

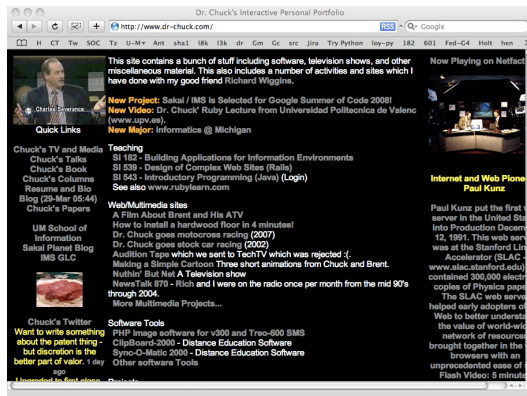
# “Viewing” Web Pages In Python

Charles Severance - [www.dr-chuck.com](http://www.dr-chuck.com)

# 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.

[http://en.wikipedia.org/wiki/Web\\_scraping](http://en.wikipedia.org/wiki/Web_scraping)



GET

Server

HTML

GET

```
charles-severances-macbook-air:Scraping csev$ 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
>>>
```

HTML

# 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

# 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;

<http://www.myspace.com/index.cfm?fuseaction=misc.terms>

8. **Content/Activity Prohibited.** The following are examples of the kind of Content that is illegal or prohibited to post on or through the MySpace Services. MySpace reserves the right to investigate and take appropriate legal action against anyone who, in MySpace's sole discretion, violates this provision, including without limitation, removing the offending Content from the MySpace Services and terminating the Membership of such violators. Prohibited Content includes, but is not limited to, Content that, in the sole discretion of MySpace:

8.22 any automated use of the system, such as, but not limited to, using scripts to add friends or send comments or messages;

Looks like a loophole... So we will play a bit with MySpace - be respectful - look but never touch..

# Web Protocols

## The Request / Response Cycle



# Web Standards

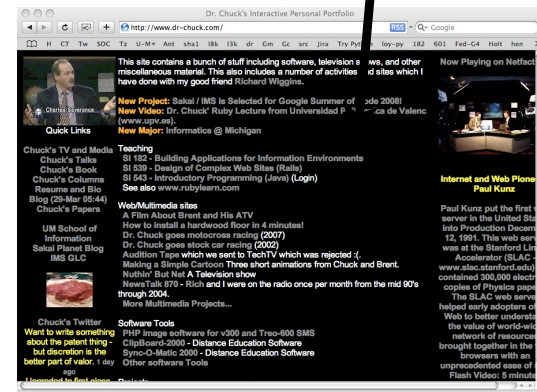
- HTML - HyperText Markup Language - a way to describe how pages are supposed to look and act in a web browser
- HTTP - HyperText Transport Protocol - how your Browser communicates with a web server to get more HTML pages and send data to the web server

# What is so “Hyper” about the web?

- If you think of the whole web as a “space” like “outer space”
- When you click on a link at one point in space - you instantly “hyper-transport” to another place in the space
- It does not matter how far apart the two web pages are

Hello there my name is Chuck.

Go ahead and click on [here](#).

