

# TopSky plugin for EuroScope

---

- *version 2.6* -

Developer Guide

## Table of Contents

<b>1 EuroScope setup .....</b>	<b>3</b>
1.1 General Settings .....	3
1.1.1 Display options .....	3
1.1.2 TAG display options .....	3
1.1.3 Miscellaneous options.....	3
1.2 Display settings dialog .....	4
1.3 Symbology dialog.....	4
1.3.1 Colors.....	4
1.3.2 Symbols.....	5
1.4 Plug-ins Dialog .....	5
1.5 Conflict Alert Settings Dialog .....	5
1.5.1 STCA Lower altitude .....	5
1.5.2 STCA Higher level.....	5
1.5.3 MTCA options .....	5
1.6 Plugin data files and internal settings .....	6
1.7 Tag families and flight lists setup .....	7
1.7.1 Tag families.....	7
1.7.2 Flight lists.....	7
1.7.3 CPDLC.....	8
1.7.4 DCL/PDC.....	9
1.7.5 Oceanic .....	10
<b>2 Plugin data files .....</b>	<b>11</b>
2.1 TopSkyAirspace.txt .....	14
2.2 TopSkyAreas.txt.....	20
2.2.1 STCA area settings .....	30
2.2.2 TCT area settings .....	31
2.2.3 AUP activation .....	34
2.2.4 Remote activation .....	35
2.3 TopSkyCPDLC.txt.....	36
2.4 TopSkyFAST.txt .....	41
2.5 TopSkyMaps.txt & TopSkyMapsLocal.txt .....	45
2.6 TopSkyMSAW.txt.....	66
2.6.1 MSAW areas .....	66
2.6.2 APM areas.....	68
2.7 TopSkyRadars.txt .....	70
2.8 TopSkySettings.txt & TopSkySettingsLocal.txt .....	73
2.9 TopSkySSRcodes.txt.....	74
2.9.1 Code 1000 assignment .....	83
2.10 TopSkySTCA.txt .....	84
2.11 TopSkySymbols.txt.....	87
2.12 TopSkyViews.txt .....	90
2.13 ICAO_Aircraft.txt .....	90
2.14 ICAO_Aircraft.json .....	91
2.15 ICAO_Airlines.txt & ICAO_Airlines_Virtual.txt.....	92
2.16 ICAO_Airports.txt .....	92
2.17 isec.txt .....	92
<b>3 Sound files.....</b>	<b>93</b>
<b>4 Cursor files .....</b>	<b>93</b>
<b>5 External access .....</b>	<b>93</b>
<b>Appendix 1: Label functions .....</b>	<b>94</b>

# 1 EuroScope setup

This chapter is meant as a guide for users who either didn't get the plugin as a part of a package including all the settings files, or who want to create their own setup.

The plugin requires the [MSVC++ Redistributable package](#) to be installed. Without it, the plugin will either fail to load at all, or crash while trying to load.

To load the plugin and setup EuroScope to make the most out of it, use the following settings in the EuroScope settings dialogs. Settings not mentioned shouldn't make any difference. Remember to save the settings when exiting ES. Use of the plugin file must be saved to each profile file separately for it to be loaded automatically.

## 1.1 General Settings

### 1.1.1 Display options

- |                                   |     |
|-----------------------------------|-----|
| • Show route when accepting       | Off |
| • Lock show route when accepting  | Off |
| • Rotate radar target symbol      | Off |
| • Rotate flight plan track symbol | Off |
| • Show CLAM warnings              | Off |
| • Show RAM warnings               | Off |

The plugin uses its own route display and the default plugin track labels don't offer any way of hiding the ES route display so the first two should be selected off. If both settings are left "on" and the track labels are not modified, the radar screen will soon be filled with aircraft routes with no way to remove them. The plugin also has its own CLAM and RAM functions.

### 1.1.2 TAG display options

- |  |     |
|--|-----|
| • Allow correlated aircraft tag untagged | On  |
| • Allow concerned aircraft tag untagged  | Off |
| • Allow assumed aircraft tag untagged    | Off |
| • Show detailed over untagged            | On  |

These settings are needed to display the correct type of track label for each aircraft.

### 1.1.3 Miscellaneous options

- |   |                |
|---|----------------|
| • Keep scratch pad content after direct | On             |
| • Highlight ASEL AC in lists            | Off (see note) |

The plugin stores the manual alerts in the scratch pad. If the first setting is selected off and a direct clearance is given, any manual alerts are removed as a result.

If the second setting is selected on, the "Toggle Route draw (with autohide)" tag function will not work in flight lists for any flights that have a label displayed on the radar screen.

## 1.2 Display settings dialog

- |                                 |     |
|---------------------------------|-----|
| • Number of history dots        | 0   |
| • Do not display aircraft below | 0   |
| • Do not display aircraft above | 0   |
| • Show leader lines             | Off |

The plugin draws its own custom history dots and leader lines so the ES default ones must be selected off. Similarly, the plugin has its own altitude filtering settings, so the ES filters should be off.

## 1.3 Symbology dialog

### 1.3.1 Colors

Whether to use the transparency settings is up to the user, and their colors should be set as desired. The plugin colors determine the track label color but the EuroScope defined color is used to draw the leader line between the position symbol and the label. Some EuroScope colors have a corresponding plugin color setting as listed below. They should be set to the same value.

<u>EuroScope color</u>	<u>Plugin color</u>
------------------------	---------------------

Aircraft colors:

- |                                    |                        |
|------------------------------------|------------------------|
| • Datablock/non concerned          | Color_Unconcerned      |
| • Datablock/notified               | Color_Coordination     |
| • Datablock/assumed                | Color_Assumed          |
| • Datablock/redundant              | Color_Redundant        |
| • Datablock/information            | Color_Warning          |
| • Datablock/detailed background    | Color_Flight_Highlight |
| • Datablock/active item background | Color_Field_Highlight  |

List colors

- |   |                        |
|---|------------------------|
| • Datablock/AC list background          | Color_Background       |
| • Datablock/AC list selected background | Color_Flight_Highlight |
| • Other/list header                     | Color_Foreground       |

Background colors:

- |                                     |                       |
|-------------------------------------|-----------------------|
| • Sector/active sector background   | Color_Active_Sector   |
| • Sector/inactive sector background | Color_Inactive_Sector |

If adjusted from the default ones, the background colors should be chosen so that all plugin colors can be seen. Medium grey colors work well with the default plugin colors (darker colors with the “COOPANS” version activated), but anything can be used as long as the plugin colors are taken into account and changed accordingly.

“Other/normal menu item” and “Other/disabled menu item” should be different colors to be able to see disabled menu items in the setup menus.

### 1.3.2 Symbols

The “Aircraft primary radar only”, “Aircraft flight plan track”, “Aircraft coasting”, “History dot” and all entries beginning with “Aircraft corr.” or “Aircraft uncorr.” are drawn by the plugin and should be set to “MOVETO 0 0” only. All other symbols are drawn by ES and should be set as desired.

## 1.4 Plug-ins Dialog

Load the plugin file (TopSky.dll). After loading it, highlight its name (“TopSky plugin”) in the list and move “Standard ES radar screen” from the “Forbidden to draw on types” box to the “Allowed to draw on types” box.

## 1.5 Conflict Alert Settings Dialog

The plugin has its own STCA and MTCD systems and doesn't use the ES default ones. Selecting the below settings will disable the warnings from the ES default systems.

### 1.5.1 STCA Lower altitude

- Show lower altitude STCA Off

### 1.5.2 STCA Higher level

- Show higher level STCA Off

### 1.5.3 MTCA options

- Show MTCA Off
- Warn if vertical separation is less 0 feet
- Warn if horizontal separation is less 0 nm

## 1.6 Plugin data files and internal settings

It is possible to adjust hundreds of different settings using the [settings files](#). Even though the ranges of acceptable values have been limited to prevent major problems, care should be taken when adjusting any settings as the results can be unexpected.

When creating a new setup, only the **ICAO** data files can be used from an old setup, as they contain only general data that can be used anywhere. All the other data files contain information specific to the original setup, and should not be used.

Some setup-specific plugin settings are listed below. At least these settings should be looked at when creating a new setup, but many of the others are important too. However, never change a setting without understanding what it does.

Setup_COOPANS	Specify if the “COOPANS” version of the plugin is to be activated
Airspace_C_Flag	Define the list of clearance flag airports
Airspace_C_No_Flag	
Airspace_P_Flag	Define the list of P-RNAV airports (applicable to “COOPANS” version only)
Airspace_P_No_Flag	
Airspace_ASSR_Type	Define the transponder code ranges
Airspace_ASSR_StartCode	
Airspace_ASSR_EndCode	
Airspace_No_DUPE_Codes	Define the transponder codes that do not produce a DUPE alert
Airspace_SIGMET_Areas	Define airspace information for the SIGMET/NOTAM functions
Airspace_NOTAM_Add	
Airspace_NOTAM_Remove	
Airspace_AOR_Airports	Define airport identifiers in the AOR
Airspace_AOR_Airports_Exclude	
Airspace_AOR_Positions	Define controller login callsigns in the AOR
Airspace_AOR_Positions_Exclude	
HTTP_GRIB_Latitude_Max	Define area to download upper wind and temperature data
HTTP_GRIB_Latitude_Min	
HTTP_GRIB_Longitude_Max	
HTTP_GRIB_Longitude_Min	
WXR_Latitude	Define the weather radar tile center point
WXR_Longitude	(for performance reasons, look into the ..._Min and ..._Max settings as well)

## 1.7 Tag families and flight lists setup

To make the plugin work as it's supposed to, there are some rules that need to be followed when creating the tag families and flight lists. Label item and color descriptions can be found in the General part of the manual, function descriptions in Appendix 1 of this document.

### 1.7.1 Tag families

Items beginning with “**List**” should not be used in tag definitions (“List” items are not hidden when a label is subject to filtering, the other ones are). Tags should be constructed only from TopSky plugin items. If this is not possible, the plugin’s label filtering functionalities may not work as they can only toggle the visibility of its own tag items

#### 1.7.1.1 General

- For every correlated and flight plan track tag type:
  - The *untagged* and *tagged* tags must contain the “**Dummy item – not detailed tag**” item
  - The *detailed* tag must contain the “**Dummy item – detailed tag**” item
- Tag items marked “**(0-1)**” are only displayed when the track is unconcerned or notified, items with “**(2+)**” when the track is in any other state. Other items are shown in all states.
- A couple of the indicator items have an “**(inactive)**” version as well. Both can be used in the same tag, as only one of them will be shown (the inactive version can then be used to toggle the indicator back to active state if required).

#### 1.7.1.2 Tagging levels

- When a tag item has an “**(unselected track)**” variant, it should be used in the *untagged* and *tagged* tagging levels, and the normal variant in the *detailed* tagging level. For tag items without specific variants, the normal variant can be used in all tagging levels.
- Items beginning with “**ET**” should be used in the *detailed* tagging level (they are only displayed in the extended label which is a sublevel of the *detailed* tagging level in the plugin, displayed by the “**Open extended tag**” function). Items with “**(not ET)**” will not be shown in the extended label. Other items are displayed in both the normal *detailed* tag and the extended label.

### 1.7.2 Flight lists

Of the plugin tag items, only the ones beginning with “**List**” should be used in flight lists (“List” items are not hidden when a label is subject to filtering, the other ones are).

The “**List\_CALLSIGN**” item is used by the plugin for display of flight legs. If that item is not set up to be used in the flight lists, adjust the plugin’s “**FlightLeg\_Refresh\_TagItem**” setting value to another item that is used.

The items beginning with “**List Sorting Item –**” are special items to allow correct sorting of the respective lists (the sorting order is selected via the Global Menu). To use this sorting capability, enter the item somewhere in the list definition, set the list to sort according to the item (left-click on the column header), and hide the column as it never displays any useful data. Then save the list definition in EuroScope.

### 1.7.3 CPDLC

The following tag items (all variants – List, ET, etc. unless otherwise specified) and functions have CPDLC related features, and should be incorporated in the relevant tag definitions and flight lists to provide the intended functionality when the plugin's datalink features are used.

#### 1.7.3.1 Tag items

- AHDG                      Displays heading uplink status
- ASSR                      Displays squawk code uplink status
- CALLSIGN                 Displays CPDLC connection status
- CFL                       Displays level uplink status
- CPDLC\_E                 Displays received CPDLC emergency messages
- CPDLC\_W                 Displays message and connection status messages
- NPT                       Displays direct-to downlink status
- NSSR                      Displays squawk code uplink status
- PFREQ                    Displays pilot monitored frequency after “monitor” uplink has been used
- PRFL                      Displays level downlink status (non-COOPANS version only)
- RFL                       Displays level downlink status (COOPANS version only)
- List RFL                 Displays level downlink status
- SI                         Displays frequency change uplink status

#### 1.7.3.2 Tag functions

- Acknowledge PFREQ                      Acknowledges the information  
(use with PFREQ tag item)
- CPDLC Warning functions               Acknowledges warning or opens a menu  
(use with CPDLC\_W tag item)
- Open AHDG menu                         Used to send/answer heading uplinks/downlinks  
and send direct-to uplinks
- Open ASP menu                          Used to send/answer speed uplinks/downlinks
- Open Callsign menu                     Contains many CPDLC related items
- Open CFL menu / Open CFL/PEL menu                      Used to send/answer level uplinks/downlinks
- Open CPDLC Current Message Window (\*)                      Opens the window  
(use with the corresponding items, except RFL in the non-COOPANS version: use with “List RFL” and “PRFL” items)
- Open CPDLC Emergency Acknowledgement menu              Opens the menu (use with CPDLC\_E tag item)
- Open Waypoint menu                     Used to send/answer direct-to uplinks/downlinks
- Send CPDLC squawk SSR                 Sends squawk code uplink

## **1.7.4 DCL/PDC**

The following tag items and functions have DCL/PDC related features, and should be incorporated in the relevant flight lists to provide the intended functionality when the plugin's datalink features are used.

### **1.7.4.1 Tag items**

- 1) List CLR/DCL/CMT      Displays DCL/PDC request status if relevant, otherwise the clearance flag
- 2) List CMT                  Displays whether a received clearance request contained free text
- 3) List DCL                  Displays DCL/PDC request status

Item 1 or 3 must be used to be able to see the request status. Item 2 may be used together with item 3 if desired.

### **1.7.4.2 Tag functions**

- Open CMT Pop-up              Opens a pop-up displaying the free text
- Open DCL Window              Opens the DCL Window used to send DCL/PDC
- Open DCL Window/PDC Window      Opens the DCL Window used to send DCL/PDC, or the PDC Window if no datalink clearance request received

## 1.7.5 Oceanic

The following tag items (all variants – List, ET, etc. unless otherwise specified) and functions have Oceanic related features, and should be incorporated in the relevant tag definitions and flight lists to provide the intended functionality when the plugin's oceanic features are enabled (setting "System\_Oceanic=1").

### 1.7.5.1 Tag items

- + Displays indicator for a time restriction present in the OCM
- List CTO Displays cleared oceanic entry time
- List MNR Displays cleared Mach number
- List NAT Displays cleared track designator
- List NBT Displays time restriction
- List NLT Displays time restriction
- O Displays indicator for non-acknowledged OCM
- List OAN Displays cleared oceanic entry point
- OFL Displays cleared oceanic cruising level

### 1.7.5.2 Tag functions

- Acknowledge OCM Acknowledges the received OCM
- Acknowledge OCM/Toggle OFL Highlight If OCM not acknowledged: acknowledges it  
Otherwise: toggles the highlight
- Acknowledge/clear OCM NBT If NBT not acknowledged: acknowledges it  
Otherwise: clears it
- Acknowledge/clear OCM NLT If NLT not acknowledged: acknowledges it  
Otherwise: clears it
- Open Oceanic Time Restriction Window Opens the window
- Toggle Oceanic Level Highlight Toggles the highlight

## 2 Plugin data files

This chapter gives guidance on developing the data files used by the plugin for various features. The data files must be located in the same folder as the plugin dll. Even though the plugin does its best to check the data for errors, some errors may get through and cause all kinds of issues, possibly leading to ES crashing, so it's important to be careful to provide correctly formed data when creating the files. Errors found in the data files by the plugin can be seen in the "Plugin Status" submenu. It is found in the "Status" or "STS" menu depending on the plugin version.

To reload a data file, left-click on the "Reload" button. Some of the data can also be visually checked for correctness by left-clicking the "View" button. The areas will then be shown on the radar screen.

### *Coordinate formats*

The following coordinate formats can be used to define a <position>, <latitude> or <longitude>:

#### <position>

Position defined in the active sector file

- Id Fix, VOR, NDB or airport identifier
- AirportId/rwyId Runway threshold

Position defined according to ARINC 424 paragraph 7.2.5 rules ( [L] is N, E, S or W )

- [L]dddd , d[L]ddd , dd[L]dd or dddd[L] Full or half degrees of latitude, full degrees of longitude
- Hddd Same as Nddd (i.e. H6030 = N6030 = 60°30'N 030°00'W)

Latitude/longitude coordinates ( first [L] is N or S, second [L] is E or W )

- dd[L]ddd[L] Full degrees of latitude and longitude
- [L]dd[L]ddd Full degrees of latitude and longitude
- ddmm[L]dddmm[L] Full degrees and minutes of latitude and longitude
- [L]ddmm[L]dddmm Full degrees and minutes of latitude and longitude
- ddmmss[L]dddmmss[L] Full degrees, minutes and seconds of latitude and longitude
- [L]ddmmss[L]dddmmss Full degrees, minutes and seconds of latitude and longitude

*Note : Pad all formats with zeroes as necessary to get correct number of digits (two for latitude degrees, three for longitude degrees and two for all minutes and seconds), The smallest unit – whether it's degrees, minutes or seconds – may have a decimal value.*

#### <latitude> or <longitude> ( [L] is N, E, S or W )

- d.m.s[L] or [L]d.m.s Degrees, minutes and seconds (sector file format)
- +d , -d or d Degrees
- d[L] , [L]d , d°[L] or [L]d° Degrees
- d°m'[L] or [L]d°m' Degrees and minutes
- d°m's"[L] or [L]d°m's" Degrees, minutes and seconds
- dddmm[L] or [L]dddmm Degrees and minutes (1)
- ddmmss[L] or [L]dddmss Degrees, minutes and seconds (1)

*Note 1 : Pad these formats with zeroes as necessary to get correct number of digits (two for latitude degrees, three for longitude degrees and two for all minutes and seconds)*

*Note 2 : In all the above formats, the smallest unit – whether it's degrees, minutes or seconds – may have a decimal value.*



## 2.1 TopSkyAirspace.txt

The file contains the QNH value to transition level tables for the QNH/TL Window, any custom runway approach lines (i.e. starting point not on the runway threshold and/or approach course not aligned with the runway centerline), lists of fixes for the “no fix” warning, lists of custom level menu values, definitions for transition altitude areas, pre-defined initial cleared levels and custom approach clearance types.

For the QNH/TL tables, the file is read one line at a time and the first matching line will be used. The following example shows a possible setup:

// Made up lines...	Comment
QNHTL:XBZZ:80	QNHTL
QNHTL:XB,XC:80,978,75,996,70	QNHTL

For the runway approach lines, any lines defined here will override the corresponding lines automatically created from the sector file data. The following line shows an example:

APPLINE:EFHK:15:153.4:HEL	AppLine
---------------------------	---------

For the custom level lists, the file is read one line at a time and the first matching line will be used. The following example shows a possible setup:

LEVELS:EETN:22,25,32,35,50	LEVELS
LEVELS:*:15,20,30	LEVELS

For the transition altitude areas, the file is read one line at a time and the first matching line will be used. The following example shows a possible setup:

TA_CIRCLE:8000:54.2:11.5:50.0	TA_Circle
TA_POLYGON:7000:54.0:10.0:54.0:20.0:52.0:20.0:52.0:10.0	TA_Polygon

## QNHTL

### QNHTL:ICAOlist:TL

**QNHTL:ICAOlist:TL<sub>1</sub>,QNH<sub>1</sub>,TL<sub>2</sub>,QNH<sub>2</sub>, ... ,TL<sub>n</sub>,QNH<sub>n</sub>,TL<sub>n+1</sub>**

Defines a variable transition level based on the QNH value for the specified airports.

- ICAOlist              Comma-separated list of airport ICAO designators. Either complete designators or one to three first letters of the designator.
- TL<sub>n</sub>                  Transition level value (text string, will be displayed exactly as written)
- QNH<sub>n</sub>                QNH value (integer value, see also below)

The first definition sets a fixed transition level, while the second one creates a table of transition levels based on QNH values.

The QNH value must be in the same format as in the METAR. If the METAR reports the QNH in hPa (Qxxxx), the values in the list must be in hPa as well. If the METAR reports the QNH in inHg (Axxxx), the values in the list must be in inHg\*100.

The TL/QNH list must contain one or more pairs of TL and QNH values followed by a TL value in the end.

The transition level is found in the following way: the actual QNH value found in the METAR is compared against the values in the list, from left to right. If the actual QNH is less than the list value, the corresponding TL value (the one before the QNH value) is used. If not, the next QNH value in the list is checked and so on. If the actual QNH value is equal to or greater than all the values in the list, the TL value after the last QNH value in the list ( $TL_{n+1}$ ) is used.

## *AppLine*

**APPLINE:AirportCode:Runway:AppCourseT:PointName**

**APPLINE:AirportCode:Runway:AppCourseT:Lat:Lon**

Defines a runway approach line with a specified approach course and end point (either as a point name or coordinates). By default, approach lines are created for all runways in the active sector file with the approach courses set on the extended runway center lines and ending at the runway thresholds. If a runway needs some other approach line instead, it can be defined here. Any runway approach line defined here will override the plugin created default one for the same runway.

