

OPIS PLIKU TRASY

Autor: Marcin Woźniak

Komentarze i poprawki: M.Czapkiewicz

na czerwono – komentarze MC

na niebiesko – nowe zmiany

na żółto – propozycje

niebieską czcionką komentarze MW,MC

Domyślny plik trasy powinien nazywać się **scene.scn**, możliwe jest wgranie innego pliku poprzez podanie jego nazwy jako parametr, np.

`eu07.exe -s scenery/testowo.scn`.

Można też zdefiniować domyślną nazwę scenerii w pliku **eu07.ini**

Parametry obiektów w scenerii mogą być oddzielane spacjami, przecinkami, średnikami, tabulatorami lub końcami linii.

1 Co może zawierać plik trasy:

Node, Event, Include, Trainset i inne.

1.1 Node

Definiuje obiekty widoczne (a ogólniej: obiekty które bierze pod uwagę funkcja Render).

Parametry podstawowe:

- **MaxDistance** – maksymalna odległość z jakiej obiekt będzie widoczny
- **MinDistance** – minimalna odległość z jakiej obiekt będzie widoczny
- **Name** – nazwa obiektu, jeśli nie potrzebujemy należy wpisać **none**
- **Type** - rodzaj obiektu

Poniżej są opisane różne obiekty typu **Node**.

Oprócz wyżej wymienionych parametrów, po **Type** są definiowane dodatkowe parametry zależne od **Type**.

1.1.1 Track

Definiuje nam tor, po którym możemy puścić obiekt **dynamic**. Kształt toru definiujemy za pomocą krzywej Beziera (jak w Corel Draw).

Parametry: **na zielono wartości domyślne (default) które powinny być w skrypcie tworzącym**

- TrackType –możliwe wartości to:
 - **normal** zwykły tor
 - **switch** zwrotnica
 - **road** droga
 - **river** rzeka

Category – czym jest ten Track (1- tor kolejowy, 2- droga, 4- rzeka) ——— 1

Uwaga! Interpretacja poniższych parametrów dotyczy się tylko toru kolejowego, dla innych obiektów będzie podana w oddzielnym pliku wraz z przykładami.

- TrackLength – długość odcinka toru
- TrackWidth – szerokość toru (potrzebne w wielu sprawach) 1.435
- Friction – statyczny współczynnik tarcia 0.15 czy ta wielkość jest prawidłowa?
- SoundDist – co ile metrów będzie odgrywany dźwięk stukotu 20
- Quality – pierwsze 4 bity – ile ton/oś, pozostałe – rezerwa 20
- DamageFlag – kombinacja stałych dtrack_* z mover.pas, np 128 oznacza brak szyn 0 dla normal, 2 dla switch
- Environment – słowo kluczowe oznaczające otoczenie toru: flat, mountains, canyon, tunnel (w zależności od tego będzie się zmieniać oświetlenie i/lub echo dźwięków)
- Visibility – jeśli tor ma być niewidoczny wpisujemy **unvis** ale normalnie powinno być **vis** i wtedy należy podać:
- Tex1 – tekstura szyn (jeśli **none** to szyny nie są rysowane) Rail_screw_used1
- TexLength – długość w [m] odpowiadająca teksturze szyny 4.0
- Tex2 – dla **track normal** tekstura podsypki z podkładami (gdy none to nie jest automatycznie rysowana) a w przypadku zwrotnicy tekstura szyn drugiego toru (dla zwrotnicy podkłady trzeba zdefiniować jako oddzielny obiekt!) TpD.tex
- TexHeight – wysokość automatycznie rysowanej podsypki (w przypadku zwrotnicy odstęp iglicy od szyny) 0.2
- TexWidth – szerokość automatycznie rysowanej podsypki od szyny do początku nachylenia (w przypadku zwrotnicy długość odbojnicy) 0.5
- TexSlope – szerokość automatycznie rysowanej podsypki w obszarze pochylenia (w przypadku zwrotnicy odległość środka odbojnicy od końca rozjazdu) 1.1

geometria toru:

- **State – (tylko w przypadku zwrotnicy) jak jest przełożona zwrotnica, 0 lub 1 (zawsze na prosto)**
- Point1 – punkt początkowy toru [x,y,z]
- Roll1 – przechyłka początku toru [deg]
- CVec1 – wektor [x,y,z]
- CVec2 – wektor [x,y,z]
- Point2 – punkt końcowy toru [x,y,z]
- Roll2 – przechyłka końca toru [deg]
- Radius1 – najmniejszy promień toru (wartości Radius i TrackLength przydatne będą dla AI dyspozytora)
- Point3 – (tylko w przypadku zwrotnicy) punkt początkowy toru [x,y,z]
- Roll3 – (tylko w przypadku zwrotnicy) przechyłka toru [deg]
- CVec3 – (tylko w przypadku zwrotnicy) wektor [x,y,z]
- CVec4 – (tylko w przypadku zwrotnicy) wektor [x,y,z]

- Point4 – (tylko w przypadku zwrotnicy) punkt końcowy toru [x,y,z]
- Roll4 – (tylko w przypadku zwrotnicy) przechyłka toru [deg]
- Radius2 – najmniejszy promień toru (tylko w przypadku zwrotnicy)
- Velocity (opcjonalny) – prędkość jakiej będzie się starał nie przekroczyć jadący przez ten tor obiekt **dynamic** jeśli jest sterowany przez AI
- Event0 (opcjonalny) – zdarzenie które zostanie uruchomione gdy **obsadzony załogą dynamic** stoi na torze.
- Event1 (opcjonalny) – zdarzenie które zostanie uruchomione gdy **obsadzony załogą dynamic** wjedzie na tor w kierunku punktu początkowego (Point1)
- Event2 (opcjonalny) – zdarzenie które zostanie uruchomione gdy **obsadzony załogą dynamic** wjedzie na tor w kierunku punktu końcowego (Point2)
- Eventall0 (opcjonalny) – zdarzenie które zostanie uruchomione gdy **jakikolwiek dynamic** stoi na torze.
- Event1 (opcjonalny) – zdarzenie które zostanie uruchomione gdy **jakikolwiek dynamic** wjedzie na tor w kierunku punktu początkowego (Point1)
- Event2 (opcjonalny) – zdarzenie które zostanie uruchomione gdy **jakikolwiek dynamic** wjedzie na tor w kierunku punktu końcowego (Point2)

Jeśli chcemy otrzymać tor prosty należy oba wektory CVec wyzerować oraz dać Radius1=0. Nazwa toru jest potrzebna tylko gdy chcemy na ten tor jakoś oddziaływać (np. przełożyć zwrotnicę) albo ustawić na nim obiekt **dynamic**. Zwykły tor nie musi mieć konkretnej nazwy tzn. można go nazwać **none**.

Przykład toru prostego o nazwie track_sb102, o długości 100m, z ograniczeniem szlakowym 40km/h:

```
node -1 0 track_sb102 track normal 100.0 1.435 20.0 19 4 flat vis
Rail_screw_used1 4.0 TpB-old1.tex 0.2 0.5 1.1
-646.0 0.2 169.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
-646.0 0.2 69.0 0.0
0
event1 test_sb102_s1
event2 test_sb102_s1
velocity 40
endtrack
```

Przykład anonimowego zakrętu w wykopie

```
node -1 0 none track normal 100.0 1.435 25.0 20 0 canyon vis
Rail_screw_used1 4.0 TpD.tex 0.3 0.6 0.9
-46.0 0.2 -65.0001 0.0
0.0 0.0 -11.3351
1.28189 0.0 11.2623
-47.9246 0.2 -98.9273 0.0
300.0
endtrack
```

Przykład zwrotnicy:

```
node -1 0 Testowo_zwr1 track switch 34.0 1.435 25.0 20 2 flat vis
Rail_screw_used1 4.0 Rail_screw_uNused1 0.2 1.5 2.5
-46.0 0.2 269.0 0 //point 1
0.0 0.0 0.0 //control vector 1
0.0 0.0 0.0 //control vector 2
-46.0 0.2 235.0 0 //point 2
0
-46.0 0.2 269.0 0 //point 1
0.0 0.0 -11.3351 //control vector 1
1.28189 0.0 11.2623 //control vector 2
-47.9246 0.2 235.073 0 //point 2
-100
endtrack
```