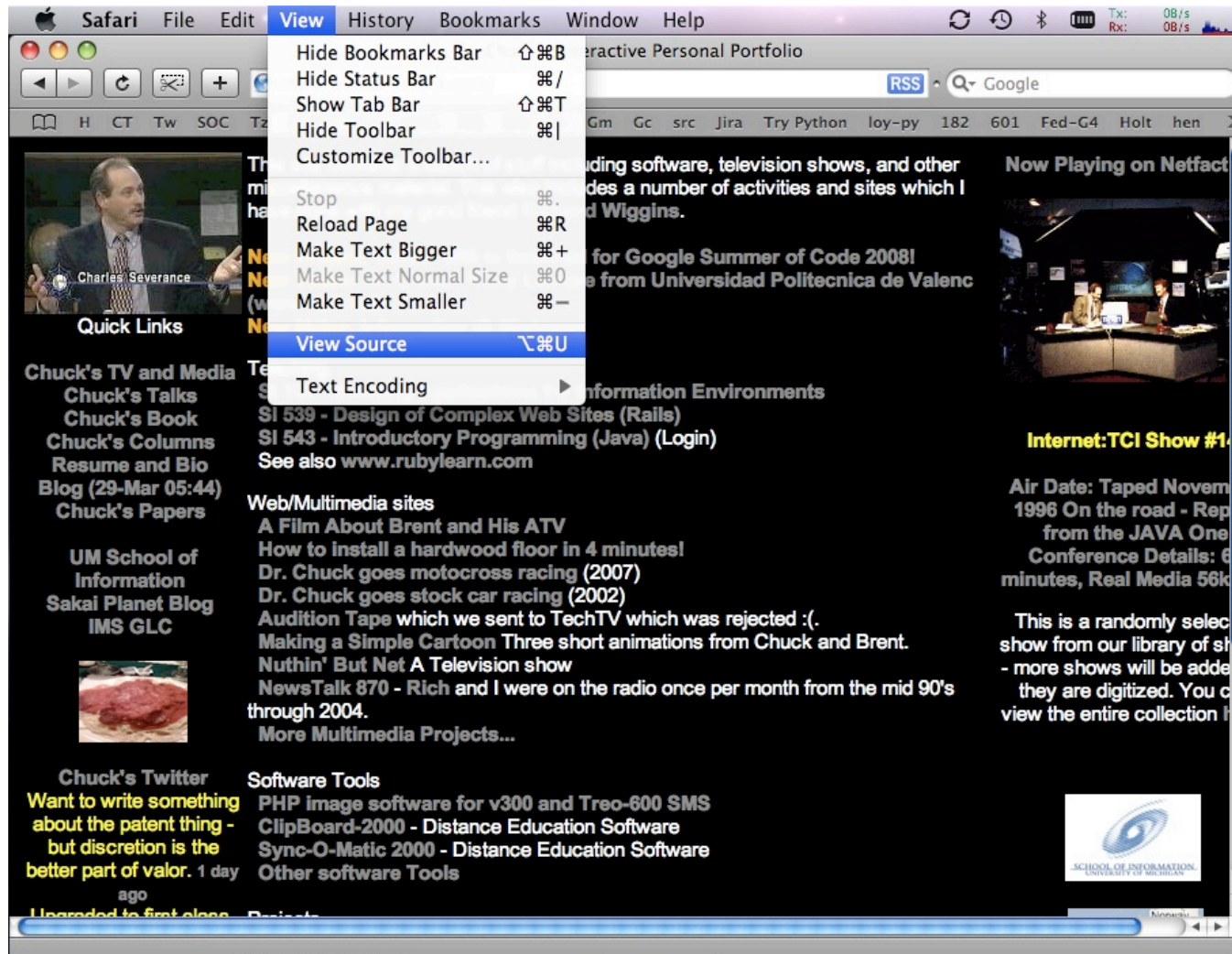


# HTML - View Source

- The hard way to learn HTML is to look at the source to many web pages.
- There are lots of less than < and greater than > signs
- Buying a good book is much easier



<http://www.sitepoint.com/books/html/>



New Project:

</strong>

<a href=http://www.sakaiproject.org/soc2008/>

Sakai / IMS is Selected for Google Summer of Code 2008!

</a>

<br>

<strong>

New Video:

</strong>

<a href=http://www.dr-chuck.com/media.php?comment=top&id=85>

Dr. Chuck' Ruby Lecture from Universidad Politecnica de Valenc (www.upv.es).

