

# Python 爬虫基础

MSC@CQU 2017 姜竣予

2017-11-25

# | CONTENTS

01 认识爬虫

02 HTTP简介

03 环境搭建

04 一个实例

# 01

---

## 认识爬虫

1. 什么是爬虫
  2. 爬虫能干什么
  3. 爬虫的本质
-

# 01

## 什么是爬虫

网络蜘蛛 (Web spider) 也叫网络爬虫 (Web crawler) , 蚂蚁 (ant) , 自动检索工具 (automatic indexer) , 或者 (在FOAF软件概念中) 网络疾走 (WEB scutter) , 是一种 “自动化浏览网络” 的程序, 或者说是一种网络机器人。它们被广泛用于互联网搜索引擎或其他类似网站, 以获取或更新这些网站的内容和检索方式。它们可以自动采集所有其能够访问到的页面内容, 以供搜索引擎做进一步处理 (分检整理下载的页面) , 而使得用户能更快的检索到他们需要的信息。 (从维基百科上抄下来的)

---

# 02

## 爬虫能干什么

- 可以创建搜索引擎 (Google, 百度)
- 可以用来抢火车票
- 带逛

可参考链接: [利用爬虫技术能做到哪些很酷很有趣很有用的事情?](#)

---

# 03

## 爬虫的本质

模仿浏览器来打开网页。

# 02

---

## HTTP简介

超文本传输协议（英文：HyperText Transfer Protocol，缩写：HTTP）是一种用于分布式、协作式和超媒体信息系统的 应用层协议[1]。

HTTP是万维网的数据通信的基础。

---

# HTTP协议

超文本传输协议（英文：HyperText Transfer Protocol，缩写：HTTP）是一种用于分布式、协作式和超媒体信息系统的应用层协议[1]。HTTP是万维网的数据通信的基础。

设计HTTP最初的目的是为了提供一种发布和接收HTML页面的方法。通过HTTP或者HTTPS协议请求的资源由统一资源标识符（Uniform Resource Identifiers，URI）来标识。

HTTP的发展是由蒂姆·伯纳斯-李于1989年在欧洲核子研究组织（CERN）所发起。HTTP的标准制定由万维网协会（World Wide Web Consortium，W3C）和互联网工程任务组（Internet Engineering Task Force，IETF）进行协调，最终发布了一系列的RFC，其中最著名的是1999年6月公布的 RFC 2616，定义了HTTP协议中现今广泛使用的一个版本——HTTP 1.1。

2014年12月，互联网工程任务组（IETF）的Hypertext Transfer Protocol Bis (httpbis) 工作小组将HTTP/2标准提议递交至IESG进行讨论[2]，于2015年2月17日被批准。[3] HTTP/2标准于2015年5月以RFC 7540正式发表，取代HTTP 1.1成为HTTP的实现标准。[4]

**#这也是我从维基百科上抄下来的**

# 03

---

## 环境搭建

1. 编辑器
  2. 浏览器
  3. Python爬虫相关的包  
(requests, urllib, lxml, beautifulsoup, scrapy...)
-

# 04

---

## 一个实例

爬取某条微博下的所有评论

---



# 请求方法

- **GET**: 向指定的资源发出“显示”请求。
- **POST**: 向指定资源提交数据, 请求服务器进行处理(例如提交表单或者上传文件)。数据被包含在请求本文中。
- **OPTIONS**: 使服务器传回该资源所支持的所有HTTP请求方法。用'\*'来代替资源名称, 向Web服务器发送OPTIONS请求, 可以测试服务器功能是否正常运行。
- **HEAD**: 与GET方法一样, 都是向服务器发出指定资源的请求。只不过服务器将不传回资源的本文部分。
- **PUT**: 向指定资源位置上传其最新内容。
- **DELETE**: 请求服务器删除Request-URI所标识的资源。
- **TRACE**: 回显服务器收到的请求, 主要用于测试或诊断。
- **CONNECT**: HTTP/1.1协议中预留给能够将连接改为管道方式的代理服务器。

# 状态码

状态代码的第一个数字代表当前响应的类型:

- 1xx消息——请求已被服务器接收, 继续处理
- 2xx成功——请求已成功被服务器接收、理解、并接受
- 3xx重定向——需要后续操作才能完成这一请求
- 4xx请求错误——请求含有词法错误或者无法被执行
- 5xx服务器错误——服务器在处理某个正确请求时发生错误

常见状态码:

- 200 OK
- 400 Bad Request
- 403 Forbidden
- 404 Not
- 500 Internal Server Error
- 503 Server Unavailable

## 关于反爬虫

- 限制IP访问频率，超过频率就断开连接。（解决办法：降低爬虫的速度在每个请求前面加上`time.sleep()`；或者不停的更换代理IP）
- 后台对访问进行统计，如果单个User-Agent访问超过阈值，予以封锁。（误伤很大，一般站点不会使用）
- 针对于cookies的。（一般网站不会用）
- 动态页面的反爬虫。（解决办法：利用phantomJS执行JS来模拟人为操作以及触发页面中的JS脚本）

**THANKS**