1.1.2 Traction

Trakcja elektryczna

Parametry: na zielono wartości domyślne (default) które są w skrypcie tworzącym

- PowerSourceName – nazwa źródła zasilania, tak sama dla każdego odcinka zasilania
- NominalVoltage – napięcie w sieci trakcyjnej bez obciążenia 3500
- MaxCurrent – prąd przy którym napięcie w sieci spadłoby o połowę 4500
- Resistivity – rezystancja styku ślizgacz-przewód 0.01
- Material – z czego zrobiony jest drut [Cu, Al] Cu
- WireThickness – grubość drutu w mm 3
- DamageFlag – flaga bitowa uszkodzeń, 1 oznacza patynę, 128 oznacza zerwanie 1
- Point1 [x,y,z] – punkt początkowy dolnego przewodu (jezdnego)
- Point2 [x,y,z] – punkt końcowy dolnego przewodu (jezdnego)
- Point3 [x,y,z] – punkt początkowy górnego przewodu (nośnego)
- Point4 [x,y,z] – punkt końcowy górnego przewodu (nośnego)

(skrypt oblicza Point3 i Point4 dodając do Point1 i Point2 zmienne h1 i h2 czyli wysokości zaczepu górnego nad dolnym, defaultowo wynoszą one 1.65 m)

- Hmin – najniższa wysokość górnego przewodu nad dolnym 0.4
- DeltaL – odstęp pomiędzy kolejnymi wieszakami
- Wires – ilość przewodów (0,1,2,3 a w przyszłości 4) 2
- WireOffset – odstęp między przewodami jezdnymi gdy Wires=3 0.04
- Visibility – jeśli trakcja ma być niewidoczna, wpisujemy unvis vis

• CurrentEvent (opcjonalny) – zdarzenie które zostanie uruchomione gdy dynamic pobiera prąd z sieci.

Uwaga – słupy/wysięgniki są odrębnymi obiektami (skrypt 3dsmax umożliwia dołączanie słupów .inc do każdego węzła linii typu Traction)

3400;0.01;4500

1.1.3 TractionPowerSource (p2) (p3) (p4) (p7) 0 (p8) (p9) 1.0 3 60.0 norecuperation end

Zasilanie trakcji elektrycznej

UWAGA – nazwa tego obiektu informuje program że wszystkie obiekty typu Traction które mają taką samą nazwę w polu PowerSourceName należą do tego samego odcinka zasilania.

- Origin [x,y,z] – położenie źródła prądu
- NominalVoltage – napięcie w sieci trakcyjnej bez obciążenia
- VoltageFrequency – częstotliwość prądu (0 dla stałego)
- InternalRes – rezystancja wewnętrzna podstacji
- MaxOutputCurrent – prąd przy którym uruchamia się bezpiecznik nadmiarowy szybki
- FastFuseTimeOut – czas po którym obwód się uruchamia ponownie po przeciążeniu
- FastFuseRepetition – ilość prób wznowienia pracy obwodu
- SlowFuseTimeOut – czas po jakim zostanie uruchomiony obwód jeśli zostanie przekroczona ilość wznowień (załączeń bezpiecznika szybkiego)
- Recuperation – czy jest odzysk prądu z sieci (jeśli nie to dać NoRecuperation)

1.1.4 Dynamic

Obiekt poruszający się po torach.