- AirportCode      Airport ICAO code
- Runway          Runway identifier
- AppCourseT      Approach track (degrees true, decimal value)
- PointName        Fix, VOR, NDB, airport code or runway (must be found in the active sector file)
- Lat              Endpoint <latitude>
- Lon              Endpoint <longitude>

*Note: the syntax to define a runway threshold as a PointName is the 4-letter ICAO airport designator followed by a forward slash and the runway identifier.*

## *Levels*

**LEVELS:AirportCode:LevelList**

Defines a custom level list to be used in all plugin level menus (AFL, CFL and RFL). The list here overrides the default list values up to the highest value in the custom list, which after the default list takes over. The airport code is compared against the login callsign, and if a matching list is found, it will be used. In case a matching list is not found, the code then compares the lists against the aircraft's departure airport until more than 1/3 of the way to the destination, and then against the destination airport.

- AirportCode      Airport ICAO code. "\*" matches any airport, other wildcards are not supported
- LevelList        Comma-separated list of levels (in 100's of feet, range 1-999)

## *TA\_Circle*

### **TA\_CIRCLE:TransAlt:Lat:Lon:Radius**

Defines a transition altitude area as a circle with a defined center point and radius. When within this area, the default EuroScope transition altitude will be overridden with the value defined here. The EuroScope transition altitude should still be set to a value representing the airspace as it is used everywhere outside the defined areas, and in parts of the code where the current aircraft position is not available or relevant.

- TransAlt              Transition altitude (feet, range 0-99999)
- Lat                    Center point <latitude>
- Lon                    Center point <longitude>
- Radius                Radius (nautical miles, range 0.1-9999.9)

## *TA\_Polygon*

### **TA\_POLYGON:TransAlt:Lat<sub>1</sub>:Lon<sub>1</sub>:Lat<sub>2</sub>:Lon<sub>2</sub>:Lat<sub>3</sub>:Lon<sub>3</sub>:...**

Defines a transition altitude area as a polygon. When within this area, the default EuroScope transition altitude will be overridden with the value defined here. The EuroScope transition altitude should still be set to a value representing the airspace as it is used everywhere outside the defined areas, and in parts of the code where the current aircraft position is not available or relevant.

- TransAlt              Transition altitude (feet, range 0-99999)
- Lat<sub>n</sub>                Polygon vertex <latitude>
- Lon<sub>n</sub>                Polygon vertex <longitude>

## *No\_Fix*

### **NO\_FIX:IcaoList:FixList**

Lists mandatory fixes in the flight plan for specific destinations. More than one *No\_Fix* line can be used to set up multiple airports, but information for one airport may not be present in more than one line.

- IcaoList              Comma-separated list of aerodrome location identifiers
- FixList                Comma-separated list of waypoint/navaid identifiers

### *Initial\_CFL*

## **INITIAL\_CFL:Value**

Starts a new initial cleared altitude rule definition. If no CFL has been set, this value will be used as the default value in the CFL menu when opened using the “Open CFL menu (DEP List)” function, and from the PDC Window. Additionally, the PDC Window may be set up to automatically assign the value when opened. To set restrictions to where this value is used, *CFL\_\** lines (see below) may be used after this line. The rule definitions are checked in the order they are in the file, and the first one matching will be used.



CFL\_FRUL

## CFL\_FRUL:FlightRules

Restricts the preceding rule definition to flights with the specified flight rules.

- FlightRules “I” or “V”

CFL-ADEP

CFL\_ADEP:Adep<sub>1</sub>:Adep<sub>2</sub>:Adep<sub>3</sub>:

Restricts the preceding rule definition to flights departing from the specified airports. "AB\*" will match any airport whose code begins with "AB". No other type of wildcard use is supported.

- Adepx Departure airport ICAO code

CFL\_DRWY

**CFL\_DRWY:RwyId<sub>1</sub>:RwyId<sub>2</sub>:RwyId<sub>3</sub>:**

Restricts the preceding rule definition to flights departing from the specified runways.

- **RwyId<sub>x</sub>** Departure runway identifier

## CFL DRWY ISACTIVE

## **CFL\_DRWY\_ISACTIVE:State**

Restricts the preceding rule definition based on the activity state of the aircraft's departure runway.

- State                      The activity state of the runway
    - “1”                      runway must be active for departures
    - “0”                      runway must not be active for departures

## *CFL\_SID*

### **CFL\_SID:Sid<sub>1</sub>:Sid<sub>2</sub>:Sid<sub>3</sub>:...**

Restricts the preceding rule definition to flights with the specified SIDs.

- Sid<sub>x</sub> SID identifier
  - “ABC1A” matches with SID ABC1A
  - “ABC\*” matches any SID beginning with “ABC”
  - “none” must not have a SID assigned
  - “\*” must have a SID assigned (any SID will match)

## *CFL\_AHDG*

### **CFL\_AHDG:Ahdg<sub>1</sub>:Ahdg<sub>2</sub>:Ahdg<sub>3</sub>:...**

Restricts the preceding rule definition to flights with the specified assigned headings.

- Ahdg<sub>x</sub> Assigned heading value
  - “DDD” assigned heading value in degrees (001-360)
  - “LLL-RRR” assigned heading range, clockwise from LLL to RRR degrees
  - “none” must not have an assigned heading
  - “\*” must have an assigned heading (any heading will match)

## *CFL\_VIA*

### **CFL\_VIA:Point<sub>1</sub>:Point<sub>2</sub>:Point<sub>3</sub>:...**

Restricts the preceding rule definition to flights routing via one of the specified points.

- Point<sub>x</sub> Point name

## *CFL\_ENGTYP*

### **CFL\_ENGTYP:EngineTypes**

Restricts the preceding rule definition to aircraft with the specified engine types.

- EngineTypes List of allowed engine types (one or more of the following characters):
  - E (Electric), J (Jet), P (Piston), R (Rocket) or T (Turboprop/turboshaft)

## *CFL\_ENGCOUNT*

### **CFL\_ENGCOUNT:EngineCounts**

Restricts the preceding rule definition to aircraft with the specified number of engines.

- EngineCounts List of allowed numbers of engines (one or more of the following characters):
  - 1-8 or C (Two engines coupled to drive a single propeller system)

## *Approach clearance type*

### **APPCLR:Icao:Rwy:AppTypes**

This line type defines the additional approach types available for selection in the CFL menu. It is possible to define runway-specific, airport-specific and global sets. When the CFL menu is opened, the code first tries to find a runway-specific set for the aircraft's arrival runway, then an airport-specific set for the arrival airport, and finally a global one. If none are found, no additional approach types will be selectable.

- Icao                      Airport ICAO code (set both Icao and Rwy to "\*" to define a global set)
- Rwy                      Runway identifier (set Rwy to "\*" to define an airport-specific set)
- AppTypes                Comma-separated list of approach types

Each approach type must be 1-4 characters long. The following types have special meaning:

- CAT2                      CAT 2 approach clearance, taken into account by the FAST function
- CAT3                      CAT 3 approach clearance, taken into account by the FAST function
- OS                        Own separation clearance, taken into account by the FAST function, and considered as a visual approach clearance by MSAW and APM safety nets (included in the CFL menu by default in the COOPANS version)

Other approach types have no other function than displaying the text as the approach clearance value.

## 2.2 TopSkyAreas.txt

This file contains the areas for the APW and SAP functionality as well as the MTCD, STCA and CLAM/RAM inhibit areas, STCA areas, TCTAs and TCT inhibit areas, and FPCAs for the MTCD function. The following example area is used to show the syntax (optional lines in grey color).

CATEGORYDEF:D:7:0:5:0:0:0	CategoryDef
// EF D101 Isosaari	Comment
AREA:T:EFD101	Name
CATEGORY:D	Category
ACTIVE:NOTAM:EFIN:EF D101	Active
ACTIVE:NOTAM:EFIN:EFD101	Active
LABEL:N059.55.08.817:E025.07.08.496:D101	Label
LIMITS:0:390	Limits
N059.54.15.000 E025.15.06.000	Coordinate
N059.53.27.000 E024.59.49.000	Coordinate
N059.56.36.000 E025.10.10.000	Coordinate

Whitespace and tab characters will be automatically stripped from the beginnings of lines, and any lines with a first non-whitespace/tab character being “{”, “}”, “;” or “/” are disregarded.

### *File version*

**//VERSION:VersionString**

When an URL has been defined to download areas data, the *file version* line is used to check if the downloaded data is newer than the existing data, and should replace it. To display the version string in a label, use “<VersionString>” without the quotes anywhere where a text string is to be drawn, the code will replace it with the active version string.

- If not present in either file, the downloaded data is used
- If present in the existing data only, the existing data is used.
- If present in the downloaded data only, the downloaded data is used
- If present in both files, the VersionStrings are compared, and the downloaded data is used if its VersionString is greater than the existing data's. Note that the comparison is done one character at a time, so “9” is greater than “10”, but “09” is less.

## *CategoryDef*

**CATEGORYDEF:Name:ActBorderStyle:ActFillColor:ActFillPattern:PreBorderStyle:PreFillColor:PreFillPattern**

This line starts an area category definition and defines its display colors. The default border color is *Active Map* for unfilled and *Active RD Map* for filled areas. The default fill color is *Active RD Infill Map*, and the fill percentage that was used in previous plugin versions is 50.

- Name Name for the category (text string)
- ActBorderStyle Active area border style, either “color” or “color/width”, where color is:
  - 0 use default color
  - 1-20 custom color (*Active Map Type X*)
- ActFillColor Active area fill color (values as in ActBorderColor)
- ActFillPattern Active area fill pattern
  - 0 no fill
  - 5, 10, 20, 25, 30, 40, 50, 60, 70, 75, 80, 90, 100 percentage to fill
  - E0 – E52 hatch fill
- PreBorderStyle Pre-active area border style (values as in ActBorderStyle)
- PreFillColor Pre-active area fill color (values as in ActFillColor)
- PreFillPattern Pre-active area fill pattern (values as in ActFillPattern)

If the line width is not defined in ActBorderStyle and PreBorderStyle, it defaults to 1.

The hatch fill values correspond to the GDI+ HatchStyle enumeration values. For example “E0” sets “HatchStyleHorizontal” and “E6” sets “HatchStyle05Percent” which can also be achieved by “5” (the numeric values for the FillPattern are just shortcuts to the percentage hatch styles)

## *CategoryDef\_Label*

**CATEGORYDEF\_LABEL:Name:LabelName:LabelMapText:LabelUserText:LabelLevels:LabelTimes:LabelNext**

This line allows adjusting the label for a previously defined category. If not set, the default values are used for the category.

- LabelName Show area name in reduced area label (1=yes, 0=no)
- LabelMapText Show map text in reduced area label (1=yes, 0=no)
- LabelUserText Show user text in reduced area label (1=yes, 0=no)
- LabelLevels Show area level limits in reduced area label (1=yes, 0=no)
- LabelTimes Show area activation times in reduced area label (1=yes, 0=no)
- LabelNext Show next activation period in reduced area label (1=yes, 0=no)

## *CategoryDef\_Times*

**CATEGORYDEF\_TIMES:Name:PreActTime**

**CATEGORYDEF\_TIMES:Name:PreActTime:PreInactTime**

This line allows adjusting the pre-active and pre-inactive times for a previously defined category. If not set, the default values are used for the category.

- Name Category name (text string)
- PreActTime Pre-active time (seconds, 0-3600)
- PreInactTime Pre-inactive time (seconds, 0-3600)

## *Name*

**AREA:AreaType:AreaName**

The first line for each area definition must be a *name* line.

- AreaType Area type (one of the following):
  - T (TSA area)
  - 1, 2, 3, 4 or 5 (TSA area)
  - 1F, 2F, 3F, 4F or 5F (TSA area)
  - M (MTCD inhibit area)
  - S (STCA inhibit area)
  - DD (CLAM/RAM inhibit area)
  - FPCA (Flight Plan Conflict Area for MTCD)
  - STCA\_AREA (STCA area)
  - TCTA (TCT area)
  - TCT\_I (TCT inhibit area)
- AreaName Area name to identify it in the relevant window (text string)

TSA area types 1F-5F are filled, 1-5 are not (the number defines the area border color). For area type T the colors and filling are defined in a *category* line. If a *category* line is not defined for a type T area, the area is not filled and a default border color is used.

Of the line types described below, area types M, S and TCT\_I only support *active* (partly, only “ACTIVE:1”), *bound*, *limits* and *coordinate*. TSA areas support all line types. CLAM/RAM inhibit areas are always active regardless of whether ACTIVE:1 is defined or not. FPCAs only support *FPCA\_seps*, *limits*, *bound*, *circle* and *coordinate*. STCA areas only support *STCA\_\**, *limits*, *bound*, *circle* and *coordinate*. TCTAs only support *TCTA\_seps*, *limits*, *bound*, *circle* and *coordinate*.

If no FPCAs are defined, MTCD is available everywhere using either the default alert values or any values specified in the settings file, but if even one FPCA is defined, MTCD is only available within the specified FPCA(s). If FPCAs overlap, the one specified earlier in the file will have priority. As FPCAs add more calculations to the MTCD code, they should be as few and as simple as possible.

If no STCA areas are defined, the global STCA parameters are used everywhere. If one or more STCA areas are defined, the parameters for each flight are based on the areas (global parameters outside all defined areas) and the more restrictive parameters are used for each conflict pair. In case of overlapping areas, the one specified first in the data file will have priority. The areas use the global STCA parameters for any parameters not specified using *STCA\_\** lines.

If no TCTAs are defined, TCT is available everywhere using either the default alert values or any values specified in the settings file, but if even one TCTA is defined, TCT is only available within the specified TCTA(s). In case of overlapping areas, the one specified first in the data file will have priority.

### *Category*

#### **CATEGORY:Name**

The *category* line is optional. It defines the colors, label settings and pre-active time for type T areas.

- Name Category name (must have been defined earlier in the file)

### *Times*

#### **TIMES:PreActTime**

#### **TIMES:PreActTime:PreInactTime**

This line allows adjusting the pre-active and pre-inactive times for the area. If not set, the category-based or default values are used.

- PreActTime Pre-active time (seconds, 0-3600)
- PreInactTime Pre-inactive time (seconds, 0-3600)

### *Label*

#### **LABEL:Lat:Lon**

#### **LABEL:Lat:Lon:LabelText**

The *label* line is optional. It defines the position for the area label, and the label's MapText item.

- Lat Label <latitude>
- Lon Label <longitude>
- LabelText Label text (text string)

### *Group*

#### **GROUP:Name**

The *group* line is optional. It defines the area's group name if desired (not used in the "COOPANS" version)

- Name Group name

## *Usertext*

### **USERTEXT:Text**

The *usertext* line is optional. It defines the default user text for the area.

- Text                          User text string

## *Active*

The *active* line is optional. If there are no *active* lines defined, the area will not be automatically activated. An area can contain more than one *active* line; if even one of them matches, the area is activated. See also the *and\_active* line type below for grouping conditions.

### **ACTIVE:1**

Activates the area automatically without any time limits when the plugin is loaded. Note that this option cannot be used together with other *active* lines as it would override any other schedule.

### **ACTIVE:SchedStartDate:SchedEndDate:SchedWeekdays:StartTime:EndTime**

### **ACTIVE:SchedStartDate:SchedEndDate:SchedWeekdays:StartTime:EndTime:Lower:Upper:UserText**

Used to set activation schedules.

- SchedStartDate      First day to activate the area
  - month and day in the format MMDD (for recurring periods every year)
  - year, month and day in the format YYMMDD (for a single period)
- SchedEndDate        Last day to activate the area, formats as above
- SchedWeekdays      Days of the week to activate the area
  - list of numbers representing the days to activate the area, for example “145” means the area will activate on Mondays, Thursdays and Fridays
  - “0” (zero) to activate the area continuously from StartTime on SchedStartDate to EndTime on SchedEndDate
- StartTime            Time to activate the area (UTC time in the format HHMM)
- EndTime             Time to deactivate the area (UTC time in the format HHMM)
- Lower                Lower limit for the area (feet, 0-999999)
- Upper                Upper limit for the area (feet, 0-999999)
- UserText            Displayed user text string

*Note: SchedEndDate and SchedWeekdays only limit the activation of the area. If the activation time extends past midnight, the area stays active until EndTime on the following day. Lower, Upper and UserText are optional fields. If any of them are specified, all the preceding fields must be included as well (i.e. if UserText is specified, both Lower and Upper must also be).*

### **ACTIVE:NOTAM:Icao:Text**

### **ACTIVE:NOTAM\_GROUP:Icao:Text**

Activates the area based on NOTAM information. The NOTAM\_GROUP version can be used for cases where for example the NOTAM text specifies “ABC123” but means all sub-areas such as “ABC123A” and

“ABC123B”. If the NOTAM\_GROUP version is used for these areas, the code checks the default lower and upper limits of the area and prevents the NOTAM activating it outside those values. The NOTAM version on the other hand overrides any area default limits with the NOTAM data.

- Icao                   ICAO location indicator that publishes activation NOTAMs for the area
- Text                   Text to search for in the NOTAM

#### **ACTIVE:AUP:rsa\_ids**

#### **ACTIVE:AUP\_GROUP:rsa\_ids**

Activates the area based on AUP information. AUP\_GROUP has the same difference from AUP as NOTAM\_GROUP has from NOTAM, see above.

- rsa\_ids               Comma-separated list of area IDs to look for in the downloaded data

#### **ACTIVE:RWY:ARR:ArrRwyList:DEP:DepRwyList**

#### **ACTIVE:RWY:ARR:ArrRwyList:NotArrRwyList:DEP:DepRwyList:NotDepRwyList**

Activates the area based on active runways. If all the specified runway states match, the area is activated. The runway identifiers must be in the format “<4-letter ICAO code><runwayID>”, for example “EFHK15”.

- ArrRwyList           Comma-separated list of runways. Enter “\*” to disregard.
- NotArrRwyList       Comma-separated list of runways. Enter “\*” to disregard.
- DepRwyList           Comma-separated list of runways. Enter “\*” to disregard.
- NotDepRwyList       Comma-separated list of runways. Enter “\*” to disregard.

#### **ACTIVE:ID:YourIdList:NotYourIdList:OnlineIdList:NotOnlineIdList**

Activates the area based on the current controller position ID, and the IDs of other online controllers. The area is activated if the current controller position ID is found in YourIdList, not found in NotYourIdList, all controllers specified in OnlineIdList and none of the controllers specified in NotOnlineIdList are online.

- YourIdList           Comma-separated list of controller IDs (enter “\*” to disregard)
- NotYourIdList       Comma-separated list of controller IDs (enter “\*” to disregard)
- OnlineIdList        Comma-separated list of controller IDs (enter “\*” to disregard)
- NotOnlineIdList      Comma-separated list of controller IDs (enter “\*” to disregard)

#### **ACTIVE:CALLSIGN:YourCallsignList:NotYourCallsignList:OnlineCallsignList:NotOnlineCallsignList**

Activates the area based on the current controller callsign, and the callsigns of other online controllers. The area is activated if the current controller callsign is found in YourCallsignList, not found in NotYourCallsignList, all controllers specified in OnlineCallsignList and none of the controllers specified in NotOnlineCallsignList are online. Partial matches and wildcards are not supported, but consecutive underscore (“\_”) characters are treated as if there was only one.

- YourCallsignList     Comma-separated list of controller callsigns (enter “\*” to disregard)
- NotYourCallsignList Comma-separated list of controller callsigns (enter “\*” to disregard)
- OnlineCallsignList   Comma-separated list of controller callsigns (enter “\*” to disregard)
- NotOnlineCallsignList Comma-separated list of controller callsigns (enter “\*” to disregard)

## *And\_Active*

To combine two or more conditions, the first condition must be defined using an *active* line (see above), and the other conditions using *and\_active* lines. The syntax for *and\_active* is the same as for *active*, the only difference is that instead of starting with “ACTIVE”, the *and\_active* line definitions start with “**AND\_ACTIVE**”. More than one set of conditions can be defined just by starting the next set with an *active* line. The following setup would create two activation rule sets, and the area would activate when either both of the first two conditions are met, or the third one.

```
ACTIVE:something  
AND_ACTIVE:something  
ACTIVE:something
```

Combining multiple lines with time-based schedules (including NOTAM- and AUP-schedules) within a rule set will not work, the plugin will not attempt to combine the schedules.

## *Bound*

### **BOUND:C:Lat:Lon:Radius**

The *Bound* line is optional but highly recommended for areas that are circle-shaped and have not been defined using the *Circle* line as it sets the necessary parameters automatically. It increases the accuracy of the calculation while at the same time reducing the number of calculations required, giving both an accuracy and a performance gain to the plugin.

The bound line should only be used for areas that are circles. The “Lat” and “Lon” coordinates (<latitude> and <longitude>) define the center point and the “Radius” (nautical miles, decimal number) the radius of the circle. The information is used to check if the aircraft is inside the area, but also the coordinate lines are still needed as they are used to draw the area on the screen (the coordinates will not be used for any calculations in this case so make sure you only use this line for circular areas!).

## *Limits*

### **LIMITS:Alt<sub>min</sub>:Alt<sub>max</sub>**

The *Limits* line is optional. It defines the default vertical limits of the area (in hundreds of feet). They can be changed as required in the area windows. When an area without default vertical limits is activated, its lower limit will be set to 0ft and upper limit to 999999ft.

