particle** i wprowadzam do niego nazwę wklejonego katalogu.

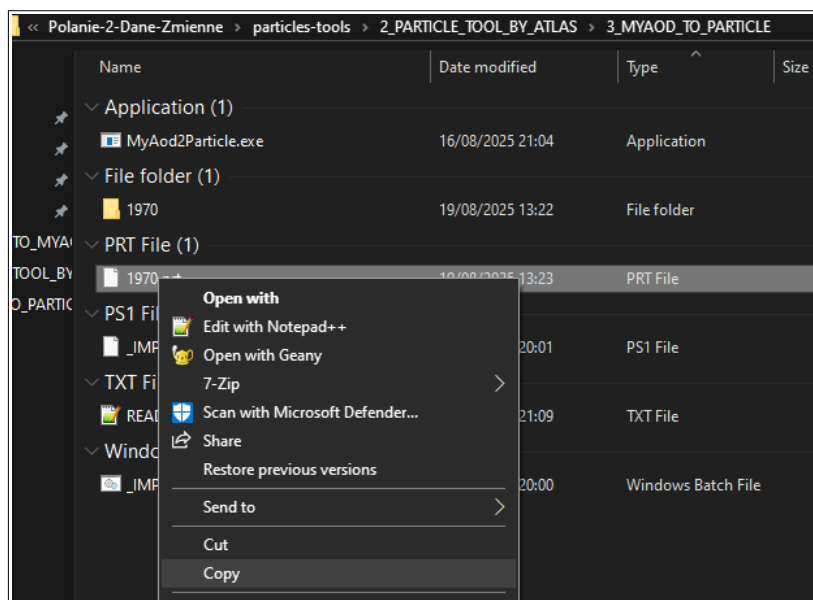


Figure 59: Kopiaję uzyskany plik **.prt**.

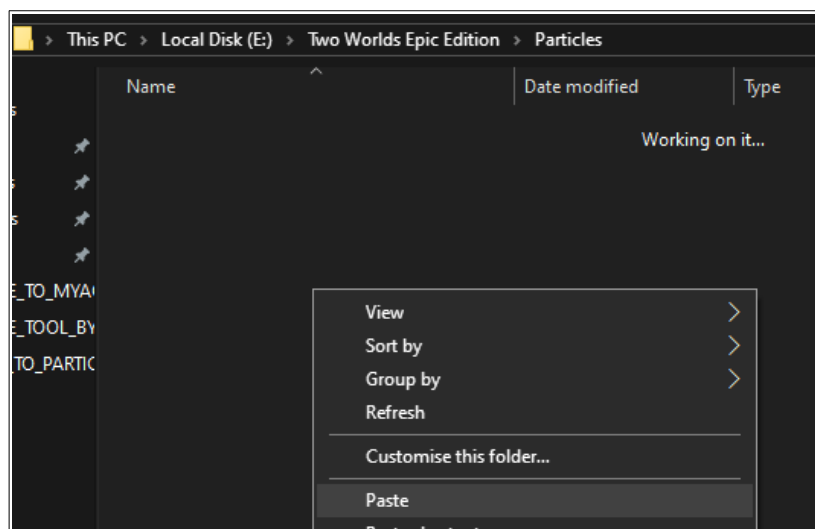


Figure 60: Wklejam do katalogu **Particles** w folderze z grą **Two Worlds**.

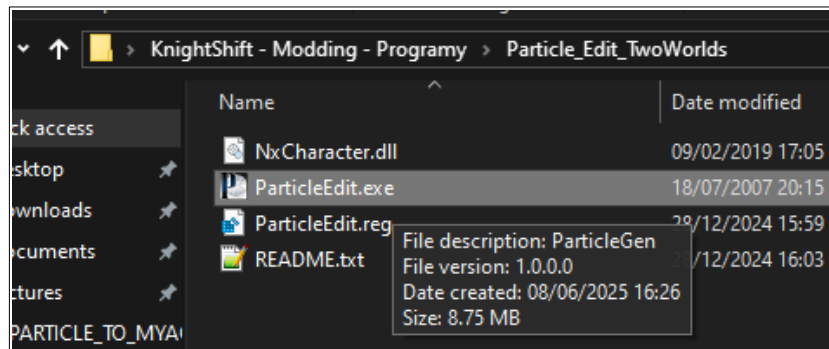


Figure 61: Uruchamiam **ParticleGena** z **Two Worlds SDK**.