Parametry:

- Directory – katalog bazowy obiektu np. PKP/EU07
- Model – plik z modelem obiektu np. EU07_HCP_zolteczolo.t3d model w pliku .mmd!
- ReplacableSkin – tekstura która zostanie podmieniona (na ogół nadwozie) – patrz model
- Type – plik (bez rozszerzenia .chk) z charakterystyką techniczną obiektu np. 303E
- Track (**tylko** jeśli nie jest częścią **TrainSet**) – nazwa toru na którym ustawiamy obiekt
- Dist – odległość początkowa (w/m pocz. toru)
- CabOccupancy – obsługa pojazdu, możliwe wartości:
 - headdriver – maszynista w pierwszej kabinie
 - reardriver – maszynista w drugiej kabinie
 - nobody – pojazd bez obsługi
 - passenger – bierny pasażer
 - conductor – konduktor
- Vel (tylko jeśli **NIE JEST** częścią **TrainSet**) – prędkość początkowa
- Coupler – (tylko jeśli **JEST** częścią **TrainSet**) – typ sprzęgu łączącego z poprzednikiem
- Load – ilość ładunku
- LoadType – nazwa ładunku – **TYLKO** gdy Load>0!

Komentarz: typ sprzęgu definiowany jest jako kombinacja bitowa flag: 1=sprzeg rzeczywisty, 2=sprzeg pneumatyczny, 4=sprzeg sterowania ukrotnionego itp. Zero oznacza brak fizycznego połączenia.

Przykład lokomotywy która na dzień dobry jedzie z prędkością 40km/h:

```
node -1 0 EU07-424 dynamic /pkp/eu07 EU07-424.tga 303E track_start 100.0 1 40.0  
0 enddynamic
```

patrz też **trainset**

mój komentarz: w katalogu można trzymać alternatywne modele i chki danej serii, jak również specyficzne tekstury, a w przyszłości dźwięki.

Przykładowo w katalogu /PKP/Bipa są 3 pliki .chk dla środkowych i skrajnych członów, w /PKP/EN57 będą pliki dla 6ra, 6s i 6rb.

Ponadto w tym katalogu znajdują się pliki *.mmd gdzie definiowane są dźwięki, wskaźniki, regulatory, modele itp.

1.1.5 Model

Wstawia model nieruchomy ale z możliwością animacji, np. semafor.

Parametry:

- Position – pozycja [x,y,z]
- Angle – kąt [deg]
- Model – model obiektu
- ReplacableSkin – jeśli w modelu istnieje tekstura o takiej nazwie to zostanie ona podmieniona
- Lights (opcjonalny) – stany świateł obiektu 0-wył, 1-wł, 2-migające
w modelu trzeba je nazwać Light_On01, Light_Off01, Light_On02, Light_Off02 itd.
Light_On01 to światło włączone, Light_Off01 to wyłączone

Przykład osadzenia modelu semafora świetlnego:

```
node -1 0 none model 10 20 4 90 SS5zpcpbY.t3d A Lights 0 0 1 0 0 0 endmodel
```

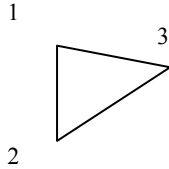
model będzie wyświetlany z teksturą A.bmp na tabliczce

Przykład osadzenia modelu latarni zwrotnicowej:

```
node -1 0 Testowo_zwr1_Wz model -44.0 0.2 234.5 0.0 WzL.t3d none endmodel
```

1.1.6 Triangles

Definiuje listę trójkątów:



Parametry:

- **ambient, diffuse, specular** – podatność materiału na oświetlenie tymi 3ma składowymi światła (np. materiał błyszczący powinien mieć wysoką składową *specular*, ściany tunelu powinny mieć wysoką składową *ambient* i bardzo niskie *diffuse* i *specular* aby światło słoneczne na nie nie padało, itp).
- Texture – nazwa pliku z teksturą obiektu
- Vertices – lista wierzchołków w formacie [x,y,z,nx,ny,nz,tu,tv], ich ilość musi być podzielna przez 3

x, y, z – współrzędne wierzchołka

nx, ny, nz – wektor normalny do płaszczyzny o długości 1

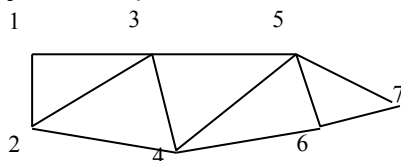
tu, tv – współrzędne tekstury w danym wierzchołku