## *Elevation*

### **ELEVATION:Min:Max**

The *elevation* line is optional. It defines the minimum and maximum ground elevation in the area (in hundreds of feet). The default values are 0, available values are from -10 to 999 (-1000ft to 99900ft). The values are used to correct the vertical limits when the area is activated by NOTAM and AGL values are specified. The Min value is used to correct the lower limit and the Max value the upper limit. The Max value may not be lower than the Min value.

## *NoCLAMRAM*

### **NOCLAMRAM**

The optional *NoCLAMRAM* line inhibits CLAM and RAM alerts inside the area when it's active.

## *NoMSAW*

### **NOMSAW**

The optional *NoMSAW* line inhibits MSAW alerts inside the area when it's active.

## *NoAIW*

### **NOAIW**

The optional *NoAIW* line inhibits AIW alerts for the area.

## *NoAPW*

### **NOAPW**

The optional *NoAPW* line inhibits APW alerts for the area.

## *APW\_Buffer\_Lat*

### **APW\_BUFFER\_LAT:BufferU:BufferLI:BufferLV**

The optional *APW\_Buffer\_Lat* line can be used to override the default lateral buffers applied to the area for APW processing. The first value is used above the setting value “*APW\_Buffer\_Lat\_SepLevel*”, the others at or below it.

- BufferU                    High level buffer (nautical miles, decimal value, range 0.0-999.0)
- BufferLI                 Low level buffer for IFR flights (nautical miles, decimal value, range 0.0-999.0)
- BufferLV                 Low level buffer for VFR flights (nautical miles, decimal value, range 0.0-999.0)

## *APW\_Buffer\_Vert*

### **APW\_BUFFER\_VERT:BufferU:BufferLI:BufferLV**

The optional *APW\_Buffer\_Vert* line can be used to override the default vertical buffers applied to the area for APW processing. The first value is used above the minimum RVSM level, the others at or below it.

- BufferU                    High level buffer (feet, integer value, range 0-9999)
- BufferLI                 Low level buffer for IFR flights (feet, integer value, range 0-9999)
- BufferLV                 Low level buffer for VFR flights (feet, integer value, range 0-9999)

## *NoSAP*

### **NOSAP**

The optional *NoSAP* line inhibits SAP alerts for the area.

## *SAP\_Buffer\_Lat*

### **SAP\_BUFFER\_LAT:BufferU:BufferLI:BufferLV**

The optional *SAP\_Buffer\_Lat* line can be used to override the default lateral buffers applied to the area for SAP processing. The first value is used above the setting value “SAP\_Buffer\_Lat\_SepLevel”, the others at or below it.

- BufferU                    High level buffer (nautical miles, decimal value, range 0.0-999.0)
- BufferLI                Low level buffer for IFR flights (nautical miles, decimal value, range 0.0-999.0)
- BufferLV                Low level buffer for VFR flights (nautical miles, decimal value, range 0.0-999.0)

## *SAP\_Buffer\_Vert*

### **SAP\_BUFFER\_VERT:BufferU:BufferLI:BufferLV**

The optional *SAP\_Buffer\_Vert* line can be used to override the default vertical buffers applied to the area for SAP processing. The first value is used above the minimum RVSM level, the others at or below it.

- BufferU                    High level buffer (feet, integer value, range 0-9999)
- BufferLI                Low level buffer for IFR flights (feet, integer value, range 0-9999)
- BufferLV                Low level buffer for VFR flights (feet, integer value, range 0-9999)

## *NoTCT*

### **NOTCT**

The optional *NoTCT* line inhibits TCT alerts for the area.

## *Coordinate*

### **COORD:Lat:Lon**

#### **Lat Lon**

Each area definition must have at least three *Coordinate* lines (or alternatively, one *Circle* line). There is practically no upper limit for the number of coordinate points, but as the required calculations increase proportionally to the number of points, it's best to keep the areas simple. The second format option has a pair of <latitude> and <longitude> values with one or more spaces between them. There may also be one or more spaces in the beginning of the line before the latitude value so it should be relatively easy to create areas from the REGIONS part of a sector file.

## *Circle*

### **CIRCLE:Lat:Lon:Radius:Spacing**

Defines a set of vertex points making up a circle (the same as defining the points one by one using *Coordinate* lines, just simpler). An area definition cannot contain both *Circle* and *Coordinate* lines.

- Lat Center point <latitude>
- Lon Center point <longitude>
- Radius Radius (in nautical miles, 0.1-9999.9)
- Spacing Vertex radial spacing (in degrees, 0.1-120.0)

## *FPCA\_Seps*

### **FPCA\_SEPS:VertSepU:VertSepL:LatSep**

Defines the vertical and lateral separation alert values for the FPCA. This is a mandatory line for a FPCA type area and not used by any other area type. These values are used for the MTCD calculations within this area.

- VertSepU Non-RVSM and above RVSM vertical separation alert value (in feet, 0-99999)
- VertSepL RVSM and below RVSM vertical separation alert value (in feet, 0-99999)
- LatSep Lateral separation alert value (in nautical miles, 0.0-100.0)

### 2.2.1 STCA area settings

These settings are used to set parameters for areas with type “STCA\_AREA”. The default value for each setting is the value of the corresponding plugin setting at the time the area is initialized.

#### *STCA\_Seps\_Lat*

##### **STCA\_SEPS\_LAT:LatSepU:LatSepL:LatSepS**

Defines the lateral separation alert values for upper level (above “STCA\_Sep\_Lat\_SepLevel”, 9500ft by default), lower level (below that level) and special (in defined final approach areas) for the STCA area.

- LatSepU              Upper lateral separation parameter (nm, 0.0-99.0), corresponding to the setting “STCA\_Coarse\_Sep\_Lat\_U”. The rest of the upper lateral separation values for the area will be set automatically based on it.
- LatSepL              Lower lateral separation parameter (nm, 0.0-99.0), corresponding to the setting “STCA\_Coarse\_Sep\_Lat\_L”. The rest of the lower lateral separation values for the area will be set automatically based on it.
- LatSepS              Special lateral separation parameter (nm, 0.0-99.0), corresponding to the setting “STCA\_CurrProx\_Sep\_Lat\_S”. The rest of the special separation values for the area will be set automatically based on it.

#### *STCA\_Seps\_Vert*

##### **STCA\_SEPS\_VERT:VertSepU:VertSepL**

Defines the vertical separation alert values for upper level (non-RVSM flights in RVSM airspace and all flights above RVSM airspace) and lower level (other flights) for the STCA area.

- VertSepU            Vertical separation parameter (feet, 0-9999), corresponding to the setting “STCA\_Coarse\_Sep\_Vert\_U”. The rest of the upper vertical separation values for the area will be set automatically based on it.
- VertSepL            Vertical separation parameter (feet, 0-9999), corresponding to the setting “STCA\_Coarse\_Sep\_Vert\_L”. The rest of the lower vertical separation values for the area will be set automatically based on it.

#### *STCA\_Alert\_Times*

##### **STCA\_ALERT\_TIMES:WarnU:WarnL:ImmU:ImmL**

Defines the alert times for the STCA area.

- WarnU              Warning time (sec, 0-300), corresponding to the setting “STCA\_WarningTime\_U”.
- WarnL              Warning time (sec, 0-300), corresponding to the setting “STCA\_WarningTime\_L”.
- ImmU              Immediate warning time (sec, 0-300),  
                       corresponding to the setting “STCA\_WarningTime\_U\_Imm”.
- ImmL              Immediate warning time (sec, 0-300),  
                       corresponding to the setting “STCA\_WarningTime\_L\_Imm”.

## 2.2.2 TCT area settings

These settings are used to set parameters for areas with type “TCTA”. The default value for each setting is the value of the corresponding plugin setting at the time the area is initialized. Some of the setting lines can optionally contain settings for the Blind Spot function in the area. If they are not specified, the values will be set to the same as the ones defined for the TCT function.

### *TCTA\_Seps\_Lat*

**TCTA\_SEPS\_LAT:LatSep:LatSepDiv**

**TCTA\_SEPS\_LAT:LatSep:LatSepDiv:BS\_LatSep:BS\_LatSepDiv**

Defines the lateral separation alert values for the TCT area.

- LatSep Lateral separation alert value for converging tracks (in nautical miles, 0.0-100.0)
- LatSepDiv Lateral separation alert value for diverging tracks (in nautical miles, 0.0-100.0)

### *TCTA\_Seps\_Vert*

**TCTA\_SEPS\_VERT:VertSepU:VertSepL**

**TCTA\_SEPS\_VERT:VertSepU:VertSepL:BS\_VertSepU:BS\_VertSepL**

Defines the vertical separation alert values for upper level (non-RVSM flights in RVSM airspace and all flights above RVSM airspace) and lower level (other flights) for the TCT area.

- VertSepU Non-RVSM and above RVSM vertical separation alert value (in feet, 0-9999)
- VertSepL RVSM and below RVSM vertical separation alert value (in feet, 0-9999)

### *TCTA\_Pred\_Time*

**TCTA\_PREDICTION\_TIME:Time**

Defines the maximum prediction time for the TCT area.

- Time Prediction time (sec, 0-600)

### *TCTA\_Alert\_Times*

**TCTA\_ALERT\_TIMES:Warn:Imm**

**TCTA\_ALERT\_TIMES:Warn:Imm:BS\_Warn**

Defines the alert times for the TCT area. For TCT, alerts where the conflict starts in less than warning time but more than immediate warning time will only be displayed when a greater than defined number of consecutive checks for the aircraft pair result in a conflict. For BS, all alerts within BS warning time will be displayed. If “BS\_Warn” is not specified, it will be set to the same as “Warn”.

- Warn Warning time (sec, 0-600)
- Imm Immediate warning time (sec, 0-600)
- BS\_Warn Blind Spot function warning time (sec, 0-600)

## *TCTA\_Route\_Params*

### **TCTA\_ROUTE\_PARAMS:TrkError:TurnZone:TurnTrkError**

Defines the lateral prediction parameters for the TCT area. Aircraft are considered to be following their predicted routes when the difference between their current track and the track to the next waypoint does not exceed *TrkError*, or *TurnTrkError* when within *TurnZone* distance from the previous or next waypoint. If these checks fail, the predictions assume the aircraft to continue on its present ground track.

- TrkError              Track error (degrees, 0-90)
- TurnZone              Distance from waypoint (nm, 0.0-10.0)
- TurnTrkError          Track error during turns (degrees, 0-90)

## *TCTA\_AHDG\_Params*

### **TCTA\_AHDG\_PARAMS:ReactTime:TurnRate:TrkError:TurnTime**

Defines the lateral prediction parameters for the TCT area. Aircraft with an assigned heading are first predicted to maintain their present track for *ReactTime*, and then turn with *TurnRate* rate to a track corresponding to the AHDG value (no wind effect considered). When the aircraft's track is within *TrkError* of the assigned heading value, the turn is considered complete. After *TurnTime* has passed, the predictions depend on whether the aircraft's track remains within *TrkError*. If so, the predictions use the current ground track. If not, the predictions assume the aircraft is not following its assigned heading, and revert to the predicted route following (if within its parameters), or the current ground track.

- ReactTime            Reaction time to heading clearance (sec, 0-60)
- TurnRate             Rate of turn (degrees per sec, 0.0-10.0)
- TrkError             Final track error (degrees, 0-90)
- TurnTime            Maximum time to complete a turn (sec, 0-600)

## *TCTA\_Time\_To\_Leave\_Level*

### **TCTA\_TIME\_TO\_LEAVE\_LEVEL:Time**

### **TCTA\_TIME\_TO\_LEAVE\_LEVEL:Time:BS\_Time**

For aircraft in level flight not at its cleared level, defines the maximum predicted time for it to start the level change. The minimum will always be zero.

- Time                Maximum time to leave the current level (sec, 0-600)

## *TCTA\_VS\_Ratios*

### **TCTA\_VS\_RATIOS:MinRatio:MaxRatio**

### **TCTA\_VS\_RATIOS:MinRatio:MaxRatio:BS\_MinRatio:BS\_MaxRatio**

Defines the vertical speed ratios for the vertical predictions of the TCT area, used for the predictions when a track is not in level flight. See below how these are used for climbs and descents.

- MinRatio              Ratio for minimum rate (0.0-1.0)
- MaxRatio              Ratio for maximum rate (0.0-1.0)

## *TCTA\_VS\_Climb*

### **TCTA\_VS\_CLIMB:MinRate:MaxRate:Threshold**

### **TCTA\_VS\_CLIMB:MinRate:MaxRate:Threshold:BS\_MinRate:BS\_MaxRate:BS\_Threshold**

Defines the vertical speeds for the vertical predictions of the TCT area. For aircraft in level flight, the predicted climb rate will be within [MinRate,MaxRate]. For aircraft with a current vertical speed less than Threshold, the predicted climb rate will be within [(1-MinRatio)\*current vs,MaxRate], and for aircraft with a higher vertical speed, [(1-MinRatio)\*current vs, (1+MaxRatio)\*current vs]

- MinRate              Minimum climb rate (ft/min, 0-99999)
- MaxRate              Maximum climb rate (ft/min, 0-99999)
- Threshold            Threshold climb rate (ft/min, 0-99999)

## *TCTA\_VS\_Descent*

### **TCTA\_VS\_DESCENT:MinRate:MaxRate:Threshold**

### **TCTA\_VS\_DESCENT:MinRate:MaxRate:Threshold:BS\_MinRate:BS\_MaxRate:BS\_Threshold**

Defines the vertical speeds for the vertical predictions of the TCT area. For aircraft in level flight, the predicted descent rate will be within [MinRate,MaxRate]. For aircraft with a current vertical speed less than Threshold, the predicted descent rate will be within [(1-MinRatio)\*current vs,MaxRate], and for aircraft with a higher vertical speed, [(1-MinRatio)\*current vs, (1+MaxRatio)\*current vs]

- MinRate              Minimum descent rate (ft/min, 0-99999)
- MaxRate              Maximum descent rate (ft/min, 0-99999)
- Threshold            Threshold descent rate (ft/min, 0-99999)

### 2.2.3 AUP activation

The “AUP” activation option in the Airspace Management Window is used to set activity periods based on AUP data, much like the “NOTAM” option sets activity periods based on NOTAM data. The format for the file can be chosen from two options. The text format is assumed when the data file location specified in the plugin settings ends with “.txt”, the JSON format otherwise. See below for the syntax.

#### 2.2.3.1 Text format

In the text format, each activation period is defined on its own line. To add an activation period, use one of the following line types:

**rsa\_id:StartDate:EndDate:0:StartTime:EndTime:Lower:Upper**  
**rsa\_id:StartDate:EndDate:0:StartTime:EndTime:Lower:Upper:UserText**

- rsa\_id Area ID as specified in the “ACTIVE:AUP:rsa\_id” line in TopSkyAreas.txt
- StartDate Activation start date (year, month and day in the format YYMMDD)
- EndDate Activation end date (year, month and day in the format YYMMDD)
- StartTime Time to activate the area (UTC time in the format HHMM)
- EndTime Time to deactivate the area (UTC time in the format HHMM)
- Lower Lower limit for the area (feet, 0-999999)
- Upper Upper limit for the area (feet, 0-999999)
- UserText Displayed user text string

The data refresh interval can be changed from the default 3600 seconds using the following line:

#### REFRESH\_INTERVAL:Interval

- Interval Data refresh interval (seconds, 30-99999)

#### 2.2.3.2 JSON format

The JSON format must contain at least the following objects and information (refresh\_interval is optional, defaults to 3600 seconds):

- notice\_info
  - valid\_wef string Schedule validity start time (ISO 8601 datetime string, UTC)
  - refresh\_interval number Data refresh interval (seconds, 30-99999)
- areas (array of activity period objects as specified below)
  - name string Area ID as specified in the “ACTIVE:AUP:rsa\_id” line in TopSkyAreas.txt
  - minimum\_fl number Area lower limit (hundreds of feet)
  - maximum\_fl number Area upper limit (hundreds of feet)
  - start\_datetime string Start time (ISO 8601 datetime string, UTC)
  - end\_datetime string End time (ISO 8601 datetime string, UTC)

The datetime formats must contain the time up to at least minutes accuracy (e.g. “2023-01-01T12:34”). The string is not parsed any further so any following information about seconds and time offsets is not taken into account.

Alternatively, “start\_datetime” and “end\_datetime” can be replaced with “start\_time” and “end\_time”. They are also string type but the format is “HH:MM” (or “HH:MM:SS” but the seconds value is disregarded). The schedule validity start time is then used to determine the start and end dates.

## 2.2.4 Remote activation

The “Remote” activation option in the Airspace Management Window is used to control area activation based on a data file whose location is defined using the “HTTP\_Areas\_Remote\_URL” setting. The file can be used to override existing as well as add new activation rules to the areas defined in the TopSkyAreas.txt file.

The data refresh interval can be changed from the default 300 seconds using the following line:

**REFRESH\_INTERVAL:Interval**

- Interval Data refresh interval (seconds, 30-99999)

The following lines affect all areas:

<b>MANUAL_ALL</b>	Sets all areas to manual mode
<b>OVERRIDE_ALL_ALL</b>	Overrides all activation types for all areas
<b>OVERRIDE_ALL_SCHED</b>	Overrides all scheduled activations for all areas
<b>OVERRIDE_ALL_NOTAM</b>	Overrides all NOTAM activations for all areas
<b>OVERRIDE_ALL_AUP</b>	Overrides all AUP activations for all areas
<b>OVERRIDE_ALL_RWY</b>	Overrides all runway activations for all areas
<b>OVERRIDE_ALL_ID</b>	Overrides all controller id activations for all areas
<b>OVERRIDE_ALL_CALLSIGN</b>	Overrides all controller callsign activations for all areas

The following lines affect only the specified area (AreaName is the area name):

<b>AreaName:MANUAL</b>	Sets the area to manual mode
<b>AreaName: OVERRIDE_ALL</b>	Overrides all activation types
<b>AreaName: OVERRIDE_SCHED</b>	Overrides all scheduled activations
<b>AreaName: OVERRIDE_NOTAM</b>	Overrides all NOTAM activations
<b>AreaName: OVERRIDE_AUP</b>	Overrides all AUP activations
<b>AreaName: OVERRIDE_RWY</b>	Overrides all runway activations
<b>AreaName: OVERRIDE_ID</b>	Overrides all controller id activations
<b>AreaName: OVERRIDE_CALLSIGN</b>	Overrides all controller callsign activations

To add new rules, the following line types can be used:

AreaName:1  
AreaName:SchedStartDate:SchedEndDate:SchedWeekdays:StartTime:EndTime  
AreaName:SchedStartDate:SchedEndDate:SchedWeekdays:StartTime:EndTime:Lower:Upper:UserText  
AreaName:AUP:rsa\_id  
AreaName:AUP\_GROUP:rsa\_id  
AreaName:NOTAM:Icao:Text  
AreaName:NOTAM\_GROUP:Icao:Text  
AreaName:RWY:ARR:ArrRwyList:DEP:DepRwyList  
AreaName:RWY:ARR:ArrRwyList:NotArrRwyList:DEP:DepRwyList:NotDepRwyList  
AreaName:ID:YourIdList:NotYourIdList:OnlinelIdList:NotOnlinelIdList  
AreaName:CALLSIGN:YourCallsignList:NotYourCallsignList:OnlineCallsignList:NotOnlineCallsignList

## 2.3 TopSkyCPDLC.txt

This file contains various data for the CPDLC and datalink departure clearance functions.

### *Departure clearance format*

```
DCL:EFHK:SID:<callsign> CLRD TO <ades> OFF <drwy> VIA <sid><cr/lf>SQUAWK <assr> <adt><cr/lf>NEXT  
FREQ <freq_next><cr/lf><stop_highlight><qn><cr/lf><rmk><cr/lf>CLIMB TO <cfl>
```

#### Type:Adep:Subtype:Text

- Type                    One of the following:
  - DCL                DCL type message (other than US)
  - DCL\_US            CPDLC DCL type message (used in USA)
  - PDC                PDC type message
- Adep                Comma-separated list of departure airport ICAO codes (or "\*" to match any airport)
- Subtype             One of the following:
  - AHDG              match when aircraft has an assigned heading
  - SID                match when aircraft has a SID
  - AHDG+SID          match when the aircraft has both
  - \*                  always match
- Text                The clearance message that is sent

Note that if both "AHDG" and "AHDG+SID" formats are defined for an airport with automatically assigned SIDs, the "AHDG" format will only be used when a SID isn't automatically assigned (an automatically assigned SID cannot be removed without changing the flight plan to prevent the automatic assignment).

As an airport typically provides either DCL or PDC (or PDC and DCL\_US) type clearances, both DCL and PDC types should not be defined here for any airport. They are requested using the same message format, and if both clearance types are defined, the plugin won't know which one to send. Having both PDC and DCL\_US for one airport is OK as they are requested with different messages.

The first definition line that matches both the *Adep* and *Subtype* will be used. The *Text* can contain some data fields that are automatically filled by the plugin. Some are highlighted if the pilot's CPDLC software supports it. The "<" character may not be used in the message definition for anything other than entering the defined data fields.

Data fields that are not highlighted:

- <day>                current UTC day (two digits)
- <hour>              hours of the current UTC time (two digits)
- <min>               minutes of the current UTC time (two digits)
- <month2>            current UTC month (two digits)
- <month3>            current UTC month (three characters: "JAN", "FEB", etc.)
- <number>            running number of sent PDCs (three digits)
- <sec>               seconds of the current UTC time (two digits)
- <wday3>            current UTC weekday (three characters: "MON", "TUE", etc.)

