一、系统要求

python3.4以上版本, 不支持python2.x

二、准备工具

requests、beautifulsoup 是爬虫两大神器，reuqests 用于网络请求，beautifusoup 用于操作 html 数据。有了这两把梭子，干起活来利索。scrapy 这样的爬虫框架我们就不用了，这样的小程序派上它有点杀鸡用牛刀的意思。此外，既然是把 html 文件转为 pdf，那么也要有相应的库支持， wkhtmltopdf 就是一个非常的工具，它可以用适用于多平台的 html 到 pdf 的转换，pdfkit 是 wkhtmltopdf 的Python封装包。首先安装好下面的依赖包

```
pip install requests
pip install beautifulsoup4
pip install pdfkit
```

三、安装 wkhtmltopdf

Windows平台直接在 http://wkhtmltopdf.org/downloads.html 下载稳定版的 wkhtmltopdf 进行安装，安装完成之后把该程序的执行路径加入到系统环境 $PATH 变量中，否则 pdfkit 找不到 wkhtmltopdf 就出现错误 "No wkhtmltopdf executable found"。Ubuntu 和 CentOS 可以直接用命令行进行安装

```
$ sudo apt-get install wkhtmltopdf  # ubuntu
$ sudo yum intsall wkhtmltopdf      # centos
```

四、 运行

```
python crawler.py
```

五、常见问题

1. SyntaxError: Missing parentheses in call to 'print'

   beautifulsoup3不支持python2,所以下载beautifulsoup是要指定 beautifusoup4

2. 如果是使用PyCharm开发, 那么运行的时候要在shell/cmd 窗口执行脚本, 直接在Pycharm中运行会找不

到 wkhtmltopdf命令

六、完整代码如下

```
# coding=utf-8
from __future__ import unicode_literals

import logging
import os
import re
import time

try:
    from urllib.parse import urlparse  # py3
except:
    from urlparse import urlparse  # py2

import pdfkit
import requests
from bs4 import BeautifulSoup

html_template = """
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
</head>
<body>
{content}
</body>
</html>

"""


class Crawler(object):
    """
    爬虫基类，所有爬虫都应该继承此类
    """
    name = None

```

```
    def __init__(self, name, start_url):
        """
        初始化
        :param name: 将要被保存为PDF的文件名称
        :param start_url: 爬虫入口URL
        """
        self.name = name
        self.start_url = start_url
        self.domain = '{uri.scheme}://{uri.netloc}'.format(uri=urlparse(self.start_url))

    @staticmethod
    def request(url, **kwargs):
        """
        网络请求,返回response对象
        :return:
        """
        response = requests.get(url, **kwargs)
        return response

    def parse_menu(self, response):
        """
        从response中解析出所有目录的URL链接
        """
        raise NotImplementedError

    def parse_body(self, response):
        """
        解析正文,由子类实现
        :param response: 爬虫返回的response对象
        :return: 返回经过处理的html正文文本
        """
        raise NotImplementedError

    def run(self):
        start = time.time()
        options = {
            'page-size': 'Letter',
            'margin-top': '0.75in',
            'margin-right': '0.75in',
            'margin-bottom': '0.75in',
            'margin-left': '0.75in',
            'encoding': "UTF-8",
```

```
            'custom-header': [
                ('Accept-Encoding', 'gzip')
            ],
            'cookie': [
                ('cookie-name1', 'cookie-value1'),
                ('cookie-name2', 'cookie-value2'),
            ],
            'outline-depth': 10,
        }
        htmls = []
        for index, url in enumerate(self.parse_menu(self.request(self.start_url))):
            html = self.parse_body(self.request(url))
            f_name = ".".join([str(index), "html"])
            with open(f_name, 'wb') as f:
                f.write(html)
            htmls.append(f_name)

        pdfkit.from_file(htmls, self.name + ".pdf", options=options)
        for html in htmls:
            os.remove(html)
        total_time = time.time() - start
        print(u"总共耗时: %f 秒" % total_time)


class LiaoxuefengPythonCrawler(Crawler):
    """
    廖雪峰Python3教程
    """

    def parse_menu(self, response):
        """
        解析目录结构,获取所有URL目录列表
        :param response 爬虫返回的response对象
        :return: url生成器
        """
        soup = BeautifulSoup(response.content, "html.parser")
        menu_tag = soup.find_all(class_="uk-nav uk-nav-side")[1]
        for li in menu_tag.find_all("li"):
            url = li.a.get("href")
            if not url.startswith("http"):
                url = "".join([self.domain, url])  # 补全为全路径
            yield url
```

```
122
123     def parse_body(self, response):
124         """
125         解析正文
126         :param response: 爬虫返回的response对象
127         :return: 返回处理后的html文本
128         """
129         try:
130             soup = BeautifulSoup(response.content, 'html.parser')
131             body = soup.find_all(class_="x-wiki-content")[0]
132 
133             # 加入标题, 居中显示
134             title = soup.find('h4').get_text()
135             center_tag = soup.new_tag("center")
136             title_tag = soup.new_tag('h1')
137             title_tag.string = title
138             center_tag.insert(1, title_tag)
139             body.insert(1, center_tag)
140 
141             html = str(body)
142             # body中的img标签的src相对路径的改成绝对路径
143             pattern = "(<img .*?src=\")(.*?)(\")"
144 
145             def func(m):
146                 if not m.group(2).startswith("http"):
147                     rtn = "".join([m.group(1), self.domain, m.group(2), m.group(3)])
148                     return rtn
149                 else:
150                     return "".join([m.group(1), m.group(2), m.group(3)])
151 
152             html = re.compile(pattern).sub(func, html)
153             html = html_template.format(content=html)
154             html = html.encode("utf-8")
155             return html
156         except Exception as e:
157             logging.error("解析错误", exc_info=True)
158 
159 
160 if __name__ == '__main__':
161     start_url = "http://www.liaoxuefeng.com/wiki/001373951630592960dd18361248578c67b8067c
162 8c017b000"
163     crawler = LiaoxuefengPythonCrawler("廖雪峰Git", start_url)
```

```
    crawler.run()
```

七、结果显示