AIRFIELD LIGHTS TOOLBOX

Airfield Lights Toolbox (or AFLT for short) is a comprehensive utility for the creation of standard airfield lights and light arrays for Microsoft Flight Simulator and Prepare3D (collectively referred to as "FlightSim") airports. AFLT includes 3D models for the usual airfield light fixtures and approach lighting configurations from which you should be able to create just about any airfield light arrangement.

The light source in AFLT's light models are BGL_LIGHTs. Obviously, this implies FS9 technology - which presents several difficulties with FSX and P3D - both Versions 1 and 2. FS9 models that use textures with alpha channels may not render properly in FSX or P3D. AFLT circumvents this issue by attaching the FS9-based light models to either FS9- or FSX-formatted bases. Due to DX10 rendering FS9 _LM textures white, AFLT will not generate useable lights for FSX in DX10 Preview Mode. P3D Versions 2.0 and 2.1 do not support BGL_LIGHTs and do not provide an alternative. Hopefully, Lockheed Martin will see fit to rectify this deficiency. If that happens, AFLT is "P3Dv2-ready".

BGL_LIGHTs are very processing-efficient - much more so than effects. Hundreds of effects together with highly-detailed scenery can reduce FPS on all but the most high-end systems to unusable levels. BGL_LIGHTs together with simple 3D models have a minimal effect on FPS. As well, it is not currently possible to create split runway-end lights, sequenced strobes ("running-rabbit"), PAPI/VASIs or wigwags with effects. Even as simple runway and taxiway edge lights, effects are not fully satisfactory.

However, AFLT is not for everyone. Single lights (runway/taxiway edge lights) must be placed individually with a tool such as ADE. Arrays (even the sample ones) may have to be adjusted for terrain. You must "turn off" the stock lighting at your airport. (AFLT includes a tool to help with this.) And, you must have a fundamental understanding of airport design and file placement in FlightSim. (While AFLT will copy the libraries/textures to their final location, you've got to "tell" AFLT where that is.) None of these are daunting, and most experienced users of FlightSim will have no difficulty. But, without this knowledge, chances are, use of AFLT will be a frustrating experience.

INSTALLING, EXECUTING and UNINSTALLING AFLT

Installation - To install AFLT, simply copy all the files from the downloaded archive into the chosen folder. AFLT does not affect the system registry.

AFLT is a Microsoft NET.Framework 3.5 application. If NET.Framework 3.5 or later is not already installed on your computer, the “redistributable” can be downloaded from the Microsoft website at no charge.

Execution - To execute AFLT, double-click on AFLT.exe.

Depending on your computer configuration and your Windows UAC setting, Vista and Windows 7 users especially and perhaps others may require administrator privileges when running AFLT. This is almost a certainty if either FlightSim or AFLT is installed on your C: drive. If you need but do not have administrator privileges, Windows will "complain" and you will not be able to access the necessary files. To run AFLT with administrator privileges, right-click *AFLT.exe*, select “Run As …” and then “administrator”. You can also specify Run as Administrator in a shortcut.

Windows 7 users may wish to run AFLT in the XP compatibility mode. Running it otherwise may result in a "this program may not have installed correctly" message when AFLT is shut-down. Despite the error message, there is no known problem - other than the annoyance factor.

Initialization - When you shut-down AFLT for the first time, an additional file, *AFLT.ini,* will be created and saved to the AFLT folder. AFLT "remembers" key settings from one session to the next in this file. The next time you run AFLT, those settings are restored.

Automatic Updates - Whenever AFLT is started, it checks the support server to determine if a more recent release is available. If so, it will download that release with your consent. The updated release must be manually installed in the normal manner.

If you decline an update, you will be asked if you wish to be advised of future updates. If you decline, the "No Automatic Updates" item in the *AFLT.ini* file will be set to "True". To reinstate automatic update checking, manually edit *AFLT.ini* to set this item to "False".

Un-Installation - To uninstall AFLT, just delete the AFLT folder and all its contents.

STOCK FILES

Airfields Lights Toolbox comes with everything you'll need to start creating airfield lighting elements.

- 3D base models (FS9 and FSX/P3D) and textures for the following (unlighted) fixtures are included:
  - omni-directional light for runway and taxiway edge lighting
  - dual-headed obstruction light
  - split light for runway ends
  - uni-directional approach light in both standalone and 1-, 2-, 3-, 4- and 5-light configurations on typical supports
  - strobe lights (uni-directional and omni-directional) both standalone and mounted on a tower up to 30'/10m high
  - wigwag
  - PAPI/VASI in 4-light PAPI (both right and left), 2- and 3-ganged VASI configurations and a standalone unit that may be used to create an array of any VASI configuration, and
  - aeronautical beacons - one that flashes red continually and another that strobes during the daytime.

  These models all have an "empty" attachpoint to which will be attached one or more BGL_LIGHTS. They, together with their textures, are saved in AFLT's *BaseModels* folder. Each model is more fully described in Appendix "A"

  There are no stock P3Dv2 base models provided (but you may add your own). P3Dv2 accepts FSX models.

- *Colors.txt* - a simple text file specifying the RGB value for each color available for use. Both the BGL_LIGHT color and the color of the model's light lens are controlled by this file. The format is simple and self-evident from an examination

- *Types.txt* - a tabular text file specifying for each light type the base model file to be used, the position(s) of the attached lights relative to the attachpoint location and other "default" data. Its format is described in Appendix "B". If you add any base models, you'll have to make corresponding entries in *Types.txt* before you will be able to use them.
- Sample array definition files for the most common types of approach lighting.

