# The CLI Module

#### Synopsis

This is a (budding) set of tools to help with writing command line and others scripts.

## Loading the tools.

This is done with the standard import and load or use statement like

cli := j\_load('cli') // OR j\_use('cli')

j\_use dumps it into the current module so you don't need a module name to access it.

## The to\_stem function

This will take a set of arguments (probably from the command line, but you can pass it whatever list you like) and it will process them into a usable stem. This means separating out the switches and flags and organizing the other arguments in order.

### Definitions

switch - an argument to the program prefixed with a marker. The immediate successor to it is its *value*. (e.g. **-format iso8601**)

flag - an argument which has no value, but is either present or not (e.g. -verbose).

Function arguments are

arg. - the list of command line arguments. If omitted this defaults to the contents of the arg() function

marker - a string that starts a flag or switch

flags. - list of flags

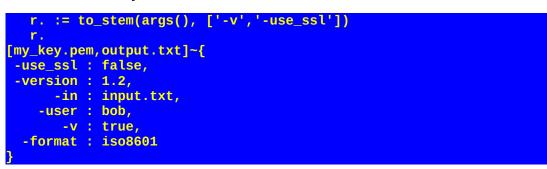
#### Examples

Let us say that we had a script that was invoked with the following:

do\_it.qdl my\_key.pem -user bob -version 1.2 -v -format iso8601 -in input.txt
output.txt

if you issue the args() command you will see the arguments are

args() [my\_key.pem, -user, bob, -version, 1.2, -v, -format, iso8601, -in, input.txt, output.txt] There is a flag (-v) and what if there is a flag like -use\_ssl that, when missing indicates the default is false? Here is what you could do



Note that access to these values is quite easy. To get the last argument issue

r.(-1)

To check if the user wants to use ssl:

if[r.'-use\_ssl'][…]

To check if the user passed in a specific switch:

if[has\_key('-format', r.)][…];

You can even specify special switches

to\_stem(args(), '-', ['--help','--version']

which tells the system to look for those flags first so you can test them for, e.g., printing help or version information.