RCLogbook

Database Import Guide

Version 4.5.2, May 2014

clevertangerine software

Table of Contents

Welcome

Overview

This guide covers the format of the text file that RCLogbook uses to support importing information into its database. This text file specifies both new records to add to the database as well as changes that RCLogbook should make to records that are currently in the database. For detailed discussion on how you can use the import functionality in the application, please consult the *RCLogbook User's Guide*.

☝ *The user's guide is available either from the link on the RCLogbook "About" screen or from the product webpage at http://www.clevertangerine.com/RCLogbook*

The import functionality that was added in RCLogbook 3.5 allows you update some, but not all, of the content in the RCLogbook database from a text file that you can edit on your Mac or PC. It is important to understand the assumptions that RCLogbook makes around *how* it expects the import functionality is used.

The import functionality **was intended** to support…

- Making batch additions to your database. For example, import allows you to incorporate existing logs into RCLogbook without having to go through the UI.
- Making many simple edits to a subset of the database content.

The import function **was not intended** to support…

- Making backups of the database. Because not all records or database fields can be imported (for example, checklist records or the image of a model), exporting, then importing, will not generally get you back to your original database.
- Removing records or information from the database.

The import functionality is **not** intended to be a substitute for basic editing that can be done through the UI.

☝ *With Great Power comes Great Responsibility. You can make a mess of your database if you are not careful. RCLogbook does its best to keep you from doing so, but please exercise extreme caution when using the import functionality.*

RCLogbook tries to be very picky about the formatting and contents of the import file in order to try to protect you from some common mistakes.

Conventions

As seen above, this guide uses the following notation to indicate a point of interest that explains a point further or brings attention to differences with a prior release.

RCLogbook 4.1 can now easily spin the black circle.

This guide typesets text file contents like this:

```
1   [SECTION]➔Locations
2   Location UID➔Location Name➔Latitude➔Longitude➔Location Notes
3   354➔Apple Computer➔37.331607➔-122.030382➔<Notes Go Here>
```

The symbol "➔" represents a tab character. The numbers in italics at the left of each line help identify the line number within the file; these numbers do not actually appear in the file. Within the body of the file, we use italic text in angle brackets to identify placeholders. In this example, the text "*<Notes Go Here>*" on line 3 does not literally appear in the file, but instead it calls out that the notes would be located at this point in the file.

Providing Import Files to RCLogbook

As mentioned above, RCLogbook interprets a user-specified text file to decide what changes need to be made to the database. This text file can be provided to RCLogbook through a variety of methods accessed through the Database tab in the RCLogbook user interface:

- Web Server – the user connects to RCLogbook through a web browser and selects a text file from their device to upload to RCLogbook.
- Dropbox – RCLogbook uses a text file from the user's Dropbox. The import file is always named `DbImport.txt` and stored in the RCLogbook folder in the Dropbox Apps folder; that is, `/Apps/RCLogbook/DbImport.txt`

Both methods operate in the same fashion once the import file is provided to RCLogbook.

Dropbox is supported on RCLogbook version 4.2 and later. In order to use Dropbox, you must have an active account with Dropbox. For more information, see http://www.dropbox.com.

Contacting Us

We welcome your feedback and suggestions. Many of the new features that have been added since 1.0 have come from suggestions by users. You can reach us on the web at http://www.clevertangerine.com or over email at support@clevertangerine.com. Links can be found on the about page reached by tapping the "i" icon in the lower right of the tab bar in the application.

Import File Format

The import file format is very similar to the text-based tab-separated value format that RCLogbook uses for its text database exports.

RCLogbook versions earlier than 3.5 used a comma-separated format.

An import file consists of a number of *lines* each terminated by one or more newline characters[1] where each *line* is divided into one or more tab-delimited *values*. By definition, *values* cannot contain newline or tab characters. Blank lines may appear in the import file but are always ignored. Here is an example of this basic structure to illustrate these concepts,

```
1   iPod→→iPhone
2
3   iPad→iMac
```

Line 1 has three values "`iPod`", "", and "`iPhone`" (back-to-back tabs imply a blank value). Line 2 is blank and has no values. Line 3 has two values "`iPad`" and "`iMac`".

An import file always begins with a set of *header lines* that establishes the file format and version of RCLogbook that can process the file. Next, the middle part of the file is made up of groups of lines, called *sections*, where the *values* describe changes to *fields* in database records. Finally, the import file ends with a set of *footer lines* that mark the end of the file.

