Scrape the Web: Strategies for programming websites that don't expect it

Presenter: Asheesh Laroia, @asheeshlaroia
(scrape-pycon@asheesh.org, +1-585-506-8865)

February 18, 2010

Outline

Intro

Meta

Hello

Hello

- You will learn neat tricks

Hello

- You will learn neat tricks
- DO NOT BECOME AN EVIL COMMENT SPAMMER

Hello

- You will learn neat tricks
- DO NOT BECOME AN EVIL COMMENT SPAMMER
- Theory, practice, and iterative development

Hello

- You will learn neat tricks
- DO NOT BECOME AN EVIL COMMENT SPAMMER
- Theory, practice, and iterative development
- Brittle? Sometimes.

Hello

- You will learn neat tricks
- DO NOT BECOME AN EVIL COMMENT SPAMMER
- Theory, practice, and iterative development
- Brittle? Sometimes.
- The comics aren't mine; ask me for references.

Format introduction

Format introduction

- I'll stand up here and talk about things.

Format introduction

- I'll stand up here and talk about things.
- You'll ask me questions.

You know what sucks?

You know what sucks?

- It sucks when everyone's thinking something and nobody's saying it.

You know what sucks?

- It sucks when everyone's thinking something and nobody's saying it.
- If I am incoherent, stop me.

“Only” three hours

“Only” three hours

- Slow me down,

“Only” three hours

- Slow me down,
- or speed me up.

“Only” three hours

- Slow me down,
- or speed me up.
- Do this with your voice or by raising your hand.

“Only” three hours

- Slow me down,
- or speed me up.
- Do this with your voice or by raising your hand.
- Don’t try to do it via Twitter.

What is screen scraping?

