Jonga Documentation

Release 0.0.4

Brendt Wohlberg

Nov 12, 2018

Contents

CHAPTER 1

Overview

Jonga is a Python package that generates a directed graph representing function calls within a block of Python code, intended for inclusion in Sphinx package documentation. There are a number of alternative packages with similar goals, including

- pycallgraph
- pyan
- snakefood

but none of them is entirely suitable for generating function/method call vizualizations for inclusion within package documentation. In particular, none of these other packages correctly identifies method classes within a hierarchy of derived classes.

Jonga is used to generate call graphs to help document the relatively complex class structure in the SPORCO package, as illustrated in this example (note that the method names are clickable, linking to the corresponding entries in the documentation).

1.1 Usage Examples

Scripts illustrating usage of the package can be found in the `examples` directory of the source distribution. These examples can be run from the root directory of the package by, for example

```
python3 examples/example1.py
```

To run these scripts prior to installing the package it will be necessary to first set the `PYTHONPATH` environment variable to include the root directory of the package. For example, in a `bash` shell

```
export PYTHONPATH=$PYTHONPATH:`pwd`
```

from the root directory of the package.

Jupyter Notebook versions of the example scripts are also available in the same directory. The notebooks can also be viewed online via nbviewer, or run interactively at binder.

1.2 Contact

Please submit bug reports, comments, etc. to brendt@ieee.org.

CHAPTER 2

Installation

The simplest way to install the most recent release of Jonga from PyPI is

```
pip install jonga
```

Jonga can also be installed from source, either from the development version from GitHub, or from a release source package downloaded from PyPI.

To install the development version from GitHub do

```
git clone git://github.com/bwohlberg/jonga.git
```

followed by

```
cd jonga
python setup.py build
python setup.py install
```

The install command will usually have to be performed with root permissions, e.g. on Ubuntu Linux

```
sudo python setup.py install
```

The procedure for installing from a source package downloaded from PyPI is similar.

Note that under Ubuntu Linux, in the commands listed above, `python` and `pip` should be replaced with `python3` and `pip3` respectively.

2.1 Requirements

The primary requirement is Python 3.3 or greater (this packages is *not* compatible with Python 2), imposed by the use of the `__qualname__` function attribute and inspect.getclosurevars. The `__qualname__` attribute could be replaced in earlier versions of Python by qualname, but there is no obvious replacement for inspect.getclosurevars, which was introduced in Python 3.3.

The other major requirement is pygraphviz. Under Ubuntu Linux 18.04, this requirement can be installed by the command

```
sudo apt-get install python3-pygraphviz
```

2.1.1 Optional

Package matplotlib is required to run the included Jupyter Notebook examples.

Packages pytest and pytest-runner are required to run the tests (`python setup.py test` or `python3 setup.py test`, depending on the operating system).

Packages sphinx, sphinx-bootstrap-theme, and numpydoc are required to build the documentation (`python setup.py build_sphinx` or `python3 setup.py build_sphinx`, depending on the operating system).

CHAPTER 3

Modules

3.1 jonga module

Call tracing for class method inheritance documentation

current_function(*frame*)
Get reference to currently running function from inspect/trace stack frame.

Parameters

frame [stack frame] Stack frame obtained via trace or inspect

Returns

fnc [function reference] Currently running function

function_qname(*fnc*)
Get qualified name of a function (the fully qualified name without the module prefix)

Parameters

fnc [function reference]

A function reference

Returns

fqn [string]

The qualified name the function

function_fqname(*fnc*)
Get fully qualified name of a function

Parameters

fnc [function reference] A function reference

Returns

fqn [string] The fully qualified name the function

current_module_name(*frame*)
Get name of module of currently running function from inspect/trace stack frame.

Parameters

frame [stack frame] Stack frame obtained via trace or inspect

Returns

modname [string] Currently running function module name

class CallTracer(*srcmodflt=None*, *dstmodflt=None*, *srcqnmflt=None*, *dstqnmflt=None*, *fnmsub=None*, *grpflt=None*, *lnksub=None*)
Bases: object

Manage construction of a call graph for methods within a class hierarchy

__init__(*srcmodflt=None*, *dstmodflt=None*, *srcqnmflt=None*, *dstqnmflt=None*, *fnmsub=None*, *grpflt=None*, *lnksub=None*)
Initialise a CallTracer object.

Parameters

srcmodflt [None or regex string, optional (default None)] A regex for call filtering based on calling function module. A function call is only recorded if the regex matches the name of the calling function module. If None, filtering is disabled.

dstmodflt [None or regex string, optional (default None)] A regex for call filtering based on caller function. A function call is only recorded if the regex matches the name of the called function module. If None, filtering is disabled.

srcqnmflt [None or regex string, optional (default None)] A regex for call filtering based on calling function qname. A function call is only recorded if the regex matches the name of the calling function. If None, filtering is disabled.

dstqnmflt [None or regex string, optional (default None)] A regex for call filtering based on caller function qname. A function call is only recorded if the regex matches the name of the called function. If None, filtering is disabled.

fnmsub [None or tuple of two regex strings, optional (default None)] A tuple of match and replace regex strings for computing graph node names from function qnames. If None, node names are function qnames.

grpflt [None or regex string, optional (default None)] A regex string for extracting part of the function fqname as a group name. If None, groups are not defined.

lnksub [None or tuple of two regex strings, optional (default None)] A tuple of match and replace regex strings for computing node href attributes from node names. If None, href attributes are not defined.

reset()
Reset record of called functions, deleting all accumulated call information

start()
Start tracing

stop()
Stop tracing

graph(*fnm=None*, *size=None*, *fntsz=None*, *fntfm=None*, *clrgen=None*, *rmsz=False*, *prog='dot'*)
Construct call graph

Parameters

fnm [None or string, optional (default None)] Filename of graph file to be written. File type is determined by the file extentions (e.g. dot for 'graph.dot' and SVG for 'graph.svg'). If None, a file is not written.

size [string or None, optional (default None)] Graph image size specification string.

fntsz [int or None, optional (default None)] Font size for text.

fntnm [string or None, optional (default None)] Font family specification string.

clrgen [function or None, optional (default None)] Function to call to generate the group colours. This function should take an integer specifying the number of groups as an argument and return a list of graphviz-compatible colour specification strings.

rmsz [bool, optional (default False)] If True, remove the width and height specifications from an SVG format output file so that the size scales properly when viewed in a web browser

prog [string, optional (default 'dot')] Name of graphviz layout program to use.

Returns

pgr [pygraphviz.AGraph] Call graph of traced function calls

class ContextCallTracer(*ct*, *pth=None*, ***kwargs*)

Bases: `object`

A wrapper class for *CallTracer* that enables its use as a context manager. At the end of the context a call graph image is generated and written to a path specified in the initialiser.

__init__(*ct*, *pth=None*, ***kwargs*)

Initialise context manager.

Parameters

ct [class:*CallTracer* object] Specify the call tracer object to be used as a context manager.

pth [string or None, optional (default None)] Specify the path of the graph image file to be written by *CallTracer.graph()* at the end of the context. A graph is not generated if it is `None`.

****kwargs** Keyword arguments for *CallTracer.graph()*

calltracer()

Return the call tracer object associated with this ContextCallTracer instance.

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

j

Index

Symbols

C

F

G

J

R

S