AILGenerator

(AILG_MSFS)

Manual

Version 1.3 beta

January 14, 2021

Nico W. Kaan

1 Introduction

The main task of AILG_MSFS is to find the titles, airline codes and aircraft types of AI Aircraft (liveries) in your installation of Microsoft Flight Simulator 2020 (MSFS). Information about these liveries is put in a file to be used by PSXT_MSFS in the matching process with live aircraft.

1.1 Technical information

AILG_MSFS is a 64bit Windows application, written in C++, for **Windows 7** or later.

AILG_MSFS reads text files coded in ANSI, UTF8 with BOM, UTF8 without BOM and UTF16. It writes ANSI or UTF8 without BOM files.

It reads and -upon your request- re-writes your aircraft.cfg files.

1.2 System Requirements

In order to run it, you need Microsoft Visual C++ Redistributable for Visual Studio 2019. The standalone installer for x64 (VC_redist.x64.exe) can be downloaded here at Microsoft.

1.3 Forum

For questions & answers and discussions use my Support Forum at AVSIM.

2 Installation

This Chapter describes how to install, update, remove and start AILG_MSFS.

2.1 Download and install

Download the setup zip file from my site. Unzip and run the installer. You should install AILG_MSFS at the PC running your Flight Simulator. You better not install in the C:\Program Files (x86) folder.

After installation, you will have a AILG_MSFS folder and a sub folder docs.

AILG_MSFS.exe is the application.

The **docs** folder contains documentation by means of the files AILG_MSFS Manual and Release Notes. The EULA_MSFS is the End-User License Agreement. This is a legally enforceable contract between you and the Author. By accepting the installation you agree to be bound by the terms of this EULA.

2.2 Update to new version and removal

Download the setup file, unzip and run the installer. It will first uninstall your previous version and then install the new one.

Removing AILG_MSFS from your PC can be done running the uninstaller from the Programs list or the apps list, just like any other Windows program.

2.3 Version Check

At each start-up, but not more than once a day, AILG_MSFS will check if there is a new version available. If so the –normally- green version message in the main window will turn yellow, and the new version number will be indicated.

This is only an indication. It is up to you when you download and install the new version. It is recommend to always update. Thus it will be easier for me to answer any questions.

It may also happen that you get the message **"error 99: This version has expired"**. In that situation you cannot go further. You have to upgrade to a new version.

2.4 Input files

Subfolder **data** contains four files with information needed by AILG_MSFS.

- **aircraft.xml**
  Contains ICAO aircraft type codes.

- **types.xml**
  Contains ICAO aircraft type codes by heavy/middle/helicopter and similar types.

- **airlines.xml**
  The most important section of this file is about the ICAO airline codes. Next to that you will also find smaller sections with Virtual airline codes, Private airline codes (3.7.1) and Government airline codes (3.7.3).

- **fixes.xml**
  Contains aircraft **type** fixes that will be **automatically** made (4.2).

You can and may add missing codes to these files yourself, but do make backup's be-cause these files will be overwritten with each new version.

I would appreciate if you post a message in my Forum about additions you have made, so I can add these in a new release (but at a slower paste then currently).

3 Searching for AI aircraft

3.1 Input Folders

Specify full paths to the folder(-s) at your PC where your AI aircraft are and then AILG_MSFS will check the aircraft.cfg files and if there are no errors it will copy the livery information.

At first start after installation AILG_MSFS will try to detect your packages folder.

See also the "auto detect MSFS" push button.

3.2 Checkboxes

The AILG_MSFS GUI comes with a number of checkboxes. The default values of all check boxes is not checked.

3.2.1 Include checkboxes

By default AILG_MSFS only accepts valid ICAO aircraft type codes. However, you may extend this with one or more of these include functions.

- country specific PVT/GA
  If checked, private aircraft liveries with a country specific private airline code (instead of PVT) will be included too. These codes can (only) be used in airport files such that you can define the right version of a private aircraft suited to the airport. Read more in section 3.7.1.

- country specific GOV/MIL
  If checked, Government/Military aircraft liveries with a country specific airline code (instead of GOV) will be included too. These codes can (only) be used in airport files such that you can define the right version of a military aircraft suited to the airport. Read more in section 3.7.3

- Virtual Airlines
  If checked, virtual airlines available in AILG_MSFS will be taken into account too. In the log file you can see what VA's are present. If you want yours to be added post a message in my Forum. The current list of VA's is shown in the log file.

