

SCASM 2.22 - Dynamic Scenery Reference



Printed Under License
From Manfred Moldenhauer.
e-mail: 100117.1465@compuserve.com
M_Moldenhauer@compuserve.com

Assumptions:

 The dynamic scenery section contains several objects which are active (moving) all the time. There is one process for every object (parallel processing). The status of every process is stored in the pattern definition table.
 The number of simultaneously moving objects may be limited but I have not tested the limit.

Notes:

 I've been often asked how to increase the visibility distance of the dynamic objects. It seems the maximum visibility distance is "hard coded" in the FS5 internal object definition and cannot be changed by any command.
 In the BGL instructions the elevation values are usually integer numbers. Only instructions which are usually used for ground operations are using the higher precision fractional number format (noted here as floating point number format).

Commands:

Area15(maxlat minlat maxlon minlon [&])
Area15(maxlat minlat maxlon minlon [maxlat2 minlat2 maxlon2 minlon2])

-
 This defines an area for the dynamic scenery section. All commands for this section must appear within Area() and End15.
 This works similar to the visual (sect 9) Area() command.
 Note: There are two boundaries in this instruction. When the first boundary is entered, the scenery is activated and it is released when leaving the second boundary. Therefore the second boundary has to be larger (or equal). If the second boundary is not defined both are set to the same value. If the '&' character is present, the second boundary is set about 1.5 degrees larger than the first. According to some user reports it seems FS5 can only handle ONE ACTIVE dynamic area. If you have more dynamic areas, define the boundaries very carefully, otherwise FS5 may not switch to new areas.
 In the FS5.0 default world you cannot add dynamic scenery to the following areas, since they are already occupied.
 San Francisco: N35:07:31 - N40:48:21, W124:22:15 - W119:37:29
 Chicago: N38:14:06 - N44:22:52, W90:22:25 - W85:37:39
 For **FS95** it is very important to define a correct second (outer) boundary.

Note: In SCASM 2.07 and later, every
 "maxlat minlat maxlon minlon" group can be replaced by:
a Lat Lon ns-dist ew-dist or
g ns-dist ew-dist

Please see SCADOC.DOC Area16() for more information about this additional parameter format.

End15

This command marks the end of an section 15 area block and tells SCASM to do some clean up work in some instructions.

-> Area15()

LatRef(lat)

-

This command defines a reference latitude. I think it's a good idea to take the same value as for the airport reference point. It seems these value is needed for the correct scaling in east-west direction.

Be sure this value is within the maxlat and minlat boundaries. This command should be the first after the Area15 definition. ATENTION, this value must be within the latitude boundaries defined in the AREA15() command and there should be only one of this command in every Area15() to avoid confusing FS.

Pattern(:Label objectcode traffic_flags mindense)

-

Label	Starting address for commands related to this object.
objectcode	Hexcode or name to define the object. If you want to enter a hex number don't forget the leading 0 ! 0FE0B Cessna 0FE0C Learjet 0FE0D bigjet (727) 0FE0E miljet, (military jet, always light gray) 0FE0F boat (sailboat) 0FE10 truck, always yellow others may cause a crash
traffic_flags	One hex digit. These bits are corresponding to the dynamic scenery option menu. 1 -> air traffic 2 -> aircraft ground traffic 4 -> airport service traffic 8 -> traffic outside airport Recommended values: 3 -> aircrafts 4 -> truck (sevice trucks) 8 -> ships (other trucks)
mindense	minimum scenery density at which this object is shown. Range 0 ... 4 (4 = maximum density)

This command defines an object pattern. It seems these instruction also defines an internal control block for every object. So this is probably something like a object (or pattern) control block.

If you want to awaken a sleeping process you have to refere to this instruction. (a :Label just in front of this Pattern()) DO NOT FORGET to insert the "Exit" command just after the last Pattern() instruction !

It is reported that the **number of objects** in one Area15() is limited to 60 (for FS95?).

Exit

This marks the end of the pattern definition table.
 Maybe this tells FS5 that there are no more pattern processes in this area to start.

SetColor(number colorcode attr)

-
number number of an object detail (0...4), see table
colorcode 2 digit hex code for the color.
attr color attribute code (hex), normal value -> F0

Sets the color of some parts of the selected object. For the color codes see the SCADOC.DOC document.

Attention, it was reported that some color codes (high numbers) can cause crashes in FS95 !

Note: Since **FS98** uses textured dynamic objects the color setting may be different from this table even if the textures are disabled.

