Dart’s Colonisation Helper

Version history (2.4)

* Added add/remove commodities from Template maker
* Tracker can now auto-generate depot files -updated text to reflect that
* New feature the \*All\* depot -an aggreagate of all depot requirements and deliveries
* ‘Archive Depot’ button added to Monitor
* ‘No depot files’ case now handled without errors
* Explicit ship pad size lookup – no longer guessing based on cargo capacity

DISCLAIMER

Please remember this is non-commercial software intended originally for the author’s personal use only. I cannot be sure my scripts will work properly or at all in all situations or system configurations. Whilst I have taken care not to include any code that might cause harm your computer, I do not accept liability for any damage direct or indirect that may be caused by using my scripts. It is provided as-is and without any warranty express or implied whatsoever.

# What is it?

Dart’s colonisation helper is a set of Powershell Scripts that helps Elite Dangerous Commanders manage the delivery of goods to colonisation depots. It tracks the purchase and movement of goods from markets to colonisation depots, optionally via a carrier. Once properly setup, the scripts will automatically track commodities being transferred, and provides a dashboard giving real-time progress – what to buy, what to deliver, what has been completed.

There are 3 scripts in the set:

* Tracker: This script analyses the journal files and updates a collection of ‘depot’ files, a ship file and a carrier file. It should be running whenever you are transferring goods for colonisation. The tracker tracks:
  + Goods bought from a market port,
  + Goods sold to a market port,
  + Goods bought off of your fleet carrier,
  + Goods sold to your fleet carrier,
  + Goods directly transferred from fleet carrier to ship’’’,
  + Goods directly transferred from ship to fleet carrier,
  + Goods unloaded at a construction site or colonisation ship (aka ‘depot’).

All other transactions are IGNORED, examples include dumping of commodities, gaining commodities through missions, salvage or mining etc. Avoid doing these kinds of activities, as you will need to manually correct any errors caused.   
The Tracker script does not have a user interface, it runs silently in the background.

* Monitor: This is the dashboard for the status of your colonisation commodity transaction . It reads the data files maintained by the tracker and displays their contents in a simple table format. If you are supplying for more than 1 depot and have more than 1 depot file, the monitor allows you to select any one of the depots at a time, plus it can display an aggregated view of all depot requirements and deliveries. The monitor can also launch an editor: this can be used to adjust commodity numbers manually if, for example, the tracker was not running when you bought/sold or transferred goods. Occasional Elite dangerous server errors will miss or rollback transactions, or the ED client will fail to correctly update commodity numbers despite writing to the journal, and this will also require manual editing to correct.
* Template maker: Each depot that you wish to manage with the Colonisation helper needs a depot file. This script is used to initialise each depot data file. It contains basic details of the depot, and a list of commodities, how mush is required and how mach has been delivered. There is no limit to how many depot files you can have, The template maker also has a feature to initialise a Carrier file from Inara data, which store the details of commodities located on your carrier. The Template make has a simple tabular user interface that allows to to select a Installation type as a a staring point, then edit it to fit your specific installation (specific quantities vary slightly for every new facility)

# Definitions

* Monitor. The Colonisation helper monitor script
* Tracker. The Colonisation helper tracker script
* Depot. Generic term for any construction site or colonisation ship where commodities can be deposited. The tracker will track cargo requirements and deliveries for as many depots at you need in individual depot files stored in the Progress directory
* Carrier. A carrier owned by you, or being used by you as intermediate storage and bulk transport for colonisation commodities. The tracker can track cargo on one carrier only in a carrier file stored in the Progress directory.
* Ship. The ship you are flying that is moving colonisation commodities. The tracker will track the ship cargo in a ship file stored in the Progress directory.
* Progress directory. Where ship, carrier and depot files are stored, directory is called progress and must be located in the same directory as the scripts.
* ED. Elite Dangerous live version with or without Odyssey DLC

# Setup

You must have Windows Powershell installed. The scripts have only been tested with version 5.1, later version may work but have not been tested. The scripts have only been tested on Windows 11, but should work on any version compatible with ED and able to run a compatible version of Powershell. The scripts are intended to be installed on the same computer that is running ED.