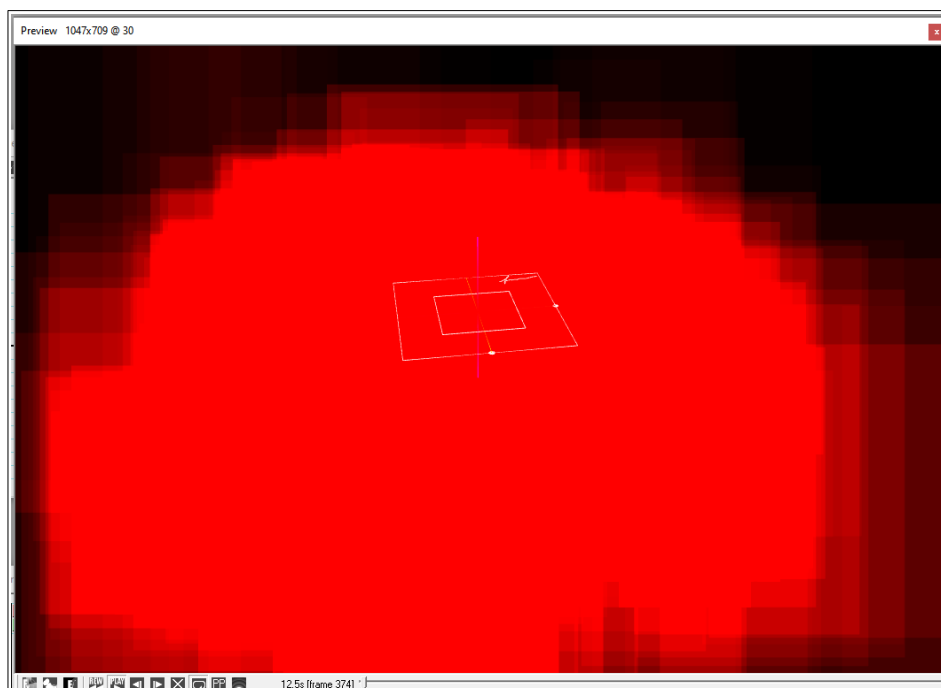


Figure 62: Jak widzimy po uruchomieniu particlesa w programie - plik działa jednak trzeba jeszcze przerzucić odpowiednią teksturę z 3D Particle Gena do Two Worlds.

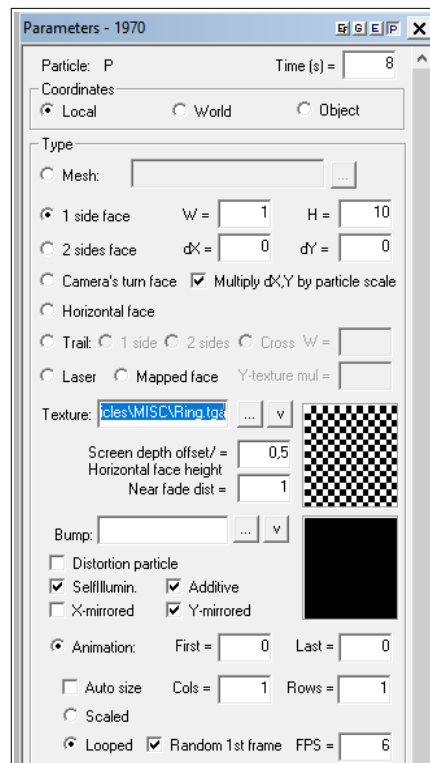


Figure 63: Możemy namierzyć uruchamianą teksturę w parametrach obiektów.

