Web Clients and Crawlers

Web Clients

alternatives to web browsers opening a web page and copying its content

Scanning files

looking for strings between double quotes parsing URLs for the server location

Web Crawlers

making requests recursively incremental development, modular design of code

MCS 275 Lecture 31
Programming Tools and File Management
Jan Verschelde, 2 April 2008

MCS 275 L-31

2 April 2008

Web Cli

alternatives to web browsers opening a web page and

canning file

looking for strings between double quotes parsing URLs for the serve

Neb Crawler

making requests recursivel incremental development, modular design of code

Web Clients and Crawlers

Web Clients

alternatives to web browsers

looking for strings between double quotes

incremental development, modular design of code

alternatives to web browsers

MCS 275 L-31

2 April 2008



Web Clients

alternatives to web browsers

Recall testing ourwebserver.py last lecture.

→ the client is a browser, e.g.: Netscape, Firefox, ...

MCS 275 L-31

2 April 2008

alternatives to web browsers

Web Clients

alternatives to web browsers

Recall testing ourwebserver.py last lecture.

 \rightarrow the client is a browser, e.g.: Netscape, Firefox, ...

But we can browse the web using scripts.

Why do we want to do this?

- 1. more efficient: no overhead from GUI
- 2. in control: request only what we need
- 3. crawl the web: request recursively
 - → operate like a search engine

How?

use urllib and urlparse modules

MCS 275 L-31

2 April 2008

Web Client

alternatives to web

opening a web page and copying its content

canning files

looking for strings between double quotes

parsing URLs for the selection

Web Crawlers



Recall testing ourwebserver.py last lecture.

 \rightarrow the client is a browser, e.g.: Netscape, Firefox, ...

But we can browse the web using scripts.

Why do we want to do this?

- 1. more efficient: no overhead from GUI
- in control: request only what we need
 → update most recent information
- 3. crawl the web: request recursively
 - → operate like a search engine

How?

use urllib and urlparse modules

MCS 275 L-31

2 April 2008

Web Client

alternatives to web browsers

opening a web page and copying its content

canning files

looking for strings between double quotes

parsing URLs for th location

Web Crawlers



 \rightarrow the client is a browser, e.g.: Netscape, Firefox, ...

But we can browse the web using scripts.

Why do we want to do this?

- 1. more efficient: no overhead from GUI
- in control: request only what we need
 → update most recent information
- 3. *crawl* the web: request recursively

 → operate like a search engine

How?

use urllib and urlparse modules

MCS 275 L-31

2 April 2008

Web Clien

alternatives to web browsers

opening a web page and copying its content

canning files

looking for strings between double quotes

parsing URLs for the

Web Crawler



Web Clients

alternatives to web browsers

Recall testing ourwebserver.py last lecture.

 \rightarrow the client is a browser, e.g.: Netscape, Firefox, ...

But we can browse the web using scripts.

Why do we want to do this?

- 1. more efficient: no overhead from GUI
- 2. in control: request only what we need
 - → update most recent information
- 3. crawl the web: request recursively
 - → operate like a search engine

How?

use urllib and urlparse modules

MCS 275 L-31

2 April 2008

web Clien

alternatives to web browsers

opening a web page and copying its content

canning files

looking for strings between double quotes

parsing URLs for the location

Web Crawle



ightarrow the client is a browser, e.g.: Netscape, Firefox, ...

But we can browse the web using scripts.

Why do we want to do this?

- 1. more efficient: no overhead from GUI
- 2. in control: request only what we need
 - → update most recent information
- 3. crawl the web: request recursively
 - → operate like a search engine

How?

use urllib and urlparse modules

MCS 275 L-31

2 April 2008

Web Clients

alternatives to web browsers

opening a web page and copying its content

Scanning file:

looking for strings between double quotes

parsing URLs for the location

Web Crawler



Recall testing ourwebserver.py last lecture.

 \rightarrow the client is a browser, e.g.: Netscape, Firefox, ...

But we can browse the web using scripts.

Why do we want to do this?

- 1. more efficient: no overhead from GUI
- 2. in control: request only what we need
 - → update most recent information
- 3. crawl the web: request recursively
 - → operate like a search engine

How?

use urllib and urlparse modules

MCS 275 L-31

2 April 2008

Web Client

alternatives to web browsers

opening a web page and copying its content

canning file

looking for strings between double quotes

parsing URLs for the

Web Crawle

Web Clients and Crawlers

MCS 275 L-31 2 April 2008

Web Client

alternatives to web browsers

opening a web page and copying its content

Scanning files

looking for strings between double quotes parsing URI's for the serve

....

making requests recursive

Web Clients

alternatives to web browsers opening a web page and copying its content

One and a class

looking for strings between double quotes parsing URLs for the server location

Web Crawlers

making requests recursively incremental development, modular design of code



Syntax:

```
urllib.retrieve( < URL >, < file name > )
```

Example

```
rom urllib import retrieve
retrieve('http://www.python.org','wpt.html
```

Opening a web page with urllib.urlopen

```
from urllib import urlopen
< object like file > = urlopen( < URL > )
data = < object like file >.read( < size >
< object like file >.close()
```

→ process web pages like we handle files

MCS 275 L-31

2 April 2008

Web Clients

alternatives to w browsers

opening a web page and copying its content

anning files

looking for strings between double quotes

location

Web Crawlers

rnaking requests recursively incremental development, modular design of code

Syntax:

```
urllib.retrieve( < URL >, < file name > )
```

Example:

```
from urllib import retrieve
retrieve('http://www.python.org','wpt.html') **king requests recursively
```

MCS 275 L-31

2 April 2008

opening a web page and copying its content

incremental development.

```
urllib.retrieve( < URL >, < file name > )
```

Example:

```
from urllib import retrieve
retrieve('http://www.python.org','wpt.html') **king requests recursively
```

Opening a web page with urllib.urlopen:

```
from urllib import urlopen
< object like file > = urlopen( < URL > )
```

MCS 275 L-31

2 April 2008

opening a web page and copying its content

incremental development.

copying its content

incremental development.

using the urllib module

Syntax:

urllib.retrieve(< URL >, < file name >)