3.2.2 Repair aircraft.cfg checkbox

With this checkbox you give AILG_MSFS permission to make changes to your aircraft.cfg files. The following changes will be made if necessary:

- [fltsim.x] numbering
  If the aircraft.cfg file is not coded in UTF16, AILG_MSFS will repair an in-consecutive [fltsim.x] section numbering in your aircraft.cfg file(-s).

- Double registration codes in the line atc_id= will be removed.
  (e.g. in atc_id=PH-AKS,PH-AKR,PH-AKS one PH-AKS will be removed)

- Formatting
  AILG_MSFS will format all your valid (*without errors*) aircraft.cfg files.
  Formatting implies:
  The file starts with the [fltsim.x] sections followed by other sections.
  The [fltsim.x] sections will be in alphabetical order of the atc_parking_codes key (= the airline code); if they are equal the ordering will be done on the registration code.
  Between [fltsim.x] sections one empty line will be inserted, extra empty lines will be removed.
  White space surrounding the title of a livery will be removed.

- registration code added to title
  AILG_MSFS will add the registration code of the livery, defined with **atc_id=regcode**, to the title of the livery (if not already present).

  This will only take place for exactly defined liveries (with one registration code), so not with more than one registration codes or with a registration code with a wild card.
  Note that this does not work for IVAO liveries.

  *Note: if the registration code is in the title, PSXT_MSFS will be able to detect an **exact** livery match.*

If there are changes due to the repair action a new aircraft.cfg file will be written after the following backups were made:
A backup file named **aircraft0.cfg** is written in the same folder as aircraft.cfg. An already existing aircraft0.cfg file is renamed to **aircraft00.cfg**.

3.3 Push buttons

The AILG_MSFS GUI has a number of push buttons.

- Detect MSFS button

  If pushed AILG will try to detect your Packages folders Official and Community. If found it will fill in two folders, the first pointing to Community and the second pointing to Official.

- Change button
  With this button you enable the folders area for editing.
  Add a folder with AI aircraft by typing the full-path name into a text box, e.g.

  D:\AI\FSPXAI

  The higher the folder in this area, the higher its priority. If the same livery appears in two folders, the livery of the lower priority folder will be skipped, see 5.1.

  When you are finished editing, push the change button again in order to make the change effective. AILG_MSFS will check immediately if the folder exists.

- Swap button
  You can re-order the folders contents with the Swap buttons (S). The higher the position of the folder, the higher its priority. If an airline-aircraft type is available in two or more folders, only the highest priority one will be put in the AI_liveries.xml file.

- Delete button
  You can delete a folder entry with the X button.

- Save and Restore buttons
  When you push the Save button, paths (not yet saved) will be saved. When you push the Restore button, all your paths will be put back in the user interface (up to 10 of course) for editing and or selecting. This is just for convenience.
  Note that these buttons will only be enabled if there is anything new to save or restore.

- Go! Button
  If all path names exist, the Go! button in the right bottom corner will be enabled. The paths to your AI folders will be saved in AILG_MSFS.xml.
  Push this button, if you’re ready with the folders. AILG_MSFS then searches for aircraft.cfg files in the folders specified.

3.4 Output folder

The list with liveries (AI_liveries.xml) is generated in the Output folder. The default value is subfolder **liveries** of AILG_MSFS.

You may change the Output folder to any folder that you have created first. Push the **change** button and type in the Output folder text box either a the name of a subfolder of AILG_MSFS or a full path to another folder. Then push the change button again to make it active. You may also enter a path to a share at a networked PC, for instance at the PC running PSXT.

3.5 How are liveries found?

AILG_MSFS finds a livery based on the following information in the [General] section of the **aircraft,cfg** file:

parameter	**is used as**
icao_type_designator	ICAO aircraft type code

If the aircraft.cfg file does not have a [General] section but a [VARIATION} section instead, AIL will look up the aircraft.cfg info from the base_container.

and it searches in (each) **[fltsim.x]** section for:

parameter	**is used as**
atc_id	Registration code(-s)
title	Title of livery
icao_airline	ICAO Airline code(-s)
atc_parking_types	Cargo aircraft info (bool), if string "**CARGO**" is found

Note that all parameters are in MSFS between double quotes, AILG skips these quotes.