Data Types

Each *field* in a *database record* carries data of a specific type[2]. As a result, when the import file specifies changes to a *field* (through a *value*), the *value* must also provide the appropriate type. The specific mapping between *fields*, *database records*, and *values*, is covered later when we discuss the specific sections that the import file supports.

[1] Specifically, the characters are a CR and LF pair, though RCLogbook is flexible about the specifics.
[2] Examples of types include a string, integer number, real number, date, etc.

The data types that can appear in the import file are as follows,

Data Type	Description
Unique ID	Database record unique identifier. Identifiers are always positive integers greater-than zero. These values are never assigned from the import file. To indicate a new record, the special value “[NEW]” can be used (RCLogbook will then automatically assign a new unique identifier to the new record).
Integer	Any integer number. For example, 97 or -13.
Integer (P)	A positive or zero integer number. For example 97 or 0.
Integer (P, NZ)	A positive and non-zero zero integer number. For example 97.
Real	Any real number. For example, 6.16 or -9.25.
Real (P)	A positive or zero real number. For example, 6.16 or 0.0.
Real List (P)	A “;”-separated list of positive or zero real numbers. For example, “6.16 ; 0.0 ; 1.23”.
Boolean	Either “Yes” or “No”.
String	A text string. Unless otherwise indicated, strings are reasonably free form (they may not, however, contain a tab character as these delimit values). Exceptions are noted later when the sections are described. To indicate an empty string, the special value “[CLEAR]” should be used (this is necessary since blank fields are always ignored).
Date	A calendar date. The expected format is indicated in the field definition in the header and is based on the current locale. For example, in the US, the default format is “M/D/YY” and December 25, 2011 is represented as “12/25/11”.
Time	A time. The expected format is indicated in the field definition in the header and is based on the current locale. For example, in the US the default format is “H:MM A” and five past one in the afternoon is represented as “1:05 PM”.
Duration (MS)	A duration in minutes and seconds in the format M:SS. The seconds, SS, must be two characters long and be between 0 and 59. For example, “3:03” is three minutes, three seconds.
Duration (MS, NZ)	A non-zero duration in minutes and seconds in the format M:SS. The seconds, SS, must be two characters long and be between 0 and 59. For example, “3:03” is three minutes, three seconds.
Duration (HMS)	A duration in hours, minutes, and seconds in the format H:MM:SS. The minutes, MM, and seconds, SS, must each be two characters long and be between 0 and 59. For example, “2:03:23” is two hours, three minutes, twenty-three seconds.

A blank *value* is always legal for any of these types. Like blank lines, RCLogbook always ignores blank *values*.

Header Line

The header consists of one line that marks the start of the file and identifies the version of RCLogbook that can process the import file. Specifically,

```
1    [INFO]➔RCLogbook Database (Release 4.1 with DB v.6)
```

RCLogbook requires that the version in the import file header match the application and database version. RCLogbook 3.5, for example, can only import from a file that is formatted for RCLogbook 3.5.

This line is always the first non-blank line in the import file.

Footer Line

The footer consists of a single line that marks the end of the import file. Specifically,

```
1    [INFO]➔End of File
```

This line is always the last non-blank line in the import file.

Sections

Sections describe the modifications to related groups of database records.

Sections are always at least three lines long. The first line is a *section marker* that marks the start of the section and identifies the type of section that follows. The second line is the *field definition* that identifies the fields that are present on each of the remaining lines of the section. Finally, the third and all subsequent lines of the section make up the *database changes* that describe how to change the database.

In the *database change* portion of the section, one or more lines represent changes to related sets of database records. Each specific section defines what constitutes a "related set" of database records. For example, assume that the database has a model record where each model record links to flight records that describe each of the model's flights. In this scenario, a model section might define the "related set" as a model and all of its associated flights. In cases where the related set maps to multiple lines from the import file, the second and subsequent lines within a group always begin with a "+" character as the first value on the line.

To help illustrate what a section looks like, consider an example database that has Model and Flight records that describe models and flights, respectively. Further, assume that each Model can have a number of flights associated with it. As a result, for a "Models" section, the "related set of database records" is a model and all of its flights. An import file for the models in this database might look like this,

```
1   [SECTION]→Models
2   Model ID→Model Name→Flight Number→Flight Date→ Flight Time→Flight Notes
3   1→3DHS SHP→1→6/1/11→8:00 AM→Pattern work
4   2→3DHS AJS→1→6/2/11→8:30 AM→First flight
5   +→→2→6/2/11→9:30 AM→Harriers
6   +→→3→6/2/11→9:45 AM→3D practice
```