- <year2> last two digits of the current UTC year
- <year4> current UTC year
- <cr/lf> Line break (CR and LF characters) for DCL clearances when using the real format  
One empty space in other cases

Data fields that are highlighted in DCL and DCL\_US clearances unless found after "<stop\_highlight>", or if the real format for DCL clearances is used:

- <adep> departure airport ICAO code
- <ades> destination airport ICAO code
- <adt> approved departure time ("ADT xxxx")
- <adt mdi> approved departure time ("ADT xxxx") or minimum dep interval ("MDI xxx")
- <ahdg> assigned heading (three digits)
- <ahdg/trk> assigned heading ("HDG xxx") or track ("TRACK xxx")
- <aobt> actual off-blocks time (four digits)
- <aobtMDI> same as <aobt>, except displays "MDI" if equal to estimated off-blocks time
- <assr> assigned transponder code (four digits)
- <atis> "ATIS " + ATIS identifier (either the aircraft reported or manually entered)
- <atis chg> "ATIS " + ATIS identifier (only if manually entered)
- <callsign> callsign of the aircraft
- <cfl> cleared altitude/FL
- <copx> exit coordination point
- <drwy> departure runway identifier
- <eobt> estimated off-blocks time (four digits)
- <freq\_dep> departure frequency
- <freq\_next> next frequency
- <freq\_own> own primary frequency
- <npt> next route point
- <qnh> "QNH " (or "ALTIM " if reported in inHg) + the QNH value (3-4 digits)
- <qnh4> same as <qnh> but always with four digits
- <qnhQ> "Q" or "A" + the QNH value (four digits)
- <rfl> requested altitude/FL
- <rmk> remarks text
- <sid> SID identifier
- <sid+npt> SID identifier, a blank space and the next route point
- <startup\_next> either "START-UP APPROVED" or "REPORT READY ON " + next frequency
- <startup\_own> either "START-UP APPROVED" or "REPORT READY ON " + own primary freq

The initial value for the frequencies in "<freq\_next>" and "<startup\_next>" is the controller's primary frequency if the track is assumed, otherwise the primary frequency of the controller whose airspace the aircraft is in currently. It can be adjusted in the DCL Window prior to sending the clearance. The frequency for "<freq\_own>" and "<startup\_own>" is always the controller's primary frequency.

Highlighted items should have some text between them to work correctly unless the second item contains static text before the actual highlighted value. In cases where no text is desired between items, for example

the underscore character “\_” can be used. Creating the clearance formats usually requires some testing to get them right.

A header part will be automatically added in front of “DCL” type, by default it is

```
CLD <hour><min> <year2><month2><day> <adep> PDC <number><cr/lf>
```

The whole clearance including the header part is sent to the aircraft, but the header is not displayed in the plugin’s message windows. Fields in the header part will not be highlighted.

Even though the header parts for FSM and CLD messages and some failure and status messages themselves can be adjusted using the plugin settings, this should be avoided as it may break functionality.

Depending on the request format, the plugin can send the status messages and the clearance either as CPDLC, or as telex (real format). For the most part, this needs no additional considerations in the setup, but for line breaks the following is recommended: for the status messages, use “<cr/lf>@” to mark a line break, and when line breaks are needed for the clearance formats, use “<cr/lf>” or “<cr/lf>@” depending on the previous and next items. If the previous item ends with or the next item starts with a highlighted item, the “@” is not needed.

The technical background for the above is: If the message is sent as telex, “<cr/lf>” sends a line break and “@” is removed. If it is sent as CPDLC, “<cr/lf>” is converted to a whitespace and “@” is sent as-is (which the pilot software interprets however it wishes – some use it to indicate a line break, others in other ways, there is no standard on it, but as at least some legacy software make no attempt to break received CPDLC messages into lines on their own, it’s a good idea to include these @’s to help them out).

### *CPDLC free text*

#### **FREETEXT:ReplyType:Text**

- ReplyType            Reply type expected from the aircraft. One of the following:
  - WU            “WILCO” or “UNABLE”
  - AN            “AFFIRM” or “NEGATIVE”
  - R            “ROGER”
  - NE            no reply expected
  - UNICOM    see below
- Text            Message text that is sent

The “UNICOM” reply type is a special message. It is meant to be used to instruct the aircraft to switch to UNICOM. The actual expected replies are either WILCO or UNABLE. When a WILCO reply is received to this message, the following happens:

- 1) all currently open CPDLC dialogues are closed and discarded
- 2) the CPDLC connection is terminated
- 3) the aircraft is set to the “Free” state (i.e. drop track command sent)

## *CPDLC login*

### **LOGIN:Login:RadioCall:ControllerID**

- Login CPDLC login to match with the controller ID (four characters, alphanumeric)
- RadioCall RTF callsign used by the controller
- ControllerID Controller ID to match with the CPDLC login

As there is no way to match a controller to a CPDLC login automatically, this list is used. It follows that it's very important to use only agreed CPDLC logins for the CPDLC connection handovers to work properly.

The "RadioCall" text is sent in the communication transfer and current ATC unit messages.

## *Auto\_RST*

### **AUTO\_RST:IDlist**

- IDlist Comma-separated list of controller IDs

When a transfer of communications is done using CPDLC, the plugin can automatically send a "RADAR SERVICE TERMINATED" message when the transfer is being done to a controller ID defined here. The file can contain more than one *Auto\_RST* line, the information in them is all added up.

## *NFREQ*

### **NFREQ:Location:IDlist<sub>1</sub>:IDlist<sub>2</sub>:IDlist<sub>3</sub>:...**

- Location Defined area or airport ICAO code
- IDlist<sub>n</sub> Comma-separated list of controller IDs (at least one list must be defined)

The NFREQ definition can be used to pre-define frequencies to display in the NFREQ menu. When the aircraft is within the specified area, or departing from the defined airport, the code will go through the lists in the defined order, and when it finds at least one of the controllers of a list online, the NFREQ menu will contain all the primary frequencies of the controllers in that list that are online. The following lists will not be evaluated.

## *Area*

### **AREA:Name**

- Name Name for the area

The *Area* line starts a new area definition, to be used with the NFREQ definitions. Every area must have at least three coordinate points, to be defined using *Coord* lines.

## *Coordinate*

### **COORD:Lat:Lon**

Defines area vertex coordinates.

- Lat                            <latitude>
- Lon                            <longitude>

## 2.4 TopSkyFAST.txt

This file contains the approach definitions for the FAST function. A runway may only belong to one approach definition. Curved approaches are not supported.

It is possible to set up parallel runway operations, either as dependent or independent. To set up independent operations, just define each runway as its own approach. For dependent operations, a specific DPA definition is used which is then used for both single runway operations to one of the specified runways, or for parallel operations whenever there is arriving traffic to both runways. It is not possible to include more than two runways to one definition, or to include a runway in more than one approach definition.

### *FAST\_Approach*

**FAST\_APPROACH:Icao:Rwy:MinAlt:MaxAlt**

**FAST\_APPROACH:Icao:Rwy:MinAlt:MaxAlt:EndLat:EndLon:AppCrsT**

- Icao                      Airport ICAO code
- Rwy                      Runway identifier
- MinAlt                  Aircraft below this altitude are considered landed
- MaxAlt                  Maximum altitude to enter an aircraft to the approach sequence
- EndLat                  Approach end point <latitude>
- EndLon                  Approach end point <longitude>
- AppCrsT                Approach course (degrees true, decimal value)

Sets up an approach for a single runway. If the latitude, longitude and course are not defined, the active sector file is searched for the defined runway and the values are set to the runway threshold coordinates and true bearing towards the opposite threshold.

### *FAST\_Approach\_DPA*

**FAST\_APPROACH\_DPA:Icao:Rwy1:Rwy2:MinAlt:MaxAlt**

**FAST\_APPROACH\_DPA:Icao:Rwy1:Rwy2:MinAlt:MaxAlt:EndLat1:EndLon1:AppCrsT1:EndLat2:EndLon2:AppCrsT2**

Same as above, but used to define a pair of runways used for dependent parallel approaches. If the difference between the two approach courses exceeds 15 degrees, the setup is not considered parallel but a sequencing tool for staggering arrivals to two runways.

The following lines can be used to override the default settings, and apply to the previously defined approach. They need to be re-entered for every approach defined.

### *Sq\_Entry\_Time*

**SEQ\_ENTRY\_TIME:Time**

- Time                      Time (minutes, 5-60, default 10) from landing to enter into sequence

## *WTC\_Type*

### **WTC\_TYPE:Type**

#### **WTC\_TYPE:Type:ListType**

- Type Wake separation type to use
  - 0 Wake turbulence category (default)
  - 1 RECAT ICAO
  - 2 RECAT EU
- ListType Wake separation type to display in FAST List if different from “Type”

## *Surveillance\_Sep*

### **SURVEILLANCE\_SEP:MinSepRed<sub>1</sub>:MinSep<sub>1</sub>:MinSepExt<sub>1</sub>**

### **SURVEILLANCE\_SEP:MinSepRed<sub>1</sub>:MinSep<sub>1</sub>:MinSepExt<sub>1</sub>:MinSepRed<sub>2</sub>:MinSep<sub>2</sub>:MinSepExt<sub>2</sub>**

- MinSepRed<sub>n</sub> Reduced minimum surveillance separation on approach (nm, 0.0-20.0, default 0.0)
- MinSep<sub>n</sub> Normal minimum surveillance separation on approach (nm, 0.1-20.0, default 3.0)
- MinSepExt<sub>n</sub> Extended minimum surveillance separation on approach (nm, 0.0-20.0, default 0.0)

This line type can be used to set the applicable minimum surveillance separation minima for the approach(es). If not defined, the minima for the second runway (2) will be set to the same values as for the first runway. If left at 0.0, the reduced and extended minima will not be available for selection in the configuration window. The reduced minima will only be used between aircraft within the reduced separation area (the normal minima will be used in other cases). The separation minima can optionally be defined using a “MinSep/Name” syntax, where “Name” is a text – 10 characters maximum – to be displayed in the configuration window instead of the numerical value of the minimum.

## *DPA\_Surveillance\_Sep*

### **DPA\_SURVEILLANCE\_SEP:MinSepDPA**

- MinSepDPA Minimum separation between tracks on different approaches (nm, 1.0-10.0)

The default value for MinSepDPA is set automatically based on the spacing between the runway centerlines, but it can be overridden here if necessary. It is only used for DPA definitions. If the difference between the approach courses exceeds 15 degrees, the MinSepDPA value will be used as the stagger distance between tracks on different approaches (if a track is on one approach at x nm from threshold, the next track on the other approach will be sequenced to x+MinSepDPA nm from that threshold)

## *Approach\_Area*

### **APPROACH\_AREA:Length:HalfWidth:Angle:TrackError**

- Length Approach area length (nm, 1.0-100.0, default 30.0)
- HalfWidth Approach area maximum half-width (nm, 1.0-100.0, default 10.0)
- Angle Angle from end point (degrees, 30.0-150.0, default 90.0)
- TrackError Maximum track error (degrees, 0.0-180.0, default 180.0)

This line defines the approach area. Angles below 90 degrees are towards the approach path.

## *Reduced\_Sep\_Area*

### **REDUCED\_SEP\_AREA:Length:HalfWidth**

- Length Approach area length (nm, 1.0-100.0, default 10.0)
- HalfWidth Approach area maximum half-width (nm, 1.0-100.0, default 1.0)

This line defines the reduced separation area.

## *Indicator\_Area*

### **INDICATOR\_AREA:Length:HalfWidth**

- Length Area length (nm, 1.0-100.0, default 20.0)
- HalfWidth Area maximum half-width (nm, 1.0-100.0, default 4.0)

This line defines the area where a track must be for the “next” indicator to be displayed

## *Sq\_Warn\_Area*

### **SEQ\_WARN\_AREA:Length:HalfWidth**

- Length Area length (nm, 1.0-100.0, default 15.0)
- HalfWidth Area maximum half-width (nm, 1.0-100.0, default 3.0)

This line defines the area where a sequence warning is displayed or a forced resorting will be done depending on the settings if a wrong sequence is observed on final.

## *CAT\_2\_3\_Gaps*

### **CAT\_2\_3\_GAPS:Cat2:Cat3**

- Cat2 CAT 2 gap (nm, 1.0-99.0, default 8.0)
- Cat3 CAT 3 gap (nm, 1.0-99.0, default 8.0)

This line defines the minimum gaps to use in front of tracks with CAT2 and CAT3 approach clearances. The defined CAT 3 gap may not be less than the CAT 2 gap.

## *Gaps*

### **GAPS:DefaultGap:GapsList**

- DefaultGap      Default gap (nm, 1.0-99.0, default 8.0)
- GapsList      Comma-separated list of gaps (nm, 1.0-99.0)

This line defines the gaps to display in the GAP Menu, and the highlighted default value when the menu is opened for a flight with no gap set.

## *DPA\_Stagger\_Dist*

### **DPA\_STAGGER\_DIST:Dist**

- Dist      Stagger distance of the approach end points (nm, -9.9 – 9.9)

For DPA setups, the sequence is first calculated on the first defined runway, and the relevant indicators are then moved to the other approach path. If the thresholds are staggered, the indicators for the second approach need to be placed nearer or further away from its threshold to properly align with the indicators for the first runway. This adjustment value is automatically calculated from the threshold coordinates and the first approach course, but if the approaches are not completely parallel or even close to, or the approach end point location to one or both of the runways is somehow unconventional, the distance may need to be adjusted accordingly. Positive values move the indicators on the second runway further from the approach end point, negative values closer to it.

## *DPA\_Gaps\_Same\_Runway*

### **DPA\_GAPS\_SAME\_RUNWAY:Normal:Cat23**

- Normal      Normal gaps applied only between tracks on same approach (0=no, 1=yes)
- Cat23      CAT2/3 gaps applied only between tracks on same approach (0=no, 1=yes)

By default, all the gap types are applied only between aircraft with the same arrival runway. This behavior can be adjusted using this line.

## 2.5 TopSkyMaps.txt & TopSkyMapsLocal.txt

These files contain the definitions for the maps in the Maps Window. The difference is that the maps in the “Local” file are automatically assigned to a folder named “LMAPS” and any map folder names in that file are disregarded. The following example area is used to show the syntax (optional lines in grey color):

COLORDEF:Approach:255:255:255	Color definition
SYMBOLDEF:Fix1	Symbol definition
MOVETO:-4:3	Moveto
LINETO:0:-4	Lineto
LINETO:4:3	Lineto
LINETO:-4:3	Lineto
// EFKU VOR app rwy 15	Comment
MAP:EFKU VOR 15	Name
FOLDER:IAP	Folder
COLOR:Approach	Color
LINE:N063.01.03.067:E027.47.04.942:N063.11.03.016:E027.39.18.377	Line
LINE:N063.01.22.882:E027.49.09.775:N063.00.43.220:E027.45.00.157	Line
LINE:N063.06.39.890:E027.45.04.332:N063.06.00.228:E027.40.53.958	Line
STYLE:Dot:1	Line style
LINE:N063.01.03.067:E027.47.04.942:N062.58.47.170:E027.48.49.840	Line
LINE:N062.58.47.170:E027.48.49.840:N062.54.47.000:E027.53.04.000	Line
SYMBOL:Fix1:N063.06.19.000:E027.43.04.000:ROVSU:20:-10	Symbol + label
SYMBOL:Fix1:N062.54.47.000:E027.53.04.000:VEKEM:20:-10	Symbol + label

The mandatory items for each map are a name, a folder it belongs to in the Maps Window, and a color with which to draw the items. Whitespace and tab characters will be automatically stripped from the beginnings of lines, and any lines with a first non-whitespace/tab character being “{”, “}”, “;” or “/” are disregarded.

### File version

#### //VERSION:VersionString

When an URL has been defined to download maps data, the *file version* line is used to check if the downloaded data is newer than the existing data, and should replace it. The line is not needed in TopSkyMapsLocal.txt, the data there will always be used. To display the version string in a map, use “<VersionString>” without the quotes anywhere where a text string is to be drawn, the code will replace it with the active version string.

- If not present in either file, the downloaded data is used
- If present in the existing data only, the existing data is used.
- If present in the downloaded data only, the downloaded data is used
- If present in both files, the VersionStrings are compared, and the downloaded data is used if its VersionString is greater than the existing data’s. Note that the comparison is done one character at a time, so “9” is greater than “10”, but “09” is less.

## *Override\_Sct\_Map*

### **OVERRIDE\_SCT\_MAP:FolderName**

### **OVERRIDE\_SCT\_MAP:FolderName\MapName**

Indicates that the automatically generated map or sector file item in question should not be displayed in the Maps Window. Can be used to hide sector file items that have custom maps to replace them, or to hide unwanted sector file items from being displayed in the Maps Window. The first option hides the whole folder, the second only a single map.

- FolderName      Folder name to hide
- MapName      Map name to hide

## *Name*

### **MAP:MapName**

The first line for each map definition must be a name line. This will identify the map in the Maps Window.

- MapName      Map name to identify it in the Maps window (text string)

## *Folder*

### **FOLDER:FolderName**

Every map must belong to a folder. There is practically no limit to how many maps a single folder can contain. The following folder names get special treatment: “ARTCC HIGH”, “ARTCC”, “ARTCC LOW”, “GEO”, “SID”, “STAR” and “FREE TEXT”. Maps assigned to those folders will not be shown in the Maps Window. Instead, if sector file items with the same names are found in the active sector file (i.e. a SID named ABC1A exists in the sector file and a map named “ABC1A” in folder “SID”), the map is assigned as additional data for that sector file item and activated/deactivated together with it.

- FolderName      Folder name to identify it in the Maps window (text string)

*Note: the folder name may not start with an empty space character, and may not contain the backslash (“\”) character. Folder names “AUTO” and “LMAPS” are reserved for plugin use.*

## *Layer*

### **LAYER:LayerNumber**

To adjust the drawing order of the maps, a layer number can be specified. Layers are drawn in increasing order, maps within the same layer in the order they are defined in the data file. Areas specified in TopSkyAreas.txt are drawn in layer 0. If a layer is not specified, the map will be placed in layer 1.

- LayerNumber      Drawing layer number (valid layers are -999...-1 and 1...999)

## *Zoom*

### **ZOOM:ZoomLevel**

A whole map or parts of it can be hidden based on the current zoom level. With a positive value set, when the radar screen is zoomed out so that there are less than the specified number of pixels per nautical mile, the lines of the map definition after this line are not read. There can be more than one *zoom* line in one map to hide parts of the map at different zoom levels. When the set value is negative, the following lines are not read when the radar screen is zoomed in more than the set value.

- **ZoomLevel** Radar screen zoom level (pixels per nautical mile, decimal value)

*Note: when there is more than one zoom line in a map, their order is important (for example "ZOOM:5" has to be before "ZOOM:10" to have any effect as with zoom below 10 pix/nm the "ZOOM:5" line will never be read if it's after the "ZOOM:10" line...)*

## *Default\_Screen-specific*

### **DEFAULT\_SCREEN-SPECIFIC**

Sets the default visibility state of all following maps to screen-specific unless an *active* line is present. This is the default value when starting to read a map data file.

## *Default\_Global*

### **DEFAULT\_GLOBAL**

Sets the default visibility state of all following maps to global (synchronized across all radar screens)

## *Screen-specific*

### **SCREEN-SPECIFIC**

Sets this map's visibility state to be screen-specific. Only available for maps with no *active* lines.

## *Global*

### **GLOBAL**

Sets this map's visibility state to be global (synchronized across all radar screens).

## *Hidden*

### **HIDDEN**

Hides this map from the maps list. Hidden maps must have at least one *active* line.

## *Filters\_Off*

### **FILTERS\_OFF**

Clears previously set ASR, callsign and ID filters. When using the filters (see below), note that setting a filter does not remove a previously active filter unless it was of the same type. It is possible to have any combination of ASR, callsign and ID filters active at the same time, and a map line is only read by the drawing code if the conditions for all the active filters are met.

## *Filter\_ASR*

### **FILTER\_ASR:ItemList** (The previous syntax “ASRDATA:ItemList” / “ASRDATA:/\*” is also still accepted)

Display of certain parts of the map can be decided based on information entered in the currently active ASR file. This line defines the “type” of the following lines in the map until the next *Filter\_ASR* line. It is then checked against any definitions in the ASR file.

- ItemList                   Comma-separated list of items

The formats to define map data in the ASR file are as follows:

**PLUGIN:TopSky plugin>ShowMapData:<ItemList>**

**PLUGIN:TopSky plugin>HideMapData:<ItemList>**

<ItemList> is a comma-separated list of items. ShowMapData draws only those parts of plugin maps affected by *Filter\_ASR* lines that contain at least one of the defined items, HideMapData hides such parts.

## *Filter\_ASR\_Off*

### **FILTER\_ASR\_OFF**

Clears a previously set ASR filter.

## *Filter\_Callsign*

### **FILTER\_CALLSIGN:YourCallsignList:NotYourCallsignList:OnlineCallsignList:NotOnlineCallsignList**

Controls whether the following map lines are read by the drawing code based on the current controller callsign, and the callsigns of other online controllers. The lines are read if the current controller callsign is found in YourCallsignList, not found in NotYourCallsignList, all controllers specified in OnlineCallsignList and none of the controllers specified in NotOnlineCallsignList are online. Partial matches and wildcards are not supported, but consecutive underscore (“\_”) characters are treated as if there was only one.

- YourCallsignList   Comma-separated list of controller callsigns (enter “\*” to disregard)
- NotYourCallsignList   Comma-separated list of controller callsigns (enter “\*” to disregard)
- OnlineCallsignList   Comma-separated list of controller callsigns (enter “\*” to disregard)
- NotOnlineCallsignList   Comma-separated list of controller callsigns (enter “\*” to disregard)

## *Filter\_Callsign\_Off*

### **FILTER\_CALLSIGN\_OFF**

Clears a previously set callsign filter.