OPERATING OVERVIEW

Since the work products are likely to relate to an individual airport, as a first step you must specify a folder where AFLT is to save its work products, called the Library Folder. A good location is your development folder for that airport. But, it may be anywhere.

The starting point for every light to be created with AFLT (generally referred to as an "element", is a base model, i.e., a 3D model of the light fixture, or support or whatever. AFLT's stock base models include both FS9 and FSX .mdl files for all usual light types. These stock models should satisfy most of your requirements. Should you need to replace or supplement these models, please note that each base model MUST include an attachpoint named "Light" (i.e. "attachpt_Light"). The attachpoint will normally be located where you want the light to be displayed, but need not be. If more than one light is to be shown, the attachpoint would normally be placed at a central location. The positions of the lights relative to the location of the attachpoint are defined in *Types.txt*.

Light elements are specified using the main pane as described later in this manual. The data entered on the main panel is captured in an .ini file for each element. The ini files, which are in plain text, are saved in the Library Folder's INI sub-folder and are named as the element to which they apply, i.e., *element*.ini. These .ini files are the starting point for the Make Library and Make Array functions.

Note that neither FSX nor P3D support nesting of attachpoints. So, for lights that are normally attached to other scenery, e.g., obstruction lights, AFLT consolidates the FS9 3D model and the generated light model into a single FS9 file. (Unfortunately, there's no way to embed BGL_LIGHTs in a FSX/P3D model.)

AFTL supports a full range of pilot-controlled lighting, PCL, through the specification of on/off criteria for a system global variable (e.g., a COMS frequency or a transponder code) and, where appropriate, airport operating hours. With FS9 base models, if PCL is used during the daytime (for example, in low visibility conditions), in addition to the lights being on, the light lens "glow". (That's what the "day-glo" texture is used for.) Unfortunately, for technical reasons, this same capability is not available with FSX/P3D models. So, generally, you'll want to use FS9 models - even for FSX or P3D unless - there's a good reason not to.

In order to be rendered by Flightsim, the elements must be collected in an object library file. AFLT's Make Library function accomplishes this. The Make Library function creates two object libraries for each FlightSim version. First is the "lights" library ("lib_AF_Lights.bgl"), which is FS9 based and is identical for all FlightSim versions. A unique version of the "bases" library ("lib_AF_Bases_*FSn*.bgl") is created for each

Flightsim version. Each light model is associated with a specific base model via the attach point.

Both libraries ("lights" and "bases") must be located in the *scenery* subfolder of the scenery with which they are to be used and the associated textures in the *texture* subfolder. (Make Library will save the object libraries and textures for you. If you choose not to use this feature, you must manually copy these items to the appropriate add-on scenery folder before the lights will be visible.) Elements in the base object library may be placed with any object placement tool, including ADE. Only the base models are placed; the light models "follow".

To create more complex light arrays like, for example, a runway end configuration or an approach lighting system, AFLT allows you to specify the contents of the array as a series of X/Y/Z positions relative to a base point specified by latitude, longitude, elevation (Z-axis) and orientation (Y-axis). These coordinates are included in a "def" file, a simple text file. A ".def" file may hold the specifications for more than one array. AFLT's Make Array function compiles a .bgl file (the "language" understood by FlightSim and P3D) which defines the placement of each element in the array. The .bgl file includes only the placement of the array elements. The elements themselves must be included in your object libraries once you generate the array

One drawback of using BGL_LIGHTs for directional light assemblies is that once the orientation is specified, it cannot be adjusted - except by re-compilation of the model, i.e., oriented elements do not respond to placement heading. So, for example, if you have two PAPI-equipped runways at an airport, you'll have to create two PAPI elements, each one oriented for its particular application.

Finally, use of AFLT involves the selection or entry of file/folder paths. In every case, the path may be selected by clicking on the associated Select button and locating the file/folder with standard Windows resources, or by entering the path directly into the textbox. For the latter, and whenever entering data into any other textbox, use the <Enter> key or move the cursor to another location to terminate the entry.