</a>

<br>

<strong>

New Major:

</strong>

<a href=http://informatics.umich.edu/>

Informatics @ Michigan

</a>

<p>

Teaching

<BR> &nbsp;

<a href=http://www.si182.com/>SI 182 - Building Applications for Information Environments</a>

<BR> &nbsp;

<a href=http://www.si539.com/ target=\_new>SI 539 - Design of Complex Web Sites (Rails)</a>

<BR> &nbsp;

<a href=http://www.si543.com target=\_new>SI 543 - Introductory Programming (Java)</a> (Login)

# HTML - Brief tutorial

Hello there my name is Chuck.

Go ahead and click on [here](#).

```
<p>  
Hello there my name is Chuck.  
</p>  
<p>  
Go ahead and click on  
<a href="http://www.dr-chuck.com">here</a>.  
</p>
```

Start a hyperlink

Where to go

What to show

End a hyperlink



New Project:

</strong>

<a href=http://www.sakaiproject.org/soc2008/>

Sakai / IMS is Selected for Google Summer of Code 2008!

</a>

<br>

<strong>

New Video:

</strong>

<a href=http://www.dr-chuck.com/media.php?comment=top&id=85>

Dr. Chuck' Ruby Lecture from Universidad Politecnica de Valenc (www.upv.es).

</a>

<br>

<strong>

New Major:

</strong>

<a href=http://informatics.umich.edu/>

Informatics @ Michigan

</a>

<p>

Teaching

<BR> &nbsp;

<a href=http://www.si182.com/>SI 182 - Building Applications for Information Environments</a>

<BR> &nbsp;

<a href=http://www.si539.com/ target=\_new>SI 539 - Design of Complex Web Sites (Rails)</a>

<BR> &nbsp;

<a href=http://www.si543.com target=\_new>SI 543 - Introductory Programming (Java)</a> (Login)

**New Project:** Sakai / IMS is Selected for Google Summer of Code 2008!  
**New Video:** Dr. Chuck' Ruby Lecture from Universidad Politecnica de Valenc (www.upv.es).  
**New Major:** Informatics @ Michigan

**Teaching**

**SI 182 - Building Applications for Information Environments**

**SI 539 - Design of Complex Web Sites (Rails)**

**SI 543 - Introductory Programming (Java) (Login)**

# HyperText Transport Protocol

- HTTP describes how your browser talks to a web server to get the next page.
- That next page will use HTML
- The way the pages are retrieved is HTTP

