Input & Output design of plugin

Jerry Chae & Irene Cho

ARGOS LABS

Contents

ARGOS LABS

Module Context (Main Class of plugin)

- Inherited from argparse.ArgumentParser module
- keyword parameters
  - owner : define owner
  - group : which group, usually second level of package
  - version : (just info, real python module version is from setup.yaml file)
  - platform : supported OS platform ('windows', 'darwin' for mac, 'linux')
  - output_type : reserved
  - display_name : showed label in STU

```
with ModuleContext(
    owner='ARGOS-LABS-DEMO',
    group='Rooman',
    version='1.0',
    platform=['windows', 'darwin', 'linux'],
    output_type='text',
    display_name='Whatsapp Rooman',
    icon_path=get_icon_path(__file__),
    description='send message whatsapp friends',
) as mcxt:
    # ##################################### for app dependent parameters
    mcxt.add_argument('msg',
                      display_name='message',
                      help='type message to send')
    mcxt.add_argument('friends', nargs='+',
                      display_name='Friends Name',
                      help='type your friends name')
    # ##################################### for app dependent options
    mcxt.add_argument('--driver',
                      choices=['Chrome', 'Firefox', 'MSIE'],
                      display_name='Chromedriver path',
                      help='type Chromedriver path')
    mcxt.add_argument('--filepath',
                      display_name='send file',
                      help='give path of img/video')
    argspec = mcxt.parse_args(args)
    return whatsapp(mcxt, argspec)
```

ARGOS LABS

Input Design

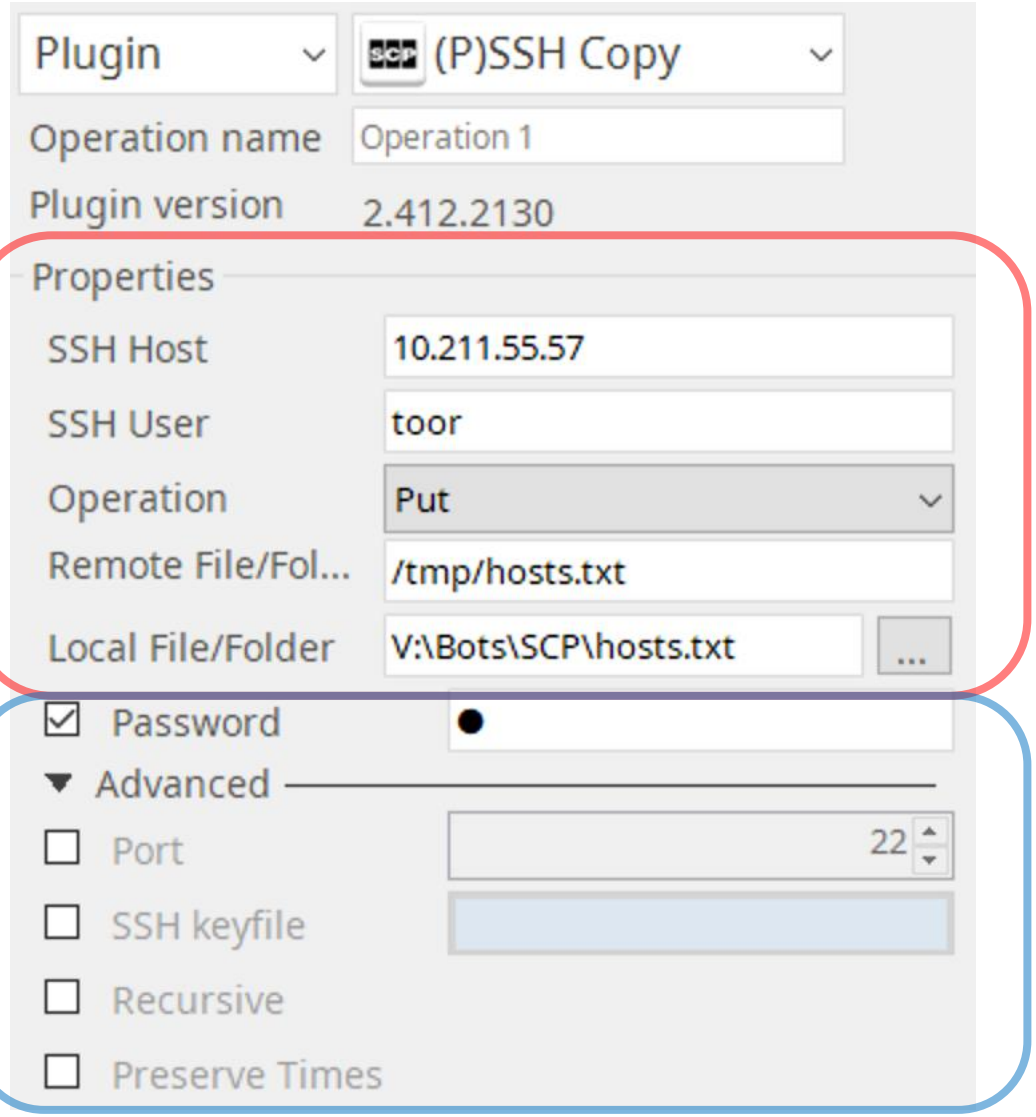
- All user input is added by add_argument() method
- Mandatory argument is parameter (not starting with '-')
- Optional arguments are options (starting with '-')

```
with ModuleContext(
    owner='ARGOS-LABS-DEMO',
    group='Rooman',
    version='1.0',
    platform=['windows', 'darwin', 'linux'],
    output_type='text',
    display_name='Whatsapp Rooman',
    icon_path=get_icon_path(__file__),
    description='send message whatsapp friends',
) as mcxt:
    # ##################################### for app dependent parameters
    mcxt.add_argument('msg',
                      display_name='message',
                      help='type message to send')
    mcxt.add_argument('friends', nargs='+',
                      display_name='Friends Name',
                      help='type your friends name')
    # ##################################### for app dependent options
    mcxt.add_argument('--driver',
                      choices=['Chrome', 'Firefox', 'MSIE'],
                      display_name='Chromedriver path',
                      help='type Chromedriver path')
    mcxt.add_argument('--filepath',
                      display_name='send file',
                      help='give path of img/video')
    argspec = mcxt.parse_args(args)
    return whatsapp(mcxt, argspec)
```