To make the structure a little more apparent, this is how the example import file described above would look if you were to bring it into your favorite spreadsheet application,

	A	B	C	D	E	F
1	[SECTION]	Models				
2	Model ID	Model Name	Flight Number	Flight Date	Flight Time	Flight Notes
3	1	3DHS SHP	1	6/1/2011	8:00 AM	Pattern Work
4	2	3DHS AJS	1	6/2/2011	8:30 AM	First flight
5	+		2	6/2/2011	9:30 AM	Harriers
6	+		3	6/2/2011	9:45 AM	3D practice

Line 1 is the section marker. Line 2 is the field definitions: in this scenario, there are Model ID, Model Name, Flight Number, Flight Date, and Flight Notes fields on a line. The remaining lines describe the changes to make to the database. Line 3 describes a change to the “3DHS SHP” model and one of its flights, while lines 4 through 6 describe changes to the “3DHS AJS” model and three of its flights.

The specific content of the field definitions row (line 2 in the file) is important. To avoid misinterpreting import data, RCLogbook checks to make sure the header matches the expected format and will flag an error in the file if it doesn’t match.

Changes to the Import Format

This section lists the changes that have been made to the import format since its original release with RCLogbook 3.5.

RCLogbook 4.5.x does not make any changes to the 4.2.x import format.

RCLogbook 4.2.x does not make any changes to the 4.1.x import format.

RCLogbook 4.1.x does not make any changes to the 4.0.x import format.

RCLogbook 4.0.x makes the following changes to the 3.8.x import format:

- Battery section adds “In Storage?” column.

RCLogbook 3.8.x makes the following changes to the 3.7.x import format:

- Cycles section adds “Discharge Cell Voltage”, “Charge Cell Voltage”, “Charge Resistance”, and “Charge Cell Resistance” columns.
- Events section adds “Discharge Cell Voltage”, “Charge Cell Voltage”, “Charge Resistance”, and “Charge Cell Resistance” columns.
- Added new “Maintenance” section.

RCLogbook 3.7.x makes the following changes to the 3.5.x import format:

- Model section adds “Retired?”, “Default Prop”, and “Default Style” columns.
- Battery section adds “Retired?” column.
- Cycles section adds “Ignore in Plots?” column.
- Events section adds “Style”, and “Propeller” columns.
- Added new “Propellers” and “Styles” sections.

Processing an Import File

RCLogbook processes each section from the import file in the order in which they appear in the file. Changes from a section are applied to the database in the order in which they appear in the section.

As a simple example, consider a database made up entirely of “Character” records. Each such record from the database describes a character and their “friends”. For the purposes of this example, we will assume that the import file uses two different types of sections to represent our database of characters: one that describes the properties of a character (the `Characters` section), and one that describes the friendships (the `Friends` section)[3].

Database changes in the `Characters` section includes the following fields,

Field	Type	Notes
Character UID	Unique ID	Internal unique ID for the character database record.
Character Name	String	Name of the character. All characters are expected to have unique names.
Sith	Boolean	Is the character a Sith?
Weapon	String	Type of weapon the character uses

Database changes in the `Friends` section includes the following fields,

Field	Type	Notes
Character Name	String	Name of the character.

[3] The import file is split in this fashion to ensure that you can define all the characters before you define any friendships – that is, it lets the import file define who could be friends before actually specifying who are friends.

Field	Type	Notes
Friend Name	String	Name of the friend.
Apprentice	Boolean	Is the friend a padawan or apprentice?

The “Friends” section allows you to use continuation lines to specify multiple friends for a character.

Starting with an empty database, assume that you begin by importing the following file (remember that not all sections need to appear in a file).

```
1  [INFO]→RCLogbook Database (Release 3.7 with DB v.4)
2  [SECTION]→Characters
3  Character UID→Character Name→Sith→Weapon
4  [NEW]→Obi Wan→→Lightsaber
5  [NEW]→Darth Sidious→Yes→Force Lightning
6  [INFO]→End of File
```

Since the database is empty, all of the `Character UID` values must be “*`[NEW]`*” to create new records. After we import this file, the database will contain the following information on two newly added characters.

	Character UID	Character Name	Sith	Weapon	Friend Name	Apprentice
Before Import	*Database Initially Empty*					
After Import	325	Obi Wan		Lightsaber		
	102	Darth Sidious	Yes	Force Lightning		

Each row in this table corresponds to a character record from the database. There are several things to take note of:

- The application always picks UIDs for the new records and there need not be any particular pattern to the values it chooses.
- If fields are empty, no changes are made to the database. For example, the “Sith” field for Obi Wan was left blank in the import file.
- This import file does not modify any friendships as it lacks a “Friends” section.