Example:

```
from urllib import retrieve
retrieve('http://www.python.org','wpt.html') **king requests recursively
```

Opening a web page with urllib.urlopen:

```
from urllib import urlopen
< object like file > = urlopen( < URL > )
data = < object like file >.read( < size > )
```

Syntax:

```
urllib.retrieve( < URL >, < file name > )
```

Example:

```
from urllib import retrieve
retrieve('http://www.python.org','wpt.html')aking requests recursively
```

Opening a web page with urllib.urlopen:

```
from urllib import urlopen
< object like file > = urlopen( < URL > )
data = < object like file >.read( < size > )
< object like file >.close()
```

→ process web pages like we handle files

opening a web page and copying its content

incremental development.

opening a web page and copying its content

Scanning files

looking for strings betwee double quotes

duon

Web Crawlers

making requests recursivel incremental development, modular design of code

```
def copypage(url,file):
```

Given the URL for the web page, a copy of its contents is written to file. Both url and file are strings.

11 11 11

11 11 11

```
import urllib
copyfile = open(file,'w'
f = urllib.urlopen(url)
while True:
   data = f.read(80)
   if data == '': break
   copyfile.write(data)
f.close()
```

```
Web Clients
```

alternatives to web browsers

opening a web page and copying its content

canning files

looking for strings between double quotes

Janon

Web Crawlers

```
def copypage(url,file):
   11 11 11
   Given the URL for the web page,
   a copy of its contents is written to file.
   Both url and file are strings.
   . . . .
   import urllib
   copyfile = open(file,'w')
   f = urllib.urlopen(url)
```

```
Web Clients
```

browsers
opening a web page and

copying its content

canning files

looking for strings between double quotes

ation

Web Crawlers

```
def copypage(url,file):
   11 11 11
   Given the URL for the web page,
   a copy of its contents is written to file.
   Both url and file are strings.
   11 11 11
   import urllib
   copyfile = open(file,'w')
   f = urllib.urlopen(url)
   while True:
      data = f.read(80)
```

```
Web Clients
alternatives to web
```

opening a web page and copying its content

canning files

looking for strings between double quotes

Jacon

Web Crawlers

```
def copypage(url,file):
   11 11 11
   Given the URL for the web page,
   a copy of its contents is written to file.
   Both url and file are strings.
   11 11 11
   import urllib
   copyfile = open(file,'w')
   f = urllib.urlopen(url)
   while True:
      data = f.read(80)
      if data == '': break
      copyfile.write(data)
```

```
Web Clients
alternatives to web
```

opening a web page and copying its content

canning files

looking for strings between double quotes

30011

Web Crawlers

```
def copypage(url,file):
   11 11 11
   Given the URL for the web page,
   a copy of its contents is written to file.
   Both url and file are strings.
   11 11 11
   import urllib
   copyfile = open(file,'w')
   f = urllib.urlopen(url)
   while True:
      data = f.read(80)
      if data == '': break
      copyfile.write(data)
   f.close()
   copyfile.close()
```

Web Clients and Crawlers

MCS 275 L-31

2 April 2008

Web Clien

browsers opening a web page and

Scanning files

looking for strings between double quotes

parsing URLs for the

Web Crawlers

making requests recursive incremental development, modular design of code

Web Clients

alternatives to web browsers opening a web page and copying its content

Scanning files

looking for strings between double quotes

Web Crawlers

making requests recursively incremental development, modular design of code

Applications to scan an HTML file:

- 1. search for particular information,
- 2. navigate to where the page refers to.

Example (1): download all .py files from $ext{http://www.math.uic.edu/}{\sim} ext{jan/mcs275/main.htm}$

Example (2): retrieve all URLs the paghttp://www.python.orgrefers to.

What is common between these two examples:

.py files and URLs appear between " and "

→ scan for all strings between double quotes

Web Client

alternatives to web prowsers ppening a web page an

canning file

looking for strings between

double quotes
parsing URLs for the serv

Web Crawlers



looking for strings between double quotes

location

Web Crawlers

making requests recursivel incremental development, modular design of code

Applications to scan an HTML file:

- 1. search for particular information,
- 2. navigate to where the page refers to.

Example (1): download all .py files from ${\sf http://www.math.uic.edu/\sim} {\sf jan/mcs275/main.htm}$

Example (2): retrieve all URLs the paghttp://www.python.orgrefers to.

What is common between these two examples:

.py files and URLs appear between " and "

→ scan for all strings between double quotes

Applications to scan an HTML file:

- 1. search for particular information,
- 2. navigate to where the page refers to.

Example (1): download all .py files from http://www.math.uic.edu/~jan/mcs275/main.html making requests recursively

looking for strings between double quotes

location

Web Crawlers

http://www.math.uic.edu/~jan/mcs275/main.html making requests recursively

incremental development, modular design of code

Applications to scan an HTML file:

- 1. search for particular information,
- 2. navigate to where the page refers to.

Example (1): download all .py files from

Example (2): retrieve all URLs the page

 $\verb|http://www.python.org| \textbf{refers to}.$

What is common between these two examples:

.py files and URLs appear between " and "

→ scan for all strings between double quotes

Applications to scan an HTML file:

- 1. search for particular information,
- 2. navigate to where the page refers to.

Example (1): download all .py files from http://www.math.uic.edu/~jan/mcs275/main.html making requests recursively

Example (2): retrieve all URLs the page http://www.python.orgrefers to.

What is common between these two examples:

.py files and URLs appear between " and "

→ scan for all strings between double quotes

Output: list of all strings between double quotes.

Recall that we read files with fixed size buffer:



For double quoted strings which run across two buffers we need another buffer.

Two buffers: one for reading strings from file, one for buffering double quoted string

- → Two functions
 - 1. read buffered data from file,
 - 2. scan the data buffer for double quoted strings

MCS 275 L-31

2 April 2008

web Clients

browsers
opening a web page and

opening a web page and copying its content

canning file

looking for strings between double quotes

parsing URLs for th

Web Crawlers



scanning for double quoted strings

Input: a file, or object like a file.

Output: list of all strings between double quotes.

Recall that we read files with fixed size buffer:



For double quoted strings which run across two buffers we need another buffer.