ARGOS LABS

Input Design and STU property (example 1)

```
# ##################################### for app dependent options
mcxt.add_argument('--port',
                  display_name='Port',
                  type=int, default=22,
                  help='port number, default is [[22]]')
mcxt.add_argument('--password', show_default=True,
                  display_name='Password',
                  input_method='password',
                  help='user password')
mcxt.add_argument('--key-filename',
                  display_name='SSH keyfile',
                  help='SSH key filename instead of password')
mcxt.add_argument('--recursive', action='store_true',
                  display_name='Recursive',
                  help='If this flag is set get or put recursively '
                       'including all files and sub folders')
mcxt.add_argument('--preserve-times', action='store_true',
                  display_name='Preserve Times',
                  help='If this flag is set preserve file time attribute')
# ##################################### for app dependent parameters
mcxt.add_argument('host',
                  display_name='SSH Host',
                  help='hostname or ip address to connect')
mcxt.add_argument('user',
                  display_name='SSH User',
                  help='user id to connect')
mcxt.add_argument('op',
                  display_name='Operation',
                  choices=['Get', 'Put'],
                  help='One of Get or Put from remote file system')
mcxt.add_argument('remote',
                  display_name='Remote File/Folder',
                  help='Remote file or folder')
mcxt.add_argument('local',
                  display_name='Local File/Folder',
                  input_method='fileread',
                  help='Remote file or folder')
```

