

Networked Programs

Chapter 12



Python for Informatics: Exploring Information
www.py4inf.com



open.michigan

Unless otherwise noted, the content of this course material is licensed under a Creative Commons Attribution 3.0 License.
<http://creativecommons.org/licenses/by/3.0/>.

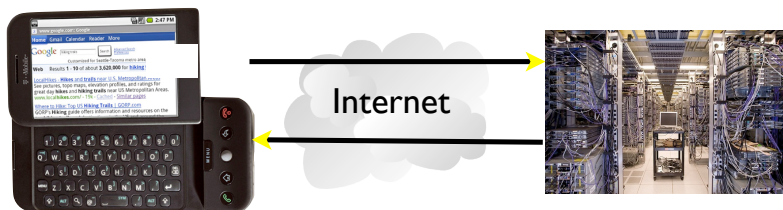
Copyright 2009, 2010, Charles Severance, Jim Eng

UNIVERSITY OF MICHIGAN

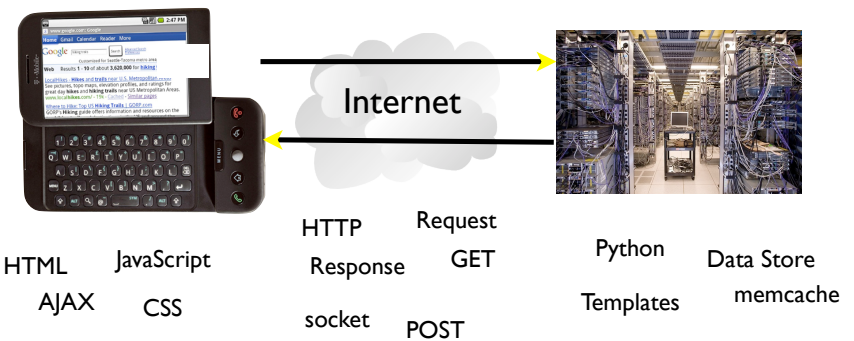


Client

Server



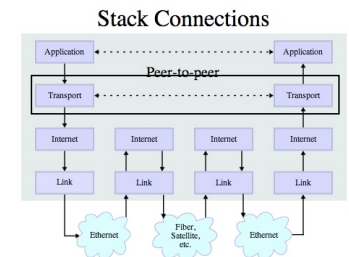
Wikipedia



Network Architecture....

Transport Control Protocol (TCP)

- Built on top of IP (Internet Protocol)
- Assumes IP might lose some data - stores and retransmits data if it seems to be lost
- Handles “flow control” using a transmit window
- Provides a nice reliable pipe



Source: http://en.wikipedia.org/wiki/Internet_Protocol_Suite

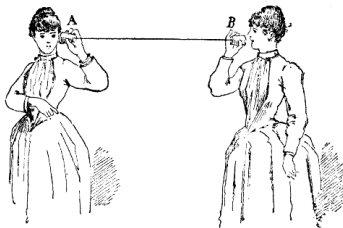


FIG. 76. Trådtelefon.



http://en.wikipedia.org/wiki/Tin_can_telephone

<http://www.flickr.com/photos/kitcowan/2103850699/>

TCP Connections / Sockets

"In computer networking, an Internet socket or network socket is an endpoint of a bidirectional inter-process communication flow across an Internet Protocol-based computer network, such as the Internet."