Two buffers: one for reading strings from file, one for buffering double quoted string

→ Two functions

- 1. read buffered data from file,
- 2. scan the data buffer for double quoted strings.

MCS 275 L-31

2 April 2008

Web Clien

alternatives to web browsers

opening a web page and copying its content

canning file

looking for strings between double quotes

parsing URLs for the se

Veb Crawlers



scanning for double quoted strings

Input: a file, or object like a file.

Output: list of all strings between double quotes.

Recall that we read files with fixed size buffer:



For double quoted strings which run across two buffers we need another buffer.

Two buffers: one for reading strings from file, one for buffering double quoted string.

→ Two functions

- 1. read buffered data from file,
- 2. scan the data buffer for double quoted strings.

MCS 275 L-31

2 April 2008

Web Cliei

alternatives to web browsers

opening a web page and copying its content

canning file

looking for strings between double quotes

location

Veb Crawlers



scanning for double quoted strings

Input: a file, or object like a file.

Output: list of all strings between double quotes.

Recall that we read files with fixed size buffer:



For double quoted strings which run across two buffers we need another buffer.

Two buffers: one for reading strings from file, one for buffering double quoted string

→ Two functions

- 1. read buffered data from file,
- 2. scan the data buffer for double quoted strings.

MCS 275 L-31

2 April 2008

Web Clien

alternatives to web browsers

opening a web page and copying its content

canning file

looking for strings between double quotes

location

Web Crawlers



scanning for double quoted strings

Input: a file, or object like a file.

Output: list of all strings between double quotes.

Recall that we read files with fixed size buffer:



For double quoted strings which run across two buffers we need another buffer.

Two buffers: one for reading strings from file, one for buffering double quoted string.

→ Two functions

- 1. read buffered data from file,
- 2. scan the data buffer for double quoted strings.

MCS 275 L-31

2 April 2008

Web Clie

alternatives to web browsers

opening a web page and copying its content

canning file

looking for strings between double quotes

parsing URLs for the location

Web Crawlers

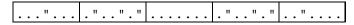


scanning for double quoted strings

Input: a file, or object like a file.

Output: list of all strings between double quotes.

Recall that we read files with fixed size buffer:



For double quoted strings which run across two buffers we need another buffer.

Two buffers: one for reading strings from file, one for buffering double quoted string

→ Two functions:

- 1. read buffered data from file,
- 2. scan the data buffer for double quoted strings.

MCS 275 L-31

2 April 2008

Web Clie

alternatives to web browsers

opening a web page and copying its content

canning file

looking for strings between double quotes

arsing URLs for the

Web Crawlers



scanning for double quoted strings

Input: a file, or object like a file.

Output: list of all strings between double quotes.

Recall that we read files with fixed size buffer:



For double quoted strings which run across two buffers we need another buffer.

Two buffers: one for reading strings from file, one for buffering double guoted str

→ Two functions:

- 1. read buffered data from file,
- 2. scan the data buffer for double quoted strings

MCS 275 L-31

2 April 2008

Web Clie

alternatives to web browsers

opening a web page and copying its content

canning file

looking for strings between double quotes

parsing URLs for the se

eb Crawlers



Output: list of all strings between double quotes.

Recall that we read files with fixed size buffer:



For double quoted strings which run across two buffers we need another buffer.

Two buffers: one for reading strings from file, one for buffering double quoted string.

→ Two functions:

- 1. read buffered data from file,
- 2. scan the data buffer for double quoted strings.

MCS 275 L-31

2 April 2008

Web CI

alternatives to web browsers

opening a web page and copying its content

canning file

looking for strings between double quotes

parsing URLs for the serve

eb Crawlers



Output: list of all strings between double quotes.

Recall that we read files with fixed size buffer:



For double quoted strings which run across two buffers we need another buffer.

Two buffers: one for reading strings from file, one for buffering double quoted string.

- → Two functions:
 - 1. read buffered data from file,
 - scan the data buffer for double quoted strings.

MCS 275 L-31

2 April 2008

Web C

alternatives to web browsers

opening a web page and copying its content

canning file

looking for strings between double quotes

location

eb Crawlers



Output: list of all strings between double quotes.

Recall that we read files with fixed size buffer:



For double quoted strings which run across two buffers we need another buffer.

Two buffers: one for reading strings from file, one for buffering double quoted string.

- → Two functions:
 - 1. read buffered data from file,
 - 2. scan the data buffer for double quoted strings.

Woh Clio

alternatives to web browsers

opening a web page and copying its content

Scanning file

looking for strings between double quotes

location

eb Crawlers



def OuotedStrings(file):

11 11 11

Web Clients

alternatives to web browsers opening a web page and

Scanning files

looking for strings between double quotes

the file and returns a list of all strings parsing URLs for the server location

on the file enclosed between double quotes web Crawlers

making requests recursively incremental development,

```
making incommondular
L = []
buffer = ''
while True:
   data = file.read(80)
   if data == '': break
   (L,buffer) = UpdateQstrings(L,buffer,data
```

Given a file object, this function scans

looking for strings between double quotes

```
def OuotedStrings(file):
    11 11 11
```

Given a file object, this function scans the file and returns a list of all strings person URLs for the sorver on the file enclosed between double quotes web Crawlers

```
L = []
buffer = ''
```

11 11 11

looking for strings between

```
def OuotedStrings(file):
    11 11 11
   Given a file object, this function scans
                                                        double quotes
   the file and returns a list of all strings passing URLs for the server
   on the file enclosed between double quotes web Crawlers
    11 11 11
   L = []
   buffer = ''
   while True:
       data = file.read(80)
       if data == '': break
```

def OuotedStrings(file):

11 11 11

looking for strings between double quotes

the file and returns a list of all strings paragraph URLs for the server

```
on the file enclosed between double quotes web Crawlers
11 11 11
L = []
buffer = ''
while True:
   data = file.read(80)
   if data == '': break
   (L, buffer) = UpdateOstrings(L, buffer, data)
return L
```

Given a file object, this function scans

location

Web Crawlers

making requests recursive incremental development, modular design of code

In ${\tt L}$ we store the double quoted strings. In ${\tt b}$ we buffer the double quoted strings.

$$L = [], b = '"...'$$

$$L = ['...', '.'], b = "$$

$$T_{i} = [' \ ' \ ' \ '] h = ['$$

$$L = ['....', '..', '...'], b = "$$

def UpdateQstrings(L,b,s):

L is a list of double quoted strings, b buffers a double quoted string, and s is the data string to be processed. Returns an update of (L,b).

location

Web Crawlers

incremental development, modular design of code

In $\[\]$ we store the double quoted strings. In $\[\]$ we buffer the double quoted strings.

$$L = [], b = '"...'$$

$$L = ['...','.'], b =$$

$$T_{i} = [' \ ' \ ' \ '] h =$$

$$L = ['....', '..', '...'], b = '"'$$

$$L = ['....', '..', '...'], b = "$$

def UpdateQstrings(L,b,s):

L is a list of double quoted strings, b buffers a double quoted string, and s is the data string to be processed. Returns an update of (L,b).