This software should be treated as being in the ‘Beta’ development and testing stage, as such you are expected to have a basic understanding of how computer files and directories work, and how to open, modify and save text based files. A basic knowledge of powershell and json files would be useful but not essential.

The scripts expect to be in a specific location on your hard drive. By default this is C:\data\scripts\colonise for all scripts. Supporting files for the scripts are located in this directory and a ‘templates’ subdirectory. Live data files ( the depot, \_ship and \_carrier files) are located in the ‘progress’ subdirectory. For standard use, you can move the ‘colonise’ directory to anywhere you like, the scripts will compensate for changes asn long as the progress and template direcories stay below the new top level directory. If you do move the top level directory, the supplied shortcuts will need to be edited, and if you are game enough to want to debug or modify these scripts, there is a $homedir variable in each script that you will need to adjust before launching in ISE. The scripts and supporting files will be delivered in a zip to preserve the correct relative directory structure and initial working files– you will need to unpack the zip file and ensure they are either located in the default ‘home directory’ or you make the necessary adjustments to the scripts.

My powershell installation is ‘default’. If your is not then I cannot guarantee the scripts will work without modifying your powershell setup. One issue I already know will cause problems is the launching of the monitor and template maker scripts– which requires a certain .NET assembly to be loaded before launching. This is not a problem if using the ISE, but to use it in a non-development environment I launch the scripts using a shortcut that preloads the necessary assembly before launching the script. I will provide the shortcuts in the zip file – copy them and edit them if you change the home directory. For information the shortcuts run the following commands:

**Template maker:**

C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -c "using namespace System.Windows.Forms;add-type -AssemblyName System.Windows.Forms; . C:\data\Scripts\Colonise\ColonisemakeTemplatesV2.ps1"

**Tracker:**

C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -c "using namespace System.Windows.Forms;add-type -AssemblyName System.Windows.Forms; . C:\data\Scripts\Colonise\ColoniseTrackerjobV2.ps1"

**Monitor:** (optional – the tracker will auto-launch the monitor, but having a shortcut is useful if you want to restart the monitor without stopping the tracker)

C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -c "using namespace System.Windows.Forms;add-type -AssemblyName System.Windows.Forms; . C:\data\Scripts\Colonise\ColoniseMonitorGUIV2.ps1"

On first setup check that the files \_ship.json and \_carrier.json exist in the progress directory. If they are not already there, **copy** them (dont move them) from the templates directory to the progress directory – these are starter files for your ship and carrier. Even if you don’t have a carrier, the \_carrrier.json file must be present (The script assumes a carrier is used but doesn’t require you to actually use it).

**If using a carrier:**

Open the \_carrier.json file in the progress directory (not templates) and locate the following line:

"registration": "AAA-AAA",

Change AAA-AAA to match exactly the registration number of the carrier that you will use for colonisation purposes (usually your own, but it doesn’t have to be). Watch out for number-letter confusion. Be careful not to change anything else, including the quotation marks, colon and trailing comma. The tracker script presently only supports 1 carrier, and this is it’s file. If the registration is incorrect the tracker will assume you are buying/selling from/to a regular market and wont update the carrier cargo data. If you make a mistake re-copy it from the template directory and start again.

If you have ANY cargo on your carrier, you should initialise the carrier file to match your current cargo. See instructions for using the template maker to load Inara data, or the use the monitor ‘Edit’ function to add colonisation cargo data manually.

If you have any colonisation related cargo on your ship, you should offload all cargo before starting the tracker first time. Alternatively (less preferred) use the edit function of the monitor script to manually adjust your ship’s cargo.

My scripts assume you are going to track at least 1 depot and will need to find at least 1 depot file to work properly. The Monitor script needs an active depot to display or it will throw errors. One or more ‘Demo’ depot files may already be present in the progress directory, or you can create your own using the template maker.