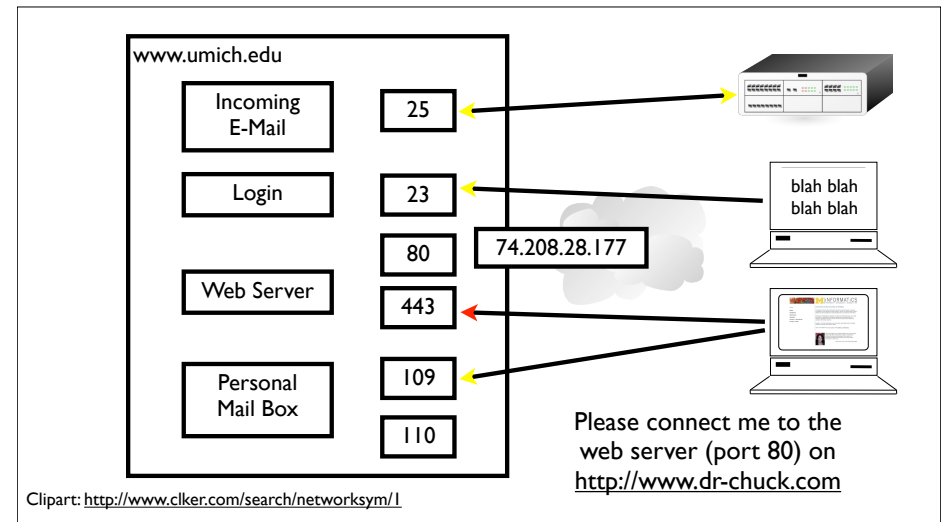


http://en.wikipedia.org/wiki/Internet_socket

TCP Port Numbers

- A port is an application-specific or process-specific software communications endpoint
- It allows multiple networked applications to coexist on the same server.
- There is a list of well-known TCP port numbers

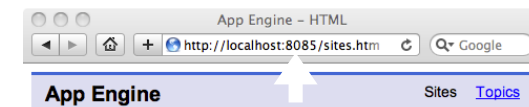
http://en.wikipedia.org/wiki/TCP_and_UDP_port



Common TCP Ports

- Telnet (23) - Login
- SSH (22) - Secure Login
- HTTP (80)
- HTTPS (443) - Secure
- SMTP (25) (Mail)
- IMAP (143/220/993) - Mail Retrieval
- POP (109/110) - Mail Retrieval
- DNS (53) - Domain Name
- FTP (21) - File Transfer

http://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers



App Engine: Sites

Here are some sites we hope you find useful:

- [Python Learn](#)
- [App Engine Learn](#)
- [Google App Engine Site](#)

Sometimes we see the port number in the URL if the web server is running on a "non-standard" port.

Sockets in Python

- Python has built-in support for TCP Sockets

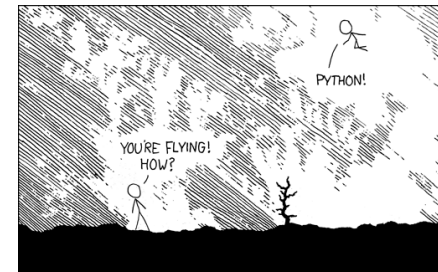
```
import socket
```

```
mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
mysock.connect( ('www.py4inf.com', 80) )
```

Host

Port

<http://docs.python.org/library/socket.html>

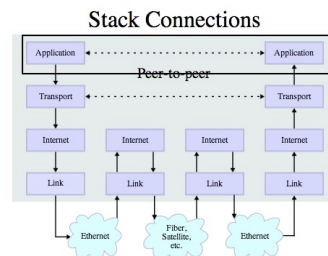


<http://xkcd.com/353/>



Application Protocol

- Since TCP (and Python) gives us a reliable socket, what do we want to do with the socket? What problem do we want to solve?
- Application Protocols
 - Mail
 - World Wide Web



Source: http://en.wikipedia.org/wiki/Internet_Protocol_Suite

HTTP - Hypertext Transport Protocol

- The dominant Application Layer Protocol on the Internet
- Invented for the Web - to Retrieve HTML, Images, Documents etc
- Extended to be data in addition to documents - RSS, Web Services, etc..
- Basic Concept - Make a Connection - Request a document - Retrieve the Document - Close the Connection

<http://en.wikipedia.org/wiki/Http>

HTTP

- The HyperText Transport Protocol is the set of rules to allow browsers to retrieve web documents from servers over the Internet

What is a Protocol?

- A set of rules that all parties follow for so we can predict each other's behavior
- And not bump into each other
- On two-way roads in USA, drive on the right hand side of the road
- On two-way roads in the UK drive on the left hand side of the road



http://	www.dr-chuck.com/	page1.htm
protocol	host	document

<http://www.youtube.com/watch?v=x2GyILq59rI>



Getting Data From The Server

- Each the user clicks on an anchor tag with an href= value to switch to a new page, the browser makes a connection to the web server and issues a "GET" request - to GET the content of the page at the specified URL
- The server returns the HTML document to the Browser which formats and displays the document to the user.