3.6 Registration codes

Normally a livery has of course only one unique Registration code. However, you may fill in multiple Registration codes separated by comma's. Thus it is easier to separate different groups of liveries for a single type without having to install all liveries. For example to differ between Normal BAW 744 liveries and BAW One World 744 liveries. Or to distinguish all freighter liveries of a certain type from the Pax liveries of that same type, or 773–ER versions from 773 versions. Another major advantage is that if the callsign is missing an aircraft may be matched upon its registration code.

The registration codes for lots of airlines in the format needed by PSXT_MSFS can be found here.

3.6.1 Wild cards in registration codes

You may use **one** wildcard (the Asterisk sign **'*'**) in a registration code at either the fourth position (first position is 1) or the last position.

3.6.1.1 Last character wild card:

Instead of

atc_id=A6-EBA,A6-EBF,A6-EBH,A6-EBK,A6-EBL,A6-EBM,A6-EBP,A6-EBR,A6-EBS,A6-EBT,A6-EBU,A6-EBV,A6-EBX,A6-EBZ,A6-ECB,A6-ECE,A6-ECF,A6-ECG,A6-ECH,A6-ECI,A6-ECJ,A6-ECK,A6-ECQ,A6-ECR,A6-ECS,A6-ECT,A6-ECU,A6-ECV,A6-ECW,A6-ECX,A6-ECY,A6-ECZ,A6-EGA,A6-EGB,A6-EGC,A6-EGD,A6-EGE,A6-EGF,A6-EGG,A6-EGH,A6-EGI,A6-EGJ,A6-EGK,A6-EGL,A6-EGM,A6-EGN,A6-EGO,A6-EGP,A6-EGQ,A6-EGR,A6-

EGS,A6-EGT,A6-EGU,A6-EGV,A6-EGW,A6-EGY,A6-EGZ,A6-ENA,A6-ENB,A6-ENC,A6-END,A6-ENE,A6-ENF,A6-ENG,A6-ENH,A6-ENI,A6-ENJ

you may specify

atc_id=A6-EB*,A6-EC*,A6-EG*,A6-EN*

This is a lot less work, saves data in PSXT_MSFS, speeds up livery search and is future proof while when Emirates adds a new 777 (A6-ENK), it will be recognised immediately without you having to change your aircraft.cfg file.

3.6.1.2 Fourth character wild card:

You may also use a wildcard for the fourth character (start counting with 1); this is handy for US carriers.

For example:

atc_id=N72*AN

3.6.1.3 Special German Helicopters "wild card":

You may use code D-H to match all German Helicopters (with registration codes D-H***).

atc_id=D-H

3.7 Non-standard (additional) ICAO airline codes

3.7.1 Generic aircraft

Generic aircraft in house livery or white livery should be given the code **ZZZ**.
Generic codes used in previous versions such as BOE, AIB, EMB, BBA and AED are translated to ZZZ automatically.

3.7.2 Private aircraft

Private (GA) aircraft often do not have an airline code. I recommend to use a non (ICAO) existing airline code **PVT** (for Private) for these type of aircraft. So change the atc_parking_codes lines to

"atc_parking_codes=PVT" or for IVAO_MTL: "title=<aircrafttypecode>PVT"

in aircraft.cfg files that define a private aircraft livery. You can use this code for parking positions in an Airport file (ICAO.txt) of PSXT_MSFS in order to see a PVT aircraft at a certain gate.

For the perfectionists there is also an option to use country specific private airline (General Aviation) codes, see checkbox 3.2.1. These codes are defined by Brian van Hout. If you want to add some, please contact him.

In this document GA Aircraft per country you will find the current set (73 codes).

3.7.3 Government aircraft

Government aircraft sometimes do not have an ICAO airline code. I recommend to use a non (ICAO) existing airline code **GOV** for these type of aircraft. So change the atc_parking_codes line to

"atc_parking_codes=GOV" or for IVAO_MTL: "title=<aircrafttypecode>GOV" in aircraft.cfg files that define a government aircraft livery. You can use this code for parking positions in an Airport file (ICAO.txt) of PSXT_MSFS in order to see a GOV aircraft at a certain gate.

For the perfectionists there is also an option to use country specific GOV/Military airline codes, see checkbox 3.2.1. These codes are defined by Brian van Hout. If you want to add some, please contact him.

In this document Military and Govt Aircraft per Country v6 you will find all military airlines including 93 new non-ICAO codes (in red colour).