*Advanced users: You can also initialise cargo by directly editing the \_ship.json and \_carrier.json files with your favourite editor or scripting tools. JSON was chosen as a ‘human-friendly’ option compatible with most script languages and data manipulation tools. The structure of the data in these should be easy enough to understand.*

# Setting up depot files

The easiest way to create a depot file for an existing construction site is to simply dock at the site wile the tracker is running, this automatically creates a depot file.

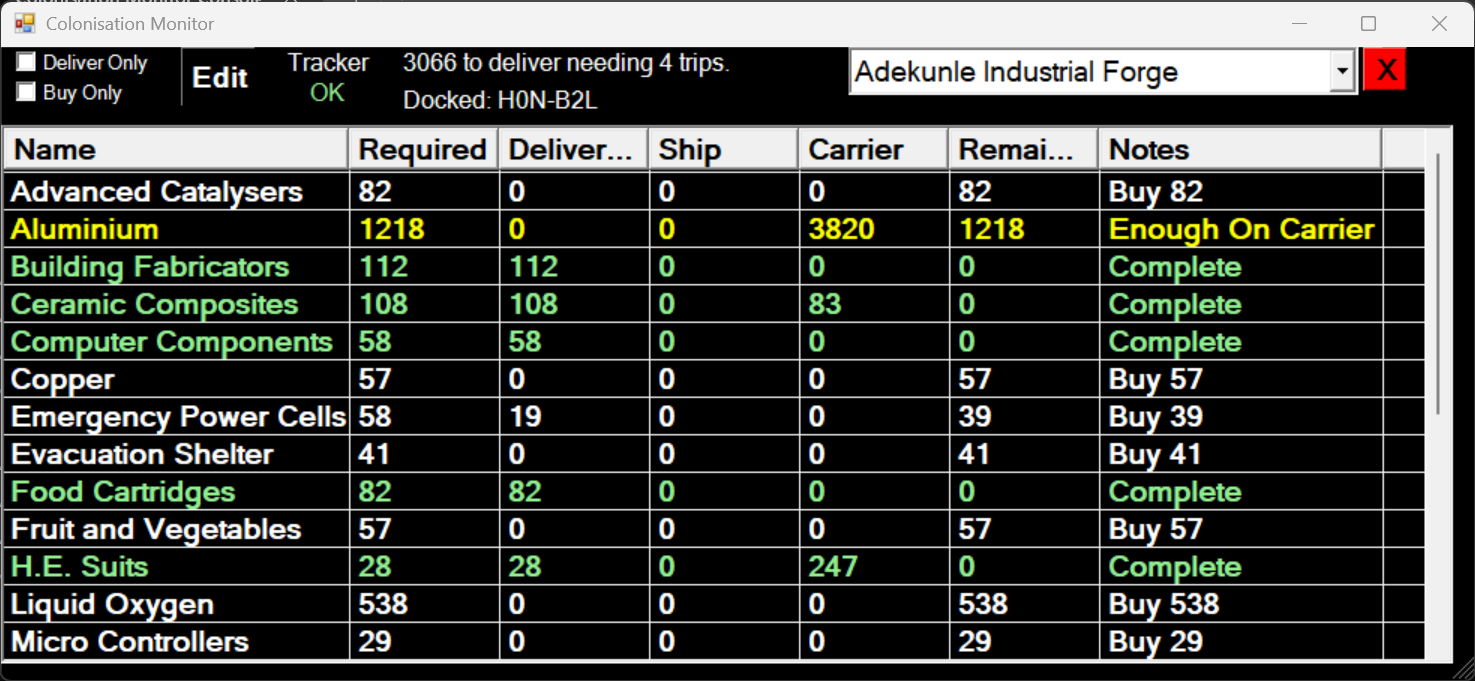
The second way to create a depot file is to use the Template maker. This could be used for example to prepare a depot file in advance of starting the construction, or to create demo files for what-if scenarios (*The Template Maker was written at a time when the ED journals did not contain any information about depots and an external method to create depot files was needed. The template maker is kept now as a legacy method and support for it may be removed in the future*)

# Using the Tracker

Running the tracker opens a text window that displays debug information only. The tracker requires no user input and provides no user interface, but a powershell console window will be present displaying debug data. The tracker needs to be running whenever ED is running and you are transferring cargo that impacts one or more of your depots.