To add some friendships, we import the following file:

```
1   [INFO]➔RCLogbook Database (Release 3.7 with DB v.4)
2   [SECTION]➔Characters
3   Character UID➔Character Name➔Sith➔Weapon
4   325➔Obi Wan Kenobi➔No
5   [NEW]➔Commander Cody➔No➔Blaster
6   [SECTION]➔Friends
7   Character Name➔Friend Name➔Apprentice
8   Obi Wan Kenobi➔Commander Cody➔No
9   [INFO]➔End of File
```

Here is how the database would look before and after importing this file.

	Character UID	Character Name	Sith	Weapon	Friend Name	Apprentice
Before Import	325	Obi Wan		Lightsaber		
	102	Darth Sidious	Yes	Force Lightning		
After Import	325	Obi Wan Kenobi	No	Lightsaber	Commander Cody	No
	102	Darth Sidious	Yes	Force Lightning		
	500	Commander Cody	No	Blaster		

There are several things to take note of:

- This file makes changes to Obi Wan's record (specifically, updating his name and sith status) in the database through line 4 since the "Character UID" line 4 specifies identifies that record.
- This file adds a new "Commander Cody" character (line 5) and then makes him a friend of "Obi Wan Kenobi" (line 8).
- In general, section order matters. In this example, the "Characters" section must proceeed the "Friends" section, since the "Friends" section uses names that might be newly added by the "Characters" section.

To see how continuation lines work, consider the following import file:

```
1   [INFO]→RCLogbook Database (Release 3.7 with DB v.4)
2   [SECTION]→Characters
3   Character UID→Character Name→Sith→Weapon
4   [NEW]→R2-D2→No→
5   [NEW]→Annakin Skywalker→Yes→Lightsaber
6   [NEW]→Darth Vader→Yes→Lightsaber
7   [SECTION]→Friends
8   Character Name→Friend Name→Apprentice
9   Obi Wan Kenobi→R2-D2→No
10  +→Annakin Skywalker→Yes
11  Darth Vader→Darth Sidious→No
12  Darth Sidious→Darth Vader→Yes
13  [INFO]→End of File
```

Here is how the database would look before and after importing this file.

	Character UID	Character Name	Sith	Weapon	Friend Name	Apprentice
Before Import	325	Obi Wan Kenobi	No	Lightsaber	Commander Cody	No
	102	Darth Sidious	Yes	Force Lightning		
	500	Commander Cody	No	Blaster		
After Import	325	Obi Wan Kenobi	No	Lightsaber	Commander Cody R2-D2 Annakin Skywalker	No No Yes
	102	Darth Sidious	Yes	Force Lightning	Darth Vader	Yes
	500	Commander Cody	No	Blaster		
	123	R2-D2	No			
	657	Annakin Skywalker	No	Lightsaber		
	700	Darth Vader	Yes	Lightsaber	Darth Sidious	No

There are several things to take note of:

- A continuation line is used to add two more friends to “Obi Wan Kenobi”. Lines 9-10 in the import file together refer to a single character record (that of Obi Wan Kenobi, specifically). In this simple database scenario, you could also accomplish this with two separate “Obi Wan Kenobi” lines.
- Continuation lines are not needed in this example if you only want to add a single friend.
- Note that there is no way to delete a friend through the import file. Any friendship relationships the “Friends” section establishes **add** to the existing relationships (for example, see how Obi Wan’s record changes).

These examples should give you a feel for how the import works. The next section of this document will look specifically at the section types that RCLogbook supports.

General Guidelines

Here are some general guidelines to help make the import file robust

- It is best to place sections in the import file in the order: `Locations`, `Propellers`, `Styles`, `Models`, `Batteries`, `Events`, and `Cycles`.
- Remember that later changes over-ride earlier changes. For example, a particular database cycle record can appear in both the `Events` section and the `Cycles` section. If you make different changes to the record in each of these sections, the last change will "stick".
- Only put things in the import file that you want to change. This implies that you should blank out the values of any fields that you are not changing in a particular set of records and also limit the contents of the import file to the set of records that you want to change.
- A template for an import file can be found on the database access page. You can also use the text export format as a guide.

Following these guidelines will help make sure the import does what you want it to do.

Section Types

The bulk of an import file is made up of "Sections" that describe changes or additions to related types of information in the database. There are several different types of sections that may appear in the import file as described below.