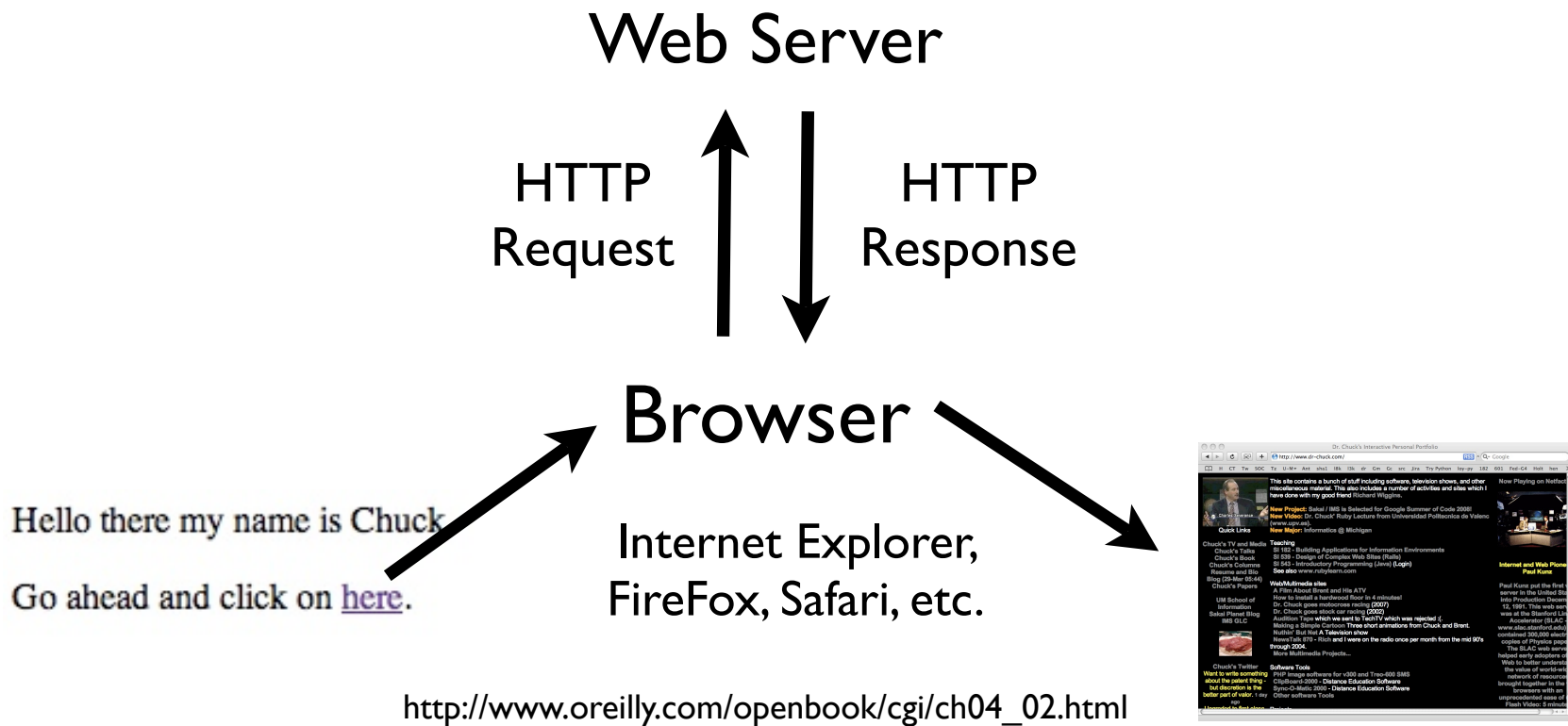


<nerdy-stuff>

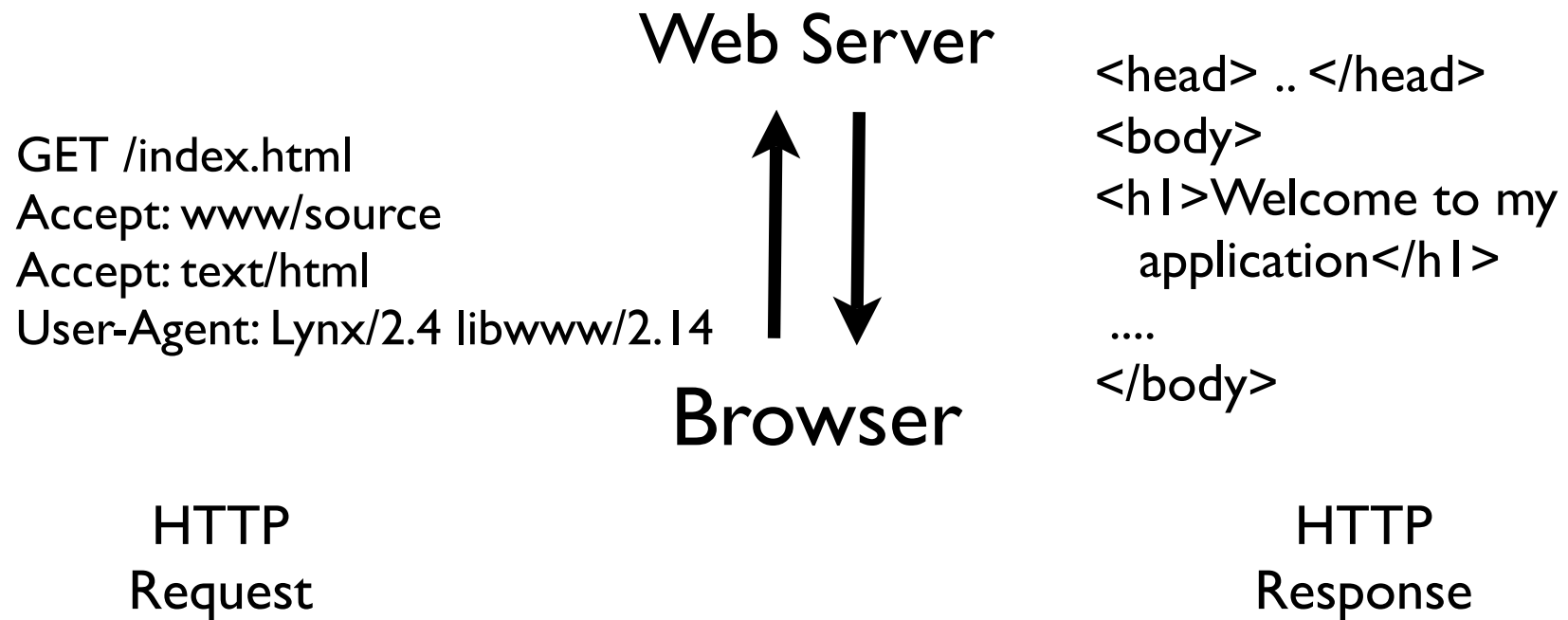
# Getting Data From The Server

- Each time 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.

# HTTP Request / Response Cycle



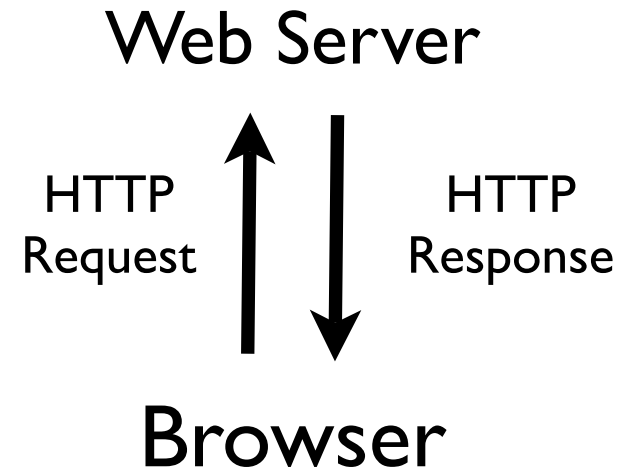
# HTTP Request / Response Cycle



[http://www.oreilly.com/openbook/cgi/ch04\\_02.html](http://www.oreilly.com/openbook/cgi/ch04_02.html)

# “Hacking” HTTP

```
Last login: Wed Oct 10 04:20:19 on ttyp2
si-csev-mbp:~ csev$ telnet www.umich.edu 80
Trying 141.211.144.188...
Connected to www.umich.edu.
Escape character is '^]'.
GET /
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
....
```



**</nerdy-stuff>**

# HTML and HTTP in Python

# Using urllib to retrieve web pages

```
charles-severances-macbook-air:Scraping csev$ 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
>>> 
```





## 18.5 urllib -- Open arbitrary resources by URL

This module provides a high-level interface for fetching data across the World Wide Web. In particular, the `urlopen()` function is similar to the built-in function `open()`, but accepts Universal Resource Locators (URLs) instead of filenames. Some restrictions apply -- it can only open URLs for reading, and no seek operations are available.

It defines the following public functions:

**`urlopen(uri[, data[, proxies]])`**

Open a network object denoted by a URL for reading. If the URL does not have a scheme identifier, or if it has `file:` as its scheme identifier, this opens a local file (without universal newlines); otherwise it opens a socket to a server somewhere on the network. If the connection cannot be made the `IOError` exception is raised. If all went well, a file-like object is returned. This supports the following methods: `read()`, `readline()`, `readlines()`, `fileno()`, `close()`, `info()` and `geturl()`. It also has proper support for the iterator protocol. One caveat: the `read()` method, if the size argument is omitted or negative, may not read until the end of the data stream; there is no good way to determine that the entire stream from a socket has been read in the general case.