In \mathbb{L} we store the double quoted strings. In b we buffer the double quoted strings.

$$L = [], b = '"...'$$

$$L = ['...','.'], b = '$$

$$L = [\cdot \dots \cdot , \cdot \dots], D = \cdots$$

$$L = ['....', '.', '...'], b = '"'$$

$$L = ['....', '.', '...', '...'], b = "$$

double quotes

In \mathbb{L} we store the double quoted strings. In b we buffer the double quoted strings.

$$L = [], b = '"...'$$

$$L = ['...','.'], b =$$

$$L = ['....', '..', '...'], b = "$$

canning files

looking for strings between double quotes

location

Web Crawlers

incremental development, modular design of code

In ${\tt L}$ we store the double quoted strings. In ${\tt b}$ we buffer the double quoted strings.

$$L = [], b = '"...'$$

$$L = ['...','.'], b = "$$

$$L = ['....', '.'], b =$$

```
def UpdateQstrings(L,b,s):
```

L is a list of double quoted strings, b buffers a double quoted string, and s is the data string to be processed. Returns an update of (L,b).

double quotes

location

Web Crawlers

making requests recursive incremental development, modular design of code

In L we store the double quoted strings. In b we buffer the double quoted strings.

$$L = [], b = '"...'$$

$$L = ['....', '.'], b = '$$

$$L = ['....','.'], b =$$

$$L = ['....', '..', '...'], b = '"'$$

$$L = ['....','..','...'], b = "$$

def UpdateQstrings(L,b,s):

L is a list of double quoted strings, b buffers a double quoted string, and s is the data string to be processed. Returns an update of (L,b).

double quotes

In \mathbb{L} we store the double quoted strings. In b we buffer the double quoted strings.

$$L = [], b = '"...'$$

$$L = ['....', '.'], b =$$

$$L = ['...','.'], b =$$

$$L = ['....', '..', '...'], b = '"$$

$$L = ['....', '..', '...', '...'], b =$$

canning files

looking for strings between double quotes

location

Web Crawlers

making requests recursivel incremental development, modular design of code

In ${\tt L}$ we store the double quoted strings. In ${\tt b}$ we buffer the double quoted strings.

$$L = [], b = '"...'$$

$$L = ['...', '.'], b = "$$

$$L = ['....', '.'], b =$$

$$L = ['....', '.', '...'], b = '"'$$

."....
$$L = ['....', '.', '...', '...'], b =$$

def UpdateQstrings(L,b,s):

L is a list of double quoted strings, b buffers a double quoted string, and s is the data string to be processed. Returns an update of (L,b).

parsing URLs for the sen

Web Crawlers

making requests recursive incremental development,

In ${\tt L}$ we store the double quoted strings. In ${\tt b}$ we buffer the double quoted strings.

$$L = [], b = '"...'$$

$$L = ['...', '.'], b = '$$

$$L = ['....', '.'], b =$$

def UpdateQstrings(L,b,s):

L is a list of double quoted strings, b buffers a double quoted string, and s is the data string to be processed. Returns an update of (L,b).

double quotes

Web Crawlers

making requests recursive incremental development, modular design of code

In L we store the double quoted strings. In b we buffer the double quoted strings.

$$L = [], b = '"...'$$

$$L = ['...','.'], b =$$

$$L = ['....','.'], b =$$

$$L = ['....', '..', '...'], b = '"'$$

$$L = ['....', '..', '...'], b = "$$

def UpdateQstrings(L,b,s):

L is a list of double quoted strings, b buffers a double quoted string, and s is the data string to be processed. Returns an update of (L,b).

looking for strings between double quotes

location

Web Crawlers

making requests recursive incremental development, modular design of code

In L we store the double quoted strings. In b we buffer the double quoted strings.

..

..

$$L = [], b = '"...'$$

$$L = ['...', '.'], b = "$$

$$L = ['....', '.', '..', '...'], b = '$$

def UpdateQstrings(L,b,s):

L is a list of double quoted strings, b buffers a double quoted string, and s is the data string to be processed. Returns an update of (L,b).

Web Client

alternatives to web browsers

opening a web page and copying its content

Scanning file

looking for strings between double quotes

location

Web Crawlers

```
def UpdateQstrings(L,b,s):
   nb = b
   for i in range(0,len(s)):
      if nb == '':
         if s[i] == '\"':
```

```
Web Clients
```

Ilternatives to web prowsers

opening a web page and copying its content

Scanning files

looking for strings between double quotes

location

Web Crawlers

```
def UpdateQstrings(L,b,s):
   nb = b
   for i in range(0,len(s)):
      if nb == '':
         if s[i] == '\"':
            nb = 'o' # 'o' is for 'opened'
      else:
```

```
Web Clients
```

alternatives to web prowsers

opening a web page and copying its content

Scanning files

looking for strings between double quotes

location

Web Crawlers

```
def UpdateQstrings(L,b,s):
   nb = b
   for i in range(0,len(s)):
      if nb == '':
         if s[i] == '\"':
            nb = 'o' # 'o' is for 'opened'
      else:
         if s[i] != '\"':
            nb += s[i]
```