## *Filter\_ID*

### **FILTER\_ID:YourIdList:NotYourIdList:OnlinelIdList:NotOnlinelIdList**

Controls whether the following map lines are read by the drawing code based on the current controller position ID, and the IDs of other online controllers. The lines are read if the current controller position ID is found in YourIdList, not found in NotYourIdList, all controllers specified in OnlinelIdList and none of the controllers specified in NotOnlinelIdList are online.

- YourIdList            Comma-separated list of controller IDs (enter "\*" to disregard)
- NotYourIdList        Comma-separated list of controller IDs (enter "\*" to disregard)
- OnlinelIdList        Comma-separated list of controller IDs (enter "\*" to disregard)
- NotOnlinelIdList    Comma-separated list of controller IDs (enter "\*" to disregard)

## *Filter\_ID\_Off*

### **FILTER\_ID\_OFF**

Clears a previously set ID filter.

## *SctData*

### **SCTDATA:Type**

### **SCTDATA:Type\Name**

### **SCTDATA:FREETEXT\Group**

### **SCTDATA:FREETEXT\Group\Name**

The *SctData* line is used to draw items from the active sector file data. Specifying only the Type or Group will draw all items. Due to limitations in the data available from EuroScope, the drawing will only look correct when an item contains a single polygon or polyline with the points in consecutive order. The items are drawn using the current map line style and color regardless of what is specified in the sector file.

- Type                    Type of item(s) to draw
  - "LOW AIRWAY", "HIGH AIRWAY", "ARTCC LOW", "ARTCC", "ARTCC HIGH", "SID", "STAR" or "GEO"
- Group                Group name of free text item(s) to draw
- Name                Name of item to draw

## *SctFileData*

**SCTFILEDATA:Type**

**SCTFILEDATA:Type\Name**

**SCTFILEDATA:FREETEXT\Group**

**SCTFILEDATA:FREETEXT\Group\Name**

The *SctFileData* line is used to draw items from a sector file specified with the *SctFilePath* line. Specifying only the Type or Group will draw all items. The items are drawn using the current map line style and color unless a specific color is set in the sector file. To force all items to be drawn using the current map color regardless of possible specific colors set in the sector file, start the line with “**SCTFILEDATA/MONO**” instead of “SCTFILEDATA”.

- Type                    Type of item(s) to draw
  - “LOW AIRWAY”, “HIGH AIRWAY”, “ARTCC LOW”, “ARTCC”, “ARTCC HIGH”, “SID”, “STAR”, “GEO”, “REGIONS”, “SECTOR” or “SECTORLINE”
- Group                Group name of free text item(s) to draw
- Name                Name of item to draw

## *SctFilePath*

**SCTFILEPATH:FileLocation**

The *SctFilePath* line sets the location for the sector file to use as the data source for the *SctFileData* lines.

- FileLocation            Location of the sector file including the file name

The path can be either absolute or relative to the folder where the plugin dll is located. If the path ends with the “\*” character, all files matching up to that and having the “.sct” extension will be considered and the one that’s alphabetically last will be chosen. (e.g. “..\\ABCD-\*” will search the parent folder of the plugin dll folder for any files starting with “ABCD-” and with “.sct” extension. If “ABCD-1234.sct” and “ABCD-1235.sct” are found, “ABCD-1235.sct” will be used).

## *Active*

The *active* line is optional. If there are no *active* lines, the map will not be automatically activated. A map can contain more than one *active* line; if even one of them is a match, the map will be activated.

Automatically activating maps cannot be placed in the following folders: "ARTCC HIGH", "ARTCC", "ARTCC LOW", "AIRWAYS H", "AIRWAYS L", "SID", "STAR", "GEO", "REGIONS" and "FREE TEXT". If no folder name is specified, an automatically activating map will be placed in the "AUTO" folder. See also the *and\_active* line type below for grouping conditions.

### **ACTIVE:1**

Activates the map automatically when the plugin is loaded. Note that this option cannot be used together with other *active* lines.

### **ACTIVE:SchedStartDate:SchedEndDate:SchedWeekdays:StartTime:EndTime**

Activates the map based on fixed activation schedules.

- SchedStartDate      First day to activate the map
  - month and day in the format MMDD (for recurring periods every year)
  - year, month and day in the format YYMMDD (for a single period)
- SchedEndDate      Last day to activate the map, formats as above
- SchedWeekdays      Days of the week to activate the map
  - list of numbers representing the days to activate the map, for example "145" means the map will activate on Mondays, Thursdays and Fridays
  - "0" (zero) to activate the map continuously from StartTime on SchedStartDate to EndTime on SchedEndDate
- StartTime      Time to activate the map (UTC time in the format HHMM)
- EndTime      Time to deactivate the map (UTC time in the format HHMM)

*Note:* *SchedEndDate* and *SchedWeekdays* only limit the activation of the map. If the activation time extends past midnight, the map stays active until *EndTime* on the following day.

### **ACTIVE:NOTAM:Icao:Text**

### **ACTIVE:NOTAM\_GROUP:Icao:Text**

Activates the map based on NOTAM information. For the difference between NOTAM and NOTAM\_GROUP see the same definitions in the Areas file.

- Icao      ICAO location indicator that publishes activation NOTAMs for the map
- Text      Text to search for in the NOTAM

### **ACTIVE:AUP:rsa\_ids**

### **ACTIVE:AUP\_GROUP:rsa\_ids**

Activates the map based on AUP information. For the difference between AUP and AUP\_GROUP see the same definitions in the Areas file.

- rsa\_ids      Comma-separated list of area IDs to look for in the downloaded data

## **ACTIVE:AREA:ActiveAreaList**

### **ACTIVE:AREA:ActiveAreaList:NotActiveAreaList**

Activates the map based on area activity. The map will be activated if all areas listed in *ActiveAreaList* and none of the areas listed in *NotActiveAreaList* are active.

- ActiveAreaList Comma-separated list of area names
- NotActiveAreaList Comma-separated list of area names

## **ACTIVE:MAP:Type:MapFolder\MapName**

Activates the map based on the state of another map. The other map must be defined earlier in the file than this map, and must not be defined as screen-specific (note that maps without an *active* line are screen-specific by default).

- Type How to activate the map (one of the following):
  - = Same state as the specified other map
  - ! Opposite state as the specified other map
- MapFolder\MapName Folder and name of the other map

## **ACTIVE:RWY:ARR:ArrRwyList:DEP:DepRwyList**

### **ACTIVE:RWY:ARR:ArrRwyList:NotArrRwyList:DEP:DepRwyList:NotDepRwyList**

Activates the map based on active runways. If all the specified runway states match, the map is activated. The runway identifiers must be in the format “<4-letter ICAO code><runwayID>”, for example “EFHK15”.

- ArrRwyList Comma-separated list of runways. Enter “\*” to disregard.
- NotArrRwyList Comma-separated list of runways. Enter “\*” to disregard.
- DepRwyList Comma-separated list of runways. Enter “\*” to disregard.
- NotDepRwyList Comma-separated list of runways. Enter “\*” to disregard.

## **ACTIVE:ID:YourIdList:NotYourIdList:OnlineIdList:NotOnlineIdList**

Activates the map based on the current controller position ID, and the IDs of other online controllers. The map is activated if the current controller position ID is found in *YourIdList*, not found in *NotYourIdList*, all controllers specified in *OnlineIdList* and none of the controllers specified in *NotOnlineIdList* are online.

- YourIdList Comma-separated list of controller IDs (enter “\*” to disregard)
- NotYourIdList Comma-separated list of controller IDs (enter “\*” to disregard)
- OnlineIdList Comma-separated list of controller IDs (enter “\*” to disregard)
- NotOnlineIdList Comma-separated list of controller IDs (enter “\*” to disregard)

## **ACTIVE:CALLSIGN:YourCallsignList:NotYourCallsignList:OnlineCallsignList:NotOnlineCallsignList**

Activates the map based on the current controller callsign, and the callsigns of other online controllers. The map is activated if the current controller callsign is found in YourCallsignList, not found in NotYourCallsignList, all controllers specified in OnlineCallsignList and none of the controllers specified in NotOnlineCallsignList are online. Partial matches and wildcards are not supported, but consecutive underscore (“\_”) characters are treated as if there was only one.

- YourCallsignList Comma-separated list of controller callsigns (enter “\*” to disregard)
- NotYourCallsignList Comma-separated list of controller callsigns (enter “\*” to disregard)
- OnlineCallsignList Comma-separated list of controller callsigns (enter “\*” to disregard)
- NotOnlineCallsignList Comma-separated list of controller callsigns (enter “\*” to disregard)

## *And\_Active*

To combine two or more conditions, the first condition must be defined using an *active* line (see above), and the other conditions using *and\_active* lines. The syntax for *and\_active* is the same as for *active*, the only difference is that instead of starting with “ACTIVE”, the *and\_active* line definitions start with “**AND\_ACTIVE**”. More than one set of conditions can be defined just by starting the next set with an *active* line. The following setup would create two activation rule sets, and the map would activate when either both of the first two conditions are met, or the third one.

```
ACTIVE:something  
AND_ACTIVE:something  
ACTIVE:something
```

Combining multiple lines with time-based schedules (including NOTAM- and AUP-schedules) within a rule set will not work, the plugin will not attempt to combine the schedules.

## *Other\_Map\_Act*

### **OTHER\_MAP\_ACT:Type:MapFolder\MapName**

### **OTHER\_MAP\_ACT:Type:MapFolder**

Activates another map or all maps in a folder based on the state of this map. The other map(s) must not have any *active* lines, and this and the other map(s) need to have the same screen-specific/global state. This only affects the state of the other map(s) when the state of this map is changed, the other map(s) can later be manually toggled on/off independently.

- Type How to activate the other map(s) (one of the following):
  - = Same state as this map
  - ! Opposite state as this map
- MapFolder\MapName Folder and name of the other map
- MapFolder Folder name

## *Color*

**COLOR:ColorName**

**COLOR:ColorName:FillColorName**

**COLOR:ColorName:FillColorName:FillBgColorName**

Every map must have at least one *color* line. It sets the color to be used to draw the subsequent drawings.

Each item within a map can be drawn with a different color simply by including a new *color* line when a color change is required. If the FillColorName is not specified, it is set to the same color as ColorName. All used color names (with some exceptions listed below) must be defined in the file using a *colordef* line.

- ColorName      Color to be used for drawing lines and texts
- FillColorName    Color to be used for filling the FILLARC, POLYGON and COORDPOLY items
- FillBgColorName    If specified, used to color the background of the filled area of COORDPOLY items with a hatch fill. If not specified, a transparent background is set.

The following plugin colors can be used as map colors without having to define them in a *colordef* line (however if the color is defined in a *colordef* line, that definition will override the plugin color definition):

- Active\_Map
- Active\_Map\_Type\_1 ... Active\_Map\_Type\_20
- Active\_RD\_Infill\_Map
- Active\_RD\_Map
- Active\_Sector
- Active\_Text\_Map
- East\_NAT\_Map
- Inactive\_Sector
- Map\_1 ... Map\_4
- Map\_Border
- Map\_Hotspot
- Map\_Info
- Map\_Land
- Map\_Symbol
- Preactive\_Map
- Preactive\_Text\_Map
- Predisplay\_Map
- Rwy\_App\_Line\_Inuse
- Rwy\_App\_Line\_Not\_Inuse
- West\_NAT\_Map

## *Color definition*

### **COLORDEF:ColorName:R:G:B**

Every color used in the maps (with some exceptions listed in the COLOR line specification) must be defined using one of these lines.

- ColorName      Color name to be used in the Color lines (text string)
- R, G and B      Color's red, green and blue component values (0-255)

## *Style*

### **STYLE:StyleName**

#### **STYLE:StyleName:Width**

The *style* line sets the line type for any subsequent *line* items within this map. It is not mandatory, a Solid type line with width 1 pixel will be drawn by default. As with the *color* line, a single map may contain any required number of *style* lines to draw different line styles within the same map. If a width is not defined, a 1-pixel width is drawn by default.

- StyleName      Style to be used
  - Default: Solid, Alternate, Dash, Dot, DashDot, DashDotDot or Null
  - Custom: a style name defined earlier in the file using a *LineStyleDef* line
- Width      Width of line (pixels)

## *Line style definition*

### **LINESTYLEDEF:StyleName:BrushStyle:HatchStyle**

#### **LINESTYLEDEF:StyleName:BrushStyle:HatchStyle:StyleArray**

Custom line styles can be defined using the *LineStyleDef* line.

- StyleName      Style name to be used in the *Style* lines (text string)
- BrushStyle      Brush style (Solid, Hatched or Null)
- HatchStyle      Hatch style (set any value when brush style is not "Hatched")
  - /      45-degree upward hatch
  - +      Horizontal and vertical crosshatch
  - X      45-degree crosshatch
  - \      45-degree downward hatch
  - -      Horizontal hatch
  - |      Vertical hatch
- StyleArray      Comma-separated array of values to define the line style if not a solid line.  
First value defines the length of the first dash, second the first space, etc. The pattern repeats as necessary when drawing the line – reversing if the number of values is odd (dashes become spaces and vice versa). A maximum of 16 values may be entered to define the style.

## *Line*

**LINE:Lat<sub>1</sub>:Lon<sub>1</sub>:Lat<sub>2</sub>:Lon<sub>2</sub>**

**LINE:StartPointName:EndPointName**

Draws a line from one point to another. Uses the previously defined line style (or solid line with 1-pixel width if no style defined).

- Lat<sub>1</sub> Start point <latitude>
- Lon<sub>1</sub> Start point <longitude>
- Lat<sub>2</sub> End point <latitude>
- Lon<sub>2</sub> End point <longitude>
- StartPointName Start point <position>
- EndPointName End point <position>

## *FontSize*

**FONTSIZE>Type:Size**

**FONTSIZE:0**

Each new map starts out with the default font size. It can be modified using the *FontSize* line. All texts after the line in that map use the new size. “FONTSIZE:0” sets the size back to the default value.

- Type Type of change
  - “=” sets a new size
  - “-” reduces the size from the default by the given amount
  - “+” increases the size from the default by the given amount
  - “\*” multiplies the size of the default by the given amount
- Size New font size (1-99)

The resulting font size is limited to values between 1 and 99.

## *FontStyle*

**FONTSTYLE:Weight:Italic:Underline:Strikethrough**

**FONTSTYLE:0**

Each new map starts out with the default font style. It can be modified using the *FontStyle* line. All texts after the line in that map use the new style. “FONTSTYLE:0” sets the style back to the default settings.

- Weight Font weight (0-1000)
  - some example values are 0=default weight, 400=normal, 700:bold
- Italic Italic (1=yes, 0=no)
- Underline Underline (1=yes, 0=no)
- Strikethrough Strikethrough (1=yes, 0=no)

## *TextAlign*

### **TEXTALIGN:Flags**

Sets the default text alignment used in the *Text* and *Symbol* lines. If defined before the first map, becomes the default alignment for all maps. If defined within a map, becomes the default alignment for all following lines of that map.

- Flags                      Combination of the following:
  - “L”, “C” or “R”              for left, center or right-aligned horizontally
  - “T”, “C” or “B”              for top, center or bottom-aligned vertically

By default, the alignment is centered both horizontally and vertically, i.e. the text label is centered on the defined position. Entering for example “LT” puts the text label’s top left corner in the defined position instead.

## *Text*

### **TEXT:Lat:Lon:Label**

### **TEXT:Lat:Lon:Label:OffsetX:OffsetY**

### **TEXT:PointName:Label**

### **TEXT:PointName:Label:OffsetX:OffsetY**

Draws a text label.

- Lat                      Label anchor point <latitude>
- Lon                      Label anchor point <longitude>
- PointName              Label anchor point <position>
- Label                   Text label (text string)
- OffsetX                Number of pixels to offset the label in the left(-) - right(+) direction
- OffsetY                Number of pixels to offset the label in the up(-) - down(+) direction

*Note : to set the text alignment for just this label, it is possible to suffix TEXT with a forward slash followed by the required alignment flags, i.e. TEXT/LT to align the label top left corner on the anchor point.*

## *Symbol*

**SYMBOL:SymbolName:Lat:Lon**

**SYMBOL:SymbolName:Lat:Lon:Label:OffsetX:OffsetY**

**SYMBOL:SymbolName:PointName**

**SYMBOL:SymbolName:PointName:Label:OffsetX:OffsetY**

Draws a predefined symbol on the screen, optionally with a text label.

- SymbolName      Name of symbol (1)
- Lat              Symbol centerpoint <latitude>
- Lon              Symbol centerpoint <longitude>
- PointName      Symbol centerpoint <position>
- Label            Text label (text string)
- OffsetX         Number of pixels to offset the label in the left(-) - right(+) direction
- OffsetY         Number of pixels to offset the label in the up(-) - down(+) direction

*Note 1:* the *SymbolName* can be a symbol type name (see *TopSkySymbols.txt*) or a symbol defined in this file with a *SymbolDef* line.

*Note 2:* to set the text alignment for just this label, it is possible to suffix *SYMBOL* with a forward slash followed by the required alignment flags, i.e. *SYMBOL/LT* to align the label top left corner on the symbol centerpoint.

## *Symbol definition*

**SYMBOLDEF:SymbolName**

The first line for each symbol definition must be a *symbol definition* line.

- SymbolName      Symbol name to use for this symbol in the Symbol lines (text string)

The symbol itself can consist of various elements, drawn by the following lines. The X and Y coordinates are relative to the symbol centerpoint, with the X axis having increasing values to the right and the Y axis having increasing values to the down direction. The commands are the same as in the EuroScope Symbology dialog with the exception of the possibility to draw elliptical arcs and the ":" separating the values here so the ES dialog can be used in most cases to test the results.

## **MOVETO:X:Y**

Sets the starting point for the next LINETO command

- X Number of pixels from the symbol centerpoint in the left(-)-right(+) direction
- Y Number of pixels from the symbol centerpoint in the up(-)-down(+) direction

## **LINETO:X:Y**

Draws a straight line from the previous position

- X Number of pixels from the symbol centerpoint in the left(-)-right(+) direction
- Y Number of pixels from the symbol centerpoint in the up(-)-down(+) direction

## **SETPIXEL:X:Y**

Paints the selected pixel

- X Number of pixels from the symbol centerpoint in the left(-)-right(+) direction
- Y Number of pixels from the symbol centerpoint in the up(-)-down(+) direction

## **ARC:X:Y:Radius:StartAngle:EndAngle**

## **ARC:X:Y:Radius<sub>X</sub>:Radius<sub>Y</sub>:StartAngle:EndAngle**

Draws a part of a circle

- X Centerpoint offset from the symbol centerpoint in the left(-)-right(+) direction
- Y Centerpoint offset from the symbol centerpoint in the up(-)-down(+) direction
- Radius Arc radius in pixels (to make a circular arc)
- Radius<sub>X</sub> Arc radius in relation to the X axis in pixels (to make an elliptical arc)
- Radius<sub>Y</sub> Arc radius in relation to the Y axis in pixels (to make an elliptical arc)
- StartAngle Arc starting angle (integer degrees, 0 degrees is at positive X-axis, increasing counterclockwise)
- EndAngle Arc ending angle (integer degrees, 0 degrees is at positive X-axis, increasing counterclockwise)

## **FILLARC:X:Y:Radius:StartAngle:EndAngle**

## **FILLARC:X:Y:Radius<sub>X</sub>:Radius<sub>Y</sub>:StartAngle:EndAngle**

Otherwise the same as ARC above but the result is filled

## **POLYGON:X<sub>1</sub>:Y<sub>1</sub>: X<sub>2</sub>:Y<sub>2</sub>:...: X<sub>n</sub>:Y<sub>n</sub>**

Draws a filled polygon with n vertices

## *Coordinate*

**COORD:Lat:Lon**

**COORD:PointName**

Defines line or polygon vertex coordinates to be used later with *CoordLine* or *CoordPoly* lines.

- Lat <latitude>
- Lon <longitude>
- PointName <position>

## *Coordinate\_PBD*

**COORD\_PBD:Lat:Lon:Bearing:Distance**

**COORD\_PBD:PointName:Bearing:Distance**

Defines line or polygon vertex coordinates to be used later with *CoordLine* or *CoordPoly* lines as a bearing and distance from a specified point.

- Lat <latitude>
- Lon <longitude>
- PointName <position>
- Bearing Bearing from the point (decimal degrees, 0.0-360.0, optionally suffixed with "M" to indicate magnetic reference, otherwise a true bearing)
- Distance Distance from the point (in nautical miles, 0.1-9999.9)

## *Coordinate\_PBX*

**COORD\_PBX:Lat1:Lon1:Bearing1:Lat2:Lon2:Bearing2**

**COORD\_PBX:PointName1:Bearing1:PointName2:Bearing2**

Defines line or polygon vertex coordinates to be used later with *CoordLine* or *CoordPoly* lines as an intersection of bearings from two specified points.

- Lat1/2 <latitude>
- Lon1/2 <longitude>
- PointName1/2 <position>
- Bearing1/2 Bearing from the point (decimal degrees, 0.0-360.0, optionally suffixed with "M" to indicate magnetic reference, otherwise a true bearing)

## *Coordinate\_AF*

**COORD\_AF:Lat:Lon:Radius:Spacing:StartAngle:Direction:EndAngle**

**COORD\_AF:PointName:Radius:Spacing:StartAngle:Direction:EndAngle**

Defines a set of vertex points making up an arc to be used later with *CoordLine* or *CoordPoly* lines. The line is expanded to the necessary number of *Coordinate* lines when the maps data file is read. This means small Spacing values may affect performance when the map is active due to the number of line segments being drawn.