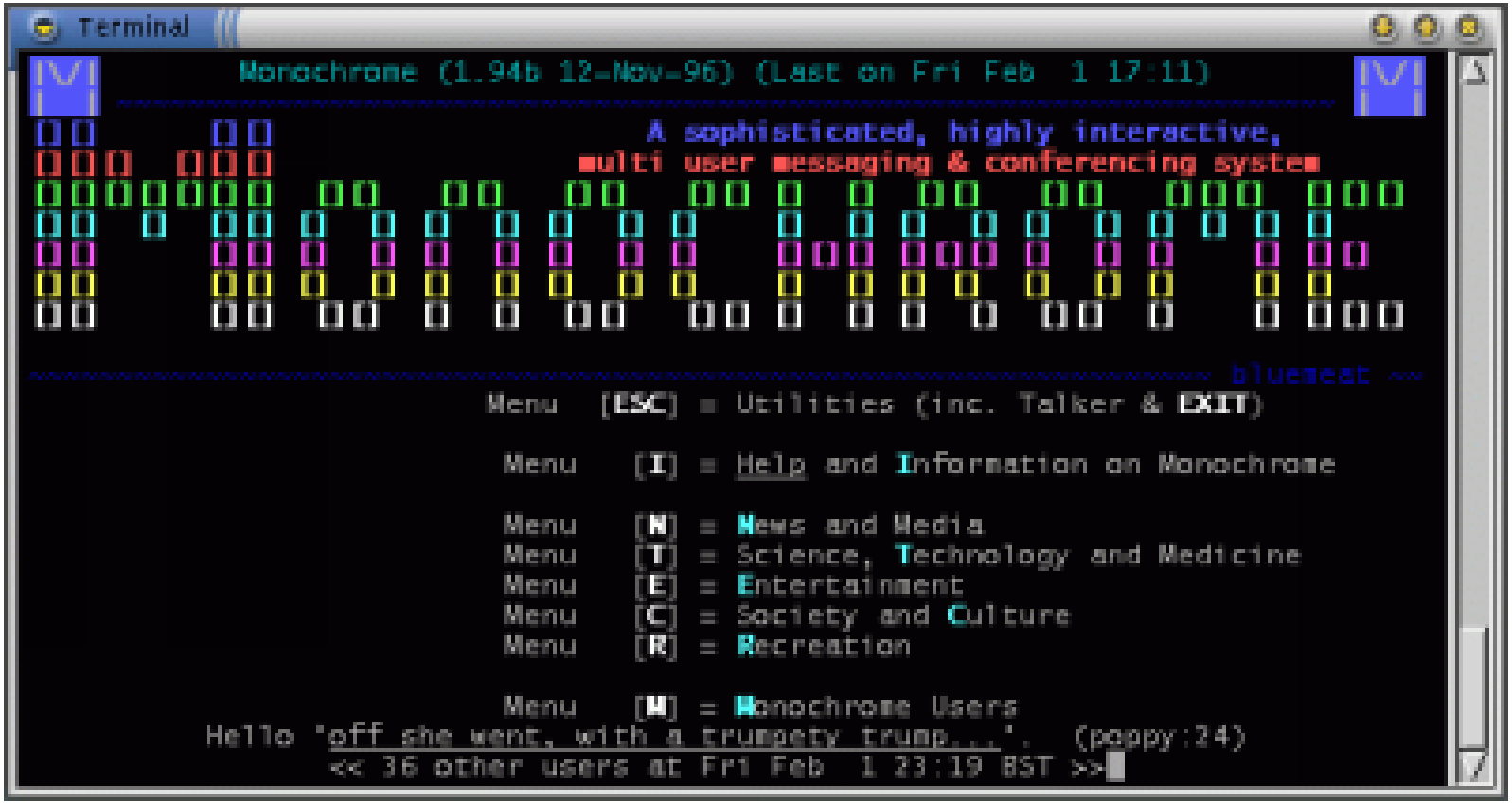
Photo


at the east end of the passage.
There is a small wicker cage discarded nearby.
light lamp
Your lamp is now on.
look
You are crawling over cobbles in a low passage. There is a dim light
at the east end of the passage.
There is a small wicker cage discarded nearby.
take cage
OK
w
You are in a debris room filled with stuff washed in from the surface.
A low wide passage with cobbles becomes plugged with mud and debris
here, but an awkward canyon leads upward and west. A note on the wall
says "Magic word XYZZY".
A three foot black rod with a rusty star on an end lies nearby.
digital VT 101

Photo

```
Monochrome (1.101w 07-May-08) (Last on Wed May 14 13:36)
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

New streamlined layout! Easier to use! New files! Extra exclamation marks!

                    Dish some dirt at <MTO> today!
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ archon ~~
                    Menu  [ESC] = Utilities (inc. Talker & EXIT)

You don't use ssh. Booo!  Menu   [I] = Help and Information on Monochrome

     Welcome to           Menu   [N] = News and Media
        the new           Menu   [T] = Science, Technology and Medicine
     version of           Menu   [E] = Entertainment
    Monochrome!           Menu   [C] = Society and Culture
 (version 1.101w)         Menu   [R] = Recreation

                          Menu   [M] = Monochrome Users

      Hello 'SexDrugs&DrumMachinesForAgRaveGeneration'.  (evilandi:4)
                << 22 other users at Sun Jan 11 19:30 BST >>
```

Brittle?


Terminal
Monochrome (1.94b 12-Nov-96) (Last on Fri Feb 1 17:11)
A sophisticated, highly interactive,
multi user messaging & conferencing system
bluemeat
Menu [ESC] = Utilities (inc. Talker & EXIT)
Menu [I] = Help and Information on Monochrome
Menu [N] = News and Media
Menu [T] = Science, Technology and Medicine
Menu [E] = Entertainment
Menu [C] = Society and Culture
Menu [R] = Recreation
Menu [M] = Monochrome Users
Hello 'off she went, with a trumpety trump...'. (poppy:24)
<< 36 other users at Fri Feb 1 23:19 BST >>

Remote procedure call

Remote procedure call

- Every time you press a key, you cause the remote computer to execute code.

Remote procedure call

- Every time you press a key, you cause the remote computer to execute code.
- Every *keypress* causes a *remote procedure call*.

Remote procedure call

- Every time you press a key, you cause the remote computer to execute code.
- Every *keypress* causes a *remote procedure call*.
  - If you understand this, you can document it as an API.

Power

Power

- We get to interact with the raw data.

Power

- We get to interact with the raw data.
- We could write our own interface.

Power

- We get to interact with the raw data.
- We could write our own interface.
- We get to programmatically interact with a system that only expect humans at the door.

Independence

Independence

- Design choices and restrictions fall away.

Power, too much

Power, too much

- WE CAN SEND SPAM!

Power, too much

- WE CAN SEND SPAM!
- Don't do that.

Outline

Programming the web

Say

The Web

The Web

- It's the twenty-first century.

The Web

- It's the twenty-first century.
- The Web is a massive, mostly-unrestricted remote procedure call system.

Mac OS “say”

Mac OS “say”

- I’m not hip enough to have “say”

Mac OS “say”

- I’m not hip enough to have “say”
- but I do have the Web

Cepstral demo

Curry

Delicious

Curry on the web

http://mehfilindian.com/LunchMenuTakeOut.htm

Beneath the covers...

Beneath the covers...

- FrontPage 6.0 is from 2003

Beneath the covers...

- FrontPage 6.0 is from 2003
- Some really ugly HTML...

Beneath the covers...

- FrontPage 6.0 is from 2003
- Some really ugly HTML...
- I like to call this 1998-style HTML

The easy way

examples/curry/trivial.py

The easy way

examples/curry/trivial.py

- urllib2.urlopen() gives you a file descriptor

The easy way

examples/curry/trivial.py

- urllib2.urlopen() gives you a file descriptor
- Now you can read() it... (and you get a big ol' byte string)

The easy way

examples/curry/trivial.py

- urllib2.urlopen() gives you a file descriptor
- Now you can read() it... (and you get a big ol' byte string)
- Test its contents for squash, and you're done.

The Web and standards

The Web and standards

- We don't have to resort to visual screen scraping.

The Web and standards

- We don't have to resort to visual screen scraping.
- The web has a standard data format for marking up page content.

The Web and standards

- We don't have to resort to visual screen scraping.
- The web has a standard data format for marking up page content.
- What is it called?

XHTML and HTML

XHTML and HTML

- It's 2010.