Each description includes a table that summarizes the fields in the section. The columns in this table include:

- *C* – The order of the field (or column number) within a row.
- *Field* – Name of the field, this appears in the field definition row of the section.
- *Type* – Type of data in the field, for the complete list of possible types, see the table in the "Data Types" section on page 5.
- *RN* – "Y" if the field is required for new records, "N" otherwise.
- *Notes* – Details of interest on the field.

The remainder of this section considers each type of section RCLogbook uses in turn.

Models

The `Models` section provides general information on the models defined in the database. Each line in the *database changes* portion of a `Models` section corresponds to a single model. The fields on each line include,

C	Field	Type	RN	Notes
1	Model UID	Unique ID	Y	Internal unique ID for the model database record.
2	Model Name	String	Y	Name of the model. All models are expected to have unique names.
3	Retired?	Boolean	N	Identifies whether or not the model is retired
4	Model Type	String	Y	Type of model. Legal values include: • Airplane • Helicopter • Boat • Car • Robot • Multicopter No other values are permitted.
5	Vendor	String	N	Vendor of the model.
6	Last Event Date (*F*)	Date	N	Date of the last event. “F” defines the format of the date based on the current locale. For example, “F” is “M/D/YY” in the US. This field must be specified if “Last Event Time” is not blank.
7	Last Event Time (*F*)	Time	N	Time of the last event. “F” defines the format of the time based on the current locale. For example, “F” is “H:MM A” in the US. This field must be specified if “Last Event Date” is not blank.
8	Total Time (H:MM:SS)	Duration (HMS)	N	Total time on the model.
9	Total Events	Integer (P)		Total number of events on the model.
10	Total Fuel (*F*)	Real (P)	N	Total fuel consumed by the model. “F” defines the expected units of the fuel. Fuel units are selected through the application preferences.
11	Logs Batteries?	Boolean	N	Identifies whether or not the model logs battery information.
12	Logs Fuel?	Boolean	N	Identifies whether or not the model logs fuel information.
13	Default Prop	String	N	Default propeller to use for events on this model. The value should match the name of one of the propellers defined in the `Propellers` section of the import file.
14	Default Style	String	N	Default event style to use for events on this model. The value should match the name of one of the styles defined in the `Styles` section of the import file.
15	Model Notes	String	N	User notes on the model.

Continuation lines cannot appear in this section.

Batteries

The `Batteries` section provides general information on the batteries defined in the database. Each line in the *database changes* portion of a `Batteries` section corresponds to a single battery. The fields on each line include,

C	Field	Type	RN	Notes
1	Battery UID	Unique ID	Y	Internal unique ID for the battery database record.
2	Battery Name	String	Y	Name of the battery.
3	In Storage?	Boolean	N	Identifies whether or not the battery is in storage state.
4	Retired?	Boolean	N	Identifies whether or not the battery is retired.
5	Vendor	String	N	Name of the vendor or manufacturer of the battery
6	Chemistry	String	N	Chemistry of the battery. Legal values include: • LiPo • LiIon • LiFe • NiCd • NiMH • Other No other values are permitted.
7	C Rating	Integer (P, NZ)	N	C rating for the battery pack.
8	Capacity (mAh)	Integer (P, NZ)	Y	Total capacity of the battery pack in mAh.
9	Cells (S)	Integer (P, NZ)	N	Number of series cells in the battery pack.
10	Cells (P)	Integer (P, NZ)	N	Number of parallel cells in the battery pack.
11	Total Cycles	Integer (P)	N	Total number of cycles on the battery pack.
12	Battery Notes	String	N	User notes on the battery.

Continuation lines cannot appear in this section.

Locations

The `Locations` section provides general information on the locations defined in the database. Each line in the *database changes* portion of a `Locations` section corresponds to a single location. The fields on each line include,

C	Field	Type	RN	Notes
1	Location UID	Unique ID	Y	Internal unique ID for the location database record.
2	Location Name	String	Y	Name of the location.
3	Latitude	Real	Y	Latitude of the location in degrees (between -180 and 180).
4	Longitude	Real	Y	Longitude of the location in degrees (between -90 and 90).

C	Field	Type	RN	Notes
5	Notes	String	N	User notes on the location.

Continuation lines cannot appear in this section.

Styles

The `Styles` section provides general information on the event styles defined in the database. Each line in the *database changes* portion of a `Styles` section corresponds to a single style. The fields on each line include,

C	Field	Type	RN	Notes
1	Style UID	Unique ID	Y	Internal unique ID for the style database record.
2	Name	String	Y	Name of the style.
3	Notes	String	N	User notes on the style.