Making an HTTP request

- Connect to the server like `www.dr-chuck.com`
 - a "hand shake"
- Request a document (or the default document)
 - GET `http://www.dr-chuck.com/page1.htm`
 - GET `http://www.mlive.com/ann-arbor/`
 - GET `http://www.facebook.com`



Browser



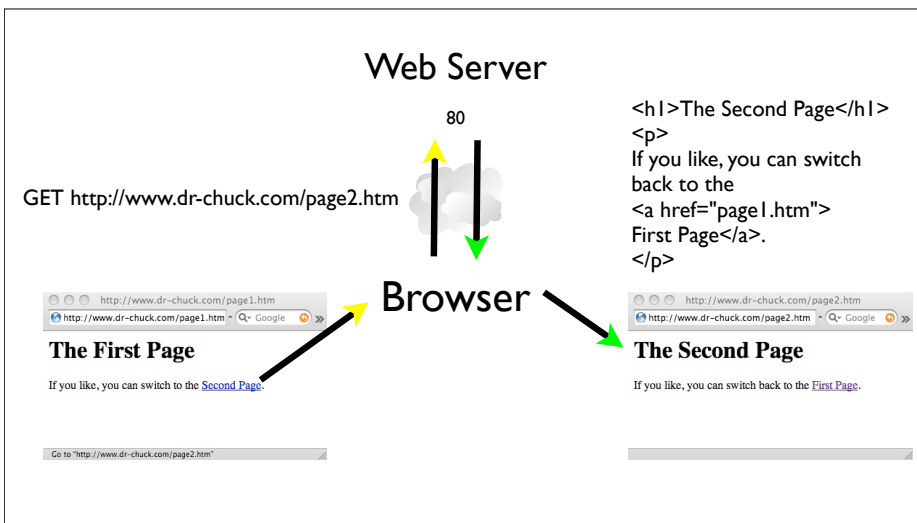
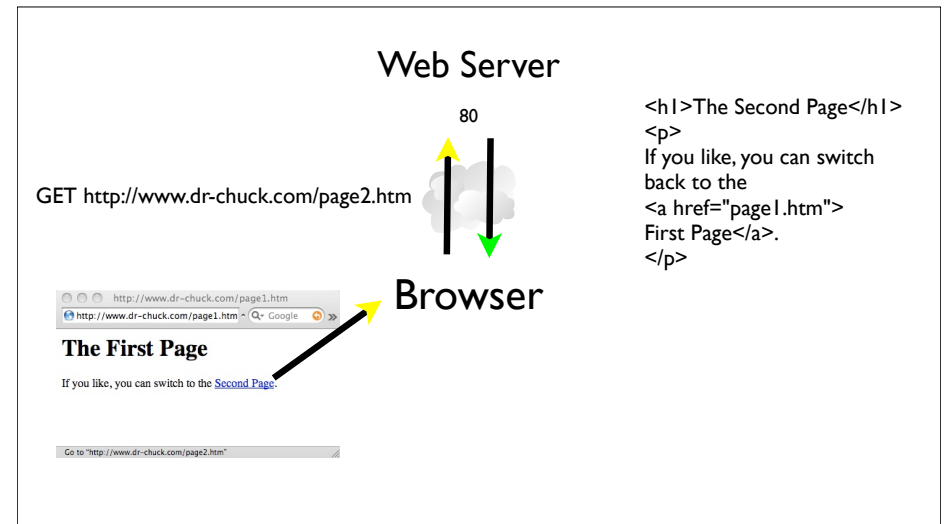
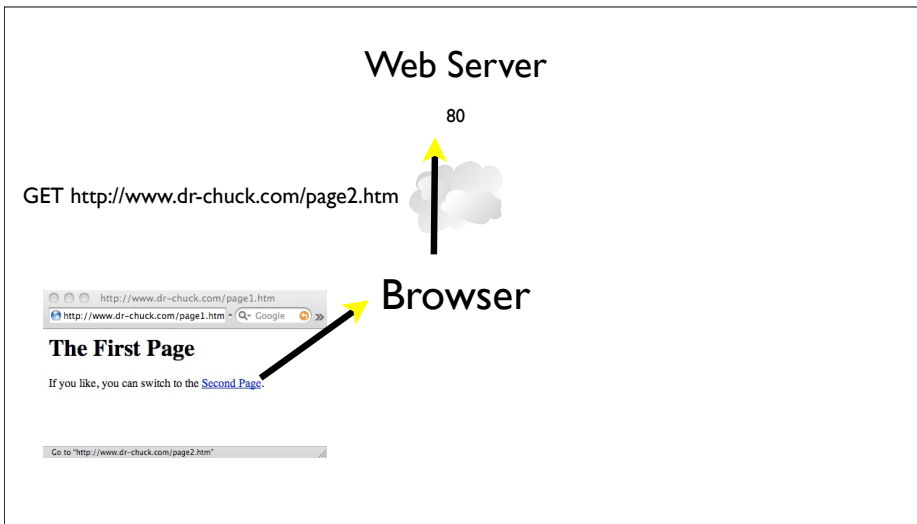
Web Server