From the BGL code in FS98 Dynlib.BGL it is known that some objects do not use the color parameter passed to them!

	Cessna	Learjet	bigjet	miljet	sailboat	truck
0	nose	fuselage	nose	---	---	---
1	fus.up	wings	fuselelage	---	---	---
2	fus.dn	tail	wings	---	---	---
3	tail	jet	tail	---	---	---
4	wings	---	jet	---	---	---

SetPos(Lat Lon alt)

-
Lat latitude
Lon longitude
alt altitude (meters/floating point)
 The altitude setting is always above MSL. It seems the altitude definition point of an object is its bottom. That means you have to add the height of the gear to the ground level (above MSL). If you forgot this your dynamic object may appear somewhere below the ground surface. I calculated the following gear heights by comparing the altitude settings in the different BGL sections:

	FS5-FS95	FS98
Cessna	1.221	1.158
Learjet	1.186	1.138
big jet	2.637	2.544
military jet	1.582	
sail boat	0.0	0.0
truck	1.102	1.102
Helicopter	---	0.733

Note: The maximum total distance an object can be moved is 32767 meters (17,7nm) from this point.

It seems that this limit does not longer exists in FS98.

Heading(degrees)

Sets the heading of an object (i.e. 22.75)

Sleep

Stops any action of this object (pattern process) until it is reactivated (->Awaken) by an other pattern process.

ATTENTION! Be very carefull with this command. Mislplaced Sleep instructions may crash FS5. Never use this as the last instruction. If in doubt place a jump after the sleep even if you never Awaken() this object. -> See example.

Hold(time)

Stops the action of this object for 'time' (integer value) seconds. No Awaken command is needed.

Note: I havn't seen any value greater than 255 in original scenery but it seems that in FS98 larger values will work.

Awaken(:Label)

Reactivates the addressed pattern process. You have to address a label at the required pattern definition entry. Do not try to Awaken() non sleeping objects.

Do not place a Sleep() command immediately after this one. This can loock up all dynamics. If your scenery does not need more movements at this point use at least a Hold(2) command before putting this object to sleep.

-> Sleep, -> Pattern()

Gear(up|down|number)

Moves the gear up or down.

Ignored for objects with fixed gear (as in FS98 Cessna).

Turn(time heading rad)

time time in seconds, integer

heading new heading in degrees (floatung point v.)

The turning is done to the shortest direction.

rad turning radius (meters, integer)

Jump(:Label)

The programm execution is continued at :Label.

Call(:Label)

The subroutine at :Label is called. Programm execution is continued there until a Return command appears. The Return command in the subroutine causes the programm to continue with the next command following the Call() instruction.

-> Return

Return

The programm execution is continued with the instruction immediately following the Call() which calls this subroutine.

-> Call()

Jump..(:Label var dezval)

-
all preliminary! Maybe renamed in future versions.
These are conditional jumps to :Label.
The variable(?) number 4 contains the dynamic scenery density code.
I had no luck when trying to test variables from the normal visual
area.
found variable number(s): 4
found values: 0, 1, 2, 3, 4
Some/all? of these jumps may have a 2nd jump condition. In the
german version of FS5.0 it seems the time of day and the air
traffic control do not influence the jump conditions.

Jump2B(:Label var dezval)

-
condition unknown, probably jump_if_greater_than, GT

Jump2C(:Label var dezval)

-
condition unknown, probably jump_if_equal, EQ

Jump2D(:Label var dezval)

-
This is a guess !
Not found in any BGL file but accepted by FS5. In my test
scenery I found the jump was always performed.
condition unknown

Jump2E(:Label var dezval)

-
condition unknown, probably jump_if_less_or_equal, LE

PitchTo(angle speed)

angle Pitch angle (fp)
+ -> nose down
- -> nose up
speed rotating speed in degrees per second (int)
+ -> move nose down
- -> move nose up

BankTo(angle speed)

angle Bank angle (fp)
+ left wing down, right wing up
- left wing up, right wing down
speed rotating speed in degrees per second (int)
+ -> move left wing down, right wing up
- -> move left wing up, right wing down

YawTo(heading speed)

heading new heading angle (fp)
speed rotating speed in degrees per second (int)
+ rotate to the right side (clockwise)

- rotate to the left side

MoveX(time dist)

time time in seconds for this movement, integer
 dist the distance to move, integer
 + move east
 - move west

MoveY(time dist)

time time in seconds for this movement
 dist the distance to move, + north, - south

