```
#!/usr/bin/env python
# -*- coding: latin-1 -*-
# ****************************************************************************
# * Software: FPDF for python                                                *
# * Version:  1.7.1                                                          *
# * Date:     2010-09-10                                                     *
# * Last update: 2012-08-16                                                  *
# * License:  LGPL v3.0                                                      *
# *                                                                          *
# * Original Author (PHP):  Olivier PLATHEY 2004-12-31                       *
# * Ported to Python 2.4 by Max (maxpat78@yahoo.it) on 2006-05               *
# * Maintainer:  Mariano Reingart (reingart@gmail.com) et al since 2008 est. *
# * NOTE: 'I' and 'D' destinations are disabled, and simply print to STDOUT  *
# ****************************************************************************

from __future__ import division, with_statement

from datetime import datetime
from functools import wraps
import math
import errno
import os, sys, zlib, struct, re, tempfile, struct

from .ttfonts import TTFontFile
from .fonts import fpdf_charwidths
from .php import substr, sprintf, print_r, UTF8ToUTF16BE, UTF8StringToArray
from .py3k import PY3K, pickle, urlopen, BytesIO, Image, basestring, unicode, exception, b

# Global variables
FPDF_VERSION = '1.7.2'
FPDF_FONT_DIR = os.path.join(os.path.dirname(__file__),'font')
FPDF_CACHE_MODE = 0 # 0 - in same folder, 1 - none, 2 - hash
FPDF_CACHE_DIR = None
SYSTEM_TTFONTS = None

PAGE_FORMATS = {
    "a3": (841.89, 1190.55),
    "a4": (595.28, 841.89),
    "a5": (420.94, 595.28),
    "letter": (612, 792),
    "legal": (612, 1008),
}

class FPDF(object):
```

```python
    def small_1(self):
        print('this function is small')

    def big_no_call(self):
        print('This function is big')
        print('This function is big')
        print('This function is big')
        print('This function is big')
        print('This function is big')
        print('This function is big')
        print('This function is big')

    def small_2(self):
        print('this function is also small')
        print('another line')

    def big_with_one_call(self):
        print('This function is big with one call')
        print('This function is big with one call')
        print('This function is big with one call')
        print('This function is big with one call')
        self.small_2()
        print('This function is big with one call')
        print('This function is big with one call')
        self.big_no_call()

    def small_3(self):
        print('this function is also small')

    def big_with_three_call(self):
        print('This function is big with three call')
        print('This function is big with three call')
        print('This function is big with three call')
        print('This function is big with three call')
        self.small_1()
        print('This function is big with three call')
        print('This function is big with three call')
        print('This function is big with three call')
        print('This function is big with three call')
        self.small_2()
        self.small_1()
        self.small_3()
```