80

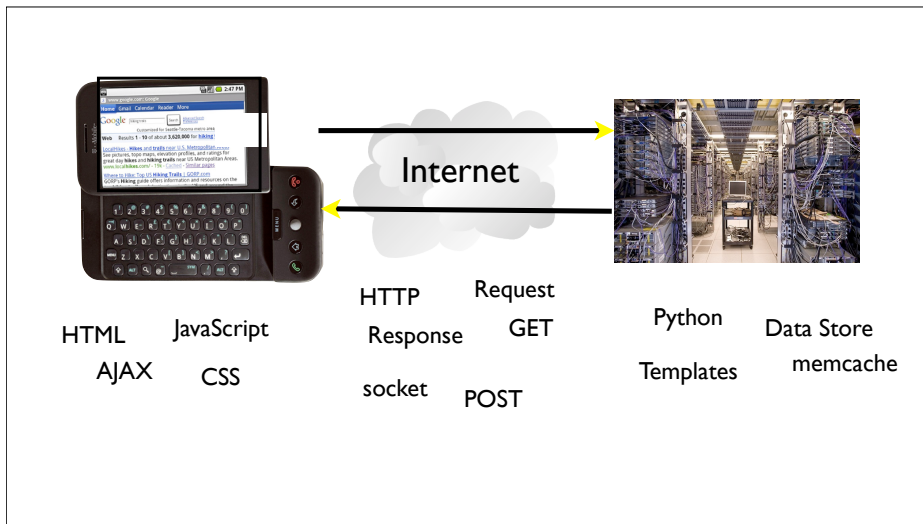


Browser





Lets Write a Web Browser!



Internet Standards

- The standards for all of the Internet protocols (inner workings) are developed by an organization
- Internet Engineering Task Force (IETF)
- www.ietf.org
- Standards are called “RFCs” - “Request for Comments”

INTERNET PROTOCOL
DARPA INTERNET PROGRAM
PROTOCOL SPECIFICATION
September 1981

The internet protocol treats each internet datagram as an independent entity unrelated to any other internet datagram. There are no connections or logical circuits (virtual or otherwise).
The internet protocol uses four key mechanisms in providing its service: Type of Service, Time to Live, Options, and Header Checksum.

Source: <http://tools.ietf.org/html/rfc791>

Network Working Group
Request for Comments: 2616
Obsoletes: 2068
Category: Standards Track

R. Fielding
UC Irvine
J. Gettys
Compaq/W3C
J. Mogul
Compaq
H. Frystyk
W3C/MIT
L. Masinter
Xerox
P. Leach
Microsoft
T. Berners-Lee
W3C/MIT
June 1999

<http://www.w3.org/Protocols/rfc2616/rfc2616.txt>

Hypertext Transfer Protocol -- HTTP/1.1

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (1999). All Rights Reserved.

Abstract

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information

5 Request

A request message from a client to a server includes, within the first line of that message, the method to be applied to the resource, the identifier of the resource, and the protocol version in use.

```
Request = Request-Line           ; Section 5.1
          *(( general-header       ; Section 4.5
            | request-header      ; Section 5.3
            | entity-header ) CRLF) ; Section 7.1
          CRLF
          [ message-body ]       ; Section 4.3
```

5.1 Request-Line

The Request-Line begins with a method token, followed by the Request-URI and the protocol version, and ending with CRLF. The elements are separated by SP characters. No CR or LF is allowed except in the final CRLF sequence.

```
Request-Line = Method SP Request-URI SP HTTP-Version CRLF
```