# Using the Monitor

This is the dashboard for your colonisation commodities and looks like this when first opened and a depot is properly setup:

Monitor Script Window

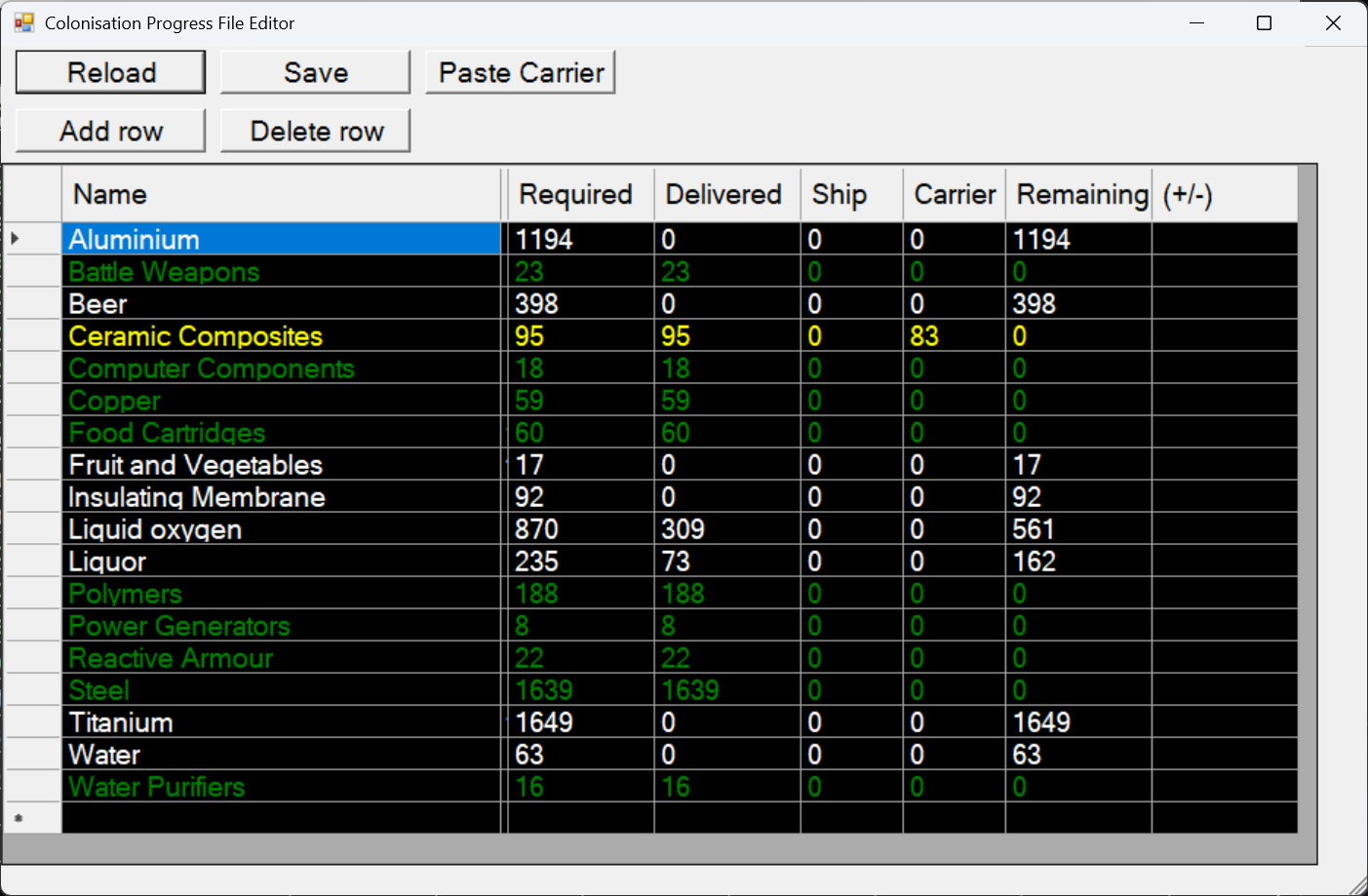
Features:

* The main table presents a list of commodities required by the Active Depot. Columns are
  + Name: Name of each commodity.
  + Required: How much is needed for the current Depot.
  + Delivered: How much has already been delivered.
  + Ship: How much is currently on your ship.
  + Carrier: How much is currently on your carrier.
  + Remaining: How much is yet to be delivered i.e. Required-Delivered.
  + Notes: Info about the status of this commodity with respect to the Active Depot.
* Table colour coding:
  + Green: delivery completed
  + Yellow: Enough bought and stored on carrier to complete delivery
  + Orange: Enough bough and stored on ship and carrier to complete delivery
  + White: Need to buy more
  + Blue: Enough on ship to complete delivery.
  + Red background: Too much on ship (overstock) This is not an error, just warning in case you did not intend to buy extra.
  + Blue background: When docked and after opening the station services commodity market the background changes to blue for commodities that you need and can be bought here. This also adds a temporary ‘Stock’ Column that shows how much is available to buy at this market. Market information will disappear when you launch to leave the station.
* “Deliver Only” checkbox: (top left) Activates a filter that only shows commodities that still need to be delivered. You can only have 1 filter selected at a time.
* “Buy Only” checkbox: (top left) Activates a filter that only shows commodities that still need to be bought. You can only have 1 filter selected at a time.
* “Edit” button: (top) Opens the editor to allow manual adjustment of quantities. This button is disabled when the active depot is \*All\*. Switch to a single depot if you need to correct an error
* “Tracker”: (top) Indicates if the state of the tracker script: Green OK if it is running, Red OFF if it is not.
* Info Lines: (top) The first line gives an overall summary of how much still needs to be delivered for the Active Depot. The second line is the name of the journal file currently being tracked by the tracker script (primarily for debug reasons)
* Active Depot Selector: (Top right) This is a drop-down list box of known depots. The Active Depot is displayed when the list is not expanded. The last entry is always \*All\* - this will generate an aggregate view across all known depots. Use this when in a buying cycle and you are buying for several depots at the same time. *Note: Regardless of which depot is active, the Tracker will still track deliveries to all depots, the Active Depot Selector only affects which depot is displayed on the monitor, and which Depot the Editor will edit.*
* Archive Depot: (Top Right, red X button) Remove the active depot from the selector list; does nothing if the active depot is \*ALL\*. A prompt will appear first to confirm this action. The depot file is not deleted, but renamed with an ‘.archive’ extension and left in the progress directory, after which the monitor and tracker will ignore it. It can be returned to use by removing the extension, or permanently deleted with file explorer.
* Inara commodity search: Double-clicking on any Commodity name in the Name column will open Inara and search for that commodity near the last known ship location. See known issues regarding search criteria limitations.

The Monitor script can only display information for 1 depot at a time. You can switch between depots at any time, but if you want to have a real-time update of 2 or more depots, just run another copy of the monitor and select a different depot. If you have more than one site under construction the \*All\* depot may be used so that you can see your total commodity requirements when , for example, you are filling a carrier from some distance away from your systems.

# Using the Editor

This is the manual editor you open from the Monitor script “Edit” button when viewing a single depot only, it will not open if the active depot is \*All\*:

Editor

The Editor replicates the monitor main window, however in the editor the numeric fields can be edited. After making changes click save and the changes are written to the appropriate files. Some columns are linked: where appropriate a change in a column will update other columns, for example updating Delivered or Required will auto-update Remaining. Updating ‘Remaining’ direclty is useful to help update the depot file with the depot’s own in-game requirements list if they get out of step.

The (+/-) column will add an amount to the Delivered column and subtract it from the Remaining column to allow for manually reporting deliveries that occurred by others or by you when the tracker was not running. To undo a delivery, use a negative number.

The Reload button will revert the values to the what they were after the last save.