Przykład trójkąta, który będzie widoczny z odległości mniejszej niż 1km:

```
node 1000 0 none triangles material ambient: 100 100 100 diffuse: 255 255 255
specular: 200 200 200 endmaterial TRACK.TGA
-51.4 0.2 0.0 -1.1094 1.6641 0.0 0.15 0.0 end
-51.4 0.2 100.0 -0.554701 0.83205 0.0 0.15 25.0 end
-48.6 0.2 0.0 0.0 2.0 0.0 0.85 0.0
endtri
```

1.1.7 Triangle_Strip

Definiuje wielokąt:

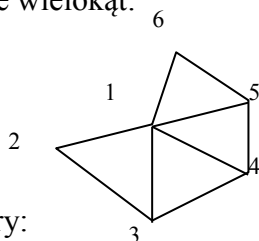


Parametry:

- Texture – nazwa pliku z teksturą obiektu
- Vertices – lista wierzchołków w formacie [x,y,z,nx,ny,nz,tu,tv]

1.1.8 Triangle_Fan

Definiuje wielokąt:



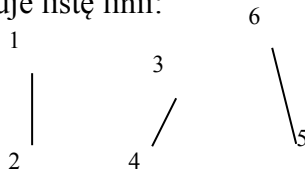
Parametry:

- Texture – nazwa pliku z teksturą obiektu

- Vertices – lista wierzchołków w formacie [x,y,z,nx,ny,nz,tu,tv]

1.1.9 Lines

Definiuje listę linii:



Parametry:

- Color – [r,g,b]
- Thickness – grubość linii
- Points – lista wierzchołków w formacie [x,y,z], ich ilość musi być podzielna przez 2

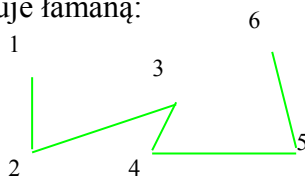
Jeśli grubość linii (liczona w pierwszym wierzchołku) odwzorowana na ekran jest mniejsza niż jeden piksel to linia rysowana jest jako częściowo przezroczysta (chyba że przezroczystość jest mniejsza niż 4%, wtedy w ogóle nie rysuje).

Przykład **czarnych linii o grubości 3mm**:

```
node 300 0 none lines 0 0 0 3
-638.0 0.0 -89.0 -638.0 1.7 -89.0
-638.0 1.7 -89.0 -630.5 1.5 -86.0
endline
```

1.1.10 Line_Strip

Definiuje łamaną:



Parametry:

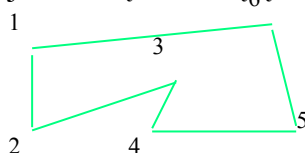
- Color – [r,g,b]
- Points – lista wierzchołków w formacie [x,y,z]

Przykład **zielonej łamanej o grubości 1mm**:

```
node 300 0 none lines 0 255 0 1
-638.0 0.0 -89.0
-638.0 1.7 -89.0
-630.5 1.5 -86.0
endline
```

1.1.11 Line_Loop

Definiuje łamaną zamkniętą:



Parametry:

- Color – [r,g,b]
- Points – lista wierzchołków w formacie [x,y,z]

Przykład **niebieskozielonej linii zamkniętej nigdy nie przezroczystej**:

```
node 300 0 none lines 0 255 128 -1
```

```
-638.0 0.0 -89.0  
-638.0 1.7 -89.0  
-630.5 1.5 -86.0  
endline
```


Obiekty niewidzialne ale należące do klasy **Node**

1.1.12 MemCell

Komórka pamięci, nie jest rysowana ale ma współrzędne X,Y,Z

Parametry:

- Position X,Y,Z: współrzędne komórki pamięci
- Command: początkowa wartość parametru (informacji) tekstowego
- Value1: początkowa wartość parametru liczbowego
- Value2: początkowa wartość drugiego parametru liczbowego
- TrackName: nazwa toru na który oddziałowuje w przypadku zmiany parametrów (można dać **none**)

Przykład:

```
node -1 0 memcell_train3 memcell 1.0 1.0 1.0 Wait_for_orders 0 0
StatAStatC_trk415`endmemcell
```

1.1.13 EventLauncher

Obiekt wyzwalany naciśnięciem klawisza albo o określonej godzinie lub raz na jakiś czas.