Except for the `info()` and `geturl()` methods, these methods have the same interface as for file objects -- see section [3.9](#) in this manual. (It is not a built-in file object, however, so it can't be used at those few places where a true built-in file object is required.)

<http://docs.python.org/lib/module-urllib.html>

- You get the entire web page when you do `f.read()` - lines are separated by a “newline” character “`\n`”

```
>>> print contents[0:150]
<html>
<head>
  <title>Dr. Chuck's Interactive Personal Portfolio</title>
<style type="text/css">
body { background: black; font-family: Arial,Hel
>>> 
```

- You get the entire web page when you do `f.read()` - lines are separated by a “newline” character “`\n`”
- We can split the contents into lines using the `split()` function

```
>>> print contents[0:150]
<html>
<head>
  <title>Dr. Chuck's Interactive Personal Portfolio</title>
<style type="text/css">
body { background: black; font-family: Arial,Hel
>>> 
```

- Splitting the contents on the newline character gives use a nice list where each entry is a single line
- We can easily write a for loop to look through the lines

```
>>> print len(contents)
95328
>>> lines = contents.split("\n")
>>> print len(lines)
2244
>>> print lines[3]
<style type="text/css">
>>>
```

for ln in lines:  
    # Do something for each line

# Parsing HTML

- We could treat the HTML as XML - but most HTML is not well formed enough to be truly XML
- So we end up with ad hoc parsing
  - For each line look for some trigger value
  - If you find your trigger value - parse out the information you want using string manipulation

# Looking for links

Hello there my name is Chuck.

Go ahead and click on [here](#).

```
<p>  
Hello there my name is Chuck.  
</p>  
<p>  
Go ahead and click on  
<a href="http://www.dr-chuck.com">here</a>.  
</p>
```

Start a hyperlink

Where to go

What to show

End a hyperlink

for ln in lines:

print "Looking at", ln

pos = ln.find('href=')

if pos > -1 :

print "\* Found link at", pos

<http://docs.python.org/lib/string-methods.html>

`find(sub[, start[, end]])`

Return the lowest index in the string where substring *sub* is found, such that *sub* is contained in the range [*start*, *end*]. Optional arguments *start* and *end* are interpreted as in slice notation. Return -1 if *sub* is not found.

\$ python links.py

Looking at <p>

Looking at Hello there my name is Chuck.

Looking at </p>

Looking at <p>

Looking at Go ahead and click on

Looking at <a href="http://www.dr-chuck.com/">here</a>.

\* Found link at 3

Looking at </p>

<http://docs.python.org/lib/string-methods.html>

`find(sub[, start[, end]])`

Return the lowest index in the string where substring *sub* is found, such that *sub* is contained in the range *[start, end]*. Optional arguments *start* and *end* are interpreted as in slice notation. Return *-1* if *sub* is not found.

```
pos = ln.find('href="')
```



```
<a href="http://www.dr-chuck.com/">here</a>.
0123
```



```
pos = ln.find('href="')
```



0123

```
<a href="http://www.dr-chuck.com/">here</a>.
```

456789



Six characters

```
etc = ln[pos+6:]
```

```
http://www.dr-chuck.com/">here</a>.
```

```
etc = ln[pos+6:]
```



endpos = 24

```
http://www.dr-chuck.com/">here</a>.  
0123456789012345678901234
```

```
endpos = etc.find('\"')  
linktext = etc[:endpos]
```

```
http://www.dr-chuck.com/
```

```
<a href="http://www.dr-chuck.com/>here</a>.
```



No closing quote

```
print "* Found link at", pos  
etc = ln[pos+6:]  
print "Chopped off front bit", etc  
endpos = etc.find("")  
print "End of link at", endpos  
linktext = etc[:endpos]  
print "Link text", linktext
```

What happens?