Web Clients

Ilternatives to web rowsers

opening a web page and copying its content

Scanning files

looking for strings between double quotes

location

Web Crawlers

```
def UpdateQstrings(L,b,s):
   nb = b
   for i in range(0,len(s)):
      if nb == '':
         if s[i] == '\"':
            nb = 'o' # 'o' is for 'opened'
      else:
         if s[i] != '\"':
            nb += s[i]
         else:
                      # do not store 'o'
            L.append(nb[1:len(nb)])
            nb = ''
```

Web Client

Iternatives to web

opening a web page and copying its content

Scanning file:

looking for strings between double quotes

location

Web Crawlers

making requests recursively incremental development, modular design of code

```
def UpdateQstrings(L,b,s):
   nb = b
   for i in range(0,len(s)):
      if nb == '':
         if s[i] == '\"':
            nb = 'o' # 'o' is for 'opened'
      else:
         if s[i] != '\"':
            nb += s[i]
         else:
                      # do not store 'o'
            L.append(nb[1:len(nb)])
            nb = ''
   return (L,nb)
```

Web Client

alternatives to web browsers

opening a web page and copying its content

Scanning file

looking for strings between double quotes

ocation

Web Crawlers

making requests recursively incremental development, modular design of code

```
def main():
   .. .. ..
   Prompts the user for a file name and
   scans the file for double quoted strings.
   . . . .
   print 'getting double guoted strings'
   name = raw input('Give file name : ')
```

Moh Clion

alternatives to web browsers

opening a web page and copying its content

Scanning file

looking for strings between double quotes

location

Web Crawlers

making requests recursively incremental development,

```
def main():
   .. .. ..
   Prompts the user for a file name and
   scans the file for double quoted strings.
   . . . .
   print 'getting double guoted strings'
   name = raw input('Give file name : ')
   file = open(name,'r')
   L = QuotedStrings(file)
   print L
   file.close()
if name ==" main ": main()
```

Web Clients and Crawlers

Woh Clionto

alternatives to web browsers opening a web page and copying its content

Scanning files

looking for strings between double quotes parsing URLs for the server location

Web Crawlers

making requests recursively incremental development, modular design of code

MCS 275 L-31

2 April 2008

Web Clients

browsers
opening a web page and

canning files

looking for strings between double quotes

parsing URLs for the server location

Web Crawlers

making requests recursive incremental development, modular design of code



canning file

looking for strings betwee double quotes

parsing URLs for the server location

Neb Crawlers

```
Recall the second example application: list all URLs referred to at http://www.python.org
```

```
Prompts the user for a web page,
and prints all URLs this page refers to
"""

print 'listing reachable locations'
page = raw_input('Give URL : ')
L = HTTPlinks(page)
print 'found %d HTTP links' % len(L)
ShowLocations(L)
```

looking for strings between double quotes parsing URLs for the server

location

Web Crawlers

incremental development, modular design of code

```
Recall the second example application: list all URLs referred to at http://www.python.org
```

```
def main():
    """
    Prompts the user for a web page,
```

Prompts the user for a web page, and prints all URLs this page refers to.

```
print 'listing reachable locations'
page = raw_input('Give URL : ')
L = HTTPlinks(page)
print 'found %d HTTP links' % len(L)
ShowLocations(L)
```

Recall the second example application:

ShowLocations(L)

anning file

looking for strings between double quotes parsing URLs for the server

location

Web Crawlers

incremental development, modular design of code

```
def main():
    """
    Prompts the user for a web page,
    and prints all URLs this page refers to.
    """
    print 'listing reachable locations'
    page = raw_input('Give URL : ')
    L = HTTPlinks(page)
```

print 'found %d HTTP links' % len(L)

list all URLs referred to at http://www.python.org

Neh Clien

alternatives to web browsers

opening a web page and copying its content

anning file

looking for strings betwee double quotes

parsing URLs for the server location

Web Crawlers

incremental development, modular design of code

```
from scanquotes import UpdateQstrings
```

```
def HTTPfilter(L):
```

Returns from the list L only those strings which begin with http.

```
.....
```

```
H = []
For s in L:
   if len(s) > 4:
      if s[0:4] == 'http': H.append(s)
```

return H

Web Cl

alternatives to web prowsers

opening a web page and copying its content

anning files

looking for strings between double quotes

parsing URLs for the server location

Veb Crawlers

```
from scanquotes import UpdateOstrings
def HTTPfilter(L):
   .. .. ..
   Returns from the list L only those strings
   which begin with http.
   11 11 11
   H = []
   for s in L:
      if len(s) > 4:
          if s[0:4] == 'http': H.append(s)
```

```
Woh Clion
```

browsers
opening a web page an

canning files

looking for strings between double quotes

parsing URLs for the server location

Web Crawlers

making requests recursivel incremental development, modular design of code

```
def HTTPlinks(url):
   .. .. ..
   Given the URL for the web page,
   returns the list of all HTTP strings.
   .. .. ..
   import urllib
   f = urllib.urlopen(url)
```

```
Web C
```

browsers opening a web page and

canning file

looking for strings between double quotes

parsing URLs for the server location

Web Crawlers

```
def HTTPlinks(url):
   .. .. ..
   Given the URL for the web page,
   returns the list of all HTTP strings.
   .. .. ..
   import urllib
   f = urllib.urlopen(url)
   L = []; b = ''
   while True:
      data = f.read(80)
      if data == '': break
```

```
Wob Clic
```

browsers opening a web page an

canning file

looking for strings between double quotes parsing URLs for the server

location

Web Crawlers

```
def HTTPlinks(url):
   .. .. ..
   Given the URL for the web page,
   returns the list of all HTTP strings.
   .. .. ..
   import urllib
   f = urllib.urlopen(url)
   L = []; b = ''
   while True:
      data = f.read(80)
      if data == '': break
       (L,b) = UpdateQstrings(L,b,data)
      L = HTTPfilter(L)
```

Web C

browsers
opening a web page and

anning file

looking for strings between double quotes parsing URLs for the server

location

Web Crawlers

```
def HTTPlinks(url):
   .. .. ..
   Given the URL for the web page,
   returns the list of all HTTP strings.
   .. .. ..
   import urllib
   f = urllib.urlopen(url)
   L = []; b = ''
   while True:
      data = f.read(80)
      if data == '': break
       (L,b) = UpdateQstrings(L,b,data)
      L = HTTPfilter(L)
   f.close()
   return L
```