- Lat Center point <latitude>
- Lon Center point <longitude>
- PointName Center point <position>
- Radius Radius (in nautical miles, 0.1-9999.9)
- Spacing Vertex radial spacing (in degrees, 0.1-120.0)
- StartAngle Arc start angle (decimal degrees, 0.0-360.0, optionally suffixed with "M" to indicate magnetic reference, otherwise a true bearing)
- Direction ">" for clockwise, "<" for counterclockwise
- EndAngle Arc end angle (decimal degrees, 0.0-360.0, optionally suffixed with "M" to indicate magnetic reference, otherwise a true bearing)

## *Coordinate\_CR*

**COORD\_CR:Course:Lat:Lon:Radial**

**COORD\_CR:Course:PointName:Radial**

Defines a vertex point as a specified course from the previously defined point to intercept a radial from a specified point be used later with *CoordLine* or *CoordPoly* lines.

- Course Intercept course (decimal degrees, 0.0-360.0, optionally suffixed with "M" to indicate magnetic reference, otherwise a true course)
- Lat <latitude>
- Lon <longitude>
- PointName <position>
- Radial Bearing from the point (decimal degrees, 0.0-360.0, optionally suffixed with "M" to indicate magnetic reference, otherwise a true bearing)

## *Coordinate\_DF*

**COORD\_DF:InitCrs:TurnDir:Radius:Spacing:Lat:Lon**

**COORD\_DF:InitCrs:TurnDir:Radius:Spacing:PointName**

Defines a set of vertex points making up a turn direct to a specified point be used later with *CoordLine* or *CoordPoly* lines.

- InitCrs                      Initial course (decimal degrees, 0.0-360.0, optionally suffixed with “M” to indicate magnetic reference, otherwise a true course)
- TurnDir                      Turn direction (“L” or “R” forcing a turn direction, anything else meaning shortest turn)
- Radius (1)                 Turn radius (in nautical miles, 0.1-999.9)
- Spacing                      Vertex radial spacing (in degrees, 0.1-120.0)
- Lat                          <latitude>
- Lon                          <longitude>
- PointName                  <position>

*Note 1: the radius can optionally be defined using a groundspeed value in knots. The value needs to be suffixed with “KTS” in that case, and the radius is automatically calculated assuming a rate of turn 3°/sec, limited to a maximum bank angle of 25°.*

## *Coordinate\_FC*

**COORD\_FC:Bearing:Distance**

Defines line or polygon vertex coordinates to be used later with *CoordLine* or *CoordPoly* lines as a bearing and distance from the previously defined point.

- Bearing                      Bearing from the point (decimal degrees, 0.0-360.0, optionally suffixed with “M” to indicate magnetic reference, otherwise a true bearing)
- Distance                    Distance from the point (in nautical miles, 0.1-9999.9)

## *Coordinate\_FD*

**COORD\_FD:Bearing:Lat:Lon:Radius**

**COORD\_FD:Bearing:PointName:Radius**

Defines line or polygon vertex coordinates to be used later with *CoordLine* or *CoordPoly* lines as a bearing from the previously defined point crossing a specified circle.

- Bearing                      Bearing from the point (decimal degrees, 0.0-360.0, optionally suffixed with "M" to indicate magnetic reference, otherwise a true bearing)
- Lat                          <latitude>
- Lon                          <longitude>
- PointName                  <position>
- Radius                      Radius of the circle (in nautical miles, 0.1-9999.9)

## *Coordinate\_HM*

**COORD\_HM:Lat:Lon:InbdCrs:TurnDir:Length:Radius:Spacing**

**COORD\_HM:PointName:InbdCrs:TurnDir:Length:Radius:Spacing**

Defines a set of vertex points making up a holding pattern to be used later with *CoordLine* or *CoordPoly* lines. The line is expanded to the necessary number of *Coordinate* lines when the maps data file is read. This means small Spacing values may affect performance when the map is active due to the number of line segments being drawn.

- Lat                          <latitude>
- Lon                          <longitude>
- PointName                  <position>
- InbdCrs                    Holding inbound course (decimal degrees, 0.0-360.0, optionally suffixed with "M" to indicate magnetic reference, otherwise a true course)
- TurnDir                    Turn direction ("L" or "R")
- Length (1)                Length of straight segments (in nautical miles, 0.1-999.9)
- Radius (2)                Radius of the turns (in nautical miles, 0.1-999.9)
- Spacing                    Vertex radial spacing in turns (in degrees, 0.1-120.0)

*Note 1:* the length can optionally be defined using a time value in minutes. The value needs to be suffixed with "MIN" in that case, and the defined groundspeed (calculated from the radius value) will be used to calculate the length.

*Note 2:* the radius can optionally be defined using a groundspeed value in knots. The value needs to be suffixed with "KTS" in that case, and the radius is automatically calculated assuming a rate of turn 3°/sec, limited to a maximum bank angle of 25°.

## *Coordinate\_Circle*

### **COORD\_CIRCLE:Lat:Lon:Radius:Spacing**

### **COORD\_CIRCLE:PointName:Radius:Spacing**

Defines a set of vertex points making up a circle to be used later with *CoordLine* or *CoordPoly* lines. The line is expanded to the necessary number of *Coordinate* lines when the maps data file is read. This means small Spacing values may affect performance when the map is active due to the number of line segments being drawn.

- Lat Center point <latitude>
- Lon Center point <longitude>
- PointName Center point <position>
- Radius Radius (in nautical miles, 0.1-9999.9)
- Spacing Vertex radial spacing (in degrees, 0.1-120.0)

## *Coordinate\_Turn*

### **COORD\_TURN:Radius:Spacing:InitialTrack:TrackChange**

Defines a set of vertex points starting from the previously defined point, making up a circle or a part of it (a circular arc) to be used later with *CoordLine* or *CoordPoly* lines. The line is expanded to the necessary number of *Coordinate* lines when the maps data file is read. This means small Spacing values may affect performance when the map is active due to the number of line segments being drawn.

- Radius (1) Radius (in nautical miles, 0.1-9999.9)
- Spacing Vertex radial spacing (in degrees, 0.1-120.0)
- InitialTrack Track to start the turn from (decimal degrees, 0.0-360.0, optionally suffixed with "M" to indicate magnetic reference, otherwise a true track)
- TrackChange Amount of turn (decimal degrees, -360.0 to 360.0, negative meaning left turn)

*Note 1: the radius can optionally be defined using a groundspeed value in knots. The value needs to be suffixed with "KTS" in that case, and the radius is automatically calculated assuming a rate of turn 3°/sec, limited to a maximum bank angle of 25°.*

## *Coordinate\_List*

### **COORD\_LIST>ListSep:LatLonSep:CoordList**

Defines a set of coordinates. The list is first split to text strings using ListSep as the separator, and then each of them is treated like a *Coord* line – a <position> if not containing LatLonSep, and a <latitude>,<longitude> pair if a LatLonSep character is found.

- ListSep Separator character between two coordinates
- LatLonSep Separator character between a latitude,longitude pair
- CoordList Coordinates (either <latitude> and <longitude> separated by LatLonSep, or <position>) separated by ListSep

## *CoordLine*

### **COORDLINE**

Draws a sequence of lines from a set of coordinates having been defined by preceding *coordinate* lines. The lines use the previously defined line style (or solid line with 1-pixel width if no style defined). After drawing, the used coordinates are discarded and new ones can be defined.

## *CoordPoly*

### **COORDPOLY:FillPattern**

Draws a polygon from a set of coordinates having been defined by preceding *coordinate* lines. The edge line is drawn with the previously defined line style (or solid line with 1-pixel width if no style defined).

- |               |  |                    |
|---------------|--|--------------------|
| ○ FillPattern | Polygon fill pattern                                 |                    |
|               | • 0  | no fill            |
|               | • 5, 10, 20, 25, 30, 40, 50, 60, 70, 75, 80, 90, 100 | percentage to fill |
|               | • E0 – E52   | hatch fill         |

The hatch fill values correspond to the GDI+ HatchStyle enumeration values. For example, “E0” sets “HatchStyleHorizontal” and “E6” sets “HatchStyle05Percent” which can also be achieved by “5” (the numeric values for the FillPattern are just shortcuts to the percentage hatch styles). After drawing, the used coordinates are discarded and new ones can be defined.

## 2.6 TopSkyMSAW.txt

This file contains the data for the MSAW and APM functions.

### 2.6.1 MSAW areas

The file is read one line at a time and the first line that contains the aircraft position returns the minimum safe altitude, so put specific small area lines at the top and large general areas to the end of the file. Setting a MSA value of "0" to an area makes it an inhibit area (no alerts even if aircraft are below 0 feet). Only define one area per line. There are five types of area definitions that are accepted:

#### *Lat/Lon box area*

**A:Lat<sub>min</sub>:Lat<sub>max</sub>:Lon<sub>min</sub>:Lon<sub>max</sub>:MSA**

An area bounded by the minimum and maximum latitude and longitude values

- Lat<sub>min</sub> Minimum <latitude>
- Lat<sub>max</sub> Maximum <latitude>
- Lon<sub>min</sub> Minimum <longitude>
- Lon<sub>max</sub> Maximum <longitude>
- MSA Minimum Safe Altitude within the area (feet, integer value)

#### *Circle*

**C:Lat:Lon:R:MSA**

A circle of radius R with center point at (Lat,Lon)

- Lat Circle center point <latitude>
- Lon Circle center point <longitude>
- R Radius of circle (nautical miles, decimal value)
- MSA Minimum Safe Altitude within the circle (feet, integer value)

#### *Lat/Lon box area list*

**L:Lat<sub>min</sub>:Lon<sub>min</sub>:ΔLat:ΔLon:N:MSA<sub>1</sub>:MSA<sub>2</sub>:...:MSA<sub>n</sub>**

A series of latitude-longitude bounded boxes. The boxes are in an east-west direction, with the first box being the westernmost.

- Lat<sub>min</sub> <latitude> of the south edge of the boxes
- Lon<sub>min</sub> <longitude> of the west edge of first box
- ΔLat Latitude size of one box (decimal degrees)
- ΔLon Longitude size of one box (decimal degrees)
- N Number of boxes
- MSA<sub>1</sub>-MSA<sub>n</sub> Minimum Safe Altitudes of the boxes (feet, integer values)

## *Polygon*

**P:N:Lat<sub>1</sub>:Lon<sub>1</sub>:Lat<sub>2</sub>:Lon<sub>2</sub>:...:Lat<sub>n</sub>:Lon<sub>n</sub>:MSA**

A polygon with  $n$  vertices at given latitude-longitude points

- N Number of vertices
- Lat<sub>n</sub> <latitude> of vertex n
- Lon<sub>n</sub> <longitude> of vertex n
- MSA Minimum Safe Altitude within the polygon (feet, integer value)

## *Sector*

**S:Lat:Lon:TRdl<sub>1</sub>:TRdl<sub>2</sub>:R<sub>min</sub>:R<sub>max</sub>:MSA**

An area defined as being between two true bearings from a point (Lat,Lon) - clockwise direction from Rdl<sub>1</sub> to Rdl<sub>2</sub> - and between distances R<sub>min</sub> and R<sub>max</sub> from the point

- Lat Point <latitude>
- Lon Point <longitude>
- TRdl<sub>1</sub> Bearing 1 (degrees true, decimal value)
- TRdl<sub>2</sub> Bearing 2 (degrees true, decimal value)
- R<sub>min</sub> Minimum distance from point (nautical miles, decimal value)
- R<sub>max</sub> Maximum distance from point (nautical miles, decimal value)
- MSA Minimum Safe Altitude within the sector (feet, integer value)

## 2.6.2 APM areas

APM areas, unlike the other MSAW areas, are defined by more than one data line. The definition for an APM area starts with an *APM* line:

### *APM*

#### **APM:Icao:Rwy:Alerts:TdpLat:TdpLon:TdpAlt:AppCrsT:Rmin**

- Icao                      Airport ICAO code
- Rwy                      Runway identifier
- Alerts                    One or more of the following characters to define which alerts to display:
  - L                      Lateral alerts
  - A                      Above path alerts
  - B                      Below path alerts
- TdpLat                  Touchdown point <latitude>
- TdpLon                  Touchdown point <longitude>
- TdpAlt                  Touchdown point altitude (feet, integer value)
- AppCrsT                Approach course (degrees true, decimal value)
- Rmin                    Alert inhibit distance from touchdown point (nautical miles, decimal value)

The approach lateral path area is then defined using either three or more *Coord* lines, or an *APM\_Area* line.

### *Coord*

#### **COORD:Lat:Lon**

- Lat                      Point <latitude>
- Lon                      Point <longitude>

### *APM\_Area*

#### **APM\_AREA:XTE\_init:XTE\_slope:Rmax**

#### **APM\_AREA:XTE\_init:XTE\_slope:XTE\_limit:Rmax**

- XTE\_init                Half-width of area at touchdown point (nautical miles, decimal value, 0.1-99.9)
- XTE\_slope              Angle by which the area opens outwards (degrees, decimal value, 0.0-45.0)
- XTE\_limit              Maximum half-width of area (nautical miles, decimal value, 0.1-999.9)
- Rmax                    Length of area from touchdown point (nautical miles, decimal value, 0.1-999.9)

The rest of the definitions below are optional, and if not specified, default values will be used.

## *APM\_Vert*

The *APM\_Vert* line is used to specify the vertical path area of the approach. The low and high angles begin from the touchdown point at its altitude minus/plus the VTE\_init value.

### **APM\_VERT:VTE\_init:AngleLow:AngleHigh**

- VTE\_init Half-height of area at touchdown point (feet, integer value, default **0**)
- AngleLow “Below path” angle (degrees, decimal value, default **2.0**)
- AngleHigh “Above path” angle (degrees, decimal value, default **4.0**)

## *APM\_Vert\_Lim*

The *APM\_Vert\_Lim* line is used to specify the lowest altitude to join the final approach path (no “below path” alerts will be displayed above this altitude), and the altitude above which aircraft are not considered as being on approach even when within the approach lateral area.

### **APM\_VERT\_LIM:JoinAlt:OverflyAlt**

- JoinAlt Joining altitude (feet, integer value, default **99999**)
- OverflyAlt Overflight altitude (feet, integer value, default **99999**)

## *APM\_TKE*

The *APM\_TKE* line is used to specify the track error limits to start monitoring the approach (less than Tolerance and within lateral path area) and to flag a lateral deviation within the lateral path area (more than Tolerance+Deviation when previously less than Tolerance).

### **APM\_TKE:Tolerance:Deviation**

- Tolerance Track error tolerance (degrees, integer value, default **20**)
- Deviation Track error deviation (degrees, integer value, default **10**)

## 2.7 TopSkyRadar.txt

The file contains the primary radar station definitions to be used for displaying raw video radar data. The following example shows the syntax (optional lines in grey color):

// Helsinki PSR	Comment
RADAR:Helsinki	Radar
POSITIONS:EFHK:EFIN	Positions
LOCATION:N060.18.56.400:E024.57.54.400	Location
ALTITUDE:335	Altitude
BEAMWIDTH:1.4	Beamwidth
PULSEWIDTH:1.0	Pulsewidth
MAXANGLE:50	Maxangle
RANGE:0.0:60.0	Range
NOTERRAIN	NoTerrain

### *Radar*

#### **RADAR:RadarName**

Each radar definition must start with a Radar line that defines the radar station name.

- RadarName              Radar station name (text string)

### *Positions*

#### **POSITIONS:Pos<sub>1</sub>:Pos<sub>2</sub>:...**

Defines the list of controller positions that use the radar station. Only one radar station can be active so the first station in the file that contains a match will be used. The logic compares the positions against the login callsign from the beginning of the string (“EF” will be a match for either “EFIN\_CTR” or “EFHK\_TWR” but not for “SAEF\_APP”)

- Pos<sub>x</sub>              Position login callsign (text string, full callsign or first x letters)

### *Location*

#### **LOCATION:Lat:Lon**

The location of the radar antenna.

- Lat                    <latitude>
- Lon                    <longitude>

## *Altitude*

### **ALTITUDE:Alt**

The radar antenna altitude above mean sea level. If not specified, a value of 0 is used.

- Alt Antenna altitude (feet AMSL, integer value)

## *Beamwidth*

### **BEAMWIDTH:Beamwidth**

Specifies the beamwidth of the radar in degrees. If not specified, a value of 1.5 will be used. The value affects how wide the radar targets will be (twice the beamwidth value).

- Beamwidth Beamwidth of the radar (degrees, decimal value)

## *Pulsewidth*

### **PULSEWIDTH:Pulsewidth**

Specifies the pulse width of the radar in microseconds. If not specified, a value of 1.0 will be used. The value affects how deep the radar targets will be (approx. 0.08nm/microsecond).

- Pulsewidth Pulse width of the radar (microseconds, decimal value)

## *Maxangle*

### **MAXANGLE:MaxAngle**

Defines the maximum elevation angle of the radar measured from horizontal level. If not specified, a value of 90 will be used (i.e. coverage all the way up to vertical)

- MaxAngle Maximum vertical angle of the radar (degrees, integer value)

## *Range*

### **RANGE:MinRange:MaxRange**

Defines the minimum and maximum detection ranges of the radar station. If not specified, a minimum value of 0 and a maximum value of 999999 will be used.

- MinRange Minimum detection range of the radar (nautical miles, decimal value)
- MaxRange Maximum detection range of the radar (nautical miles, decimal value)

## *Ceiling*

### **CEILING:Ceiling**

Defines the maximum detection altitude of the radar. If not specified, a value of 999999 will be used.

- Ceiling                    Maximum detection altitude of the radar (feet AMSL, integer value)

## *NoTerrain*

### **NOTERRAIN**

Sets the radar station to disregard ESE file radar data and to have no terrain masking.

When *NoTerrain* or *terrain* lines are not defined, the displayed targets are based on the primary radar coverage defined in the ESE file. For reasonable results, a primary radar should be defined at the same position in the ESE file as the one in this data file.

With the *NoTerrain* definition, all targets within the radar's visibility area will be displayed regardless of the ESE file radar definitions.

## *Terrain*

### **TERRAIN:ARC:StartAngle:EndAngle:Distance:Elevation**

### **TERRAIN:LINE:StartLat:StartLon:EndLat:EndLon:Elevation**

Sets the radar station to disregard ESE file radar data and defines a terrain obstruction for the radar, either as a radial arc or a line segment of constant elevation value. The arc format is more performance-friendly so it's the preferred choice whenever possible.

When one or more *terrain* lines are defined, all targets within the radar's visibility area that are not masked by the defined terrain will be displayed regardless of the ESE file radar definitions.

- StartAngle                Arc starting angle (degrees true, decimal value)
- EndAngle                 Arc ending angle (degrees true, decimal value)
- Distance                Arc radius from radar station (nautical miles, decimal value)
- StartLat                Line starting point <latitude>
- StartLon                Line starting point <longitude>
- EndLat                 Line ending point <latitude>
- EndLon                 Line ending point <longitude>
- Elevation               Terrain elevation (feet AMSL, decimal value)

## 2.8 TopSkySettings.txt & TopSkySettingsLocal.txt

These two files allow changing the plugin settings. The difference between them is that the settings in the first file are loaded every time, while the settings in the “Local” file are only loaded by performing a “Sign In” which is by default done automatically, but this can be inhibited in the settings.

The available settings, their default values and acceptable ranges are described in an Excel spreadsheet provided together with this document.

The settings in the files can be either general or login callsign specific. General settings have to be located at the beginning of the file before any login callsign specific ones. Login callsign specific ones are defined by creating sections starting with a line that contains a text string in square brackets above them.

The login callsign specific settings are checked by comparing the text string against the login callsign. If the login callsign contains the text, any settings after that are loaded until a new line with text in square brackets is found which after the check is done again. If not, all the settings in that section are skipped.

An example TopSkySettings.txt file, and how it's interpreted by the plugin for login callsign “EFIN\_D\_CTR”:

Setting1=0	
Setting2=123	
[_CTR]	New section start
Setting2=100	
Setting3=0	
[EFIN_]	New section start
Setting2=200	
[ESSA_TWR]	New section start
Setting2=300	
Setting4=0	

- The first part of the file has no callsign restrictions, so the settings are valid for any callsign
  - “Setting1” is set to “0” and “Setting2” to “123”
- The first section (“\_CTR”) is a match
  - “Setting2” will be changed to “100” and “Setting3” is set to “0”.
- The second section (“EFIN\_”) is also a match
  - “Setting2” is changed once again, this time to “200”.
- The last section (“ESSA\_TWR”) is not a match so the settings there won’t be applied.

Settings can appear in the file more than once, and be set more than once depending on how the file is laid out. As the file is always read in the order it is written, the more specific sections should be at the bottom (like in the example, any \_CTR callsign will get a different “Setting2” value than the other callsigns, but if it happens to be EFIN\_CTR or EFIN\_<anything here>\_CTR, the value is different from the other \_CTR callsigns).

Sections can also start with a position ID enclosed in curly brackets – a section starting with “{D}” will be read if the current position ID is “D”. Partial matches are not supported.

When the plugin detects that the login callsign has changed, the settings files are automatically reloaded.

## 2.9 TopSkySSRcodes.txt

The file contains the SSR code range and area definitions to be used when assigning transponder codes via the plugin. The following example shows the syntax (optional lines in grey color):

AREA:EF1	Area
RADIUS:15.0	Radius
N060.00.00.000 E025.00.00.000	Coordinate
// domestic secondary	Comment
RANGE:3201:3277	Range
ADES:EF	ADES
PROTECTION:EF1	Protection
PRIORITY:-1	Priority
IFR	IFR

The plugin's SSR code assignment system checks the flightplan, finds out which code ranges are available for that flightplan, checks for codes already in use, and then assigns one of the available codes. Transponder codes ending with "00" are not assigned.