Nie jest rysowany ale ma współrzędne X,Y,Z

Parametry:

- Position X,Y,Z: współrzędne wyzwalacza
- Radius: maksymalna odległość obserwatora od obiektu, -1 oznacza brak sprawdzania odległości
- Key: kod klawisza (tylko literowe, none= brak reakcji na klawisze)
- Time: **godzina w formacie hh:mm albo** ze znakiem minus: okresowość wyzwalania w sekundach, zero oznacza brak reakcji czasowej)
- Event1: zdarzenie wyzwalane przy naciśnięciu klawisza gdy SHIFT nie jest naciśnięty albo gdy upłynął określony czas
- Event2: zdarzenie wyzwalane przy naciśnięciu klawisza gdy SHIFT jest naciśnięty opcjonalnie, po słowie *condition*
- MemCell: nazwa komórki pamięciowej
- Parameters – String, Val1, Val2 – wartości którym się muszą równać wartości komórki pamięciowej żeby zdarzenia zostały wysłane do kolejkovania

Przykład - patrz:

scenery/ zwrL34R300M.inc

Zdarzenia czyli:

1.2 Event

Definiuje zdarzenia które służą do sterowania obiektami, np. zmienić sygnał na semaforze czy przełożyć zwrotnice.

Parametry:

- Name – nazwa zdarzenia

Część nazwy może być parametrem, np.

```
event; (P1) sem_anim12 animation 0 kszt2.t3d rotate Ramie01 0 45 0 80 endevent
```

oznacza, jeśli taka deklaracja jest w pliku semkszt2.inc, że jak damy w scenerii wywołanie semkszt2 Raba_A to zdarzenie będzie miało nazwę Raba_A_sem_anim11.

- EventType – typ zdarzenia
- Delay – opóźnienie przy uruchamianiu zdarzenia
- ObjectName – nazwa obiektu do którego odnosi się zdarzenie
- dodatkowe parametry zależne od EventType:

może być ujemne, wtedy wywoływane jest cyklicznie, ale to jest nieprzetestowane

1.2.1 Lights

Zmienia światła w danym przez ObjectName modelu

Parametry:

- Lights – stany świateł obiektu 0-wyłącz, 1-włącz, 2-migające

Przykład:

```
event sem10_light11 lights 0.0 sem10 2 0 0 1 0 endevent
```

1.2.2 Animation

dokonyje rotacji lub translacji fragmentu modelu, nazwa modelu jest w ObjectName

Parametry:

- AnimationType – rodzaj animacji: rotate/translate
- SubModel – nazwa fragmentu modelu podlegającego animacji
- X, Y, Z – wartości kątów lub przesunięć
- AnimationSpeed – prędkość animacji

Przykład:

jeśli model jest osadzony (patrz 1.1.3) w ten sposób:

```
node -1 0 Testowo_A model 100.0 0.2 20.0 180 sem_kszt2.t3d endmodel
```

i jego fragment nazwany jest Ramie01 to rotacja tego ramienia o 45deg wokół osi Y z prędkością 40deg/s definiuje się:

```
event Testowo_A_sem_anim21 animation 0 Testowo_A rotate Ramie01 0 -45 0 40 endevent
```

translacja jeszcze nie działa

TrackVel

Zmienia przypisaną prędkość do toru.

Parametry:

- Velocity – prędkość która zostanie przypisana do toru.

Przykład:

```
event zwr_1_wbok trackvel 0.0 t_zwr_1 40.0 endevent
```

1.2.3 UpdateValues

Ładuje informacje do komórki pamięci.