4 Search results

AILG_MSFS reports next to each folder how many valid liveries were found. Below the folders area the search results are shown.

Note that even if AILG_MSFS does not show any errors there still can be problems with your AI aircraft.

4.1 Invalid airline codes

The number of invalid airlines codes found. See also section 6.4

4.2 Fixed aircraft type codes

Some obviously wrong Aircraft type codes will be fixed automatically. Fixes are read from the file data/fixes.xml.

The fix is applied to the information that is stored in AI_Liveries.xml, your aircraft.cfg file remains untouched.

4.3 Invalid aircraft type codes

The number of invalid aircraft type codes found. See also section 6.4

4.4 Invalid aircraft.cfg

The number of aircraft.cfg files that could not be processed, for instance because of non-consecutive numbered [fltsim.x] sections. See the log file for details.

4.5 Empty aircraft.cfg

The number of aircraft.cfg files that did not contain a [fltsim.x] section. See the log file for details.

4.6 Invalid [fltsim.x]

The number of [fltsim.x] sections that could not be processed, most often because of a missing **atc_parking_codes** value. See this file (section 6.4) for detailed info.

4.7 Total valid found

The total number of liveries that were valid for use in PSXT_MSFS.

5 Finalizing the search results

AILG_MSFS continuous with executing some extra functions of which the results are shown below the horizontal white line.

5.1 Skipped on priority

A livery of the same type and the same airline in a lower priority folder will be skipped if it has no registration code or a registration code that is already present at a higher priority folder. Liveries that were ignored are written to a file, see section 6.3.

5.2 Skipped on reg

A livery with a registration code that is already present will be skipped. Liveries that were ignored are written to a file, see section 6.4.

5.3 % without reg

The percentage of liveries that have no registration code. Note that for optimal livery matching in PSXT_MSFS a registration code is a great help.

The liveries that have no registration code are written to a file, see section 6.3.

5.4 Unique airlines and unique types

Just for information.

5.5 Number of liveries written to AI_liveries.xml

At the end all valid liveries will be written to the file **AI_liveries.xml**, see section 6.1.

6 Output files

The **AI_liveries.xml** file is of course the **primary output** of this application.

With the other files it will be easier for you to track/pinpoint, and manually repair errors in your AI packages. You should do that if you want to make the best out of your AI package, otherwise you can also choose to go on without paying attention to it.
Even with red lines in the UI you can just go on and start PSXT_MSFS!

6.1 AI_liveries.xml in the Output folder

The file AI_liveries.xml is to be used by PSXT. You should not make any changes in it.

Important!

Make the liveries accessible in PSXT_MSFS:
Change the Liveries folder parameter in the PSXT UI (see PSXT_MSFS Manual section 4.2.1) into a path to the Output folder of AIL. If PSXT_MSFS and AILG_MSFS are at the same PC, you can just push the **connect button.**

6.2 Log.txt

The file log.txt, in the AILG_MSFS folder, logs the last session(-s). The log file of the previous session will be renamed to Log0.log.

6.3 Files with more information about your liveries

- **info\AirlineData.txt**
  Gives an overview by Airline, Type and Folder of the liveries by their Registration code(-s). This may help you to see what you miss. It also helps in organising the Registration codes, by introducing wild cards.
- **info\AircraftCfgWithNoFltsimSection.txt**
  This file gives all the aircraft.cfg file(-s) that are "empty", having no [fltsim.x] section.
- **info\EmptyRegistrations.txt**
  This file gives all the [fltsim.x] sections that have no (valid) atc_id key.
- **info\LiveriesSkippedOnPriority.txt**
  It shows the livery that was skipped and the livery that was already in (at a higher priority).

6.4 Files with information about errors in your liveries

- **errors\InvalidAirlineCodes.txt**
  This file contains invalid airline codes.
- **errors\InvalidAircraftCodes.txt**
  This file contains invalid aircraft type codes.
- **errors\IncompleteFltsimSections.txt**
  The file contains the incomplete [fltsim.x] sections; a section is incomplete: if the title key is not unique, if the texture key is invalid, if the model key is invalid, if the sim key is invalid, if there is no or an invalid value for the atc_parking_codes key, if there is more than one title key, if there is more than one atc_id key, if there is more than one atc_parking_codes key.
  If incomplete the livery is skipped.
- **errors\LiveriesSkippedOnRegistration.txt**
  It shows the Registration code and the livery that was skipped, and the Title of the livery that was already in.