Python Scripts

The Python API makes it easy to write custom scripts. It consists of two modules, `ALget.py` and `ALset.py`, that are used for reading input files and writing output files.

There are several ready-made functions available, but many of the scripts only use the following 6 functions:

- ALget(filename)
- ALget.value()
- ALset.setRecordField
- ALset.setMessage
- ALset.setExitAction
- ALset.save

Mastering the functions listed above will help you to write very powerful scripts. For more details on the ALget and ALset functions, visit the wiki articles entitled ALget Common Use and ALset Common Use. You can also find valuable information for writing scripts on the Script Structure and Basics page, which will help you include all required functions.

Current_State and Old_State

Records that are currently being edited are considered to be in `"current_state"`. There is always at least one record in "`current_state`", but only one of those current state records can be supported as the input for a given script: the record used as the input is referred to as the current record. You can obtain the current record name directly with an `ALget.currentStateRecordsNamesList()[0]` function.

In addition, the script may be able to use "`old_state`", which provides the old field values of the records. These are stored in an array that is ordered the same way as the current states, so the old values for the first current state record can be obtained with a similar function and list value, `ALget.oldRecordNamesList()[0]`. These old values are the values before the current edit of the records, assuming the script is being triggered by an edit to a record. These old values are provided so you can easily see, check, and validate user changes to fields and values.

recordName

Many of the functions take a parameter recordName as an input variable. If recordName is omitted from the input variable, the first of the current state records is used by default, and then is used to populate the output file.

Custom Distribution

If you would like to use your own Python distribution with scripts rather than the default distribution, you can define the location of your preferred python.exe using the Location of external Python directory global variable. For example, you might set it to something like `c:\Python34`.