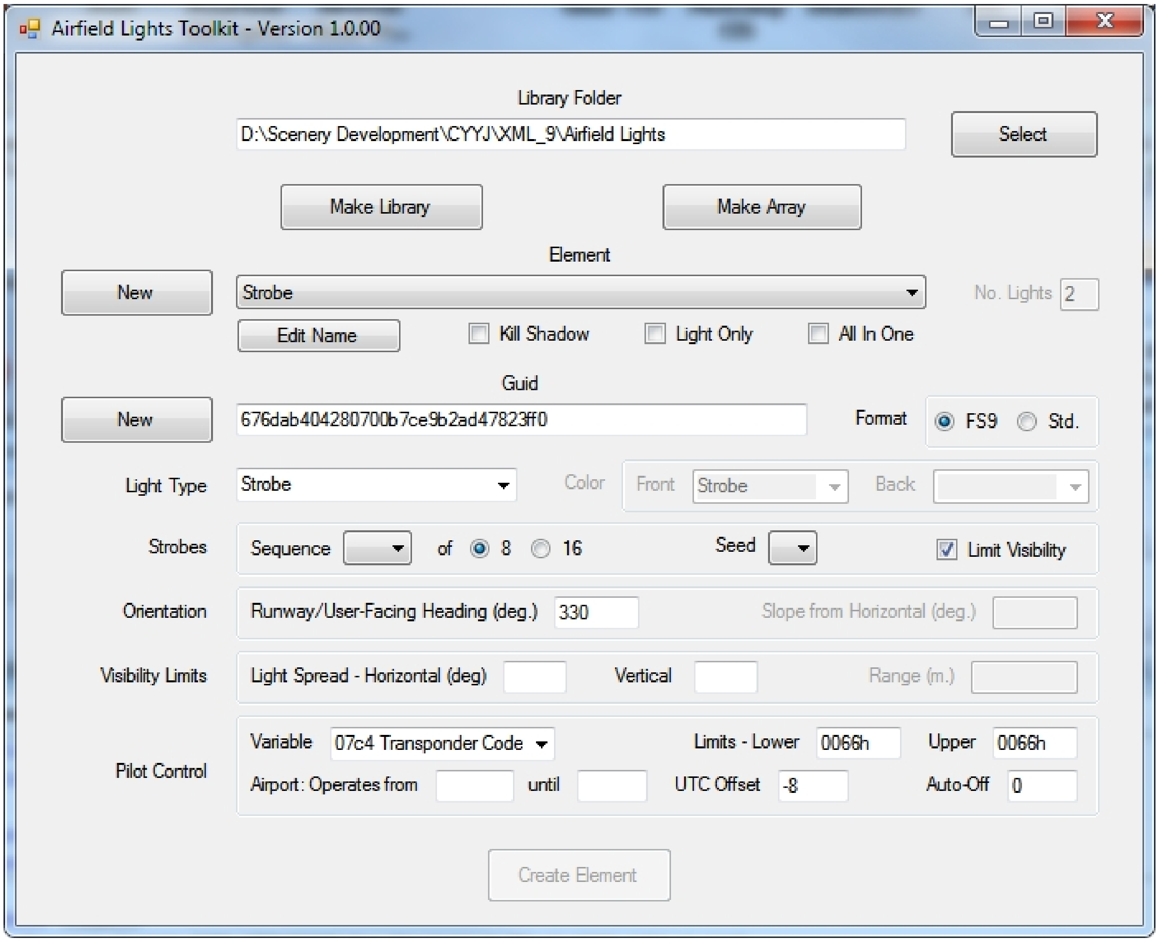
CREATING A LIGHT ELEMENT

AFLT's main panel is shown below.

The steps in creating a light element are as follows. They can be performed in any order after the first two steps.

- Specify the Library Folder where the elements are to be saved. Only the top folder need be specified. AFLT will look after the sub-folder organization. (When you restart AFLT, this field will be initialized to the last-used Library Folder
- Click the New button associated with the Element combo-box and enter a name for the new element, or select an existing element using the combo-box. If an existing element, the panel will be initialized with the last entered value for each parameter. If you need to change the name of an element after initial entry, do so using the Edit Name button. (When the name changes, numerous references need to be updated. Do not attempt to change element names manually.)
- Select the model type from the Model Type combo-box.

- If the Color combo-boxes are enabled, select from the available colors. (The contents of these combo-boxes is determined by the *Colors.txt* file. If neither combo-box is enabled, a standard color will be used (e.g., amber for wig-wags, red and white for VASI/PAPI, etc.)
- If the model type is Strobe, specify:
    - the flash sequence as 1-8 of 8 or 1-16 of 16,
    - whether the visibility is to be limited (default is omni-directional) and
    - an arbitrary "seed" value (1-8) which controls the firing of the first flash in the sequence (so all running rabbits at an airports do not have identical sequence start times).



Library Folder: D:\Scenery Development\CYYJ\XML_9\Airfield Lights

Element: Strobe

No. Lights: 2

☐ Kill Shadow ☐ Light Only ☐ All In One

Guid: 676dab404280700b7ce9b2ad47823ff0

Format: ● FS9 ○ Std.

Light Type: Strobe

Color: Front: Strobe Back:

Strobes: Sequence of ● 8 ○ 16 Seed ☑ Limit Visibility

Orientation: Runway/User-Facing Heading (deg.) 330 Slope from Horizontal (deg.)

Visibility Limits: Light Spread - Horizontal (deg) Vertical Range (m.)

Pilot Control: Variable 07c4 Transponder Code Limits - Lower 0066h Upper 0066h

Airport: Operates from until UTC Offset -8 Auto-Off 0

Main Panel

- If the model type is Beacon, the Strobe fields will be replaced with the following:
    - the period of the blinking light in seconds, i.e., the time between the start of two successive "blinks",
    - the on-time of the light, in seconds, and
    - optionally, the number of strobe flashes/minute during the day.
- Specify the number of BGL_LIGHTs to be used at each location. Normally this will be 1, or 2 for strobes. But use of more may be necessary in special

- Enter the Orientation and Visibility Limit parameters when those text boxes are enabled. (For new elements, the Visibility Limits, where applicable, will be initialized to default values from Types.txt.) If these text boxes are not enabled, the selected light type has no use for such data.
- Check Kill Shadow if you do not need a shadow. Otherwise:
    - if a custom shadow .mdl file exists in the folder along with the selected base model (its name will be suffixed with "_Shad"), it will be used (custom shadows may be used to improve processing efficiency), or
    - if no custom shadow file exists, the shadow incorporated in the base model file (if any) will appear.

  (Shadows consume processing resources. If a shadow isn't necessary - like, perhaps, for an obstruction light which is generally well away from the user aircraft - "kill" it. Shadows should always be suppressed for lights on towers/supports where part of the tower/support is buried. Otherwise, FlightSim will display a shadow for the below-ground portion as well.)
- Finally, click the Create Element button. AFLT will place a file in the libraries INI folder specifying the elements characteristics. AFLT will also create a test model to ensure all required resources are available. If all is well, a confirmation message will be displayed for a short time.

As mentioned earlier, FSX and P3D do not support "nesting" of attachpoints. Hence, lights that are normally attached to other objects, such as obstruction lights, cannot use AFLT's "attached light" scheme. If the All in One box is checked, the light is embedded in the base model, rather than being attached to it. Unfortunately, this capability is only available with FS9 base models.