Save button will write the changes. Cannot be un-done. You must save to keep changes, closing the editor without saving will leave the progress un-changed.

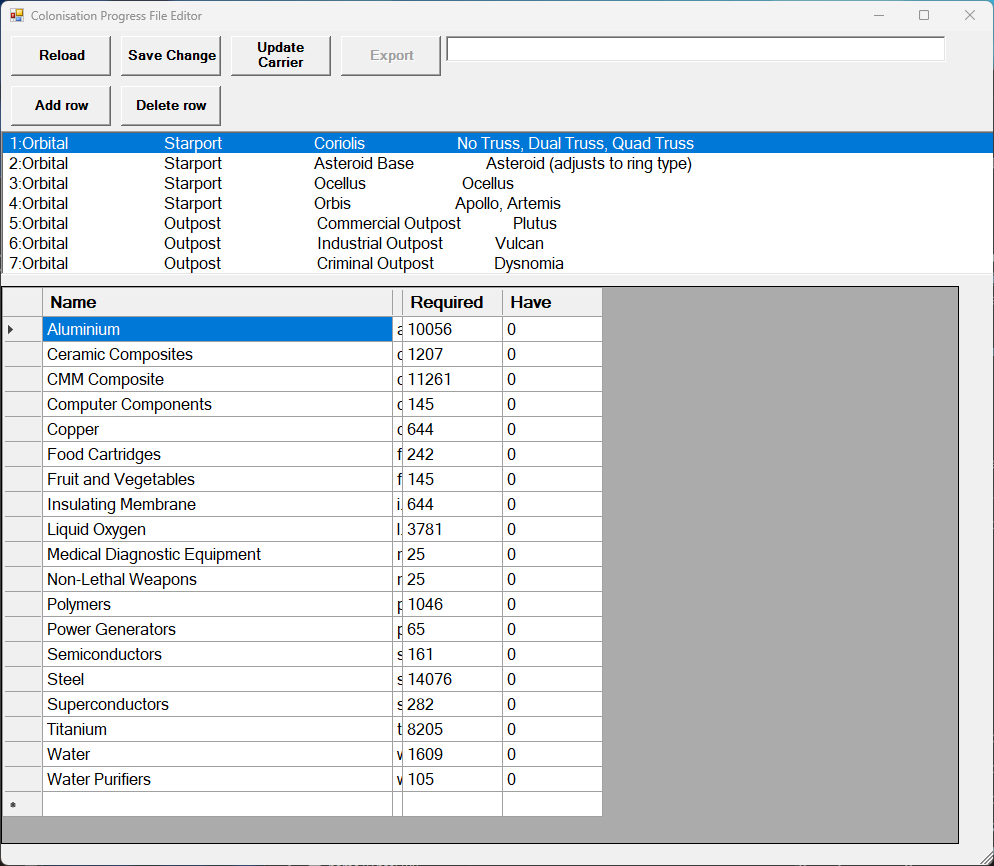
Paste Carrier. (Also found in the template maker.) Use this when the actual cargo and carrier file data is different. Requires an Inara account linked to your commander. The method for using the ‘Paste Carrier’ function:

* Open Inara and refresh your commander data,
* Navigate to the carrier’s cargo page (from your Commander page: Hangar, Fleet Carrier, Cargo),
* right-click ‘Select-All’,
* Copy the selected data (whole page),
* Return to the template maker and click on “Update Carrier”,
* If you have not properly captured the carrier data you will get an error message and nothing will change.

# Using the template maker

The Template maker was created to make depot files before Frontier Developments updated the journal files to include detailed construction site requirements and progress. The template maker is retained for the situation where you might want to create a template before the construction site is started, or you want to start collecting commodities before your first visit to the site. Support for this tool may be dropped in the future.