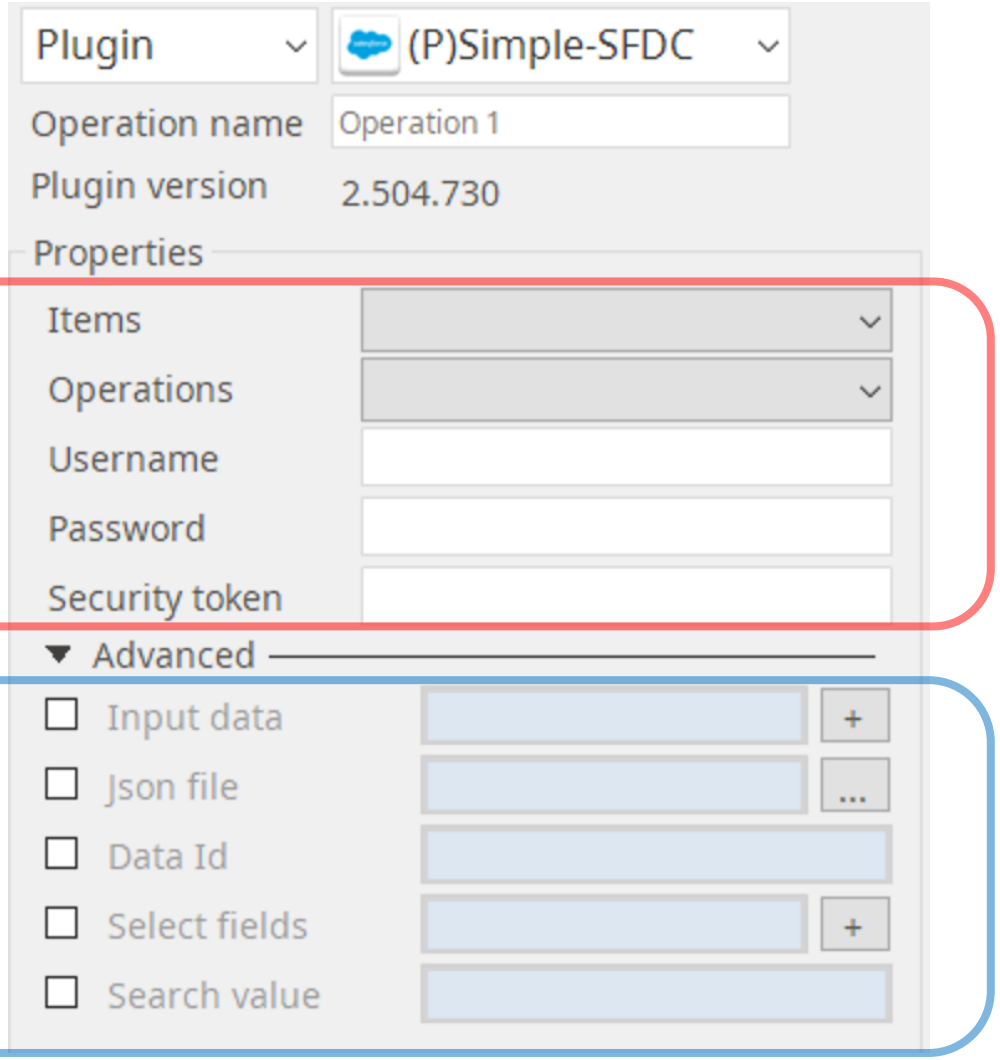


ARGOS LABS

Input Design and STU property (example 2)

```
# ######################################## for app dependent options
mcxt.add_argument('--data', action='append', display_name='Input data',
                  help='data to create or insert')
# ------------------------------------------------------------------------
mcxt.add_argument('--jsonfile', display_name='Json file',
                  input_method='fileread',
                  help='Json inputfile')
# ------------------------------------------------------------------------
mcxt.add_argument('--id', display_name='Data Id',
                  help='id for data')
# ------------------------------------------------------------------------
mcxt.add_argument('--fieldnames', action='append',
                  display_name='Select fields',
                  help='select query fieldnames')
# ------------------------------------------------------------------------
mcxt.add_argument('--value', display_name='Search value',
                  help='search specific value and return Id')
```