Check Light Only for lights that are to be displayed on or directly over a runway or taxiway, such as runway centerline lights or approach lights that extend onto the runway or, perhaps the last in the sequence of ODALS for a displaced threshold. If checked, no base model will be displayed.

PILOT-CONTROLLED LIGHTING

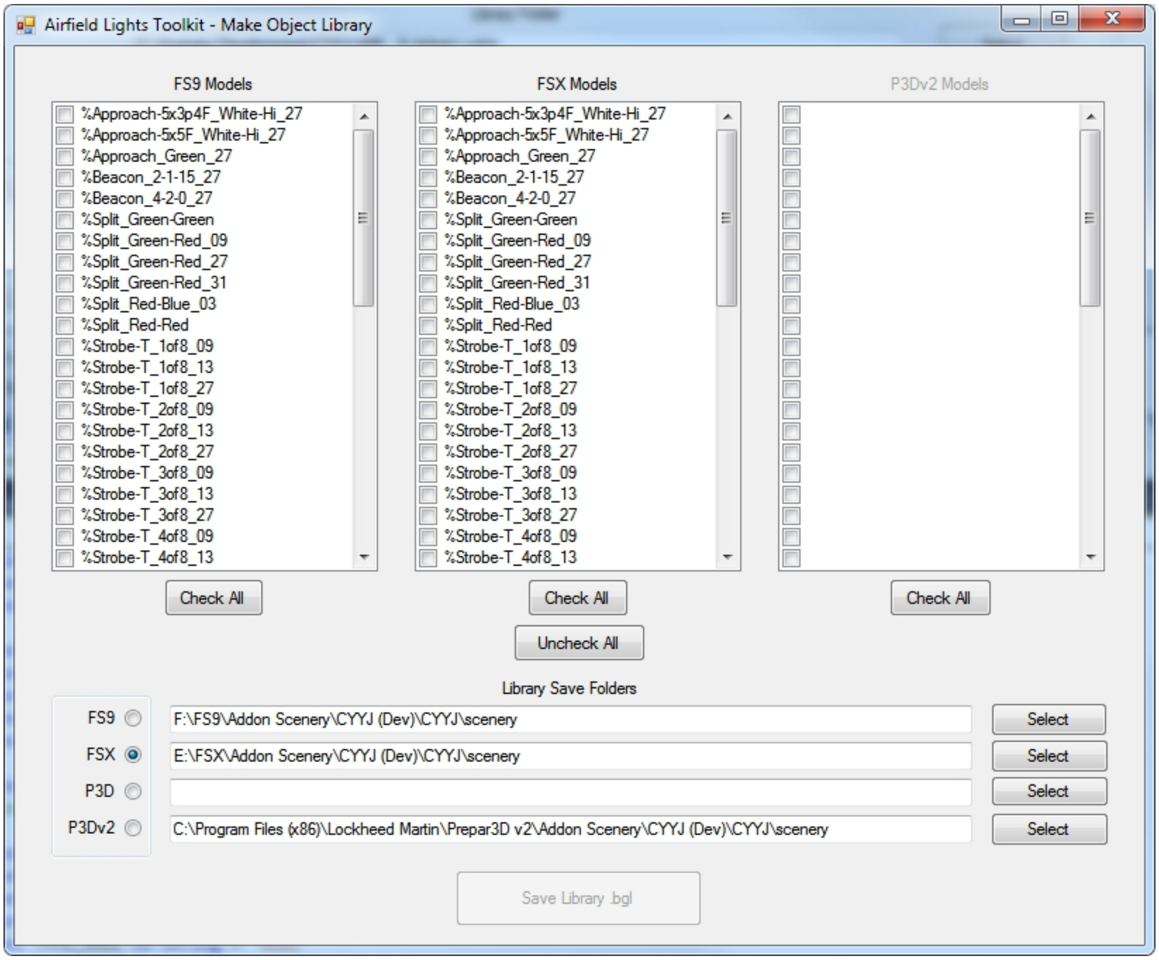
While specified on the main dialog along with the light elements, pilot-controlled lighting (PCL) is an airport attribute and applied to all elements in the library folder. If required at this airport, select a pilot-controlled lighting variable from the combo box. If the variable you wish to use is not listed in the combo box, enter its global variable hexadecimal address into the combo-box's text box. (A complete list of accessible-to-scenery variables is available from fsDeveloper.com here.) Then, specify the "ON" range of the variable, i.e., the upper and lower limits between which the lights are to be on. (Normally, a single value serves both purposes. Also, where applicable, enter the airport operating hours (i.e., the hours outside of which the lights will normally be off) in local time and the airport's UTC offset in hours (decimal hours if necessary). Finally, if you want to be able to deselect the lights-on criteria but have the lights remain on, enter the number of minutes after which the lights will automatically be turned off. **Please be aware, pilot-controlled lighting based on airport operating hours is "expensive" in processing overhead and should be used only if required.**

So that you will know which light elements respond to pilot-controlled lighting, the PCL parameters are only enabled for light elements which they affect. For example, PAPIs or wigwags are not affected by PCL and, therefore the PCL parameters text boxes are not enabled when specifying PAPIs or wigwags. So, to specify PCL criteria, ensure a runway/taxiway/approach light type is selected.

There is no save control for PCL parameters. They are saved automatically. PCL using the then current PCL parameters is applied when the object libraries are created. So, to implement a revised pilot-controlled lighting scheme, simply specify the new PCL scheme and regenerate the applicable object libraries using the Make Library function.

CREATING A LIBRARY OF ELEMENTS

Click the Make Library button on the main panel and you'll be presented with the dialog below. Array elements are prefaced with "%".