An URL consists of 6 parts

protocol://location/path:parameters?query#frag

Given URL u, urlparse.urlparse(u) returns 6-tuple.

```
def ShowLocations(L):
    """
    Shows the locations of the URL in L.
    """
    from urlparse import urlparse
    for h in L:
        p = urlparse(h)
        print p[1]
```

MCS 275 L-31

2 April 2008

Web Clie

browsers opening a web page and

canning files

looking for strings between double quotes

parsing URLs for the server location

Web Crawlers

An URL consists of 6 parts

```
protocol://location/path:parameters?query#frag
```

Given URL u, urlparse.urlparse(u) returns 6-tuple.

```
def ShowLocations(L):
    """
    Shows the locations of the URL in L.
    """
    from urlparse import urlparse
    for h in L:
        p = urlparse(h)
        print p[1]
```

MCS 275 L-31

2 April 2008

Web C

alternatives to web browsers opening a web page and

anning files

ooking for strings between ouble quotes

parsing URLs for the server location

Web Crawlers

Web Clients and Crawlers

MCS 275 L-31 2 April 2008

alternatives to web

opening a web page and copying its content

Scanning files

looking for strings between double quotes parsing URLs for the serve

Mob Crowlers

Web Crawlers

making requests recursively incremental development, modular design of code

Web Clients

alternatives to web browsers opening a web page and copying its content

Scanning files

looking for strings between double quotes parsing URLs for the server location

Web Crawlers

making requests recursively

making requests recursively

opening a web page and copying its content

canning files

looking for strings between double quotes parsing URLs for the serve

Web Crawlers

making requests recursively

incremental development, modular design of code

Scanning HTML files and browsing:

- 1. given a URL, open a web page,
- compute the list of all URLs in the page,
- 3. for all URLs in the list do:
 - 3.1 open the web page defined by location of URL.
 - 3.2 compute the list of all URLs on that page
- → continue recursively, crawling the web

Things to consider:

- 1. remove duplicates from list of URLs,
- 2. do not turn back to pages visited before
- 3. limit the levels of recursion,
- 4. some links will not work.

Similar to finding a path in a maze, but now we are interested in all intermediate nodes along the path.

making requests recursively

Scanning HTML files and browsing:

- given a URL, open a web page,
- compute the list of all URLs in the page,
- 3. for all URLs in the list do:
 - 3.1 open the web page defined by location of URL.
 - 3.2 compute the list of all URLs on that page.
- ightarrow continue recursively, $\emph{crawling}$ the web

Things to consider:

- 1. remove duplicates from list of URLs,
- do not turn back to pages visited before
- 3. limit the levels of recursion,
- 4. some links will not work.

Similar to finding a path in a maze, but now we are interested in all intermediate nodes along the path.

Wah Clients

alternatives to web browsers

opening a web page and copying its content

canning files

double quotes
parsing URLs for the serve

Nob Crowlere

web Crawlers making requests recursively



- given a URL, open a web page,
- compute the list of all URLs in the page,
- 3. for all URLs in the list do:
 - 3.1 open the web page defined by location of URL.
 - 3.2 compute the list of all URLs on that page.
- → continue recursively, crawling the web

Things to consider:

- 1. remove duplicates from list of URLs,
- 2. do not turn back to pages visited before
- 3. limit the levels of recursion,
- 4. some links will not work.

Similar to finding a path in a maze, but now we are interested in all intermediate nodes along the path.

Moh Clionto

alternatives to web browsers

opening a web page and copying its content

anning files

double quotes

parsing URLs for the serve

Mah Cravilara

making requests recursively



making requests recursively

making requests recursively

Scanning HTML files and browsing:

- given a URL, open a web page,
- compute the list of all URLs in the page.
- for all URLs in the list do:
 - 3.1 open the web page defined by location of URL,
 - 3.2 compute the list of all URLs on that page.

- 2. do not turn back to pages visited before.

- given a URL, open a web page,
- compute the list of all URLs in the page.
- for all URLs in the list do:
 - 3.1 open the web page defined by location of URL,
 - 3.2 compute the list of all URLs on that page.
- → continue recursively, crawling the web

- 2. do not turn back to pages visited before.

making requests recursively



- given a URL, open a web page,
- compute the list of all URLs in the page,
- 3. for all URLs in the list do:
 - 3.1 open the web page defined by location of URL,
 - 3.2 compute the list of all URLs on that page.
- → continue recursively, crawling the web

Things to consider:

- remove duplicates from list of URLs,
- do not turn back to pages visited before
- limit the levels of recursion,
- 4. some links will not work.

Similar to finding a path in a maze, but now we are interested in all intermediate nodes along the path.

Male Oliena

alternatives to web browsers

opening a web page and copying its content

canning files

looking for strings between double quotes

cation

Neb Crawlers making requests recursively



- given a URL, open a web page,
- compute the list of all URLs in the page,
- 3. for all URLs in the list do:
 - 3.1 open the web page defined by location of URL,
 - 3.2 compute the list of all URLs on that page.
- ightarrow continue recursively, $\emph{crawling}$ the web

Things to consider:

- remove duplicates from list of URLs,
- 2. do not turn back to pages visited before,
- limit the levels of recursion,
- 4. some links will not work.

Similar to finding a path in a maze, but now we are interested in all intermediate nodes along the path.

Web C

browsers
opening a web page an

copying its content

canning file

looking for strings between double quotes parsing URLs for the server

Web Crawlers

making requests recursively

- 1. given a URL, open a web page,
- 2. compute the list of all URLs in the page,
- 3. for all URLs in the list do:
 - 3.1 open the web page defined by location of URL,
 - 3.2 compute the list of all URLs on that page.
- → continue recursively, crawling the web

Things to consider:

- 1. remove duplicates from list of URLs,
- 2. do not turn back to pages visited before,
- limit the levels of recursion,
- 4. some links will not work.

Similar to finding a path in a maze, but now we are interested in all intermediate nodes along the path.