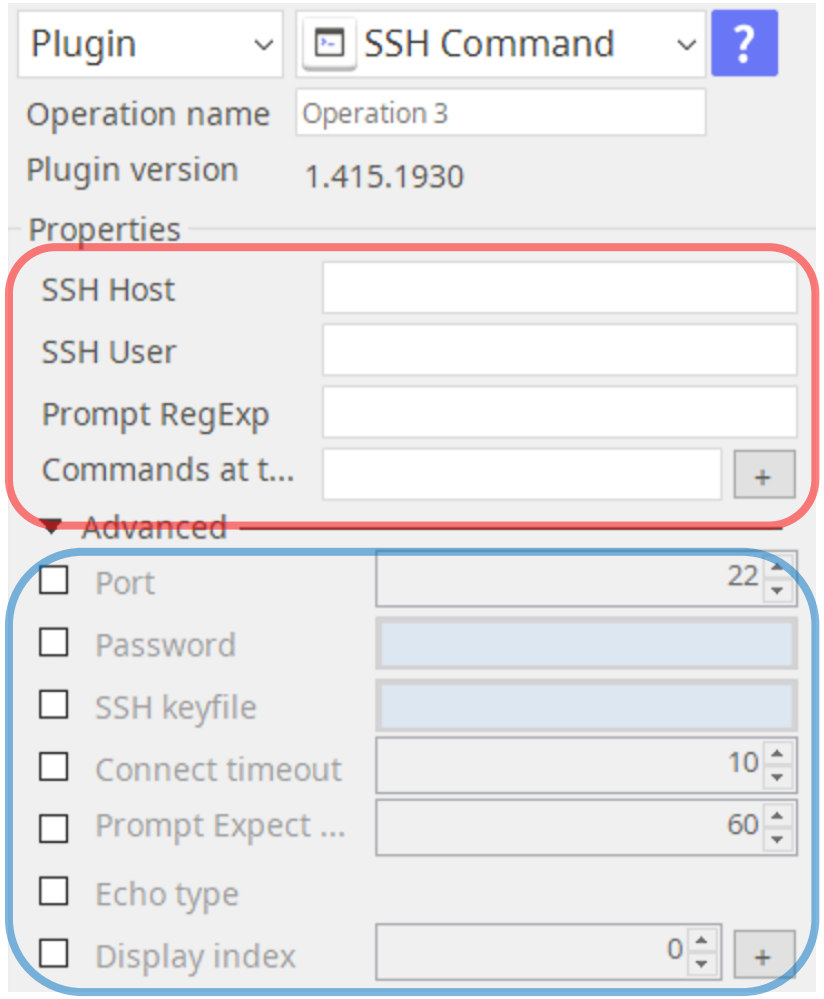
```
# ######################################## for app dependent parameters
mcxt.add_argument('op2',
                  display_name='Items', choices=SimpleAPI.OP_TYPE2,
                  help='Simple salesforce class type')
# ------------------------------------------------------------------------
mcxt.add_argument('op',
                  display_name='Operations', choices=SimpleAPI.OP_TYPE,
                  help='Simple salesforce type of operation')
# ------------------------------------------------------------------------
mcxt.add_argument('username', display_name='Username', help='username')
# ------------------------------------------------------------------------
mcxt.add_argument('password', display_name='Password',
                  input_method='password', help='password')
# ------------------------------------------------------------------------
mcxt.add_argument('security_token', display_name='Security token',
                  input_method='password', help='security_token')
```



ARGOS LABS

Input Design and STU property (example 3)

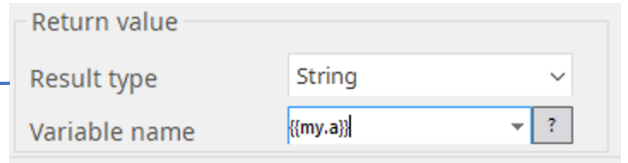
```
# ###################################### for app dependent options
mcxt.add_argument('--port',
                  display_name='Port',
                  type=int, default=22,
                  help='port number, default is [[22]]')
mcxt.add_argument('--password',
                  display_name='Password',
                  input_method='password',
                  help='user password')
mcxt.add_argument('--key-filename',
                  display_name='SSH keyfile',
                  help='SSH key filename')
mcxt.add_argument('--connect-timeout',
                  display_name='Connect timeout',
                  type=int, default=10,
                  help='connection timeout, default is [[10]]')
mcxt.add_argument('--expect-timeout',
                  display_name='Prompt Expect timeout',
                  type=int, default=60,
                  help='expectation string waiting timeout, default is [[60]]')
mcxt.add_argument('--echo-display', action='store_true',
                  display_name='Echo On',
                  help='If this flag is set echo all command output')
mcxt.add_argument('--display-index', nargs='+', type=int,
                  display_name='Display index',
                  help='1-based index to display the output from command list')
# ###################################### for app dependent parameters
mcxt.add_argument('host',
                  display_name='SSH Host',
                  help='hostname or ip address to connect')
mcxt.add_argument('user',
                  display_name='SSH User',
                  help='user id to connect')
mcxt.add_argument('prompt',
                  display_name='Prompt RegExp',
                  help='prompt to expected')
mcxt.add_argument('command', nargs='+',
                  display_name='Commands at terminal',
                  help='one or more command to execute')
```