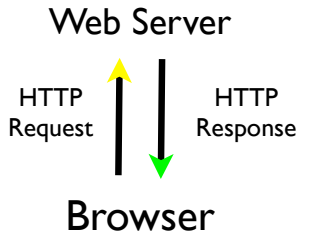

Making an HTTP request

- Connect to the server like www.dr-chuck.com
 - a "hand shake"
- Request a document (or the default document)
 - GET <http://www.dr-chuck.com/page1.htm>
 - GET <http://www.mlive.com/ann-arbor/>
 - GET <http://www.facebook.com>

“Hacking” HTTP

```
$ telnet www.dr-chuck.com 80
Trying 74.208.28.177...
Connected to www.dr-chuck.com.
Escape character is '^]'.
GET http://www.dr-chuck.com/page1.htm
<h1>The First Page</h1>
<p>
If you like, you can switch to the
<a href="http://www.dr-chuck.com/page2.htm">
Second Page</a>.
</p>
```

Port 80 is the non-encrypted HTTP port



Accurate Hacking in the Movies

- Matrix Reloaded
- Bourne Ultimatum
- Die Hard 4
- ...



http://nmap.org/movies.html
http://www.youtube.com/watch?v=Zy5_gYu_isg

```
$ telnet www.dr-chuck.com 80
Trying 74.208.28.177...
Connected to www.dr-chuck.com.
Escape character is '^J'.
GET http://www.dr-chuck.com/page1.htm
<h1>The First Page</h1>
<p>
If you like, you can switch to the
<a href="http://www.dr-chuck.com/page2.htm">
Second Page</a>.
</p>
Connection closed by foreign host.
```



```
si-csev-mbp:tex csev$ telnet www.umich.edu 80
```

```
Trying 141.211.144.190...
```

```
Connected to www.umich.edu.
```

```
Escape character is '^['.
```

```
GET /
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
```

```
<head>
```

```
<title>University of Michigan</title>
```

```
<meta name="description" content="University of Michigan is one of the top universities of the world, a diverse public institution of higher learning, fostering excellence in research. U-M provides outstanding undergraduate, graduate and professional education, serving the local, regional, national and international communities." />
```

```
...
<link rel="alternate stylesheet" type="text/css" href="/CSS/accessible.css"
media="screen" title="accessible" />
<link rel="stylesheet" href="/CSS/print.css" media="print,projection" />
<link rel="stylesheet" href="/CSS/other.css"
media="handheld, tty, tv, braille, embossed, speech, aural" />
...
```

```
<dl>
<dt><a href="http://ns.umich.edu/htdocs/releases/story.php?id=8077">
</a><span class="verbose"></span></dt>
<dd><a href="http://ns.umich.edu/htdocs/releases/story.php?
id=8077">Scientists harness the power of electricity in the brain</a></dd>
</dl>
```



As the browser reads the document, it finds other URLs that must be retrieved to produce the document.

The big picture...



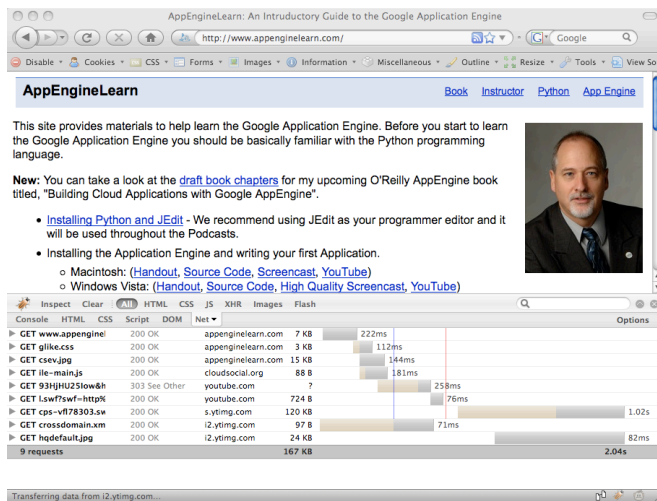
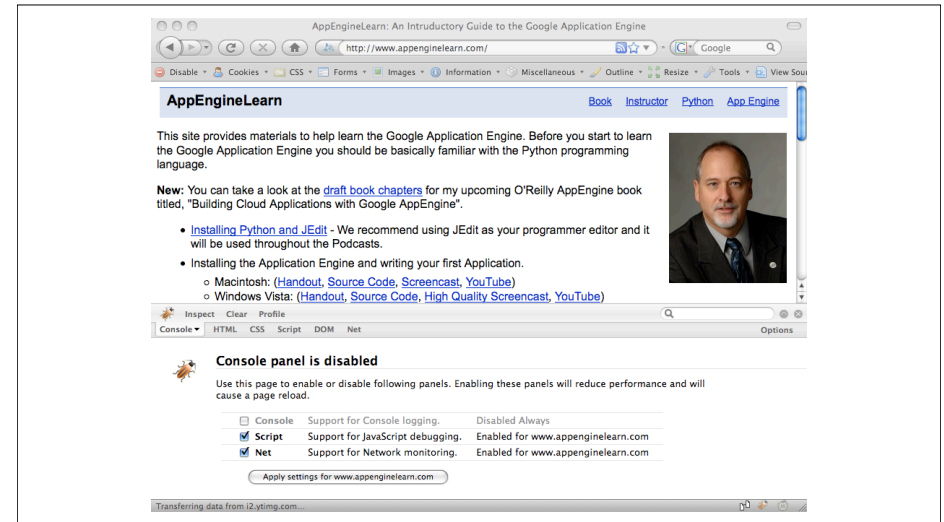
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//
EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
<title>University of Michigan</title>
```