Web Clier

alternatives to web browsers

opening a web page and copying its content

canning files

looking for strings between double quotes parsing URLs for the server

arsing URLS for the ser

Web Crawlers making requests recursively



making requests recursively

2 April 2008

making requests recursively

- Scanning HTML files and browsing:
 - given a URL, open a web page,
 - compute the list of all URLs in the page.
 - for all URLs in the list do:
 - 3.1 open the web page defined by location of URL,
 - 3.2 compute the list of all URLs on that page.
- → continue recursively, crawling the web

Things to consider:

- remove duplicates from list of URLs,
- do not turn back to pages visited before,
- limit the levels of recursion,
- some links will not work.



- 1. given a URL, open a web page,
- compute the list of all URLs in the page,
- 3. for all URLs in the list do:
 - 3.1 open the web page defined by location of URL,
 - 3.2 compute the list of all URLs on that page.
- ightarrow continue recursively, $\emph{crawling}$ the web

Things to consider:

- remove duplicates from list of URLs,
- 2. do not turn back to pages visited before,
- limit the levels of recursion,
- 4. some links will not work.

Similar to finding a path in a maze, but now we are interested in all intermediate nodes along the path.

...

alternatives to web browsers opening a web page an

canning files

looking for strings between double quotes parsing URLs for the server

making requests recursively



Web Clients and Crawlers

Web Clients

alternatives to web browsers
opening a web page and copying its content

Scanning files

looking for strings between double quotes parsing URLs for the server location

Web Crawlers

making requests recursively incremental development, modular design of code

MCS 275 L-31

2 April 2008

Web Client

browsers opening a web page and

Scanning files

looking for strings between double quotes

Mob Crowler



depth of the recursion tree.

2 April 2008

Lists all locations of web servers that can incremental development. modular design of code

L-31 MCS 275 Wed 2 Apr 2008 : webcrawler.py

Prompts the user for a URL and the maximal

reached starting from the user given URL.

depth of the recursion tree.

use what we have developed so far

Lists all locations of web servers that can

incremental development. modular design of code

from scanquotes import UpdateOstrings from scanhttplinks import HTTPfilter, HTTPlinks

L-31 MCS 275 Wed 2 Apr 2008 : webcrawler.py

Prompts the user for a URL and the maximal

reached starting from the user given URL.

depth of the recursion tree.

2 April 2008

Lists all locations of web servers that can incremental development. modular design of code

from scanquotes import UpdateOstrings from scanhttplinks import HTTPfilter, HTTPlinks

L-31 MCS 275 Wed 2 Apr 2008 : webcrawler.py

Prompts the user for a URL and the maximal

reached starting from the user given URL.

Still left to write:

- 1. management of list of server locations,
- recursive function to crawl the web.

def NewLocations(L,V):

```
.. .. ..
Given the list L of new URLs and the
list of already visited locations,
returns the list of new locations.
locations not yet visited earlier.
11 11 11
```

def NewLocations(L,V):

```
Web Clier
```

alternatives to web prowsers

opening a web page and copying its content

canning file

looking for strings between double quotes

parsing URLs for the s

Web Crawlers

```
.. .. ..
Given the list L of new URLs and the
list of already visited locations,
returns the list of new locations.
locations not yet visited earlier.
11 11 11
from urlparse import urlparse
newL = []
```

def NewLocations(L,V):

..

```
Web Clien
```

alternatives to web browsers

opening a web page and copying its content

canning file

looking for strings between double quotes

parsing URLs for the se

Web Crawlers

incremental development, modular design of code

```
list of already visited locations,
returns the list of new locations.
locations not yet visited earlier.
11 11 11
from urlparse import urlparse
newL = []
for h in L:
   p = urlparse(h)
   loc = p[1]
```

Given the list L of new URLs and the

canning file

looking for strings between double quotes

parsing URLs for the serve

Web Crawlers

king requests recursively

incremental development, modular design of code

```
def NewLocations(L,V):
   .. .. ..
   Given the list L of new URLs and the
   list of already visited locations,
   returns the list of new locations.
   locations not yet visited earlier.
   11 11 11
   from urlparse import urlparse
   newL = []
   for h in L:
      p = urlparse(h)
      loc = p[1]
      if not loc in newL:
          if not loc in V:
```

def NewLocations(L.V):

..

```
Web Client
```

alternatives to web prowsers

opening a web page and copying its content

Scanning file

looking for strings between double quotes

parsing URLs for the

Web Crawlers

incremental development, modular design of code

```
list of already visited locations,
returns the list of new locations.
locations not yet visited earlier.
11 11 11
from urlparse import urlparse
newL = []
for h in L:
   p = urlparse(h)
   loc = p[1]
   if not loc in newL:
      if not loc in V:
         newL.append(loc)
return newl
```

Given the list L of new URLs and the

copying its content

canning file

looking for strings between double quotes

Web Crawlers

making requests recursive incremental development, modular design of code

```
Add an exception handler:
```

```
def HTTPlinks(url):
   .. .. ..
   Given the URL for the web page,
   returns the list of all HTTP strings.
   . . . .
   import urllib
   try:
      print 'opening ' + url + ' ...'
      f = urllib.urlopen(url)
```

looking for strings between double quotes

parsing URLs for the

Web Crawlers

incremental development, modular design of code

Add an exception handler:

```
def HTTPlinks(url):
   .. .. ..
   Given the URL for the web page,
   returns the list of all HTTP strings.
   . . . .
   import urllib
   try:
      print 'opening ' + url + ' ...'
      f = urllib.urlopen(url)
   except:
      print 'opening ' + url + ' failed'
      return []
```

To open a web page we also need to specify the protocol.

We apply urlparse.urlunparse

```
>>> from urlparse import urlunparse
>>> urlunparse(('http','www.python.org',
... '','','',''))
'http://www.python.org'
```

We must provide a 6-tuple as argument ...

MCS 275 L-31

2 April 2008

Web Clien

browsers
opening a web page and

canning files

looking for strings between double quotes

Web Crawlers

making reguests recursively



To open a web page we also need to specify the protocol.

We apply urlparse.urlunparse

```
>>> from urlparse import urlunparse
>>> urlunparse(('http','www.python.org',
... '','','',''))
'http://www.python.org'
```

We must provide a 6-tuple as argument ...