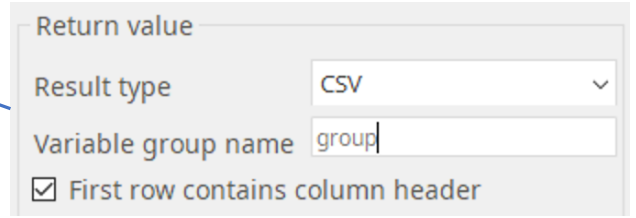


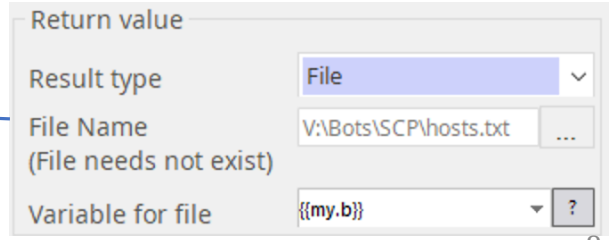
ARGOS LABS

Output design

- Output for plugin is the output stream to stdout
  - print('abc')
  - sys.stdout.write('d,e,f')
- If plugin has error or exception output stream to stderr
  - sys.stderr.write('Error:...')
- PAM interprets the output stream as one of these
  - String into variable
  - CSV with variable group
  - File path to save the result and variable for file







ARGOS LABS

Plugin return

- Return 0 for success
- Return non 0 for failure

```
############################################################
@func_log
def xml_extract(mcxt, argspec):
    """
    plugin job function
    :param mcxt: module context
    :param argspec: argument spec
    :return: True
    """
    mcxt.logger.info('>>>starting...')
    try:
        r = get_xpath_data(argspec.xml, argspec.xpath,
                           include_header=argspec.header)
        return 0 if r > 0 else 2
    except Exception as err:
        msg = str(err)
        mcxt.logger.error(msg)
        sys.stderr.write('%s%s' % (msg, os.linesep))
        return 1
    finally:
        sys.stdout.flush()
        mcxt.logger.info('>>>end...')
```

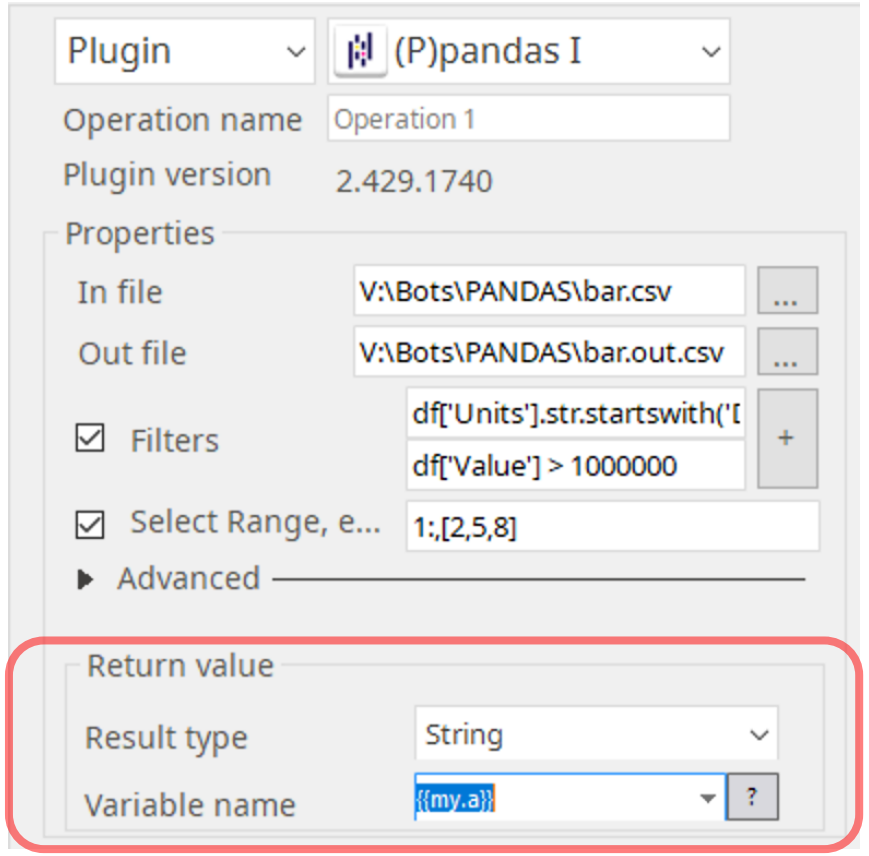
ARGOS LABS

PAM & plugin

- When PAM executes plugins in a different way from the native operations
- PAM executes plugins as a subprocess with user properties and advanced properties (same concept as CLI)
- PAM checks if the result of plugin is 0 or not
  - if 0 then stdout output stream will be processed one of String, CSV, File
  - if not 0 then stop the bot and stderr error stream will be showed up as error result (Error message must be kind plain English)