Continuation lines cannot appear in this section.

Propellers

The `Propellers` section provides general information on the propellers defined in the database. Each line in the *database changes* portion of a `Propellers` section corresponds to a single propeller. The fields on each line include,

C	Field	Type	RN	Notes
1	Propeller UID	Unique ID	Y	Internal unique ID for the propeller database record.
2	Name	String	Y	Name of the propeller.
3	Vendor	String	N	Latitude of the location in degrees (between -180 and 180).
4	Blades	Integer (P, NZ)	Y	Number of blades on the propeller, must be at least 2.
5	Diameter	Real (P, NZ)	Y	Diameter of the propeller in the selected units.
6	Pitch	Real (P, NZ)	Y	Pitch of the propeller in the selected units.
7	Units	String	N	Units that the diameter and pitch are measured in. Legal values include: • in • cm No other values are permitted.
5	Notes	String	N	User notes on the propeller.

Continuation lines cannot appear in this section.

Cycles

The `Cycles` section provides general information on the cycles defined in the database for various batteries. Each line in the *database changes* portion of a `Cycles` section corresponds to a single cycle of a battery. The fields on each line include,

C	Field	Type	RN	Notes
1	Battery Name	String	Y	Name of the battery that the cycle is for. This name must exactly match one of the batteries in the database.
2	Cycle UID	Unique ID	Y	Unique ID for the cycle in the battery, this cycle must be associated with the battery identified by the “Battery Name” field when editing an existing cycle.
3	Cycle #	Integer (P, NZ)	N	The cycle number for the battery. This field is shown for reference and is appropriately updated automatically. It may be left blank for new batteries.
4	Discharge Date (*F*)	Date	Y	Date of the last discharge. “F” defines the expected format of the date based on the current locale. For example, “F” is “M/D/YY” in the US. This field must be specified if “Discharge Time” is not blank.
5	Discharge Time (*F*)	Time	Y	Time of the last discharge. “F” defines the expected format of the time based on the current locale. For example, “F” is “H:MM A” in the US. This field must be specified if “Discharge Date” is not blank.
6	Discharge Duration (M:SS)	Duration MS	Y	Duration of discharges in minutes.
7	Discharge Voltage (V)	Real (P)	N	Final battery pack resting voltage after all discharges.
8	Discharge Cell Voltage (V)	Real List (P)	N	Final battery pack per-cell resting voltage after all discharges. The list is in order of P then S, so for a 2S/2P pack, values in the list are in the order “1P/1S ; 1P/2S ; 2P/1S ; 2P/2S”.
9	Charge Date (*F*)	Date	N	Date of the last charge. “F” defines the expected format of the date based on the current locale. For example, “F” is “M/D/YY” in the US. This field must be specified if “Charge Time” is not blank. The charge date and time should be later than the discharge date and time.
10	Charge Time (*F*)	Time	N	Time of the last charge. “F” defines the expected format of the time based on the current locale. For example, “F” is “H:MM A” in the US. This field must be specified if “Charge Date” is not blank. The charge date and time should be later than the discharge date and time.
11	Charge Amount (mA)	Integer (P)	N	Amount of energy put back into battery pack during the charge. This value must be greater-than zero.
12	Charge Voltage (V)	Real (P)	N	Final battery pack resting voltage after all charges.

C	Field	Type	RN	Notes
13	Charge Cell Voltage (V)	Real List (P)	N	Final battery pack per-cell resting voltage after all charges. The list is in order of P then S, so for a 2S/2P pack, values in the list are in the order “1P/1S ; 1P/2S ; 2P/1S ; 2P/2S”.
14	Charge Resistance (mΩ)	Real (P)	N	Final battery pack internal resistance after all charges.
15	Charge Cell Resistance (mΩ)	Real List (P)	N	Final battery pack per-cell resting voltage after all charges. The list is in order of P then S, so for a 2S/2P pack, values in the list are in the order “1P/1S ; 1P/2S ; 2P/1S ; 2P/2S”.
16	Ignore in Plots?	Boolean	N	Indicates whether or not the cycle should be ignored and not included in performance plots.
17	Cycle Notes	String	N	User notes on the cycle.

Continuation lines cannot appear in this section.

Cycles are expected to have a charge date and time that always follows the discharge date and time. When editing an existing cycle through the import file, the import file may not change either the charge or discharge date and time such that this expectation is not met. Doing so will cause RCLogbook to report an error.

The import file does not allow you to move an existing cycle from one battery to another.