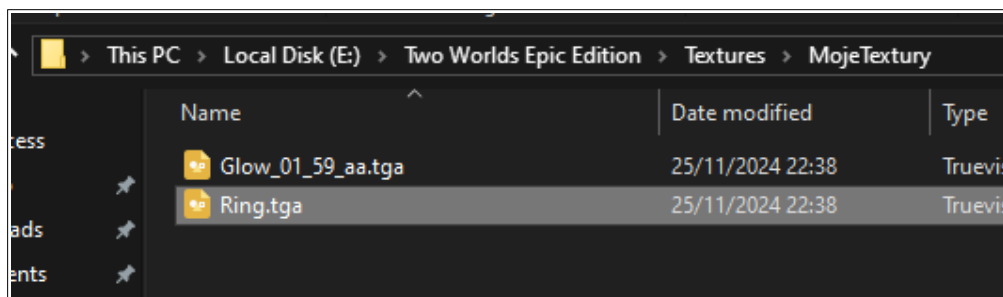


Figure 64: Wrzucamy do katalogu Textures w Two Worlds wymaganą teksturę (ja sobie zrobiłem dodatkowy katalog wewnątrz o nazwie MojeTexturey).

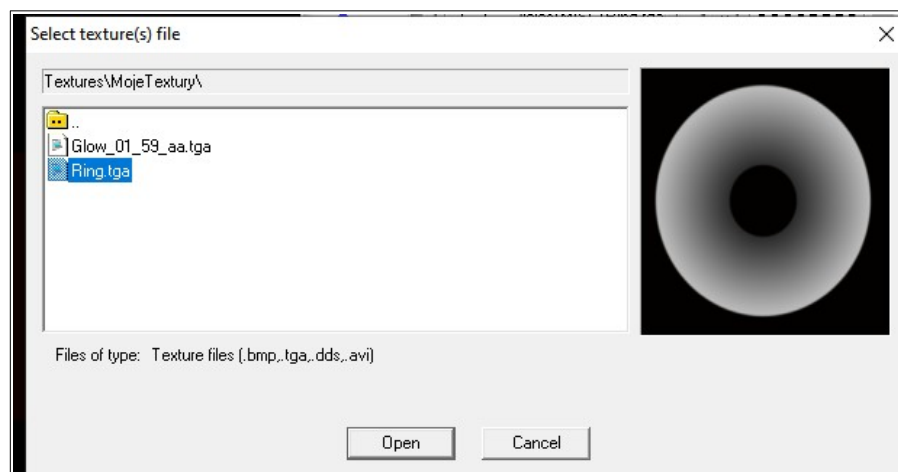


Figure 65: W programie do particlesów wybieramy dorzuconą teksturę.