Output Design (example 1)

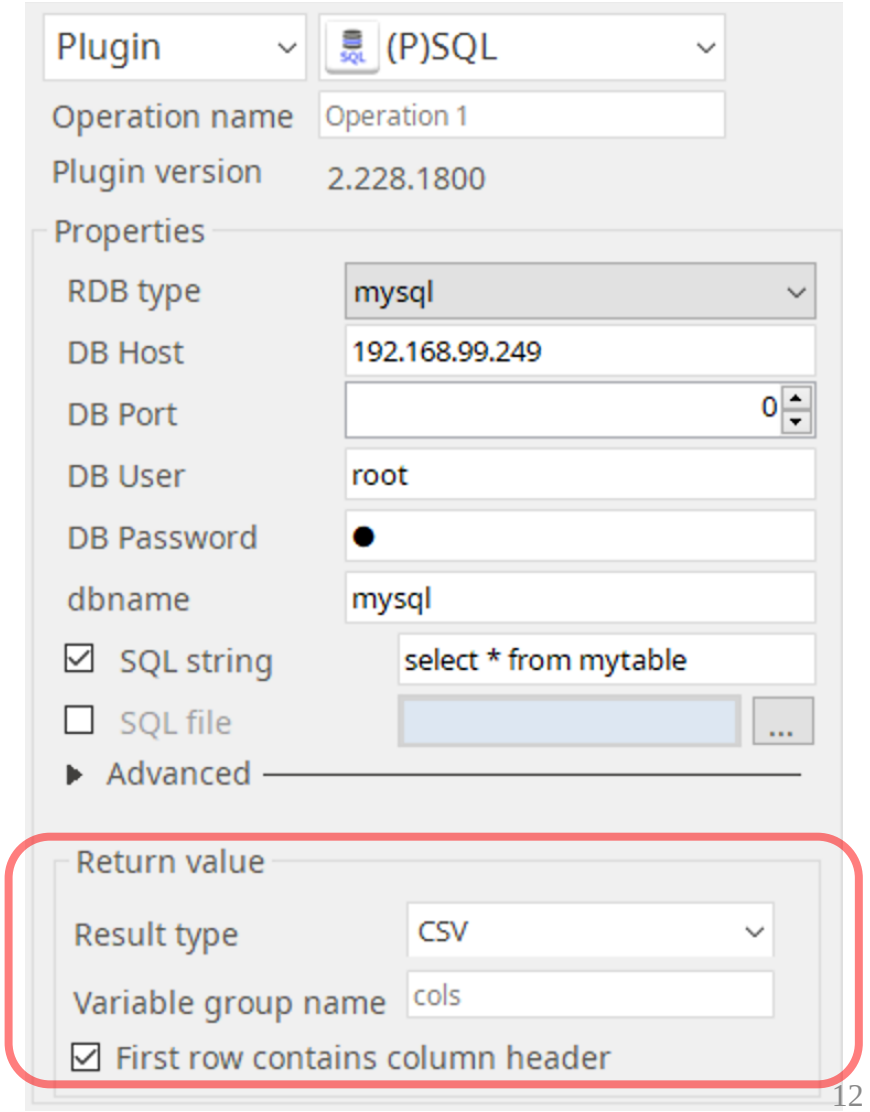
```
print(os.path.abspath(argspec.out_file), end='')
```



ARGOS LABS

Output Design (example 2)

```
# ============================================================
def sql_select(self, sql):
    self.logger.debug('MySQL.sql_select: sql=<%s>' % sql)
    with self.conn.cursor() as cursor:
        cursor.execute(sql)
        # num_fields = len(cursor.description)
        field_names = [x[0] for x in cursor.description]
        # noinspection PyUnusedLocal
        field_types = [x[1] for x in cursor.description]
        print('%s' % ','.join(field_names))
        cwr = csv.writer(sys.stdout, lineterminator='\n')
        rows = cursor.fetchall()
        for row in rows:
            cwr.writerow(row)
```