Looking at <a href="http://www.dr-chuck.com/">here</a>.

\* Found link at 3

Chopped off front bit http://www.dr-chuck.com/">here</a>.

End of link at -1

Link text http://www.dr-chuck.com/">here</a>

Remember that string  
position -1 is one from the  
right end of the string.

Hello Bob

012       -1

```
print "* Found link at", pos
etc = ln[pos+6:]
print "Chopped off front bit", etc
endpos = etc.find("")
print "End of link at",endpos
linktext = etc[:endpos]
print "Link text", linktext
```

The final bit with a bit of paranoia in the form of a try / except block in case something goes wrong.

No need to blow up with a traceback - just move to the next line and look for a link.

```
for ln in lines:
    print "Looking at", ln
    pos = ln.find('href="')
    if pos > -1 :
        linktext = None
        try:
            print "* Found link at", pos
            etc = ln[pos+6:]
            print "Chopped off front bit", etc
            endpos = etc.find('"')
            print "End of link at",endpos
            if endpos > 0:
                linktext = etc[:endpos]
        except:
            print "Could not parse link",ln
            print "Link text", linktext
```

```
python links.py
Looking at <p>
Looking at Hello there my name is Chuck.
Looking at </p>
Looking at <p>
Looking at Go ahead and click on
Looking at <a href="http://www.dr-chuck.com/">here</a>.
* Found link at 3
Chopped off front bit http://www.dr-chuck.com/">here</a>.
End of link at 24
Link text http://www.dr-chuck.com/
Looking at <a href="http://www.dr-chuck.com/">here</a>.
* Found link at 3
Chopped off front bit http://www.dr-chuck.com/">here</a>.
End of link at -1
Link text None
Looking at </p>
```

My Space

# Basic Outline

- # Make a list of a few friends as a starting point

- # For a few pages

  - # Pick a random friend from the list

  - # Retrieve the myspace page

  - # Loop through the page, looking for friend links

  - # Add those friends to the list

- # Print out all of the friends



<http://profile.myspace.com/index.cfm?fuseaction=user.viewprofile&friendid=125104617>


User Shortcuts:  People  powered by Google

[Home](#) | [Browse](#) | [Search](#) | [Invite](#) | [Film](#) | [Mail](#) | [Blogs](#) | [Favorites](#) | [Forum](#) | [Groups](#) | [Events](#) | [MySpaceTV](#) | [Music](#) | [Comedy](#) | [Classifieds](#)

**MYSPEACE MUSIC** [Music Videos](#) | [Directory](#) | [Search](#) | [Top Artists](#) | [Shows](#) | [Music Forums](#) | [Music Classifieds](#) | [Artist Signup](#)

**Burning The Night Sky [Vote To Get Us On Warped!]**  
Rock / Screamo

**"ENDLESS TRAGEDY EP available now!"**



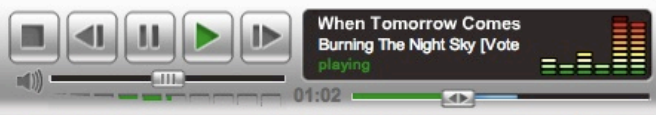
Lansing, Michigan  
United States

Profile Views: 76546


Last Login: 3/30/2008

View My: [Pics](#) | [Videos](#)

Contacting Burning The Night Sky [Vote To Get Us On Warped!]  
**MESSAGE** **FORWARD**  
**ADD** **FAVORITES**



**Total Plays: 84993** **Downloads Today: 0** **Plays Today: 16**



Endless Tragedy EP  
2008

[When Tomorrow Comes](#) Plays: 4863  
[Download](#) | [Comment](#) | [Lyrics](#) | [Add](#)