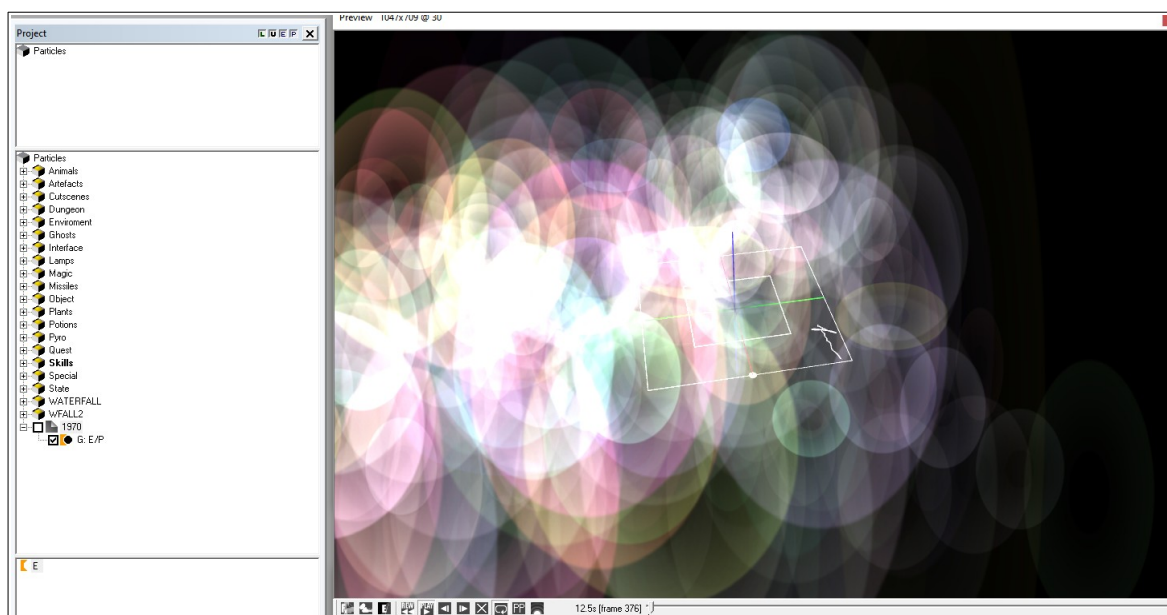


Figure 66: Po zapisaniu particlesa i zrestartowaniu programu możemy zauważyć efekt końcowy.

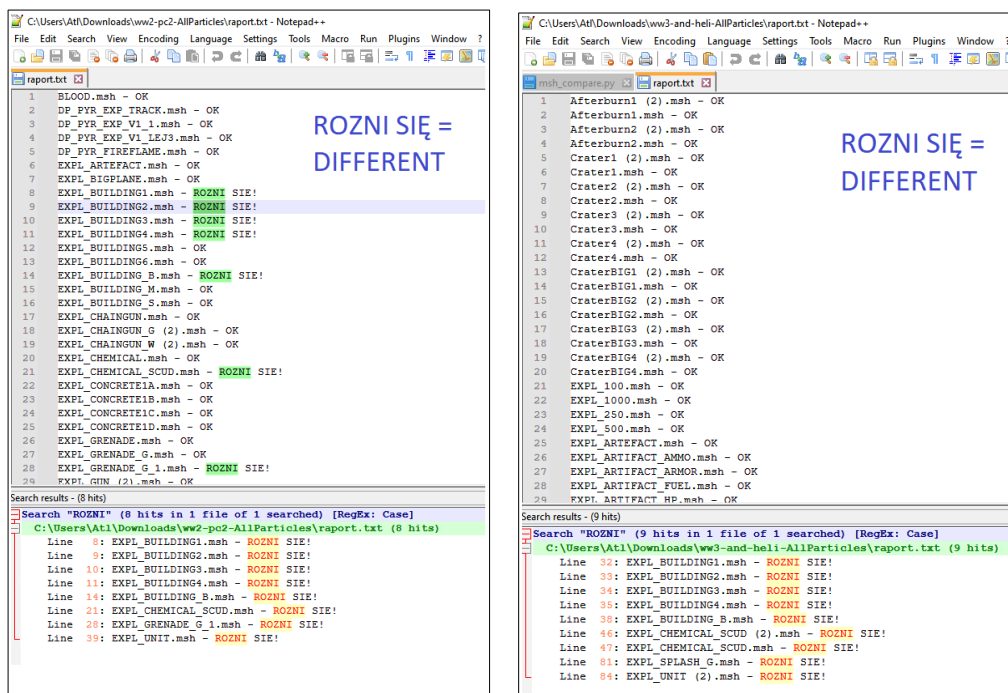
Dodatkowe informacje:

1. Wyniki testów:

Programy, a dokładnie funkcje eksportu i powrotnego importu były testowane na:

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

Dlaczego nie ma 100% w dwóch ostatnich wynikach?



Spowodowane jest to tym, że w pliku **.aod** standardowo **Time** jest zapisywane na dwóch wartościach typu **float**, a kompilator **Aod2Msh** dodatkowo wykonuje na tych wartościach pewne obliczenia matematyczne i “castuje” czyli zmienia typ wartości zmiennej na **int64_t** by później zapisać wartość na **int32_t**. Powoduje to utratę informacji, która wywołuje właśnie różnicę w polu, w którym zapisana jest wartość **Time** w **Dynamic particlesie**. Różnicę w wartościach w pliku binarnym zaprezentowano na rysunku na następnej stronie.