The example area above designates the code range 3201-3277 available for IFR traffic with destinations starting with "EF" and not entering the area "EF1". In addition, the range is defined to have a priority level of -1, to be used only if there are no available codes found in higher priority ranges.

### *Mode S global*

#### **MODE\_S\_GLOBAL**

If this line appears anywhere in the file, it indicates that the identity of mode S capable aircraft will be maintained using mode S information everywhere. See also *Mode S* line to limit the availability to specified area(s), and [Code 1000 assignment](#) for information on A1000 assignment.

### *Mode S airports*

#### **MODE\_S\_AIRPORTS:IcaoList**

This line lists airports capable of using code A1000 for aircraft identification. If not defined, all airports are considered capable. More than one *Mode S airports* line can be used, the information in them is combined. The airspace's capability must also be defined using a *Mode S global* line or areas containing *Mode S* lines. See also [Code 1000 assignment](#) for information on A1000 assignment.

- IcaoList      Comma-separated list of airport codes (partial matches from beginning are supported, such as "EF" matching all airports beginning with "EF")

## *Mode S airports exclude*

### **MODE\_S\_AIRPORTS\_EXCLUDE:IcaoList**

This line lists airports not capable of using code A1000 for aircraft identification (exceptions to *Mode S airports*). If not defined, all airports are considered capable. More than one *Mode S airports exclude* line can be used, the information in them is combined. See also [Code 1000 assignment](#) for information on A1000 assignment.

- IcaoList              Comma-separated list of airport codes (partial matches from beginning are supported, such as “EF” matching all airports beginning with “EF”)

## *Group*

### **GROUP:GroupName:Item<sub>1</sub>:Item<sub>2</sub>:Item<sub>3</sub>:...**

The *group* line can be used as a shortcut to writing a large number of text entries. It can be used in line types where lists of text strings are used. To use a group in a line, enter “GROUP\_<groupname>” like any other text string. It will be automatically expanded to the list of text strings in the group definition.

- GroupName              Name for the group
- Item<sub>x</sub>                Text strings

*Note: the item separator to be used here is the colon (:), regardless of what's used in the target line type.*

## *Area*

### **AREA:AreaName**

### **AREA:AreaName:Bottom:Top**

Each area definition must start with an *area* line that defines the area name. The area must be defined in the file before it is referred to in a code range definition. The area names are case sensitive.

- AreaName                Area name to use in the code assignment rules (text string)
- Bottom                  Area bottom altitude (feet, defaults to -1000ft)
- Top                     Area top altitude (feet, defaults to 999999ft)

## *Mode S*

### **MODE\_S**

This line indicates that inside this area the identity of mode S capable aircraft will be maintained using mode S information. If this line is not present, the area will be non-mode S capable. See [Code 1000 assignment](#) for information on A1000 assignment.

## *Radius*

### **RADIUS:Radius**

If the area is a circle, it can be defined as a center point and a distance from it. In this case the area definition needs the *radius* line and one *coordinate* line (see below). All other area shapes need to be defined as polygons using three or more *coordinate* lines, and then the *radius* line shall not be used.

- Radius                      Area radius (nautical miles, decimal value)

## *Coordinate*

### **COORD:Lat:Lon**

#### **Lat Lon**

Each area definition must have either at least three *coordinate* lines, or one *coordinate* line and a *radius* line (see above). There is practically no upper limit for the number of coordinate points, but as the required calculations increase proportionally to the number of points, it's best to keep the areas simple. In the second version, the *Lat* and *Lon* values need to be separated by one or more spaces, and the line can optionally have one or more spaces before the *Lat* value.

- Lat                        <latitude>
- Lon                        <longitude>

## *Callsign\_Code*

### **CALLSIGN\_CODE:Callsign:Code**

A specific code can be forced to a callsign using this line. The defined code will always be assigned to this callsign even if it results in a duplicate code assignment, except when the code is already assigned to this callsign, in which case another suitable code is assigned.

- Callsign                    FPL callsign
- Code                        Assigned code (4 octal digits, 0000-7777)

## *Range*

### **RANGE:StartCode:EndCode**

This line is the only mandatory line for a code range definition and must always be the first line in a definition. It starts the definition by specifying the range of codes in it.

- StartCode                  First code in the range (4 octal digits, 0001-7777)
- EndCode                    Last code in the range (4 octal digits, 0001-7777)

## *Adhoc*

### **ADHOC**

This line causes the codes in this range to be assigned only to flightplans where the departure, destination or both are empty.

## *Priority*

## PRIORITY:Level

This line sets the priority level of this range. When more than one range of codes is suitable for a flight, codes are assigned based on the range priority level, with codes from lower priority ranges assigned only when no codes are available in higher priority ranges. The default priority level of a code range is zero.

- Level Priority level (-3 to +3, +3 being the highest priority)

*IFR*

IFR

This line causes the codes in this range to be assigned only to IFR flightplans.

VFR

VFR

This line causes the codes in this range to be assigned only to VFR flightplans.

GAT

GAT

This line causes the codes in this range to be assigned only to GAT (general air traffic) flightplans. By default, a flightplan is categorized as GAT when “RMK/OAT” is not found in the remarks section.

OAT

OAT

This line causes the codes in this range to be assigned only to OAT (operational air traffic) flightplans. By default, a flightplan is categorized as OAT when “RMK/OAT” is found in the remarks section.

## *Direction*

## DIRECTION:TTrk1:TTrk2

This line limits the code assignment to flights having a track between the two specified true tracks (clockwise direction from Trk1 to Trk2). If *via* and/or *NotVia* lines are also present in the code range, the track to be checked is the outbound track from the specified point(s). If not, the tracks are checked against the aircraft's planned track from its present position.

- TTrk1 Start angle for the track range (degrees true, decimal value)
  - TTrk2 End angle for the track range (degrees true, decimal value)

## *ADEP*

### **ADEP:ICAOcode:ICAOcode:ICAOcode:...**

This line limits the code assignment to flights departing from one of the defined airports. The whole ICAO airport code is not needed; the match can also be done on the first one or more letters, e.g. entering "EF" will match all airports with ICAO designators beginning with "EF". The *ADEP* line can contain one or more airport codes and one code range definition can also have more than one *ADEP* line if necessary.

- ICAOcode              Airport ICAO code (complete or partial)

## *NotADEP*

### **NOTADEP:ICAOcode:ICAOcode:ICAOcode:...**

This line limits the code assignment to flights not departing from any of the defined airports. Otherwise the format and limitations are the same as in the *ADEP* line.

- ICAOcode              Airport ICAO code (complete or partial)

## *ADES*

### **ADES:ICAOcode:ICAOcode:ICAOcode:...**

This line limits the code assignment to flights arriving at one of the defined airports. Otherwise the format and limitations are the same as in the *ADEP* line.

- ICAOcode              Airport ICAO code (complete or partial)

## *NotADES*

### **NOTADES:ICAOcode:ICAOcode:ICAOcode:...**

This line limits the code assignment to flights not arriving at any of the defined airports. Otherwise the format and limitations are the same as in the *ADEP* line.

- ICAOcode              Airport ICAO code (complete or partial)

## *Local*

### **LOCAL:ICAOcode:ICAOcode:ICAOcode:...**

This line limits the code assignment to local flights (ADEP=ADES) from one of the defined airports. The whole ICAO airport code is not needed; the match can also be done on the first one or more letters, e.g. entering "EF" will match all airports with ICAO designators beginning with "EF". The *local* line can contain one or more airport codes and one code range definition can also have more than one *local* line if necessary. The *local* line cannot be used together with *ADEP* or *ADES* lines.

- ICAOcode              Airport ICAO code (complete or partial)

## *Via*

### **VIA:Point:Point:Point:....**

This line limits the code assignment to flights routing via at least one of the defined points. The point can be anywhere along the flightplan. One or more points can be defined in one *via* line and one code range definition can contain more than one *via* line if necessary.

- Point                          Point name (Fix, VOR, NDB or airport)

## *NotVia*

### **NOTVIA:Point:Point:Point:....**

This line limits the code assignment to flights not routing via any of the defined points. Otherwise the format and limitations are the same as in the *via* line.

- Point                          Point name (Fix, VOR, NDB or airport)

## *AreaVia*

### **AREAVIA:AreaName:AreaName:AreaName:....**

This line limits the code assignment to flights routing via at least one of the defined areas. The area(s) must have been defined earlier in the data file. One or more areas can be defined in one *AreaVia* line and one code range definition can contain more than one *AreaVia* line if necessary.

- AreaName                      Area name (text string)

## *Protection*

### **PROTECTION:AreaName:AreaName:AreaName:....**

This line limits the code assignment to flights not routing via any of the defined areas. Otherwise the format and limitations are the same as in the *AreaVia* line.

- AreaName                      Area name (text string)

## *ADEParea*

### **ADEPAREA:AreaName:AreaName:AreaName:....**

This line limits the code assignment to departing from one of the defined areas. The area(s) must have been defined earlier in the data file. If the ADEP is within more than one area laterally, the code will pick the area with the lowest bottom altitude (the one found first in the data file in case of two or more areas with equal bottom altitudes). One or more areas can be defined in one *ADEParea* line and one code range definition can contain more than one *ADEParea* line.

- AreaName                      Area name (text string)

## *NotADEParea*

### **NOTADEPAREA:AreaName:AreaName:AreaName:...**

This line limits the code assignment to flights not departing from any of the defined areas. Otherwise the format and limitations are the same as in the *ADEParea* line.

- AreaName              Area name (text string)

## *ADESarea*

### **ADESAREA:AreaName:AreaName:AreaName:...**

This line limits the code assignment to flights arriving in one of the defined areas. The area(s) must have been defined earlier in the data file. If the ADES is within more than one area laterally, the code will pick the area with the lowest bottom altitude (the one found first in the data file in case of two or more areas with equal bottom altitudes). One or more areas can be defined in one *ADESarea* line and one code range definition can contain more than one *ADESarea* line.

- AreaName              Area name (text string)

## *NotADESarea*

### **NOTADESAREA:AreaName:AreaName:AreaName:...**

This line limits the code assignment to flights not arriving in any of the defined areas. Otherwise, the format and limitations are the same as in the *ADESarea* line.

- AreaName              Area name (text string)

## *Unit*

### **UNIT:LoginCallsign:LoginCallsign:LoginCallsign:...**

This line limits the code assignment based on your network login callsign. The whole callsign is not needed; the match can also be done on the first one or more letters, e.g. entering “EFIN” will match callsigns beginning with “EFIN”. The *unit* line can contain one or more callsigns and one code range definition can also have more than one *unit* line if necessary.

- LoginCallsign        Current login callsign (complete or partial)

## *NotUnit*

### **NOTUNIT:LoginCallsign:LoginCallsign:LoginCallsign:...**

This line limits the code assignment to network login callsigns other than the specified ones. Otherwise, the format and limitations are the same as in the *unit* line.

- LoginCallsign        Current login callsign (complete or partial)

## *Callsign*

### **CALLSIGN:Callsign:Callsign:Callsign:...**

This line limits the code assignment based on the aircraft's callsign. The whole callsign is not needed; the match can also be done on the first one or more letters, e.g. entering "FIN" will match callsigns beginning with "FIN". The *callsign* line can contain one or more callsigns and one code range definition can also have more than one *callsign* line if necessary.

- Callsign                      Aircraft callsign (complete or partial)

## *NotCallsign*

### **NOTCALLSIGN:Callsign:Callsign:Callsign:...**

This line limits the code assignment to aircraft callsigns other than the specified ones. Otherwise, the format and limitations are the same as in the *callsign* line.

- Callsign                      Aircraft callsign (complete or partial)

## *ATYP*

### **ATYP:Atyp:Atyp:Atyp:...**

This line limits the code assignment based on the aircraft's type. The whole type is not needed; the match can also be done on the first one or more letters, e.g. entering "A3" will match types beginning with "A3". The *ATYP* line can contain one or more types and one code range definition can also have more than one *ATYP* line if necessary.

- Atyp                         Aircraft type designator (complete or partial)

## *NotATYP*

### **NOTATYP:Atyp:Atyp:Atyp:...**

This line limits the code assignment to aircraft types other than the specified ones. Otherwise, the format and limitations are the same as in the *ATYP* line.

- Atyp                         Aircraft type designator (complete or partial)

## *Descr*

### **DESCR>List**

This line limits the code assignment based on the aircraft's description.

- List                    Allowed description letters. Any combination of the following can be used:
  - L                  landplane
  - S                  seaplane
  - A                  amphibian
  - H                  helicopter
  - G                  gyrocopter
  - T                  tilt-wing aircraft
  - ?                  unknown

## *NotDescr*

### **NOTDESCR>List**

This line limits the code assignment to aircraft with descriptions other than the specified ones. Otherwise, the format is the same as in the *descr* line.

- List                    Forbidden description letters

## *Remarks*

### **REMARKS:Text:Text:Text:...**

This line limits the code assignment based on the flightplan remarks. When all the specified text strings are found in the remarks section, the line is a match. The *remarks* line can contain one or more text strings and one code range definition can also have more than one *remarks* line if necessary (in this case it is enough that one of the lines is a match for the code range to be used).

- Text                  Text string to look for in the flightplan remarks

## *NotRemarks*

### **NOTREMARKS:Text:Text:Text:...**

Same as above but limits the code assignment to flightplans whose remarks section contains none of the specified text strings. When more than one *NotRemarks* line is used in a code range, the range is used when even one of the lines is a match.

- Text                  Text string to look for in the flightplan remarks

### **2.9.1 Code 1000 assignment**

The plugin will only assign the mode S conspicuity code A1000 when specific conditions are met. These conditions are:

- The aircraft must be mode S equipped
- The predicted future flightpath must enter at least one defined mode S area
- The predicted future flightpath must not enter any non-mode S areas
- If any mode S airports are defined, the aircraft's destination must be one of them
- If any non-mode S airports are defined, the aircraft's destination must not be one of them

Additionally, if the aircraft has not yet departed:

- If any mode S airports are defined, the aircraft's departure airport must be one of them
- If any non-mode S airports are defined, the aircraft's departure airport must not be one of them

For the flightpath check, if the data file contains overlapping areas containing a given point in space, the mode S capability at that point is determined according to the area defined first in the data file. If a checked point is not within any defined area, it will be disregarded.

## 2.10 TopSkySTCA.txt

The file contains the definitions for final approach areas where smaller (usually 2.5nm) separation can be used, as well as STCA exclusion areas needed for parallel approaches and departures. The following example is used to show the syntax (optional lines in grey color):

FINALAPP:EFHK:04L	FinalApp
// EFHK 04's	Comment
SOIR:EFHK:04L/2300:04R/3300	SOIR
NOZ1:N060.18.37.790:E024.54.30.240	NOZ 1
NOZ1:N060.19.21.880:E024.53.08.680	NOZ 1
NOZ1:N060.12.36.560:E024.38.23.030	NOZ 1
NOZ1:N060.11.52.460:E024.39.44.310	NOZ 1
NOZ2:N060.18.49.470:E024.55.54.640	NOZ 2
NOZ2:N060.18.05.350:E024.57.16.110	NOZ 2
NOZ2:N060.11.20.250:E024.42.29.640	NOZ 2
NOZ2:N060.12.04.370:E024.41.08.450	NOZ 2

### *FinalApp*

#### FINALAPP:AirportICAO:RwyID:Range:XTE:Lat:Lon:CourseT

This line creates a final approach area where smaller separation values for STCA are used.

- AirportICAO      Airport ICAO code
- RwyID              Runway identifier
- Range              Maximum range from the approach end point (nm, decimal value, range 0.0-99.0)
- XTE                Maximum cross-track error from the approach course (nm, decimal value, range 0.0-99.0)
- Lat                 Approach end point <latitude>
- Lon                 Approach end point <longitude>
- CourseT           Approach course (degrees true, decimal value)

Only the ICAO code and RwyID are mandatory to be defined, so for example “FINALAPP:EFHK:04L” creates an approach area for that runway with default values. The default values for the other items are:

- Range              10.0 nm
- XTE                0.5 nm
- Lat, Lon           Runway threshold coordinates from the active sector file
- CourseT           Runway true bearing calculated from the threshold coordinates

Note that if one or more of the optional values needs to be changed, the line must include all the values up to the last changed value.

An STCA alert between two aircraft will be inhibited when all of the following are true:

- Both aircraft are planned to land on the runway in question
- Both aircraft are inside the same approach area

- Both aircraft have ground tracks within 10 degrees of the approach course
- The aircraft closer to the runway is not wake turbulence category HEAVY or SUPER, or a B757
- The aircraft further from the runway is in the same or higher wake turbulence category

## *SOIR*

### **SOIR:AirportICAO:RwyID1:RwyID2:Range:WidthIn:WidthOut:MinSep**

This line starts the definition.

- |               |   |
|---------------|---|
| ○ AirportICAO | Airport ICAO code   |
| ○ RwyID1      | Identifier of the runway on the left side in the direction of flight            |
| ○ RwyID2      | Identifier of the runway on the right side in the direction of flight           |
| ○ Range       | Length of the NOZ (nm, decimal value, range 0.0-99.0)                           |
| ○ WidthIn     | Width of the NOZ toward the other runway (nm, decimal value, range 0.0-99.0)    |
| ○ WidthOut    | Width of the NOZ away from the other runway (nm, decimal value, range 0.0-99.0) |
| ○ MinSep      | Minimum separation between tracks   |

Only the ICAO code and runway identifiers are mandatory to be defined. The default values for the other items are:

- Range 10.0 nm
- WidthIn / WidthOut 0.3 nm
- MinSep 0.33 nm

Note that if one or more of the optional values needs to be changed, the line must include all the values up to the last changed value. A custom polygon can also be defined as the NOZ, see the NOZ1 and NOZ2 lines.

An optional altitude value (feet AMSL) can be appended to the runway ID field to be used as the cleared level when the actual cleared level is “cleared for approach”. The format is “RwyIDx/altitude”, e.g. “22L/3000”. This may help to prevent nuisance STCA alerts when an aircraft with an approach clearance is not yet established on the final approach.

For arrivals, an STCA warning will be inhibited when all of the following conditions are true:

- One track is within NOZ1 and the other within NOZ2
- The separation between the tracks is not less than MinSep
- The ground tracks of both aircraft are within a specified (default 10 degrees) tolerance from the approach tracks

For departures, the MinSep value does not apply and the track error limit is fixed at 10 degrees and only valid toward the other runway. Both runways also need to be set as active for arrival (for an arrival setup) or departure (for a departure setup) in the EuroScope runway setup. Both arrival and departure setups can be active simultaneously.

If the airport has more than two runways where simultaneous approaches are performed, each possible pair must be defined separately. For example, if an airport can do simultaneous approaches on any combination of runways 01L, 01C and 01R, three definitions are required (one for 01L/01C, one for 01L/01R and one for 01C/01R).

## *Departure*

### **DEPARTURE**

This optional line defines the setup as a departure setup. If not present, the setup is an arrival setup.

## *Approach course 1 and 2*

### **CRS1:CourseT**

### **CRS2:CourseT**

This optional line is used to define the approach course if it is different from the runway bearing. For it to have any effect, the setup must be an arrival setup and the NOZ for the corresponding runway must also be defined manually (see below).

- CourseT                          Approach course for the runway (degrees true, decimal value)

## *Final length 1 and 2*

### **FINAL1:Length**

### **FINAL2:Length**

This optional line, used for approaches with curved segments inside the NOZ, is used to define the length of the final straight part of the approach. The track error check from the approach course will only be done within this distance from the threshold.

- Length                              Final segment length for the approach (nm, decimal value, 0.0-99.0)

## *NOZ 1 and 2*

### **NOZ1:Lat:Lon**

### **NOZ2:Lat:Lon**

These optional lines are used to define vertices for custom polygons to override the default NOZ areas for the runways.

- Lat                                <latitude>
- Lon                                <longitude>

## *NTZ*

### **NTZ:Lat:Lon**

This optional line is used to define vertices for a custom polygon to override the default NTZ area. It may only be used in an arrival setup. The NTZ is used to extend STCA alerting below its normal lower limit altitude when an aircraft is inside the NTZ.

- Lat                                <latitude>
- Lon                                <longitude>

## 2.11 TopSkySymbols.txt

This file makes it possible to change the default symbols drawn by the plugin. The following example symbol shows the syntax:

// distance marker	Comment
SYMBOL:MARKER	Type
ELLIPSE:0:0:2	Definition

### *Symbol definition*

#### **SYMBOL:SymbolType**

The first line for each symbol definition must be a type line.

- **SymbolType**      Symbol type (one of the following):

##### Track symbols

▪ PRIMARY	PSR tracks
▪ PRIMARY_DIV	Diverging PSR tracks
▪ NODAPS	Tracks without valid DAPs
▪ NODAPS_DIV	Diverging tracks without valid DAPs
▪ NODAPS_SPI	SPI tracks without valid DAPs
▪ DAPS	Tracks with valid DAPs
▪ DAPS_DIV	Diverging tracks with valid DAPs
▪ DAPS_SPI	SPI tracks with valid DAPs
▪ ADSB	ADS-B only tracks
▪ ADSB_DIV	Diverging ADS-B only tracks
▪ ADSB_SPI	SPI ADS-B only tracks
▪ UNCONTROLLED	Uncontrolled tracks
▪ COASTED	Coasting tracks
▪ FPASD_UNCONTROLLED	Uncontrolled FPASD tracks
▪ FPASD_CONTROLLED	Controlled FPASD tracks

##### Other symbols

▪ HISTORY	History dot symbol
▪ AIRPORT	Airport symbol
▪ HOTSPOT	Area hotspot symbol
▪ FIX	Fix symbol
▪ NDB	NDB symbol
▪ VOR	VOR symbol
▪ MARKER	Runway approach line distance marker symbol
▪ LEADER	Leader line used in automatically generated maps

The symbol itself can consist of various elements, drawn by the following lines. The X and Y coordinates are relative to the symbol centerpoint, with the X axis having increasing values to the right and the Y axis having increasing values to the down direction. The commands are the same as in the EuroScope Symbology dialog with the exception of the possibility to draw elliptical arcs and the ":" separating the values here so the ES dialog can be used in most cases to test the results.