MoveZ(time dist)

time time in seconds for this movement
 dist the distance to move, + up, - down

MoveXY(time X_dist Y_dist)**MoveXZ(time X_dist Z_dist)****MoveZY(time Z_dist Y_dist)****Move3D(time X_dist Z_dist Y_dist)**

time time in seconds for this movement, int
 X_dist distance to move, + east, - west
 Y_dist distance to move, + north, - south
 Z_dist distance to move, + up, - down
 int

AccelerateX(time rate X_dist)**AccelerateY(time rate Y_dist)****AccelerateZ(time rate Z_dist)****AccelerateXY(time rate X_dist Y_dist)****AccelerateXZ(time rate X_dist Z_dist)****AccelerateZY(time rate Z_dist Y_dist)****Accelerate3D(time rate X_dist Z_dist Y_dist)**

time time in seconds for this movement, int
 rate this value controls the acceleration.
 Range 15.0000 ... 0.0002
 Often found values are: 2.0, 1.0, 0.5 !
 2.0 increasing speed
 1.0 constant speed
 0.5 decreasing speed
 X_dist distance to move, + east, - west
 Y_dist distance to move, + north, - south
 Z_dist distance to move, + up, - down
 int

D..()

You can also use the D##() binary commands. See -> scadoc.txt

Dynamic Scenery Instructions for FS98

-
 This command must be used just before the first MoveToPos1() or MoveToPos5() command to ensure the correct timing of those commands. It seems that this command must be repeated when a MoveToPos.. sequence is interrupted by any other timer controlled instruction as Move.. (Gear() seem to be allowed).

MoveToPos1(Lat Lon elev pitch bank heading) FS98

-
 This command moves the dynamic object from its current position to this new position within 1 second.

-
elev new elevation in meters (integer)
pitch new pitch angle, - nose up, + nose down
bank new bank angle (floating point number)
 + left wing down, right wing up
 - left wing up, right wing down
heding new heading angle, (fp number)

MoveToPos5(Lat Lon elev pitch bank heading) FS98

-
 This command moves the dynamic object from its current position to this new position within 5 seconds.

TaxiTo(Lat Lon elev heading) FS98

-
 This command moves the dynamic object in a smooth curve and with an slow speed to this final parking position. Usually this position is identical to that one defined in the SetPos() command for this object.

-
elev final elevation in meters, (floating point value)
heading final heading (fp value)

MonitorRwy(:Label Lat Lon elev hdg len wid) FS98

-
 The specified area is monitored. If no other aircraft is in this area a jump to :Label is performed. This command is usually used to check if the runway is free.

-
elev elevation in meters. Since only integer numbers are accepted chose the nearest lower number which fits the runway elevation.
hdg runway heading in degrees, (i.e. 54.37)
len 1/2 runway length in meters (integer)
wid 1/2 runway width in meters (integer)

-
 Example:
 ...
 :Loop

```

    MonitorRwy( :free Lat Lon 120 57.3 800 25 )
    Hold( 2 )      ; let's wait before try again may help
                  ; to save some framerate
    Jump( :Loop ) ; try again
    ;
:free
    ...           ; ok, lets start operation

```

MonitorPos(Lat Lon val1 :Label val2) **FS98**

- preliminary command format, parameters may change !

-
val1 unknown, maybe the radius of the tested area in meters
val2 unknown, maybe an object or density code.
 (found values 1, 2 & 3)

-
 This specific spot is monitored. If your aircraft is not there
 a jump to :Label is performed.
 Also see the Refuel() command.

Refuel(x) **FS98**

-
x maybe the refueling speed in gal/s ???

-
 Move a fuel truck to a position near your aircraft refueling
 point and then use this command. The fuel truck then moves to
 a position in front of your aircraft, stops if your tank is
 not full, and then continues a circle back to the initial
 position of this command.

This command is often found together with MonitorPos().
 See the following code sequence:

```

...
:LAB1
    SetPos( 00:00:00 00:00:00 181.848 )
    Heading( 90.00 )
:LAB2
    MonitorPos( 00:00:00 00:00:00 16 :LAB3 1 )
    Hold( 2 )
    Jump( :LAB2 )

:LAB3
    Hold( 5 )
    MoveX( 12 61 )
    Turn( 3 0.00 10 )
    MoveY( 13 129 )
    Refuel( 60 )
    MoveY( 10 -101 )
...

