
binary_c-python

Jeff Andrews, Robert Izzard, David Hendriks

Nov 11, 2019

CONTENTS:

1	binaryc_python_utils	1
1.1	custom_logging_functions module	1
1.2	functions module	2
1.3	stellar_types module	2
2	Indices and tables	3
	Python Module Index	5
	Index	7

BINARYC_PYTHON_UTILS

1.1 custom_logging_functions module

`custom_logging_functions.autogen_C_logging_code(logging_dict)`

Function that autogenerates PRINTF statements for binaryc. input is a dictionary where the key is the header of that logging line and items which are lists of parameters that will be put in that logging line

Example:

```
{ 'MY_STELLAR_DATA':  
  [  
    'model.time',  
    'star[0].mass',  
    'model.probability',  
    'model.dt'  
  ]  
}
```

`custom_logging_functions.binary_c_log_code(code)`

Function to construct the code to construct the custom logging function

`custom_logging_functions.binary_c_write_log_code(code, filename)`

Function to write the generated logging code to a file

`custom_logging_functions.compile_shared_lib(code, sourcefile_name, outfile_name)`

Function to write the custom logging code to a file and then compile it.

`custom_logging_functions.create_and_load_logging_function(custom_logging_code)`

Function to automatically compile the shared library with the given custom logging code and load it with ctypes

returns: memory adress of the custom logging function in a int type.

`custom_logging_functions.from_binary_c_config(config_file, flag)`

Function to run the binaryc_config command with flags

`custom_logging_functions.return_compilation_dict()`

Function to build the compile command for the shared library

inspired by binaryc_inline_config command in perl

TODO: this function still has some cleaning up to do wrt default values for the compile command # <https://developers.redhat.com/blog/2018/03/21/compiler-and-linker-flags-gcc/>

returns:

- string containing the command to build the shared library

`custom_logging_functions.temp_custom_logging_dir()`

Function to return the path the custom logging library shared object and script will be written to.

Makes use of `os.makedirs exist_ok` which requires python 3.2+

1.2 functions module

`functions.create_arg_string(arg_dict)`

Function that creates the arg string

`functions.get_arg_keys()`

Function that return the list of possible keys to give in the arg string

`functions.get_defaults()`

Function that calls the binaryc get args function and cast it into a dictionary All the values are strings

`functions.parse_output(output, selected_header)`

Function that parses output of binaryc when it is construction like this: DAVID_SINGLE_ANALYSIS t=0 mass=20

You can give a 'selected_header' to catch any line that starts with that. Then the values will be put into a dictionary. TODO: Think about exporting to numpy array or pandas

`functions.run_system(**kwargs)`

Wrapper to run a system with settings

This function determines which underlying python-c api function will be called based upon the arguments that are passed via kwargs.

- if `custom_logging_code` or `custom_logging_dict` is included in the kwargs then it will
- if

`functions.run_system_with_log(**kwargs)`

Wrapper to run a system with settings AND logs the files to a designated place defined by the `log_filename` parameter.

1.3 stellar_types module

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`

PYTHON MODULE INDEX

C

`custom_logging_functions`, 1

f

`functions`, 2

S

`stellar_types`, 2

INDEX

A

`autogen_C_logging_code()` (*in module custom_logging_functions*), 1

B

`binary_c_log_code()` (*in module custom_logging_functions*), 1

`binary_c_write_log_code()` (*in module custom_logging_functions*), 1

C

`compile_shared_lib()` (*in module custom_logging_functions*), 1

`create_and_load_logging_function()` (*in module custom_logging_functions*), 1

`create_arg_string()` (*in module functions*), 2

`custom_logging_functions` (*module*), 1

F

`from_binary_c_config()` (*in module custom_logging_functions*), 1

`functions` (*module*), 2

G

`get_arg_keys()` (*in module functions*), 2

`get_defaults()` (*in module functions*), 2

P

`parse_output()` (*in module functions*), 2

R

`return_compilation_dict()` (*in module custom_logging_functions*), 1

`run_system()` (*in module functions*), 2

`run_system_with_log()` (*in module functions*), 2

S

`stellar_types` (*module*), 2

T

`temp_custom_logging_dir()` (*in module custom_logging_functions*), 1