Parametry:

- Command – łańcuch znaków
- Value1 – jakaś liczba
- Value2 – jakaś druga liczba

Uwaga – jeśli któryś z 3 powyższych parametrów jest * to dany parametr komórki pamięciowej nie zostanie uaktualniony (można selektywnie uaktualniać)

Przykłady:

```
event start3b updatevalues 30.0 memcell_train3 SetVelocity 50 -1 endevent
```

(po 30 sekundach wpisze komendę SetVelocity(50,-1) do komórki memcell_train3)

albo w pliku .inc:

```
event (p1)_sem_info_shunt2 updatevalues 1.0 (p1)_sem_mem ShuntVelocity 40 0
```

(po 1 sekundzie wpisze komendę ShuntVelocity(40,0) do komórki o nazwie (p1)_sem_mem gdzie P1 jest nazwą semafora definiowaną na zewnątrz pliku .inc

a może wykolejnica?

```
event Wk1_1 updatevalues 0 Wk1_status DamageFlag 128 1 endevent
```

wykorzystanie komórki pamięciowej do informacji o drodze przebiegu:

```
event Testowo_Wjazd1-Zaczyniek updatevalues 0.0 Testowo_status1 Wjechał 1 * endevent
```

1.2.4 GetValues

Pobiera informacje z komórki pamięci i wysyła do obiektu *dynamic*.

ObjectName w tym przypadku to nazwa komórki pamięci.

Parametry Command, Value1, Value2 oraz współrzędne komórki pamięci X,Y,Z są przekazywane obiektowi który wywołuje zdarzenie GetValues

Przykłady:

```
event StatAStatC_szlak getvalues 1.0 StatAStatC_szlak_mem endevent
```

(z komórki o nazwie StatAStatC_szlak_mem wysyłana jest jej zawartość do obiektu który wjechał na tor w którym była deklaracja event1 StatAStatC_szlak

albo w pliku .inc definiującym semafor:

```
event (p1)_sem_info getvalues 1.0 (p1)_sem_mem endevent
```

(z komórki o nazwie (p1)_sem_mem wysyła informacje o aktualnej prędkości tego semafora)

1.2.5 PutValues

Wysyła statyczne informacje do obiektu *dynamic* (z pominięciem komórki pamięci)

Przykłady:

```
W9-start.inc  
W9-stop.inc itp
```

1.2.6 Multiple

Pozwala wywołać więcej zdarzeń. ObjectName jest na ogół nieużywane, chyba że na końcu listy zdarzeń będzie parametr condition

Parametry:

- Events – lista zdarzeń do wywołania
opcjonalnie po słowie kluczowym condition:
- ConditionType – typ warunku:
trackoccupied trackfree propability memcompare
dwa pierwsze: wyzwalane są gdy tor określony w ObjectName jest zajęty lub wolny, propability wyzwalane jest jeśli wylosowana liczba jest mniejsza niż parametr z zakresu 0...1,
memcompare wyzwalane jest gdy zawartości komórki pamięciowej określonej w ObjectName równe są podanej liście parametrów:
- Parameters – String, Val1, Val2 (tylko w przypadku memcompare – wszystkie 3 wartości muszą być równe wartościom z komórki pamięciowej, chyba że któreś z nich jest *)

Przykład:

```
event semA_S13 multiple 0 none semA_light13 semA_S13set endevent
```

(wywołuje dwa zdarzenia, jedno ustawia światła na semaforze, drugie definiuje jego prędkość)

albo zdarzenia warunkowe:

```
event przejazd_otwieraj multiple 2.0 tornaprzejezdzie przejazd_1_sygn1  
przejazd_1_sygn2 condition trackfree
```

wyzwalane jest gdy tor o nazwie tornaprzejezdzie jest wolny;

```
event Zaczyniek-Testowo1 multiple 3.0 Testowo-status Testowo-Zatwierdz Testowo-zwr1+  
Testowo_ToA_os2 Testowo_A_S5 Testowo_D_S1 condition memcompare Rozwiazany * *
```

wyzwalane jest jeśli pierwszy parametr komórki pamięciowej Testowo-status jest słowem Rozwiazany

1.2.7 Switch

Zmienia przełożenie zwrotnicy.

Parametry:

- State – stan na który należy przełączyć zwrotnice 0 lub 1

Przykład:

```
event Testowo_zwr1+ switch 0.0 Testowo_zwr1 1 endevent
```

1.2.8 Sound

Odtwarza dźwięk z pliku .wav

Parametry:

- X,Y,Z – położenie dźwięku
- WaveFile – nazwa pliku z dźwiękiem

Przykład:

```
node 100 0 dzwon1 sound -8.0 0.0 4.0 CrossingBell1.wav endsound
```

1.3 TrainSet

Służy do ustawiania składów. Patrz też: dynamic.