FS9 Models:
- ☐ %Approach-5x3p4F_White-Hi_27
- ☐ %Approach-5x5F_White-Hi_27
- ☐ %Approach_Green_27
- ☐ %Beacon_2-1-15_27
- ☐ %Beacon_4-2-0_27
- ☐ %Split_Green-Green
- ☐ %Split_Green-Red_09
- ☐ %Split_Green-Red_27
- ☐ %Split_Green-Red_31
- ☐ %Split_Red-Blue_03
- ☐ %Split_Red-Red
- ☐ %Strobe-T_1of8_09
- ☐ %Strobe-T_1of8_13
- ☐ %Strobe-T_1of8_27
- ☐ %Strobe-T_2of8_09
- ☐ %Strobe-T_2of8_13
- ☐ %Strobe-T_2of8_27
- ☐ %Strobe-T_3of8_09
- ☐ %Strobe-T_3of8_13
- ☐ %Strobe-T_3of8_27
- ☐ %Strobe-T_4of8_09
- ☐ %Strobe-T_4of8_13

FSX Models:
- ☐ %Approach-5x3p4F_White-Hi_27
- ☐ %Approach-5x5F_White-Hi_27
- ☐ %Approach_Green_27
- ☐ %Beacon_2-1-15_27
- ☐ %Beacon_4-2-0_27
- ☐ %Split_Green-Green
- ☐ %Split_Green-Red_09
- ☐ %Split_Green-Red_27
- ☐ %Split_Green-Red_31
- ☐ %Split_Red-Blue_03
- ☐ %Split_Red-Red
- ☐ %Strobe-T_1of8_09
- ☐ %Strobe-T_1of8_13
- ☐ %Strobe-T_1of8_27
- ☐ %Strobe-T_2of8_09
- ☐ %Strobe-T_2of8_13
- ☐ %Strobe-T_2of8_27
- ☐ %Strobe-T_3of8_09
- ☐ %Strobe-T_3of8_13
- ☐ %Strobe-T_3of8_27
- ☐ %Strobe-T_4of8_09
- ☐ %Strobe-T_4of8_13

FS9 ○ F:\FS9\Addon Scenery\CYYJ (Dev)\CYYJ\scenery
FSX ● E:\FSX\Addon Scenery\CYYJ (Dev)\CYYJ\scenery
P3D ○
P3Dv2 ○ C:\Program Files (x86)\Lockheed Martin\Prepar3D v2\Addon Scenery\CYYJ (Dev)\CYYJ\scenery

Make Library Dialog

Click the Click All button for the version that contains most of the base models you plan to use. If you select FS9 models but, for example, intend to use a certain FSX model, just check that model in the FSX list. The corresponding FS9 model will be unchecked automatically. To make this task easier, the lists are sequenced identically with blank spaces for base models that don't exist for that version. As well, all list-boxes scroll together.

Enter the path(s) to the scenery folder(s) into which the libraries are to be placed - if you want AFLT to save them there. The object libraries are always saved to the Library folder.) Then, select the FlightSim/P3D version with which the library is to be used. If you have selected to generate a FS9 library, the FSX and P3Dv2 list-boxes are disabled. If for a FSX library, only the P3Dv2 list-box is disabled. For P3Dv2, all lists are enabled.

FS9 object libraries always incorporate FS9 base models. To the extent possible, you'll want to use FS9 models with FSX and P3D as well ("dayglo" textures can only be applied to FS9 models.). However, for transparent objects such as towers, the FSX version may have to be used to be properly rendered in FSX/P3D. You be the judge!

For models to be rendered properly in FlightSim, the object libraries must be saved in the same *\scenery* folder as the airport .bgl with which they are to be used and the required textures in the companion *\texture* folder. If a Library Save path is specified, AFLT will copy the object libraries and textures to the proper folders. AFLT requires textures for the base models to be in the *Base Models\Textures* folder. If the required textures are not found there, an error message will be issued you'll have to transfer the textures manually.

When all desired items have been selected, (don't forget to include array elements) click the Make Library button. That's all there is to it! AFLT will create object libraries containing the checked elements and copy the object libraries and required textures to the specified add-on scenery folder ready for placement with any standard object placement utility. Two libraries are created for each selected FlightSim version - a "lights" library and a "bases" library. The "lights" library will always be named *lib_AF_Lights.bgl*. The name of the bases library will include the FlightSim version for which it was compiled as well as a suffix "(for *xxx*)" if the version to which it is applied is different. For example, when FS9 bases are used with FSX, the filename would be *lib_AF_Bases_FS9 (for FSX).bgl* which will alert you to a FS9 file intentionally used with FSX.

If no errors, a confirmation message is issued. Any required textures not found in the designated texture folder (if specified) will be listed in place of the confirmation message. The object library ".bgl" files (one for lights and one for each version of the bases) are also saved in the Library Folder. So it's easy to transfer them later if, for instance, the intended add-on scenery folder doesn't exist when you create the library.

While the object library is being generated you will notice a small dialog rapidly flashing on and off. This is the compiler. (Each light element is compiler twice - once for the light and once for the base.)

CREATING LIGHT ARRAYS

Creation of light arrays requires an array definition (".def") file.

A ".def" file is a text file setting out a reference point for one or more arrays and, within each array, the location of each light element relative to that *reference point*. The format of array definition files is described in Appendix "C". These files are saved in the *\Arrays* subfolder of your AFLT folder. You may add your own.