```
@import "/CSS/graphical.css"/;
p.text strong, .verbose, .verbose p, .verbose h2{text-
indent:-876em;position:absolute}
p.text strong a{text-decoration:none}
p.text em{font-weight:bold;font-style:normal}
div.alert{background:#eee;border:1px solid red;padding:
5em;margin:0 25%}
a img{border:none}
.hot br, .quick br, dl.feature2 img{display:none}
div#main label, legend{font-weight:bold}
```



Firebug reveals the detail...

- If you haven't already installed the Firebug FireFox extension you need it now
- It can help explore the HTTP request-response cycle
- Some simple-looking pages involve lots of requests:
 - HTML page(s)
 - Image files
 - CSS Style Sheets
 - Javascript files



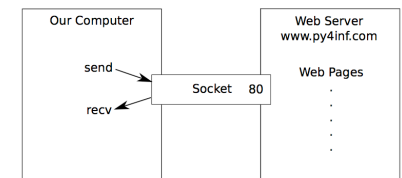
An HTTP Request in Python

```
import socket
```

```
mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
mysock.connect(('www.py4inf.com', 80))
mysock.send('GET http://www.py4inf.com/code/romeo.txt HTTP/1.0\n\n')
```

```
while True:
    data = mysock.recv(512)
    if ( len(data) < 1 ) :
        break
    print data

mysock.close()
```



```
HTTP/1.1 200 OK
Date: Sun, 14 Mar 2010 23:52:41 GMT
Server: Apache
Last-Modified: Tue, 29 Dec 2009 01:31:22 GMT
ETag: "143c1b33-a7-4b395bea"
Accept-Ranges: bytes
Content-Length: 167
Connection: close
Content-Type: text/plain
```

```
But soft what light through yonder window breaks
It is the east and Juliet is the sun
Arise fair sun and kill the envious moon
Who is already sick and pale with grief
```

HTTP Header

```
while True:
    data = mysock.recv(512)
    if ( len(data) < 1 ) :
        break
    print data
```

HTTP Body

Making HTTP Easier With urllib

Using urllib in Python

- Since HTTP is so common, we have a library that does all the socket work for us and makes web pages look like a file

```
import urllib

fhand = urllib.urlopen('http://www.py4inf.com/code/romeo.txt')

for line in fhand:
    print line.strip()
```

<http://docs.python.org/library/urllib.html>

`urllib.py`

```
import urllib

fhand = urllib.urlopen('http://www.py4inf.com/code/romeo.txt')

for line in fhand:
    print line.strip()
```

```
But soft what light through yonder window breaks
It is the east and Juliet is the sun
Arise fair sun and kill the envious moon
Who is already sick and pale with grief
```

<http://docs.python.org/library/urllib.html>

`urllib.py`

Like a file...

```
import urllib

fhand = urllib.urlopen('http://www.py4inf.com/code/romeo.txt')

counts = dict()
for line in fhand:
    words = line.split()
    for word in words:
        counts[word] = counts.get(word,0) + 1
print counts
```

urlwords.py

Reading Web Pages

```
import urllib

fhand = urllib.urlopen('http://www.dr-chuck.com/page1.htm')
for line in fhand:
    print line.strip()

<h1>The First Page</h1>
<p>
If you like, you can switch to the
<a href="http://www.dr-chuck.com/page2.htm">
Second Page</a>.
</p>
```

urllib1.py

Going from one page to another...