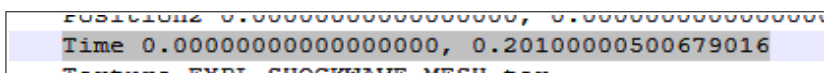


Figure 67: Time zapisane w pliku **.myaod**

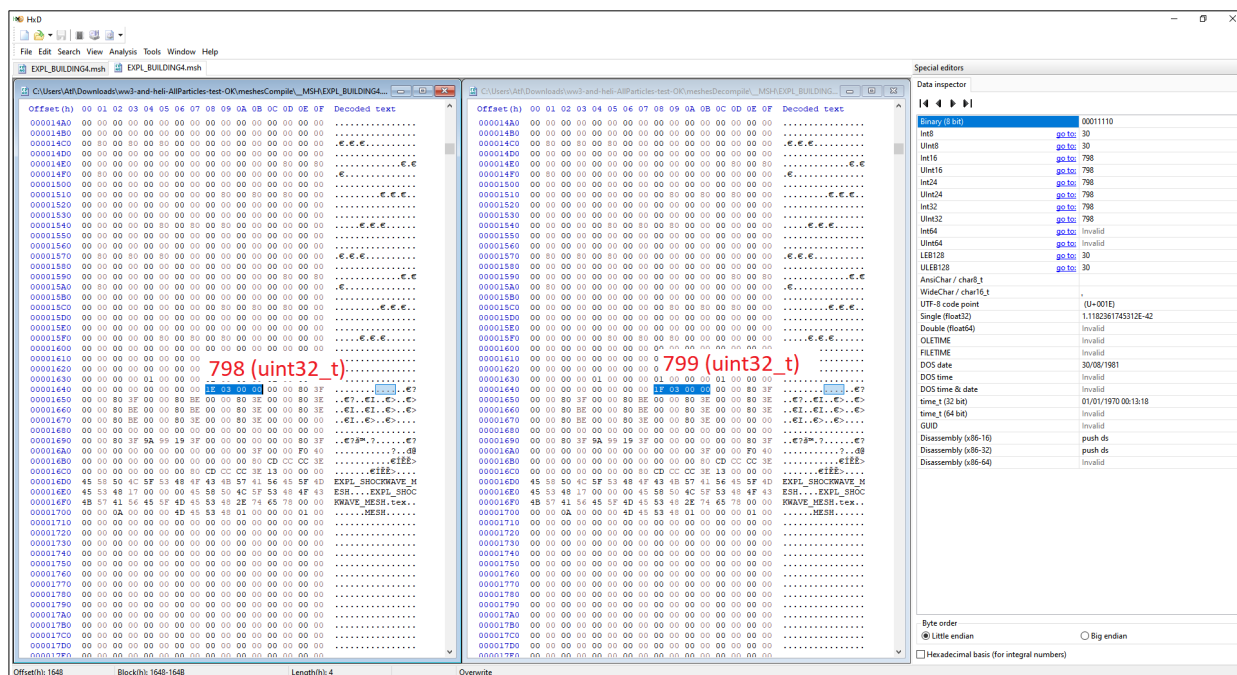


Figure 68: Widoczna różnica w wartościach wywołana dziwnymi obliczeniami matematycznymi i zmianami typu zmiennej.

Nie powoduje to jakichś olbrzymich problemów, ale wypadało zwrócić na to uwagę. Dowodem na występowanie takiego kodu, powodującego takie różnice - są przedstawione niżej obrazki.

```
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: Zrzut ekranu ukazujący kod pobierania Time po dekompilacji Aod2Msh w 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: Moja wersja w kodzie MyAod2Particle.

