```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
"""This module's docstring summary line.

This is a multi-line docstring. Paragraphs are separated with blank lines.
Lines conform to 79-column limit.

Module and packages names should be short, lower_case_with_underscores.

See http://www.python.org/dev/peps/pep-0008/ for more PEP-8 details and
http://wwd.ca/blog/2009/07/09/pep-8-cheatsheet/ for an up-to-date version
of this cheatsheet.
"""
import os
import sys

import some_third_party_lib
import some_other_third_party_lib

import your_local_stuff
import more_local_stuff
import dont_import_two, modules_in_one_line     # IMPORTANT!

_a_global_var = 2   # so it won't get imported by 'from foo import *'
_b_global_var = 3

A_CONSTANT = 'ugh.'

# 2 empty lines between top-level funcs + classes
def some_function():
    pass


class FooBar(object):
    """Write docstrings for ALL public classes, funcs and methods.

    Class and exception names are CapWords.
    """

    a = 2
    b = 4
    _internal_variable = 3
    class_ = 'foo'      # trailing underscore to avoid conflict with builtin

    # this will trigger name mangling to further discourage use from outside
    # this is also very useful if you intend your class to be subclassed, and
    # the children might also use the same var name for something else; e.g.
    # for simple variables like 'a' above. Name mangling will ensure that
    # *your* a and the children's a will not collide.
    __internal_var = 4

    # NEVER use double leading and trailing underscores for your own names
    __nooooooodontdoit__ = 0

    # don't call anything:
    l = 1
    O = 2
    I = 3

    # some examples of how to wrap code to conform to 79-columns limit:
    def __init__(self, width, height,
                 color='black', emphasis=None, highlight=0):
        if width == 0 and height == 0 and \
           color == 'red' and emphasis == 'strong' or \
           highlight > 100:
            raise ValueError("sorry, you lose")
        if width == 0 and height == 0 and (color == 'red' or
                                           emphasis is None):
            raise ValueError("I don't think so -- values are %s, %s" %
                             (width, height))
        Blob.__init__(self, width, height,
                      color, emphasis, highlight)

        # empty lines within method to enhance readability; no set rule
        short_foo_dict = {'loooooooooooooooooooong_element_name': 'cat',
                          'other_element': 'dog'}

        long_foo_dict_with_many_elements = {
            'foo': 'cat',
            'bar': 'dog'
        }

    # 1 empty line between in-class def'ns
    def foo_method(self, x, y=None):
        """Method and function names are lower_case_with_underscores.

        Always use self as first arg.
        """
        if x == 4:          # x is blue <== USEFUL 1-liner comment
            x, y = y, x     # inverse x and y <== USELESS COMMENT
        dict['key'] = dict[index] = {x: 2, 'cat': 'not a dog'}
        c = (a + b) * (a - b)

    @classmethod
    def bar(cls):
        """Use cls!"""
        pass


# a 79-char ruler:
#234567891123456789212345678931234567894123456789512345678961234567897123456789
```