```
import urllib

fhand = urllib.urlopen('http://www.dr-chuck.com/page1.htm')
for line in fhand:
    print line.strip()

<h1>The First Page</h1>
<p>
If you like, you can switch to the
<a href="http://www.dr-chuck.com/page2.htm">
Second Page</a>.
</p>
```

Google

```
import urllib

fhand = urllib.urlopen('http://www.dr-chuck.com/page1.htm')
for line in fhand:
    print line.strip()
```

Parsing HTML (a.k.a Web Scraping)

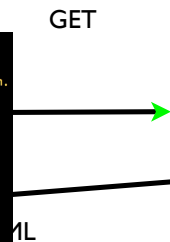
What is Web Scraping?

- When a program or script pretends to be a browser and retrieves web pages, looks at those web pages, extracts information and then looks at more web pages.
- Search engines scrape web pages - we call this “spidering the web” or “web crawling”

http://en.wikipedia.org/wiki/Web_scraping
http://en.wikipedia.org/wiki/Web_crawler



```
charles-severance@macbook-air:Scraping$ python
Python 2.5 (r25:51918, Sep 19 2006, 08:49:13)
[GCC 4.0.1 (Apple Computer, Inc. build 5341)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import urllib
>>> f = urllib.urlopen("http://www.dr-chuck.com/")
>>> contents = f.read()
>>> f.close()
>>> print len(contents)
95328
>>> print contents[0:30]
html>
head>
<title>Dr. C
>>>
```



Why Scrape?

- Pull data - particularly social data - who links to who?
- Get your own data back out of some system that has no “export capability”
- Monitor a site for new information
- Spider the web to make a database for a search engine

Scraping Web Pages

- There is some controversy about web page scraping and some sites are a bit snippy about it.
- Google: facebook scraping block
- Republishing copyrighted information is not allowed
- Violating terms of service is not allowed

<http://www.facebook.com/terms.php>

User Conduct

You understand that except for advertising programs offered by us on the Site (e.g., Facebook Flyers, Facebook Marketplace), the Service and the Site are available for your personal, non-commercial use only. You represent, warrant and agree that no materials of any kind submitted through your account or otherwise posted, transmitted, or shared by you on or through the Service will violate or infringe upon the rights of any third party, including copyright, trademark, privacy, publicity or other personal or proprietary rights; or contain libelous, defamatory or otherwise unlawful material.

In addition, you agree not to use the Service or the Site to:

- harvest or collect email addresses or other contact information of other users from the Service or the Site by electronic or other means for the purposes of sending unsolicited emails or other unsolicited communications;
- use the Service or the Site in any unlawful manner or in any other manner that could damage, disable, overburden or impair the Site;
- use automated scripts to collect information from or otherwise interact with the Service or the Site;

The Easy Way - BeautifulSoup

- You could do string searches the hard way
- Or use the free software called BeautifulSoup from www.crummy.com

<http://www.crummy.com/software/BeautifulSoup/>

Place the BeautifulSoup.py file in the same folder as your Python code...

```
import urllib
from BeautifulSoup import *

url = raw_input('Enter - ')
html = urllib.urlopen(url).read()
soup = BeautifulSoup(html)

# Retrieve a list of the anchor tags
# Each tag is like a dictionary of HTML attributes

tags = soup('a')

for tag in tags:
    print tag.get('href', None)
```

urllinks.py

```
<h1>The First Page</h1>
<p>
If you like, you can switch to the
<a href="http://www.dr-chuck.com/page2.htm">
Second Page</a>.
</p>
```

```
html = urllib.urlopen(url).read()
soup = BeautifulSoup(html)
tags = soup('a')
for tag in tags:
    print tag.get('href', None)
```

```
python urllinks.py
Enter - http://www.dr-chuck.com/page1.htm
http://www.dr-chuck.com/page2.htm
```

Summary

- The TCP/IP gives us pipes / sockets between applications
- We designed application protocols to make use of these pipes
- HyperText Transport Protocol (HTTP) is a simple yet powerful protocol
- Python has good support for sockets, HTTP, and HTML parsing