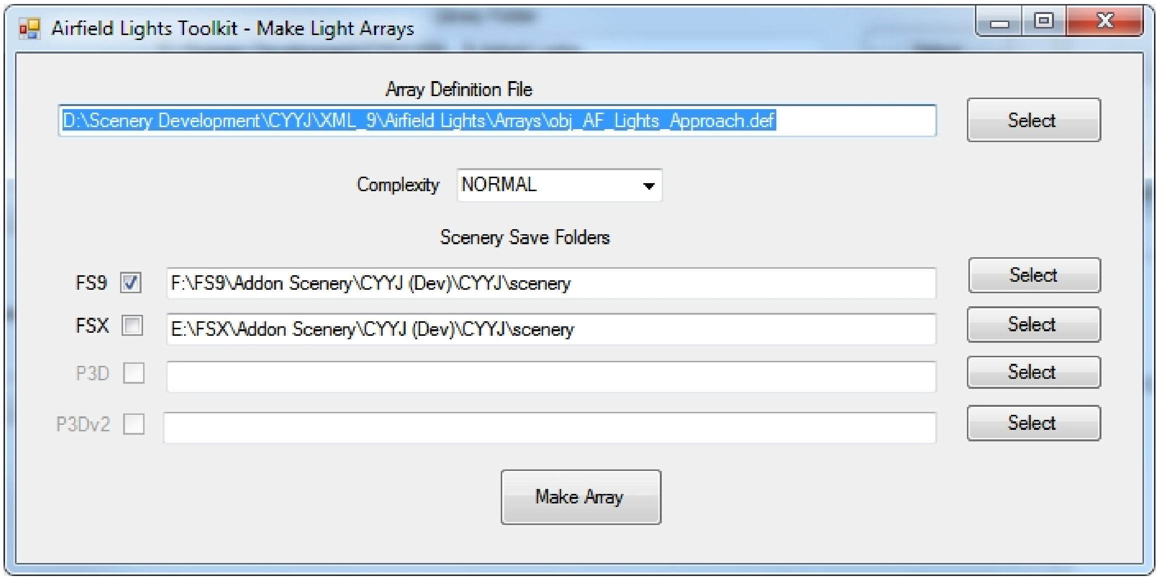
Whether you create your own array definition or modify one of the samples, you must determine the geographic location to be used as the *reference point*. Usually, the reference point will be the center of the threshold of the runway to which the array elements pertain. You can determine this location using an airport editor, such as ADE. If you are enhancing a stock airport, position the user aircraft over the center of the threshold and record the latitude, longitude and elevation using a utility such as TCalc or SlewMaster by the author.

To add one or more light arrays to your airport, click on the Make Array button on the main panel. The dialog shown below will open.

To generate an array, select:

- the name of the ".def" file to be used,
- scenery complexity,
- the scenery folder(s) where the array ".bgl" is to be saved,
- the FlightSim/P3D version(s) with which the array is to be used, and

click Make Array.



Array Definition File
D:\Scenery Development\CYYJ\XML_9\Airfield Lights\Arrays\obj_AF_Lights_Approach.def

Complexity NORMAL

Scenery Save Folders

FS9 ☑ F:\FS9\Addon Scenery\CYYJ (Dev)\CYYJ\scenery
FSX ☐ E:\FSX\Addon Scenery\CYYJ (Dev)\CYYJ\scenery
P3D ☐
P3Dv2 ☐

Make Array Dialog

As during library creation, while the array is being generated you will notice a small dialog rapidly flashing on and off. This is the compiler. (Each element in the array is test-compiled to ensure all required resources are available.)

The supplied ".def" files assume nominal, uniform spacing of the various light elements. Certainly for test purposes, and probably also for production, these spacing will be adequate. However, in the real world, there are numerous reasons why nominal spacing isn't used. It's up to you to decide whether spacing needs to be modified and, if so, to edit the .def file. Also, depending on the profile of the terrain under the approach, you may have to edit the elevations.

Once you have generated the array and saved the resulting .bgl file to the relevant scenery folder, create a library including the array elements and "fire-up" FlightSim. If things aren't exactly as you intended, don't panic. Review your array definition (all the answers will be there), correct it and regenerate the array.

SPECIAL CIRCUMSTANCES

VASI - Siting of VASI elements will require a little extra work on your part. VASI configurations are airport dependent and comprise two independent (for the purposes at hand) sets of elements, a downwind bar(s) (closest to the runway threshold) and an upwind bar(s). You'll have to make and place them individually.

The downwind bar is normally aimed 0.5 degrees below the intended glideslope. The upwind bar is aimed along the glideslope. The distance between the upwind and downwind bars, which may vary between 500 and 1000' (150-300m), is airport dependent. The nominal separation is 700' (215m). In these circumstances, you'll need to use two VASI2 or VASI3 models.

Alternately, you could create an array in which all 4, 6 or whatever number of elements are used, the elements being placed automatically.

PREPARING YOUR AIRPORT FOR AIRFIELD LIGHTS TOOLBOX LIGHTS

Unfortunately, it's not easy to replace individual lighting elements for stock airports. But, its not a daunting task either, and it is straightforward. Using ADE or some other airport definition tool:

- open each runway and remove all lighting and visual aids (VASI/PAPI) that you are replacing
- open each taxiway and deselect edge lighting and, finally
- remove any stock apron edge lighting.

Alternately, if you have access to your airport in XML format, simply drag that file onto AFLT's ClearLights.exe. It will allow you to selectively remove various categories of lights from your airport and re-compile your airport. While simpler than opening each lighted element with an airport editor, remember that if you later modify the airport, you'll have to once again remove all the replaced lights. However, if you plan to distribute two versions of your airport, one with stock lights for lower-end systems, the other with custom 3D lights, this little utility will allow you to do so while maintaining only one (i.e., the stock-lighted) version of the airport.

SUBSTITUTION/ADDITION OF BASE MODELS

You may replace or add to the provided base models and add new light types to AFLT.