Running the template maker script opens the following Window:



To create a new depot:

* Select a depot type from the list of available types.
* Edit the required quantities to match your depot. The provided required values are typical but are often wrong.
* If you have already delivered to this depot you can either adjust the ‘Required’ quantity or set the delivered quantity in the ‘Have’ column.
* Warning: do not click save change – with will write the file back into the master templates database– use this button only to save a change that will affect future depots that uses this template.
* Enter the name of the depot name into the text box next to “Export” using these rules:
  + The name for a colonisation ship is “System Colonisation Ship <system name> for example the depot name for the system colonisation ship in HIP 1234 would be “System Colonisation Ship HIP 1234”
  + The name for an orbital or surface facility is the name of the COMPLETED facility, for example the depot name for “Planetary Construction Site: Li Tourist Haven” is just “Li Tourist Haven”   
    *Note : Any errors with the name will cause the tracker to NOT update the file when deliveries are made to this depot. The monitor will tell you when you dock at a depot whether the tracker has a depot file so you have a chance to correct it before delivering. (see monitor usage section for more details)*
  + Click Export (only available when the adjacent text box has something in it). The export button creates a depot file into the progress directory with a name matching the depot name and a .json extension. The name is also embedded in the file – if the name needs to be corrected after creating the file, the file name and the embedded ‘name’ field needs to be changed

**To update the carrier file with the latest data from Inara:**

*Only applicable if you are registered with Inara and allow Inara to update from ED servers.*

Do this when you have moved cargo on or off the carrier without the tracker running to capture changes and the carrier file no longer matches the actual carrier cargo. Its a good idea to do this whenever starting a new depot to clean up any discrepancies. *Note: this tool is also available in the Monitor script.*

* Open Inara and refresh your commander data
* Navigate to the carrier’s cargo page (from your Commander page: Hangar, Fleet Carrier, Cargo)
* right-click ‘Select-All’
* Copy the selected data (whole page)
* Return to the template maker and click on “Update Carrier”
* If you have not properly captured the carrier data you will get an error message and nothing will change.

**Other features:**

* Add row: Will allow you to add a commodity to the table. Clicking will bring up a list of commodities not already present in the table, select one or more then ‘Add’
* Delete Row: Will allow you to delete a commodity from the table. Select the whole row you want to delte (not just a cell) then click then ‘Delete row’.
* Reload: Reset the table to the last saved state
* Save Change: This will permanently update the template for this construction type. Use with caution. Generally only exporting should be required, but this feature may be need to correct any changes to construction requirements

Currently there is no direct method to create new templates. The database file would need to be hand edited to insert additional construction types if any new ones are needed.

# Known Issues

* I don’t track everything that might cause a cargo change on your ship. This includes mining, dumping/ejecting/destroying, salvage operations, loss or gain during combat, using the refinery contact to convert ore into refined goods. This is not an exhaustive list. Avoid doing these while using the tracker script, or be prepared to edit the data if you do. The script will autocorrect some missed transactions if and when the appropriate journal entries occur. I have listed what I do track elsewhere in this document, and I believe I have covered all the activities you would expect to do during a colonisation supply effort.
* I dont track what other commanders are doing to your depots or your carrier. For that you will need to keep an eye on the outstanding requirements list for your depots in-game and the carrier’s current cargo either in game or with Inara. For the carrier you can use the import carrier function or manually with the editor. The depot corrections are manual only using the editor.
* Server issues sometimes delay or rolls back cargo related operations. I have noticed that Transferring, buying, selling etc just after docking when station services is supposedly available, but before the ship has completed the docking animation and sounds during a high server-lag period might mean the transaction is lost despite the relevant entries going into the journal. Repeating the transaction means a duplication in the journal and therefore by the tracker that then needs to be manually fixed.
* I don’t catch some obvious errors like missing \_carrer or \_ship files. I have assumed the \_carrier.json and \_.ship.json files will always be present.
* The monitor can now archive a depot file when no longer wanted to hide it from the depot selector. Deletion of unwanted depot files is up to the user using Windows provided tools for managing files.
* Inara commodity search: The search criteria is hard coded to the usual defaults. Except for location (set to current system), commodity (what yo clicked on) and minimum pad size (set according to the ship model). New ships added in the future need to be added to the padlookup.csv file. Each use of this features opens another tab on the default browsers – it does not reuse the current tab, so remember to close the Inara search when finished.