## **MOVETO:X:Y**

Sets the starting point for the next LINETO command

- X Number of pixels from the symbol centerpoint in the left(-)-right(+) direction
- Y Number of pixels from the symbol centerpoint in the up(-)-down(+) direction

## **LINETO:X:Y**

Draws a straight line from the previous position

- X Number of pixels from the symbol centerpoint in the left(-)-right(+) direction
- Y Number of pixels from the symbol centerpoint in the up(-)-down(+) direction

## **SETPIXEL:X:Y**

Paints the selected pixel

- X Number of pixels from the symbol centerpoint in the left(-)-right(+) direction
- Y Number of pixels from the symbol centerpoint in the up(-)-down(+) direction

## **ARC:X:Y:Radius:StartAngle:EndAngle**

## **ARC:X:Y:Radius<sub>X</sub>:Radius<sub>Y</sub>:StartAngle:EndAngle**

Draws a part of a circle or ellipse

- X Centerpoint offset from the symbol centerpoint in the left(-)-right(+) direction
- Y Centerpoint offset from the symbol centerpoint in the up(-)-down(+) direction
- Radius Arc radius in pixels (to make a circular arc)
- Radius<sub>X</sub> Arc radius in relation to the X axis in pixels (to make an elliptical arc)
- Radius<sub>Y</sub> Arc radius in relation to the Y axis in pixels (to make an elliptical arc)
- StartAngle Arc starting angle (integer degrees, 0 degrees is at positive X-axis, increasing counterclockwise)
- EndAngle Arc ending angle (integer degrees, 0 degrees is at positive X-axis, increasing counterclockwise)

## **FILLARC:X:Y:Radius:StartAngle:EndAngle**

## **FILLARC:X:Y:Radius<sub>X</sub>:Radius<sub>Y</sub>:StartAngle:EndAngle**

Otherwise the same as ARC, but the result is filled

**ELLIPSE:X:Y:Radius****ELLIPSE:X:Y:Radius<sub>x</sub>:Radius<sub>y</sub>**

Otherwise the same as FILLARC, but always draws a complete circle or ellipse

**FILLRECT:Left:Top:Right:Bottom**

Draws a filled rectangle

- |          |   |
|----------|---|
| ○ Left   | Left edge offset from the symbol centerpoint in the left(-)-right(+) direction  |
| ○ Top    | Top edge offset from the symbol centerpoint in the up(-)-down(+) direction      |
| ○ Right  | Right edge offset from the symbol centerpoint in the left(-)-right(+) direction |
| ○ Bottom | Bottom edge offset from the symbol centerpoint in the up(-)-down(+) direction   |

**POLYGON:X<sub>1</sub>:Y<sub>1</sub>: X<sub>2</sub>:Y<sub>2</sub>:...: X<sub>n</sub>:Y<sub>n</sub>**

Draws a filled polygon with n vertices

## 2.12 TopSkyViews.txt

The file contains the definitions for the items in the View Window. Two types of definitions are allowed, enter only one definition per line:

### *Lat/Lon box*

**VIEW:ViewName:Lat<sub>min</sub>:Lon<sub>min</sub>:Lat<sub>max</sub>:Lon<sub>max</sub>**

An area bounded by the minimum and maximum latitude and longitude values. The resulting screen area covers at least the required coordinates, possibly more depending on the screen shape.

- ViewName              Name to identify the view in the View Window
- Lat<sub>min</sub>              Minimum <latitude>
- Lon<sub>min</sub>              Minimum <longitude>
- Lat<sub>max</sub>              Maximum <latitude>
- Lon<sub>max</sub>              Maximum <longitude>

### *Centerpoint and range*

**VIEW:ViewName:Lat:Lon:Range**

**VIEW:ViewName:PointName:Range**

An area defined by a range from a centerpoint. The resulting screen area will be centered on the required centerpoint and will show at least the required distance to every direction from the centerpoint.

- ViewName              Name to identify the view in the View Window
- Lat                    Centerpoint <latitude>
- Lon                    Centerpoint <longitude>
- PointName            Centerpoint <position>
- Range                The displayed range (nautical miles, decimal value)

*Note: the syntax to define a runway threshold as a PointName is the 4-letter ICAO airport designator followed by a forward slash and the runway identifier.*

## 2.13 ICAO\_Aircraft.txt

This file contains aircraft type information, and is used by the Flight Plan Window and Create APL Window for checking entries and as a data source for the Document Viewer Window. The format of the file is the same as in the one used by EuroScope.

As an alternative to having the aircraft data, this file can contain the location of another file that has the data. In that case, this file should have only one line, containing the data file location. The location can be defined as absolute or relative. Relative locations starting with ".\" or "..\" are relative to the plugin folder. The pointed file must contain the aircraft data, not a path to another file.

## 2.14 ICAO\_Aircraft.json

This file contains more detailed information on the aircraft types. It is a JSON file containing an array of objects with the following keys:

Key	Data type	Description
○ ICAO	string	Type designator (mandatory item)
○ Description	string	Three-character description <ul style="list-style-type: none"><li>▪ First character – description: A (Amphibian), G (Gyrocopter), H (Helicopter), L (Landplane), S (Seaplane) or T (Tiltrotor)</li><li>▪ Second character – engine count: 1-8 or C (Two engines coupled to drive a single propeller system)</li><li>▪ Third character – engine type: E (Electric), J (Jet), P (Piston), R (Rocket) or T (Turboprop/turboshaft)</li></ul>
○ WTC	string	Wake turbulence category <ul style="list-style-type: none"><li>▪ L, M, H or J</li></ul>
○ WTG	string	ICAO wake turbulence group <ul style="list-style-type: none"><li>▪ A, B, C, D, E, F or G</li></ul>
○ RECAT-EU	string	RECAT-EU wake turbulence group <ul style="list-style-type: none"><li>▪ A, B, C, D, E or F</li></ul>
○ Wingspan	number	Wingspan in meters
○ Length	number	Length in meters
○ Height	number	Height in meters
○ MTOW	number	Maximum take-off weight in kilograms
○ Use	string	Typical use(s) for the aircraft <ul style="list-style-type: none"><li>▪ One or more of the following: A (Airliner/commuter), B (Business/corporate), C (Cargo), H (Helicopter, other than military), I (Military helicopter), M (Military, other than helicopter), P (Private), T (Military tanker/transport)</li></ul>
○ IATA	string	IATA designator
○ IATA_cargo	string	IATA designator when used as cargo aircraft
○ Manufacturer	string	Manufacturer name
○ Model	string	Aircraft model name(s) for this type designator

The “ICAO” key is the only mandatory one. Keys that are irrelevant or whose values are not known can be left out.

## 2.15 ICAO\_Airlines.txt & ICAO\_Airlines\_Virtual.txt

These files contain the radiotelephony callsigns to be displayed in the track labels, and is used as a data source for the Document Viewer Window. By default, if an identifier is found in both files, the information in the "ICAO\_Airlines.txt" file is used. The following example line shows the syntax:

**AAB**      **Abelag Aviation**      **ABG**      **Callsign definition**

The format of the files is the same as in the “ICAO\_Airlines.txt” file provided with EuroScope. Only one callsign must be defined per line.

## *Callsign definition*

**ThreeLetterID<tab>OperatorName<tab>Callsign**



As an alternative to having the callsign data, this file can contain the location of another file that has the data. In that case, this file should have only one line, containing the data file location. The location can be defined as absolute or relative. Relative locations starting with ".\" or "..\" are relative to the plugin folder. The data in the pointed file must be in the above format, and the pointed file must contain the callsign data, not a path to another file.

## 2.16 ICAO\_Airports.txt

This file contains airport information, and is used as a data source for the Document Viewer Window. The format of the file is the same as in the one used by EuroScope.

As an alternative to having the airport data, this file can contain the location of another file that has the data. In that case, this file should have only one line, containing the data file location. The location can be defined as absolute or relative. Relative locations starting with “`\`” or “`..\`” are relative to the plugin folder. The pointed file must contain the airport data, not a path to another file.

## 2.17 isec.txt

This file contains waypoint information, and is used as a data source for the plugin maps (currently only for the auto-generated NAT maps). The format of the file is the same as in the one used by EuroScope.

As an alternative to having the airport data, this file can contain the location of another file that has the data. In that case, this file should have only one line, containing the data file location. The location can be defined as absolute or relative. Relative locations starting with “`\`” or “`..\`” are relative to the plugin folder. The pointed file must contain the airport data, not a path to another file.

### 3 Sound files

The plugin uses the following sound files if present in the same folder as the plugin dll:

- |                           |  |
|---------------------------|--|
| ○ TopSkySoundAPW.wav      | APW warning sound                              |
| ○ TopSkySoundCoord.wav    | Coordination received sound (ROF, RTI and TIP) |
| ○ TopSkySoundCoordACP.wav | Coordination accepted sound (RTI and TIP)      |
| ○ TopSkySoundCoordRJC.wav | Coordination rejected sound (ROF, RTI and TIP) |
| ○ TopSkySoundCPDLC.wav    | CPDLC/DCL message sound                        |
| ○ TopSkySoundSTCA.wav     | STCA warning sound                             |

The limitations for the used files are that the file must fit into available physical memory and be playable by an installed waveform-audio device driver. If a specific file is not found, no sound is played.

### 4 Cursor files

Six custom cursors are used:

- |                              |   |
|------------------------------|---|
| ○ TopSkyCursorArrowLeft.cur  | General purpose cursor                            |
| ○ TopSkyCursorArrowRight.cur | Used when over an opened menu                     |
| ○ TopSkyCursorCross.cur      | Used when drawing vectors, selecting points, etc. |
| ○ TopSkyCursorMove.cur       | Used when moving windows                          |
| ○ TopSkyCursorStop.cur       | Used when no input is allowed in that area        |
| ○ TopSkyCursorResize.cur     | Used when resizing windows                        |

If even one of the cursor files is not found in the plugin dll folder, none of the custom cursors will be used.

### 5 External access

It is possible to trigger a transponder code assignment from another plugin, using “StartTagFunction” with the following parameters:

- |                       |                          |
|-----------------------|--------------------------|
| ○ sFunctionPlugInName | TopSky plugin            |
| ○ functionID          | 667                      |
| ○ sItemString         | Callsign of the aircraft |

The “sCallsign” parameter is not used by the plugin, but as StartTagFunction will set that text string (case sensitive!) as the ASEL callsign, it is recommended to set a reasonable value to it – for example the current ASEL callsign, the same as sItemString, or an empty string. NULL will cause an exception.

The function will trigger an automatic assignment if all the following conditions are met (otherwise the function will do nothing):

- SSR menu is not open
- A code assignment is currently not in progress
- At least 2 seconds have elapsed from the previous call to this function
- Both a valid CFlightPlan and a CRadarTarget are found for the requested callsign
- You are a controller
- The aircraft is not tracked by someone else

## Appendix 1: Label functions

The following table lists the available tag function names and functionality.

Name	Function
<b>General purpose functions</b>	
Ack RJC or FAIL AHDG coord / Clear value / Assign present heading	If there has been a rejected or failed tactical AHDG coordination, acknowledges the warning Else if the AHDG field contains data, clears it Else if enabled in settings, assigns “present heading”
Ack RJC or FAIL ARC coord / Clear value	If there has been a rejected or failed tactical ARC coordination, acknowledges the warning Else if the ARC field contains data, clears it
Ack RJC or FAIL ASP coord / Clear value	If there has been a rejected or failed tactical ASP coordination, acknowledges the warning Else if the ASP field contains data, clears it
Acknowledge AIW alert	Acknowledges AIW alert
Acknowledge AMAN highlight	Acknowledges the AMAN values highlight
Acknowledge PFREQ	Acknowledges (clears) the PFREQ label field
Acknowledge SID allocation	Acknowledges new SID allocation
Acknowledge STAR allocation	Acknowledges new STAR allocation
Assume/Transfer	Assumes track if a transfer is in progress, Initiates transfer if track is assumed
Clear approach clearance	Clears approach clearance flag
DSQ number decrement	Decrement the DSQ number by one
DSQ number increment	Increments the DSQ number by one
Edit ATIS letter	Opens pop-up to edit received ATIS letter
Edit FText	Opens pop-up to edit FText field
Edit OP_TEXT	Opens pop-up to edit OP_TEXT field
Edit OP_TEXT2	Opens pop-up to edit OP_TEXT2 field
Find ASSR	Starts Find Track function with this track's ASSR
Find PSSR	Starts Find Track function with this track's PSSR
FSQ Re-sequence	Starts a FAST re-sequence
FSQ Remove	Removes track from FAST sequence
Invoke SEP tool	Starts new SEP tool from this track
Invoke SEP tool with VSEP	Starts new SEP tool with VSEP from this track (COOPANS version only)
No function	No function. This should be set as the left-click function for any field that has any right-click function and no left-click function is needed.
Open AFL menu	Opens the AFL menu if no surveillance-based altitude information is available
Open AFL menu / Toggle AFL highlight	Opens the AFL menu if no surveillance-based altitude information is available, otherwise toggles highlight of AFL field
Open AFL menu / Toggle enhanced AFL display	Opens the AFL menu if no surveillance-based altitude information is available, otherwise toggles the enhanced AFL display (showing both pressure and barometric altitude)

Name	Function
Open AHDG menu	Opens the AHDG menu  (DEP List) : default value is the departure rwy heading
Open ARC menu	Opens the ARC menu
Open ASP menu	Opens the ASP menu
Open ASSR menu	Opens the ASSR menu
Open Callsign menu	Opens the Callsign menu
Open CFL menu	Opens the CFL menu  (DEP List) : default value is the initial climb level if specified in the airspace data file
Open CFL/PEL menu	Opens the CFL or PEL menu depending on the flight state
Open DSQ menu	Opens the DSQ menu
Open Extended label	Opens the extended label
Open Extended label (click) / Toggle minimized label (shift+click)	<b>Click</b> opens the extended label, <b>Shift+click</b> toggles between normal and minimized label (the minimized label only displays settings-file-defined label items, other ones are hidden)
Open FPL Window	Opens FPL Window
Open Group SSR Code Window	If the aircraft belongs to a formation flight, opens the Group SSR Code Window
Open Group SSR Code Window / ASSR menu	If the aircraft belongs to a formation flight, opens the Group SSR Code Window. Otherwise, opens the ASSR menu
Open NFREQ menu	Opens the NFREQ menu
Open PDC Window	Opens PDC Window
Open PEL menu	Opens the PEL menu
Open RFL menu	Opens the RFL menu
Open Stack Manager Window	Opens Stack Manager Window for this flight's holding point
Open SQ menu	Opens the SQ menu
Open Tactical Info Window	Opens Tactical Info Window (non-COOPANS version only)
Open Tactical transfer menu	Opens the Tactical transfer menu (non-COOPANS version only)
Open Time menu (x)	Opens the Time menu  (ATD), (ETD), (SLOT), (EOBT), (NBT) and (NLT) variants are available and should be used together with the corresponding list item
Open Vertical Aid Window	Opens Vertical Aid Window for this flight
Open Waypoint menu	Opens the Waypoint menu
Open XFL menu	Opens the XFL menu
Reset ATD to ETD	Sets actual departure time equal to estimated departure time in the flight plan
Toggle ACF field color	Toggles specific coloring for the ACF field
Toggle ADES display	Toggles display of ADES field in unselected label
Toggle ADES highlight	Toggles highlight of ADES field
Toggle AFL highlight	Toggles highlight of AFL field
Toggle ATYP display	Toggles display of ATYP field in unselected label

Name	Function
Toggle ATYP highlight	Toggles highlight of ATYP, N/ATYP and ATYP/W fields
Toggle ATYP/W display	Toggles display of ATYP/W field in unselected label
Toggle Callsign highlight	Toggles highlight of Callsign field
Toggle Check indicator (x)	Toggles the check indicator  (ETWR), (LOAD), (SEL), (SIL), (VFR), (TML1), (TML2) and (Resect) variants are available and should be used together with the corresponding list item
Toggle CRC display	Toggles display of CRC field in unselected label
Toggle DBPS display	Toggles display of DBPS field in unselected label
Toggle DGS display	Toggles display of DGS field in unselected label
Toggle DHDG display	Toggles display of DHDG field in unselected label
Toggle DIAS display	Toggles display of DIAS field in unselected label
Toggle DMACH display	Toggles display of DMACH field in unselected label
Toggle DRC display	Toggles display of DRC field in unselected label
Toggle DSFL display	Toggles display of DSFL field in unselected label
Toggle EAT given indicator	Toggles EAT given indicator
Toggle enhanced AFL display	Toggles enhanced AFL display (showing both pressure and barometric altitude)
Toggle EST/DEP/ABT	<p>Not departed:</p> <ul style="list-style-type: none"> <li>○ If clearance flag not set and ground state not "DEPA", opens Departure Coordination Window</li> <li>○ Else, opens the Time menu to set the actual departure time</li> </ul> <p>Departed:</p> <ul style="list-style-type: none"> <li>○ Sets actual departure time equal to estimated departure time in FPL</li> <li>○ Sets ground state to "TAXI"</li> </ul>
Toggle FCOPX display	Toggles display of FCOPX field in unselected label
Toggle FLTID/TSSR highlight	Toggles highlight of FLTID/TSSR and TSSR fields
Toggle Freq	Toggles Freq indicator
Toggle GS display	Toggles display of GS field in unselected label
Toggle GS highlight	Toggles highlight of GS field
Toggle Inbound clearance flag	Toggles Inbound clearance flag
Toggle Label display	Toggles hiding the track label for this flight (will only stay hidden if the flight is holding)
Toggle Level Band Highlight (x)	<p>Toggles Level Band Highlight function</p> <p>(AFL) : this track's level band from AFL to PEL/CFL (XFL) : this track's level band from XFL to PEL/CFL</p>
Toggle Mark	Toggles Mark indicator
Toggle Military coordination flag	Toggles Military coordination flag
Toggle minimised label	Toggles between normal and minimized label (the minimized label only displays settings-file-defined label items, other ones are hidden)
Toggle Mode S data	Enables display of mode S DAPs for this track when DAPs are not selected for display in labels. Disabled automatically when label becomes unselected.
Toggle N/ATYP display	Toggles display of N/ATYP field in unselected label

Name	Function
Toggle No Fix flag	Toggles No Fix flag
Toggle RAK indicator	Toggles RAK indicator
Toggle RFL display	Toggles display of RFL field in unselected label
Toggle Route draw (x)	Toggles display of Flight Leg  (MTCD+SAP) : displaying both types of conflicts (MTCD only) : displaying only MTCD conflicts (SAP only) : displaying only SAP conflicts  (with autohide) : automatically removes the Flight Leg from display when the mouse cursor is no longer over the label
Toggle RWY display	Toggles display of ARWY field in unselected label
Toggle SI frequency	Toggles SI field between displaying the controller ID and the primary frequency of the controller
Toggle TCT legs display	Toggles the display of TCT legs for this aircraft
Toggle TML1 HOLD/XHOLD	Toggles Traffic Management List 1 HOLD/XHOLD (COOPANS version only)
Toggle TML2 HOLD/XHOLD	Toggles Traffic Management List 2 HOLD/XHOLD (COOPANS version only)
Toggle Units	Toggles flight units between imperial and metric
Toggle WTC display	Toggles display of WTC field in unselected label
Toggle WTC highlight	Toggles highlight of WTC field
Toggle WTG highlight	Toggles highlight of WTG field
Datalink functions	
CPDLC Warning functions	Function to deal with items in the CPDLC_W field
Open CMT Pop-up	Opens the CMT Pop-up, displaying freetext remarks contained in the datalink clearance request
Open CPDLC Current Message Window (x)	Opens the CPDLC Current Message Window if the field in question contains a reply or request with a reason ("due to" something)  (AHDG), (ASP), (RFL), (NPT) and (CPDLC_W)
Open CPDLC Emergency Acknowledgement menu	Opens the Emergency Acknowledgement menu if the CPDLC_E field contains something
Open DCL Window	Opens the Departure Clearance Window
Open DCL Window / Open PDC Window	If a datalink departure clearance request has been received, opens the Departure Clearance Window Else, Opens the Pre-Departure Clearance Window
Send CPDLC Squawk SSR	If the flight is CPDLC connected and an ASSR code exists, sends a "SQUAWK <ASSR>" CPDLC message
Oceanic functions	
Acknowledge OCM	Acknowledges a new or changed clearance
Acknowledge OCM / Toggle OFL Highlight	If OCM not acknowledged, acknowledges it Else, toggles Oceanic Level Highlight function
Acknowledge/clear OCM NBT	If NBT not acknowledged, acknowledges it Else, clears the NBT field
Acknowledge/clear OCM NLT	If NLT not acknowledged, acknowledges it Else, clears the NLT field
Open Oceanic Time Restriction Window	Opens the Oceanic Time Restriction Window

Name	Function
Toggle Oceanic Level Highlight	Toggles Oceanic Level Highlight function