```

Dbx(42 s)

ACS(s)

FS98

```

-
s      state
        1 = off !!
        0 = ON
-

```

The new FS98 dynamic objects seem to have an Anti Collision System. For some purposes this system can be switched off. In the default scenery this ACS is usually switched off after an take off of an aircraft and it is usually enabled again just before touch down or lowering the gear. Usually it is also switched off for service trucks just before the TaxiTo() command is used to put them back to the parking area. If 2 dynamic objects come too close together with ACS on the whole dynamic scenery can change to an 'static' scenery. The default for ACS seem to be permanently ON (if not generally disabled from the menu system). Maybe ASC OFF will result in a higher frame rate.

CallDLibObj(:Label X Y id0 id1 id2 id3) **FS98**

```

-
This command activates an dynamic object from a library file
and passes some FS internal parameters to it.
The following ID parameters are examples from the FS98 default
dynamic object library named DYNLIB.BGL.
For more information about library objects please see OBJLIB.DOC.
-

```

```

X & Y   Traffic flags like in the Pattern() command
          The ID's are 32 bit hex numbers. These are used to specify
          the library file and the wanted object in this file
id0      00C2CdA9      for DYNLIB.BGL
id1      A9E100AA      for DYNLIB.BGL
id2      596711D0      for Dynlib.BGL
id3      ...           for the following objects in DYNLIB.BGL
          B1A18300      Cessna
          B1A18301      Lear45
          B1A18305      Helicopter, color parts 0 ... 7
          B1A18307      G_Truck 1 (gas truck)
          B1A18308      G_Truck 2 (last object, color parts 0 .. 7)

```

D_35(:Label X Y <Footprint>) **obsolete !**

```

-
This command is replaced by CallDLibObj()
This is not the final implementation of that command!
The new dynamic objects are defined in the file DYNLIB.BGL
and are identified by an long byte string which I will call
its fingerprint.
-

```

```

X & Y   Traffic flags like in the Pattern() command
Footprints (replace the ??):
          A9 CD C2 00 AA 00 E1 A9 D0 11 67 59 ?? 83 A1 B1
??      00 = Cessna
          01 = Lear45
          05 = Helicopter, color parts 0 ... 7

```

```

07 = G_Truck 1
08 = G_Truck 2 (last object, color parts 0 .. 7)

```

```
D_40( Lat Lon elev flag vall hdg len wid ) FS98
```

```

-
elev    elevation in meters (integer)
flag    unknown, in the default scenery always 0 or 1
vall    unknown, it seems this is the speed of the
           aircraft in meters per second.
           1 kts = 0.5144 m/s
hdg    heading (fractional numbers accepted)
len     1/2 of the length in meters (integer)
wid     1/2 of the width in meters (integer)
-

```

The purpose of this instruction is unknown.
This instruction is found in the landing pattern of an
dynamic object aircraft just before lowering the gear.
The position is abt. the middle of the runway. The
parameters **hdg**, **len** and **wid** matches pretty close to the
runway size.

This is an typical code sequence:

```

...
MoveXY( 67 -2689 273 )
PitchTo( 3.84 1.64 )
Move3D( 73 -2618 -184 266 )
D_40( 39:17:21 -94:42:24 532 0 36 -85.24 705 25 )
Gear( 1 )
Move3D( 18 -655 -46 67 )
PitchTo( 0.00 -1.64 )
Move3D( 9 -297 -12 30 )
AccelerateXY( 37 0.5000 -827 87 )
Return

```

```

-----

; =====
;
; This is a minimal dynamic scenery file.
; Helgoland / Duene
;
Header( 1 54:20 54:00 8:20 7:30 )
LatRange( 54:00 54:20 )
Area15( 54:20 54:00 8:20 7:30 )
      LatRef( 54:11:13 )
:Cessna_0
      Pattern( :CS0 Cessna 3 0 )
:Cessna_1
      Pattern( :CS1 Cessna 3 0 )
Exit ; Do not forget this command !

```

```
:CS0
; parking Cessna
SetColor( 0 7 )
SetColor( 1 7 )
SetColor( 2 4 )
SetColor( 3 4 )
SetColor( 4 4 )
SetPos( 54:11:19.7494 7:55:01.1519 4.221 )
Heading( 20 )
Sleep                               ; sleep forever
Jump ( :CS0 )
```

```
:CS1
SetColor( 0 5 )
SetColor( 1 5 )
SetColor( 2 4 )
SetColor( 3 4 )
SetColor( 4 4 )
SetPos( 54:11:17.2821 7:54:56.1099 4.221 )
Heading( 33.1 )
Sleep                               ; sleep (parking) forever
Jump( :CS1 )                        ; <- important
```

End15

For more examples please see the sources in SCXMPL.ZIP (distributed as a separate file).