It is recommended that you group all cycles for a given battery together and place the cycles in most- to least-recent order (by their discharge date and time). This organization will help the import process go more quickly.

RCLogbook handles cycle numbering automatically. Though there is a cycle number field in the import (column 3), it is there primarily for your reference. When editing cycles, the app will determine the appropriate cycle number and update the database as needed.

In addition, when adding a new cycle RCLogbook automatically updates the owning battery appropriately. For example, the total cycle count will be updated as will the charge state and last cycle time.

Events

The `Events` section is the most complex section in the import format primarily because the changes it describes can cover several different types of database records.

The `Events` section provides general information on the events defined in the database for various models along with any battery and cycle information associated with the event. Lines in the *database change* portion of an `Events` section corresponds to a single event along with its associated model, batteries, and cycles. The fields on each line include,

C	Field	Type	RN	Notes
1	Model Name	String	Y	Name of the model that the event is for. This name must exactly match one of the models in the database.
2	Event UID	Unique ID	Y	Unique ID for the event in the model, this event must already be associated with the model identified by the “Model Name” field when editing an existing event.
3	Event #	Integer (P, NZ)	N	The event number for the model. This field is shown for reference and is appropriately updated automatically. It may be left blank for new events.
4	Event Date (*F*)	Date	Y	Date of the event. “F” defines the expected format of the date based on the current locale. For example, “F” is “M/D/YY” in the US. This field must be specified if “Event Time” is not blank.
5	Event Time (*F*)	Time	Y	Time of the event. “F” defines the expected format of the time based on the current locale. For example, “F” is “H:MM A” in the US. This field must be specified if “Event Date” is not blank.
6	Location	String	N	Name of the location that the event takes place at. This name must exactly match one of the locations in the database.
7	Duration (M:SS)	Duration MS	Y	Duration of the event.
8	Fuel (*F*)	Real (P)	N	Total fuel consumed during the event.
9	Style	String	N	Event style for this event. The value should match the name of one of the styles defined in the `Styles` section of the import file.
10	Propeller	String	N	Propeller used in this event. The value should match the name of one of the propellers defined in the `Propellers` section of the import file.
11	Event Notes	String	N	User notes on the event.
12	Battery Name	String	N	Name of the battery that the event uses. This name should match exactly one of the batteries in the database.
13	Cycle UID	Unique ID	N	Unique ID for the cycle in the battery, this cycle must be associated with the battery identified by the “Battery Name” field.
14	Cycle #	Integer (P, NZ)	N	The cycle number for the battery. This field is shown for reference only and is appropriately updated automatically. It may be left blank for new batteries.
15	Discharge Date (*F*)	Date	N	Date of the last discharge. “F” defines the expected format of the date based on the current locale. For example, “F” is “M/D/YY” in the US. This field must be specified if “Discharge Time” is not blank.
16	Discharge Time (*F*)	Time	N	Time of the last discharge. “F” defines the expected format of the time based on the current locale. For example, “F” is “H:MM A” in the US. This field must be specified if “Discharge Date” is not blank.

C	Field	Type	RN	Notes
17	Discharge Duration (*F*)	Duration MS	N	Duration of discharges in minutes.
18	Discharge Voltage (V)	Real (P)	N	Final battery pack resting voltage after all discharges.
19	Discharge Cell Voltage (V)	Real List (P)	N	Final battery pack per-cell resting voltage after all discharges. The list is in order of P then S, so for a 2S/2P pack, values in the list are in the order “1P/1S ; 1P/2S ; 2P/1S ; 2P/2S”.
20	Charge Date (*F*)	Date	N	Date of the last charge. “F” defines the expected format of the date based on the current locale. For example, “F” is “M/D/YY” in the US. This field must be specified if “Charge Time” is not blank. The charge date and time should be later than the discharge date and time.
21	Charge Time (*F*)	Time	N	Time of the last charge. “F” defines the expected format of the time based on the current locale. For example, “F” is “H:MM A” in the US. This field must be specified if “Charge Date” is not blank. The charge date and time should be later than the discharge date and time.
22	Charge Amount (mA)	Integer (P)	N	Amount of energy put back into battery pack during the charge. This value must be greater-than zero.
23	Charge Voltage (V)	Real (P)	N	Final battery pack resting voltage after all charges. The list is in order of P then S, so for a 2S/2P pack, values in the list are in the order “1P/1S ; 1P/2S ; 2P/1S ; 2P/2S”.
24	Charge Cell Voltage (V)	Real List (P)	N	Final battery pack per-cell resting voltage after all charges.
25	Charge Resistance (mΩ)	Real (P)	N	Final battery pack internal resistance after all charges.
26	Charge Cell Resistance (mΩ)	Real List (P)	N	Final battery pack per-cell resting voltage after all charges. The list is in order of P then S, so for a 2S/2P pack, values in the list are in the order “1P/1S ; 1P/2S ; 2P/1S ; 2P/2S”.
27	Ignore in Plots?	Boolean	N	Indicates whether or not the cycle should be ignored and not included in performance plots.
28	Cycle Notes	String	N	User notes on the cycle.