ARGOS LABS

Argparse

- What is Argparse?
  - Argparse is a python module that gets command line arguments into one's program
  - Get the required spec using add_argument method

- The add_argument() method
  - Define how a single command-line argument should be parsed
  - Distinguished by an optional argument, like –f or –foo, or a positional argument, also called dependent options, like a list of filenames
  - Positional arguments are shown as default parameters in STU and optional argument are located under 'Advanced' section

Parameters of add_argument

Parameters	Type	Usage	Example
name or flags	str	A name of a list of option strings	foo (positional argument) --foo (optional argument)
action	Store	Store the argument's value (default action)	parser.add_argument('--foo')
	Append	Store a list and appends each argument value to a list	parser.add_argument('--foo', action='append')
	Count	Count the number of times a keyword argument occurs	parser.add_argument('--verbose', '-v', action='count', default=0)
	Store_true Store_false	Special case of 'store_const' used for storing the values True or False	parser.add_argument('--foo', action='store_true')

Parameters of add_argument

Parameters	Type	Usage	Example
nargs	Default	Relate a different number of command-line arguments to a single action	
	N (an integer)	Gather the N arguments into a list	parser.add_argument(‘--foo’, nargs=2)
	?	• Consume one argument and produce a single item • Print value from ‘default’ if there is no argument • In optional argument, the value from ‘const’ will be printed if there is no argument	parser.add_argument(‘--foo’, nargs=‘?’, const=‘c’, default=‘d’)
	*	Gather all command-line arguments present into a list (recommend one positional argument)	parser.add_argument(‘-bar’, nargs=‘*')
	+	Same as ‘*’ but it will give an error message if there wasn’t at least one argument	parser.add_argument(‘foo’, nargs=‘+')

Parameters of add_argument

Parameters	Type	Usage	Example
const	value	Hold constant values that are not read from the command line, but required for the various actions	parser.add_argument(‘--foo’, const=2)
default	value	Specify a value if the command-line argument is not present	parser.add_argument(‘--foo’, default=2)
type		Allow any necessary type-checking and type conversions to be performed	
	Str	String (default)	parser.add_argument(‘--foo’, type=str)
	Float	Float	
	Int	Int	

ARGOS LABS

Parameters of add_argument

Parameters	Type	Usage	Example
choices	list of str	Pass a container object and allows the arguments to be selected from a restricted set of values	parser.add_argument(‘foo’, choices=[‘rock’, ‘peer’, ‘scissors’])
help	str	A string which contains a brief description of the argument	parser.add_argument(‘--foo’, help=‘foo the bars before frobbling’)
dest	str	Determine the name of the attribute added by ArgumentParser actions	>parser.add_argument(‘bar’) >parser.add_argument(‘bar’)

Parameters of add_argument

Parameters	Type	Usage	Example
input_method	password	The way for the user to securely enter a password in STU	parser.add_argument(‘foo’, input_method=password)
	fileread	Select the file which stored on the user’s computer in STU	parser.add_argument(‘foo’, input_method=fileread)
display_name	str	Display the name of a parameter in STU	parser.add_argument(‘foo’, display_name=‘Foo’)
show_default	True/False	If show_default=True, the parameter will always be shown as a default value	parser.add_argument(‘--foo’, show_default=True)

Parameters of add_argument

Parameters	Type	Usage	Example
min_value or greater_eq	value	Check if value is greater than or equal otherwise error	parser.add_argument(‘foo’ type=int, min_value=1)
max_value or less_eq	value	Check if value is less than or equal otherwise error	parser.add_argument(‘foo’, max_value=100)
min_value_ni or greater	value	Check if value is greater than otherwise error	parser.add_argument(‘foo’, greater=0)
max_value_ni or less	value	Check if value is less than otherwise error	parser.add_argument(‘foo’, less=‘zz’)

ARGOS LABS

Parameters of add_argument

Parameters	Type	Usage	Example
equal	value	Check if value is equal otherwise error	parser.add_argument(‘foo’,equal=‘abc’)
not_equal	value	Check if value is not equal otherwise error	parser.add_argument(‘foo’,not_equal=55)
re_match	str	Check if str is matching with regular expression (example: ‘123.4.56.1’)	parser.add_argument(‘foo’, re_match=r'^\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}$')

ARGOS LABS

Happy Automation!

ARGOS LABS