Base models for AFLT **MUST** use the texture *obj_AF_Lights.bmp* (found in the *Base Models\Textures* folder) for the lens' of any colored lights and each must contain an attachpoint names "Light" (i.e., *attachpoint_Light*). Beyond those two requirements you are free to design your models as you wish.

Place each version (FS9/FSX/P3Dv2) of your model in the applicable Base Models folder and, if you use non-stock textures, place a copy of your new texture (32 bit, DXT and/or DDS formats as necessary in the *Base Models\Textures* folders.

If you are simply replacing a stock base model (using the same name) and your attachpoint is in the same location as in the stock model, you need do nothing further. However, if you have moved the attachpoint, have renamed the file or are adding a new base model, you will have to make a new, or update an existing, entry in Types.txt. (Please see Appendix "B").

To add a new light type, create a corresponding new entry in Types.txt. Of course, if the new type requires a new base model, you'll have to provide that as well.

SUPPORT

Airfield Lights Toolbox's support forum is located in the Scenery Design - Airport Design support area at http://www.fsdeveloper.com. Please direct your problem reports, suggestions for improvement and other comments there. When you report problems, please include relevant details. In particular, the AFLT version number, the exact error message and a summary of what you were doing at the time are likely to be particularly helpful. If you are asking for support with a particular lighting issue, please attach the complete local library folder. Without it, it's unlikely I can be of much help.

I also have a support website at http://stuff4fs.com for all my airports and development utilities. (Navigate to the User Applications / AFLT page.) Among other things, the site lists all known problems with the latest release. The most recent release of Airfield Lights Toolbox is available from that site as are occasional development releases of new features.

While I can’t promise to resolve every issue you report or include every feature addition you propose, I will undertake to support and enhance Airfield Lights Toolbox in a manner consistent with it becoming and remaining the tool of choice airfield lighting for Microsoft Flight Simulator.

END USER LICENSE AGREEMENT

As used in this end user license agreement, the terms "Airfield Lights Toolbox" and "AFLT" shall be construed as encompassing the full contents of the downloadable archive (.zip) file created and/or posted for distribution on “download sites” by the author, including without limitation: the executable file AFLT.exe, the associated user manual, the files Colors.txt, Types.txt and the sample array definition files, and any derivates of any of the foregoing.

You are granted a free, non-exclusive right solely to install and use Airfield Lights Toolbox on your computer system(s) for your own personal enjoyment and, subject to what follows and the rights of others, to use and distribute work products created or modified with Airfield Lights Toolbox (“derivative files”).

You may not:

- upload Airfield Lights Toolbox in whole or in part to any file distribution system,
- reverse engineer, disassemble or decompile any part of Airfield Lights Toolbox,
- incorporate Airfield Lights Toolbox in whole or in part into any commercial product or facility, “shareware” or “freeware”, or
- use Airfield Lights Toolbox or any derivative files in the development, marketing or support of, or incorporate derivative files in, any commercial product

without the express written permission of the author. Use as may be permitted for commercial purposes may be subject to a license fee.

Your use of Airfield Lights Toolbox is entirely at your own risk. You assume and are responsible for any and all liabilities and damages arising therefrom no matter how caused.

By installing or otherwise using any part of Airfield Lights Toolbox, you are deemed to have agreed to the foregoing.

Appendix "A" - Base Models Supplied

The following base models - both FS9 and FSX versions - are supplied and their use is defined in *Types.txt:*

Approach.mdl - uni-directional light for approach lighting systems threshold-lines

Approach-n x spacing.*mdl* - approach lights mounted on 20' supports (which may be buried to the extent necessary) "n" is the number of lights mounted on the support and spacing is the nominal spacing in feet except for 4x2p5 and 5x3p4 where the spacing is 2.5' and 3.4' (40") respectively. The range of lights should be adequate to configure most approach lighting systems specified by the FAA.

Obstruction.mdl - dual obstruction light assembly on a 5' post

Omni.mdl - omni-directional light for taxiway and runway edge lighting application

PAPI.mdl - 4 light PAPI system. Same model used for both PAPI_Right and PAPI_Left

Split.mdl - bi-directional light for runway end applications

Strobe.mdl - strobe light mounted on a 5' pole

Strobe-T.mdl - strobe light mounted on a 30' tower

Tower_Beacon.mdl - a 50m' high tower. You may chooses to have a blinking light atop it or a blinking light at night and a strobe during the day

VASI.mdl - single VASI

VASI-2.mdl - 2-unit VASI with 16' separation

VASI-3.mdl - 3-unit VASI with 16' separation

WigWag.mdl - typical wigwag with two alternately-blinking amber lights

Elements on towers or supports may be set to any height. The unneeded portion of the tower/support will be "buried").

None of these models will cast shadows normally. Custom shadows (provided - model file name suffixed with "_Shad") may be applied to all models except *Approach*-n x spacing.*mdl*, *Tower_Beacon.mdl*, *Obstruction.mdl, Strobe.mdl and*, *Strobe-T.mdl. Obstruction.mdl* is generally used in locations where a shadow would add little realism In the other cases, a portion of the model usually will be buried but would cast a show anyway.

You may add to this set of base models or replace individual models. It is recommended you not change the names of stock elements. If you do, or if you add new elements, you will have to add or edit the corresponding line in Types.txt. Since models are regenerated during library creation, to "retrofit" a model, simple edit Types.txt as necessary and rebuild the libraries that use it.

<u>Appendix "B" - Types.txt Format</u>

Each line in Types.txt describes a model type, as follows:

Column 1 - Light Type. These names are loaded into the Light Type combobox at startup.