MCS 275 L-31

2 April 2008

Web Client

browsers opening a web page and

canning files

looking for strings between double quotes

Wah Crawlers

ved Clawlers



To open a web page we also need to specify the protocol.

We apply urlparse.urlunparse

```
>>> from urlparse import urlunparse
>>> urlunparse(('http','www.python.org',
... '','','','))
'http://www.python.org'
```

We must provide a 6-tuple as argument ..

MCS 275 L-31

2 April 2008

Web Clie

browsers
opening a web page and

copying its content

anning file

looking for strings between double quotes

. . .

veb Crawlers naking requests recursively

To open a web page we also need to specify the protocol.

We apply urlparse.urlunparse

```
>>> from urlparse import urlunparse
>>> urlunparse(('http','www.python.org',
... '','','','))
'http://www.python.org'
```

We must provide a 6-tuple as argument ...

MCS 275 L-31

2 April 2008

Web C

browsers
opening a web page and

copying its content

canning file

looking for strings between double quotes

Janon

Veb Crawlers making requests recursively

canning file

looking for strings between double quotes

parsing URLs for the s

Web Crawlers

making requests recursive incremental development, modular design of code

```
$ python webcrawler.py
crawling the web ...
Give URL : http://www.uic.edu
give maximal depth : 2
opening http://www.uic.edu ...
opening http://www.w3.org ...
opening http://www.csail.mit.edu ...
opening http://www.ercim.org ...
opening http://jigsaw.w3.org ...
opening http://validator.w3.org ...
opening http://www2008.org ...
opening http://www.bicc.com.cn ...
opening http://www.primelife.eu ...
```

. it takes a while ..

total #locations : 538

canning file

looking for strings between double quotes

location

Web Crawlers

incremental development, modular design of code

```
$ python webcrawler.py
crawling the web ...
Give URL : http://www.uic.edu
give maximal depth: 2
opening http://www.uic.edu ...
opening http://www.w3.org ...
opening http://www.csail.mit.edu ...
opening http://www.ercim.org ...
opening http://jigsaw.w3.org ...
opening http://validator.w3.org ...
opening http://www2008.org ...
opening http://www.bicc.com.cn ...
opening http://www.primelife.eu ...
```

.. it takes a while ..

total #locations : 538

canning file

looking for strings between double quotes

Web Crawlers

making requests recursive incremental development, modular design of code

```
$ python webcrawler.py
crawling the web ...
Give URL : http://www.uic.edu
give maximal depth: 2
opening http://www.uic.edu ...
opening http://www.w3.org ...
opening http://www.csail.mit.edu ...
opening http://www.ercim.org ...
opening http://jigsaw.w3.org ...
opening http://validator.w3.org ...
opening http://www2008.org ...
opening http://www.bicc.com.cn ...
opening http://www.primelife.eu ...
```

.. it takes a while ..

total #locations : 538

Web Clien

browsers
opening a web page and

canning files

looking for strings between double quotes

ation

Neb Crawlers

```
def main():
   .. .. ..
   Prompts the user for a web page,
   and prints all URLs this page refers to.
   . . . .
   print 'crawling the web ...'
   page = raw input('Give URL : ')
   k = input('give maximal depth : ')
```

```
Web Cli
```

browsers
opening a web page and

canning files

looking for strings between double quotes

cation

Web Crawlers

```
def main():
   .. .. ..
   Prompts the user for a web page,
   and prints all URLs this page refers to.
   . . . .
   print 'crawling the web ...'
   page = raw input('Give URL : ')
   k = input('give maximal depth : ')
   L = crawler(page,k,[])
   print 'reachable locations :', L
   print 'total #locations :', len(L)
if name ==" main ": main()
```

Web Clien

alternatives to web

opening a web page and copying its content

canning file

looking for strings between double quotes

parsing URLs for t

Web Crawler

```
def crawler(url,k,V):
   .. .. ..
   Returns the list V updated with the
   list of locations reachable from the
   given url using k steps.
   . . . .
   from urlparse import urlunparse
   L = HTTPlinks(url)
```

```
2 April 2008
```

```
....
```

alternatives to web browsers

opening a web page and copying its content

canning file

looking for strings between double quotes

parsing URLs for t

Web Crawlers

```
def crawler(url,k,V):
   .. .. ..
   Returns the list V updated with the
   list of locations reachable from the
   given url using k steps.
   . . . .
   from urlparse import urlunparse
   L = HTTPlinks(url)
   newL = NewLocations(L,V)
   newV = V + newI
```

```
2 April 2008
```

alternatives to web browsers

opening a web page and copying its content

Scanning file

looking for strings between double quotes

parsing URLs for th

Web Crawlers

```
def crawler(url,k,V):
   .. .. ..
   Returns the list V updated with the
   list of locations reachable from the
   given url using k steps.
   . . . .
   from urlparse import urlunparse
   L = HTTPlinks(url)
   newL = NewLocations(L,V)
   newV = V + newI
   if k == 0:
      return newV
```

def crawler(url,k,V):

```
.. .. ..
Returns the list V updated with the
list of locations reachable from the
given url using k steps.
. . . .
from urlparse import urlunparse
L = HTTPlinks(url)
newL = NewLocations(L,V)
newV = V + newI
if k == 0:
   return newV
else:
   for loc in newL:
      u = urlunparse(('http',loc,'','',''))
      newV = crawler(u,k-1,newV)
   return newV
```

Summary + Assignments

We covered more of chapter 14 in *Making Use of Python*. Assignments:

- Write script to download all .py files from http://www.math.uic.edu/~jan/mcs275/main.html
- Limit the search of the crawler so that it only opens pages within the same domain. For example, if we start at a location ending with edu, we only open pages with locations ending with edu.
- 3. Adjust webcrawler.py to search for a path between two locations. The user is prompted for two URLs. Crawling stops if a path has been found.

Assignments collected on Friday 4 April: #1 of L-26, #2 of L-27, #4 of L-28, #1 of L-29, #2 of L-30.

2 April 2008

Web Clients
alternatives to web
browsers

opening a web page and copying its content

canning files

ooking for strings between double quotes parsing URLs for the server

Web Crawlers
making requests recursivel
incremental development,
modular design of code