

UMLgen Experiments

This is the result of some experimental work (or hacks) about generating UML-like class and object diagrams from Python code. The example below is about all that can be done right now with UMLgen, not much, but also not bad for a one day hack. The package does neither exist (being just a bunch of functions right now), nor is it documented. Just that much more information: it is using Piddle and PDFgen! If you're interested, let me know and/or watch comp.lang.python.announce!

Dinu C. Gherman, gherman@europemail.com

```
class Base1:
    pass

class Base2:
    classVar1 = "Some class variable"
    def foofoo(self):
        "A useless method."
        pass

class MyClass (Base1, Base2):
    "Just a hacked class."
    classVar2 = "Another class variable"
    def __init__(self, darg1=None):
        "__init__ doc"
        self.aString = "spam"
        self.eggs = Base2()
        return
    def foo(self, parg, darg1="barVal1",
            darg2="barVal2", **dict):
        "foo doc"
        return
    def bar(self, *dict):
        "bar doc"
        return
    def foobar(self, *arr, **dict):
        "foobar doc"
        return

c = MyClass()
c.aNewVar = 1

# Diagrams are then generated with code
# similar to this:
# drawUMLClassDiagram(Base1, canvas)
# drawUMLClassDiagram(Base2, canvas)
# drawUMLClassDiagram(MyClass, canvas)
#
# d = UMLObjectDiagram(c)
# d.draw(x, y, canvas)
```

Base1
__module__ = '__main__'
__doc__ = None

Base2
__doc__ = None
__module__ = '__main__'
classVar1 = 'Some class variable'
foofoo(self)

MyClass (Base1, Base2)
__doc__ = 'Just a hacked class.'
classVar2 = 'Another class variable'
__module__ = '__main__'
classVar1 = 'Some class variable'
__init__(self, darg1=None)
foo(self, parg, darg1, darg2='barVal2', **dict)
foobar(self, *arr, **dict)
bar(self, *dict)
foofoo(self)

? : MyClass (Base1, Base2)
__doc__ = 'Just a hacked class.'
classVar2 = 'Another class variable'
__module__ = '__main__'
aNewVar = 1
eggs = <__main__.Base2 instance at 7a9ca0>
aString = 'spam'
classVar1 = 'Some class variable'
__init__(self, darg1=None)
foo(self, parg, darg1, darg2='barVal2', **dict)
foobar(self, *arr, **dict)
bar(self, *dict)
foofoo(self)