Parametry:

- TrainName – nazwa pociągu (taka sama jak nazwa pliku *.txt z rozkładem jazdy)
- Track – nazwa toru na którym ustawiamy skład
- Dist – odległość początkowa
- Vel – prędkość początkowa

Przykład pociągu ciągniętego przez dwie EU07 w trakcji ukrotnionej:

```
trainset PE2307 StatB_track03 170.0 0.0
node -1 0 player_train dynamic PKP\EU07 4E 0.0 headdriver 7 0 enddynamic
node -1 0 player_train dynamic PKP\EU07 4E 0.0 nobody 3 0 enddynamic
node -1 0 5051-503320-2 dynamic PKP\Bipa Bipa-A 0.0 nobody 3 10 Passengers enddynamic
node -1 0 5051-503321-7 dynamic PKP\Bipa Bipa-CD 0.0 nobody 3 25 Passengers enddynamic
node -1 0 5051-503322-1 dynamic PKP\Bipa Bipa-CD 0.0 nobody 3 8 Passengers enddynamic
node -1 0 5051-503323-5 dynamic PKP\Bipa Bipa-B 0.0 nobody 0 9 Passengers enddynamic
endtrainset
```

1.4 Include

Dołącza plik z opcjonalnymi parametrami

Parametry:

- FileName – nazwa pliku do dołączenia
- Parameters – lista parametrów

Ciągi znaków **(p1)**, **(p2)**, **(p3)** ... w pliku dołączanym zostanie zastąpiony odpowiednimi parametrami.

Przykład:

```
include drzewo.inc Pinel.tga 42.0 0.0 5.0 45 5 2 end
```

Plik drzewo.inc wygląda tak:

```
//-----drzewo-----  
//param: tekstura, x, y, z, kat, wysokosc, rozpietosc  
origin (p2) (p3) (p4)  
    rotate 0 (p5) 0  
  
    node 500 0 none triangle_strip (p1)  
    0,0,(p7)          0,0,0 1,1    end  
    0,(p6),(p7)       0,0,0 1,0    end  
    0,0,-(p7)         0,0,0 0,1    end  
    0,(p6),-(p7)      0,0,0 0,0    end  
    0,0,(p7)          0,0,0 1,1    end  
    0,(p6),(p7)       0,0,0 1,0  
    endtri  
  
    node 500 0 none triangle_strip (p7)  
    -(p7),0,0         0,0,0 1,1    end  
    -(p7),(p6),0      0,0,0 1,0    end  
    (p7),0,0          0,0,0 0,1    end  
    (p7),(p6),0       0,0,0 0,0    end  
    -(p7),0,0         0,0,0 1,1    end  
    -(p7),(p6),0      0,0,0 1,0  
    endtri  
  
    rotate 0 0 0  
endorigin
```

1.5 Origin

Przesuwa obiekty o wektor, przykład powyżej.

1.6 Rotate

Obraca obiekty o zadane kąty, przykład powyżej.

1.7 Description

Opis scenerii, tekst pomiędzy Description a EndDescription jest ignorowany przez program eu07.exe, ale przydatny dla innych programów typu loader scenerii itp.

1.8 Inne

1.8.1 Fog

~~Definiuje mgłę, parametry: początek, koniec~~

1.8.2 atmo

Definiuje kolor tła (R,G,B 0...1) oraz mgłę: początek, koniec, kolory R,G, B mgły.

```
atmo 0.5 0.6 1.0 300 1200 0.7 0.8 1.0 endatmo
```

Kolory R,G, B mają być z zakresu 0...1.

1.8.3 light

Definiuje pozycje x,y,z oraz kolor RGB (0...1) składowej ambient, diffuse i specular światła dziennego

```
light -500 500 200 0.5 0.45 0.45 0.50 0.55 0.54 0.50 0.95 0.94 0.90  
endlight
```

1.8.4 Camera

Definiuje pozycję kamery w przypadku trybu freefly, parametry: X,Y,Z oraz kąty.