Column 2 - Name of base model used for this light

Column 3 - Model type. These names are predefined in the system:

- APPROACH - uni-directional light. The supported versions may be placed up to 10m. (30') above the ground.
- BEACON - a simple blinking red light (approx. 30 flashes/minute) up to 50m (160') above ground level. Optionally, you may specify a strobe during the day
- OMNI - omni-directional light in a simple housing
- PAPI - a sequenced four unit PAPI in a typical housing
- SPLIT - bi-directional light in a simple housing
- STROBE - sequentially-flashing bluish lights up to 10m. (30') above the ground in a simple housing
- VASI - white above/red below specified glide-path angle in a typical housing
- WIGWAG - alternate blinking lights in a typical housing

Column 4 - Default horizontal spread of light(s) in degrees. A blank entry implies full (360 degree) visibility.

Column 5 - Default vertical spread of lights, centered about light tilt if any. A blank entry imply full visibility.

Column 6 - Longitudinal range (in meters) of light <u>used to limit</u> visibility. A blank entry implies full visibility

Column 7 - "True" or "Yes" if light is subject to pilot control; otherwise "False" or "No"

Column 8 - M (metres) / F(feet) indication pertaining to following column

Column 9 - X/Y/Z offset(s) from attachpoint (in feet or meters). Values for each axis are separated by ",". If more than one light to be displayed, subsequent positions are separated by ";".

Appendix "C" - Array Definition Format

It is suggested you refer to one of the sample array definition files when reading this appendix.

Each array definition begins with "<" and must be terminated with "/>", much like a SceneryObject in a scenery ".xml" file. The start and termination characters may be on a line by themselves or included (in the appropriate location) in one of the other lines. Comments (lines beginning with ";" or "//") may be interspersed between array specifications.

The first non-comment line (i.e., the "header line") of the array definition specifies the reference point for the array. It must include:

- latitude,
- longitude,
- elevation,
- heading, and
- an optional "tag"

all separated by "|".

Latitude and longitude can be in any of the standard formats (*DD.dddddd*, *DD MM.mmmm* or *DD MM SS.ss*). Elevation may be in feet or meters but, if in feet, must be suffixed with "F". Heading (true) must be a number between 0 and 359.99. For approach lighting systems, use the runway heading. Be as precise as possible in all cases.

The "tag", if entered, will usually be the applicable runway number - but it may be anything so long as it is unique. (The system-generated filename for an oriented array element includes a "tag", i.e., a unique identifier, to differentiate it from other similar, but differently-oriented, elements at the same airport.) If AFLT requires a "tag" for any element in an array and none is provided, the array reference heading will be used.

The definition of each element in the array occupies a single line. It comprises:

- Light Type,
- the X/Y/Z position in meters or feet relative to the reference point, and
- supplementary data (where necessary), such as color, strobe sequence, etc.

Dimensions in feet must be suffixed with "F". Dimensions in meters may be suffixed with "M" for clarity. Y-values are the distances from the reference point when travelling along the specified heading. Thus, for an approach lighting system, if the associated runway heading is used, the Y value for all points prior to the threshold will be (-)ve whereas those on the runway or beyond the threshold will be (+)ve. X-values are (-)ve when left of the (extended) runway centerline and (+)ve if to the right when facing in the direction of the specified heading. Z-values are relative to the elevation at the reference point.

A ".def" file format would, then, be similar to the following:

; Title of Array 1
< *latitude* | *longitude* | *elevation* | *heading* | *tag*

element name | X-value | Y-value | z-Value | supplementary data
element name | X-value | Y-value | z-Value | supplementary data
element name | X-value | Y-value | z-Value | supplementary data | alternate tag />

Supplementary data need only include color data - and then only for elements for which color is specified as the first item below. (Colors must be one of the names specified in the Colors.txt file.) The remainder is optional and, if not provided, default data (as specified in Types.txt) will be used. Supplementary data comprising two or more items, they must be delineated by a forward slash ("/"). Individual items may be but, unless all following items are also omitted, an empty field (i.e., "//") must be included for each item.

The possible contents of the supplementary data field depend on the type of element:

Light Type	Modifier
Approach	Color / No. of BGL_LIGHTs (bulbs) / horizontal spread / vertical spread / tilt
Beacon	Beacon period (sec.) / on for (sec.) / strobes per minute during day / no. of bulbs
Omni	Color / no. of bulbs
PAPI/VASI	No. of bulbs
Split	Front color / Back color / no. of bulbs
Strobe	Sequence number / no. of flashes (8 or 16) / seed value / no. of bulbs / horizontal spread / vertical spread
all other types	Color / no. of bulbs

For strobes, the *seed value* is an arbitrary number (1-8) which determines the timing of the first flash of each set of sequenced strobes ("running rabbit") so that there will be slight variations in the timing from one running rabbit to the another

AFLT creates a name for each element in the array by concatenating the *light-type* and *color* or other distinguishing data and suffixing that with the *tag* or *heading* from the header line. This should be adequate in most cases. But, if you wish, you may specify an alternate tag for any item. It may be any character string.

AFLT includes sample "def" files for approach lighting systems commonly encountered, including ALSF-2, SSALR, MALSR , MALSF and ODALS in its *Sample Defs* folder. Use these as your starting point. Unless you need to alter the Z-value because of the surrounding terrain, all you need do is specify the relevant values in header line.

The FAA recommends that the light plane of an approach lighting system be horizontal whenever possible. The sample ".def" files reflect this. However, we don't live in a perfect world. If you want your approach system model to closely reflect that portion of the real world it simulates, it may be necessary for you to establish the elevation of each light bar and enter it into the ".def" file. But, unless the slopes are drastic, the "default" plane should suffice.