```

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: Pobieranie i obliczanie **Time** na podstawie wartości w pliku binarnym w programie **Particle2MyAod.exe**, gdzie **dynamic_particle_data.time[]** to tablica typu **float**.

2. Wpisy o teksturach w Dynamic particlesach z Earth 2150:

We wpisach tekstur w plikach **.myaod** z Dynamic particlesem z gry **Earth 2150** można zauważyć dopisaną nazwę katalogu **Textures** do ścieżki z teksturą.

```

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: Dodatkowa nazwa katalogu **Textures**.

W Dynamic particlesach z innych gier można zauważyć brak takiego wpisu. Spowodowane jest to tym, że **Aod2Msh** z **Earth 2150** dopisuje nazwę katalogu **Textures** przed nazwą textury. Zatem jeżeli chcemy użyć pliku **.myaod** z Dynamic particlesem z gry **Earth 2150** w innych grach to należy poprawić samodzielnie tą ścieżkę. To samo tyczy się używania **Aod2Msh** z **Earth 2150** na pliku **.myaod** z **Earth 2150**.

Figure 73: Dowód na to, że **Aod2Msh** z **Earth 2150** dopisuje napis **Textures** przed nazwą textury.

3. Dlaczego należy zmieniać gamerate w programie ParticleEdit z Earth 2160?:

Standardowo program **ParticleEdit** z **Earth 2160** ma ustawioną wartość gamerate na 30. Po wrzuceniu particlesa z innym gamerate i zapisaniem pliku może dojść do sytuacji, w której animacje po prostu będą się dziwnie zachowywać np. obracający się przedmiot w ostatnich paru sekundach animacji zacznie spadać w dół co wcześniej nie miało miejsca. Ze względu na problemy z gamerate – zaleca się wcześniejszą zmianę gamerate przed wrzuceniem jakiegokolwiek pliku i zapisaniem wszystkich efektów. Poradnik jak prawidłowo zmienić gamerate opisałem w jednym ze scenariuszy używania **Particle Toola**.

4. Brak obsługi Boxes oraz Slots:

W niektórych plikach **.aod** można było zauważyć wpisy typu Slots albo Boxes (rysunek niżej). Moje programy **Particle2MyAod** oraz **MyAod2Particle** nie obsługują tych instrukcji. Dlaczego moje programy tego nie obsługują? A to dlatego, że za tymi całymi Boxami i Slotsami kryją się jakieś dziwne obliczenia, których jeszcze w pełni nie zrozumiałem. Wszelka zmiana tych wartości odbywa się przez zmianę plików z końcówką **_extra_data.cpp** w katalogu z particlesem. Jeśli chcecie dokonać jakiś zmian to niestety sami musicie na razie odkrywać/interpretować ten wstępny blok danych z pliku **.msh**.

```
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: Przykładowy plik **.aod** z Boxami.

5. Dlaczego tylko częściowo można manipulować krzywymi w particlesach z KnightShift w ParticleEdicie z Earth 2160?

Niestety ze względu na nie zrozumienie w 100% zależności pomiędzy **teselacjami** (**teselate**), a **krzywymi** (**curve**), w niektórych **ParticleEmitterach** z **KnightShift** uruchomionych w **ParticleEdicie** (**E2160**) można zauważyć, że punkty krzywych wychodzą poza graf co uniemożliwia manipulowanie nimi. W przypadku **KnightShift** i plików **.msh** z **ParticleEmitterami** przechowywane są tylko teselacje, ponieważ krzywe w momencie konwersji z **.aod** do **.msh** zostają usunięte. Program **Particle2MyAod.exe** dokonuje interpolacji na teselacjach tworząc punkty krzywych i krzywe, które potem wypisuje do **.myaod**, jednak nie jest to w 100% idealne tak jak wyżej napisałem i część punktów na grafie wychodzi poza wykres, co uniemożliwia manipulację. Ten problem da się poprawić, ale należy lepiej zrozumieć zależności pomiędzy punktami krzywych i teselacjami. Jeśli komuś się uda znaleźć tą zależność to proszę się kontaktować ze mną:

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