Continuation lines may appear in this section to allow you to associate multiple cycles with a particular event (for example, to represent events where the model uses multiple batteries).

The table above is divided into three parts:

- Column 1 – Defines the model that the event is associated with. This column is always required.

- Column 2-11 – Defines the parameters of the event itself. These columns are always required.
- Column 12-28 – Defines the batteries and cycles associated with the event. These columns are optional for events that do not have associated batteries (for example, an event associated with a model that does not track batteries).

Multiple batteries are associated with an event by using a continuation lines with columns 12-28 setup for the subsequent batteries. To see how this works, we will look at a few examples.

Here is an `Events` section that contains two flights on a model that only uses a single battery that the pilot tracks (in this picture, columns D through J and N through W are hidden for brevity).

	A	B	C	K	L	M	N	X
1	[SECTION]	Events						
2	Model Name	Event UID	Event #	Event Notes	Battery Name	Cycle UID	Cycle #	Cycle Notes
3	Extra 300SHP	364	94	Flight 94	TP 2200/3S #1	351	10	A cycle
4	Extra 300SHP	360	93	Flight 93	TP 2200/3S #3	100	30	A cycle

Row 3 and 4 each correspond to two different flights where each flight adds a single cycle to a single battery.

Next, consider an `Events` section for a model that uses two batteries (again, we hide some of the columns for brevity).

	A	B	C	K	L	M	N	X
1	[SECTION]	Events						
2	Model Name	Event UID	Event #	Event Notes	Battery Name	Cycle UID	Cycle #	Cycle Notes
3	F-14 EDF	200	30	Flight 30	TP 2200/3S #1	351	10	Cycle A
4	+				TP 2200/3S #2	348	14	Cycle B
5	F-14 EDF	194	29	Flight 29	TP 2200/3S #4	345	30	Cycle A
6	+				TP 2200/3S #8	330	28	Cycle B

Row 3 and 4 describe the first flight while rows 5 and 6 describe the second flight. On the continuation lines, the model and event sections (columns B through I) of the line are blank. In this case, flight 30 used two batteries with the cycles lines 3 and 4 describe.

RCLogbook handles event and cycle numbering automatically. Though there are event and cycle number fields in the import (columns 3 and 14, respectively), they are there primarily for your reference. When editing events, the app will determine the appropriate event and cycle numbers and update the database as needed.

It is recommended that you group all events edits for a given model together and place the events in most- to least-recent order (by date and time). This organization will help the import process go more quickly.

Maintenance

The `Maintenance` section provides general information on the maintenance log entries defined in the database for various models. Lines in the *database change* portion of an `Maintenance` section corresponds to a single maintenance log event along with its associated model. The fields on each line include,

C	Field	Type	RN	Notes
1	Model Name	String	Y	Name of the model that the maintenance event is for. This name must exactly match one of the models in the database.
2	Log UID	Unique ID	Y	Unique ID for the maintenance event in the model, this event must already be associated with the model identified by the “Model Name” field when editing an existing event.
3	After Event	Integer (NZ)	Y	Event number (e.g., flight for aircraft) after which the maintenance occurred.
4	Date	Date	Y	Date of the maintenance event. “F” defines the expected format of the date based on the current locale. For example, “F” is “M/D/YY” in the US. This field must be specified if “Time” is not blank.
5	Time	Time	Y	Time of the last maintenance date. “F” defines the expected format of the time based on the current locale. For example, “F” is “H:MM A” in the US. This field must be specified if “Date” is not blank..
6	Action	String	Y	Action for maintenance log event.
7	Notes	String	N	User notes on the maintenance log event.

Continuation lines cannot appear in this section.

The import file does not allow you to move an existing log entry from one model to another.

About clevertangerine software

clevertangerine software is a very small software company made up of crafty *Citrus Reticulata* who are committed to developing cool and useful software for the iOS. Though they are just hack pilots, the fruit also love to fly and aspire to someday be able to 3D with the middle of the pack. RCLogbook was the result of tiring of scribbling on scraps of paper at the field to track battery usage.