Abstract syntax tree

All Versions

Examples

Analyze functions in a python script

This analyzes a python script and, for each defined function, reports the line number where the function began, where the signature ends, where the docstring ends, and where the function definition ends.

```
#!/usr/local/bin/python3

import ast
import sys

""" The data we collect.  Each key is a function name; each value is a dict
with keys: firstline, sigend, docend, and lastline and values of line numbers
where that happens. """
functions = {}

def process(functions):
    """ Handle the function data stored in functions. """
    for funcname,data in functions.items():
        print("function:",funcname)
        print("\tstarts at line:",data['firstline'])
        print("\tsignature ends at line:",data['sigend'])
        if ( data['sigend'] < data['docend'] ):
            print("\tdocstring ends at line:",data['docend'])
        else:
            print("\tno docstring")
        print("\tfunction ends at line:",data['lastline'])
        print()

class FuncLister(ast.NodeVisitor):
    def visit_FunctionDef(self, node):
        """ Recursively visit all functions, determining where each function
        starts, where its signature ends, where the docstring ends, and where
        the function ends. """
        functions[node.name] = {'firstline':node.lineno}
        sigend = max(node.lineno,lastline(node.args))
        functions[node.name]['sigend'] = sigend
        docstring = ast.get_docstring(node)
        docstringlength = len(docstring.split('\n')) if docstring else -1
        functions[node.name]['docend'] = sigend+docstringlength
        functions[node.name]['lastline'] = lastline(node)
        self.generic_visit(node)
```

Syntax

Parameters

Remarks