[I'm Not Who You Wan...](#) Plays: 5645  
[Download](#) | [Comment](#) | [Lyrics](#) | [Add](#)

[Purpose Is Overrated](#) Plays: 359  
[Download](#) | [Comment](#) | [Lyrics](#) | [Add](#)

[There's A Fine Line](#) Plays: 8449  
[Download](#) | [Comment](#) | [Lyrics](#) | [Add](#)

[open player in a new window](#)


```
Source of http://profile.myspace.com/index.cfm?fuseaction=user.viewprofile&friendid=125104617
1 match 17855 Done

<tr>
    <td width="435" style="word-wrap:break-word">&nbsp;&nbsp;&nbsp;<span
class="orangetext15">Burning The Night Sky [Vote To Get Us On Warped!]'s Friend Space (Top 40)</span></td>
</tr>
</table>
<table cellpadding="5" cellspacing="0" width="435" align="center" border="0">
    <tr>
        <td bgcolor="ffffff" colspan="4" width="435" style="word-wrap:break-word">
            <span class="btext">
                Burning The Night Sky [Vote To Get Us On Warped!]&nbsp;&nbsp;has <span
class="redbtext">17855</span> friends.
            </span>
        </td>
    </tr>
    <tr>
        <td>
            <table width="435" border="0" cellpadding="5" cellspacing="0" align="center">
                <tr>
                    <td bgcolor="FFFFFF" align="center" valign="top" width="1">
                        <table border="0" cellspacing="0"
                            <tr>
                                <td
                                    bgcolor="FFFFFF" align="center" valign="top" width="107" style="word-wrap:break-word">
                                        href="http://profile.myspace.com/index.cfm?fuseaction=user.viewprofile&friendid=51910594"
                                        id="ctl00_Main_ctl00_UserFriends1_FriendRepeater_ctl00_friendLink">Steven</a>&nbsp;&nbsp;
                                    </td>
                                <td>
                                    &nbsp;&nbsp;<a
                                        href="http://profile.myspace.com/index.cfm?


```

&nbsp;<a href="http://profile.myspace.com/index.cfm?fuseaction=user.viewprofile&friendid=51910594" id=

Trigger string  
(friendurl)



Frend ID



```
# Look for friends
pos = line.find(friendurl)
if pos > 0 :
    # print line
    try:
        rest = line[pos+len(friendurl):]
        print "Rest of the line", rest
        endquote = rest.find('"')
        if endquote > 0 :
            newfriend = rest[:endquote]
            print newfriend
```

```
# Make an empty list
friends = list()
friends.append("125104617")
friends.append("51910594")
friends.append("230923259")
```

```
if newfriend in friends :
    print "Already in list", newfriend
else :
    print "Adding friend", newfriend
    friends.append(newfriend)
```

Demo

# Assignment 10

- Build a simple Python program to prompt for a URL, retrieve data and then print the number of lines and characters
- Add a feature to the myspace spider to find the average age of a set of friends.

```
charles-severances-macbook-air:assn-10 csev$ python returl.py
Enter a URL:http://www.dr-chuck.com
Retrieving: http://www.dr-chuck.com
Server Data Retrieved 95256 characters and 2243 lines
Enter a URL:http://www.umich.edu/
Retrieving: http://www.umich.edu/
Server Data Retrieved 26730 characters and 361 lines
Enter a URL:http://www.pythonlearn.com/
Retrieving: http://www.pythonlearn.com/
Server Data Retrieved 95397 characters and 2241 lines
Enter a URL:
charles-severances-macbook-air:assn-10 csev$
```

# Summary

- Python can easily retrieve data from the web and use its powerful string parsing capabilities to sift through the information and make sense of the information
- We can build a simple directed web-spider for our own purposes
- Make sure that we do not violate the terms and conditions of a web site and make sure not to use copyrighted material improperly