XHTML and HTML

- It's 2010.
- Surely XHTML has won by now.

“Extract some information”

“Extract some information”

- HTML

“Extract some information”

- HTML
- vs. XHTML (2000)

“Extract some information”

- HTML
- vs. XHTML (2000)
- Both are trees of tags; both can be visualized in FireBug.

“Extract some information”

- HTML
- vs. XHTML (2000)
- Both are trees of tags; both can be visualized in FireBug.
- ...did XHTML win?

Outline

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?
  - 16.5K

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?
  - 16.5K
- HTML to XHTML ratio?

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?
  - 16.5K
- HTML to XHTML ratio?
  - 2:1

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?
  - 16.5K
- HTML to XHTML ratio?
  - 2:1
- Transitional vs. Strict/Frameset:

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?
  - 16.5K
- HTML to XHTML ratio?
  - 2:1
- Transitional vs. Strict/Frameset:
  - 10:1

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?
  - 16.5K
- HTML to XHTML ratio?
  - 2:1
- Transitional vs. Strict/Frameset:
  - 10:1
- How many in "Quirks" mode?

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?
  - 16.5K
- HTML to XHTML ratio?
  - 2:1
- Transitional vs. Strict/Frameset:
  - 10:1
- How many in "Quirks" mode?
  - 85%

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?
  - 16.5K
- HTML to XHTML ratio?
  - 2:1
- Transitional vs. Strict/Frameset:
  - 10:1
- How many in "Quirks" mode?
  - 85%
- What's more popular? TITLE or BODY?

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?
  - 16.5K
- HTML to XHTML ratio?
  - 2:1
- Transitional vs. Strict/Frameset:
  - 10:1
- How many in "Quirks" mode?
  - 85%
- What's more popular? TITLE or BODY?
  - TITLE

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?
  - 16.5K
- HTML to XHTML ratio?
  - 2:1
- Transitional vs. Strict/Frameset:
  - 10:1
- How many in "Quirks" mode?
  - 85%
- What's more popular? TITLE or BODY?
  - TITLE
- What percent validate in general?

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?
  - 16.5K
- HTML to XHTML ratio?
  - 2:1
- Transitional vs. Strict/Frameset:
  - 10:1
- How many in "Quirks" mode?
  - 85%
- What's more popular? TITLE or BODY?
  - TITLE
- What percent validate in general?
  - ca. 4.13%

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?
  - 16.5K
- HTML to XHTML ratio?
  - 2:1
- Transitional vs. Strict/Frameset:
  - 10:1
- How many in "Quirks" mode?
  - 85%
- What's more popular? TITLE or BODY?
  - TITLE
- What percent validate in general?
  - ca. 4.13%
- What percent of web pages that have validation badges validate?

Stats pop quiz

(Stats from the MAMA survey published by Opera <http://dev.opera.com/articles/view/mama-key-findings/>.)

- Average page size?
  - 16.5K
- HTML to XHTML ratio?
  - 2:1
- Transitional vs. Strict/Frameset:
  - 10:1
- How many in "Quirks" mode?
  - 85%
- What's more popular? TITLE or BODY?
  - TITLE
- What percent validate in general?
  - ca. 4.13%
- What percent of web pages that have validation badges validate?
  - ca. $\frac{1}{2}$

Outline

The web: Round one

Parsing considerations

A showcase of some of your options

A showcase of some of your options

- An example of valid HTML (written by hand) (examples/parsing/)

A showcase of some of your options

- An example of valid HTML (written by hand) (examples/parsing/)
  - Parsed with HTMLParser

A showcase of some of your options

- An example of valid HTML (written by hand) (examples/parsing/)
  - Parsed with HTMLParser
- An example of invalid HTML (cooked by hand) (examples/parsing/invalid-html/)

A showcase of some of your options

- An example of valid HTML (written by hand) (examples/parsing/)
  - Parsed with HTMLParser
- An example of invalid HTML (cooked by hand) (examples/parsing/invalid-html/)
  - Parsed with HTMLParser

A showcase of some of your options

- An example of valid HTML (written by hand) (examples/parsing/)
  - Parsed with HTMLParser
- An example of invalid HTML (cooked by hand) (examples/parsing/invalid-html/)
  - Parsed with HTMLParser
- An example of valid XHTML (written by hand) (examples/parsing/valid-xhtml/)

A showcase of some of your options

- An example of valid HTML (written by hand) (examples/parsing/)
  - Parsed with HTMLParser
- An example of invalid HTML (cooked by hand) (examples/parsing/invalid-html/)
  - Parsed with HTMLParser
- An example of valid XHTML (written by hand) (examples/parsing/valid-xhtml/)
  - Parsed with xml.dom.minidom

A showcase of some of your options

- An example of valid HTML (written by hand) (examples/parsing/)
  - Parsed with HTMLParser
- An example of invalid HTML (cooked by hand) (examples/parsing/invalid-html/)
  - Parsed with HTMLParser
- An example of valid XHTML (written by hand) (examples/parsing/valid-xhtml/)
  - Parsed with xml.dom.minidom
  - Parsed with HTMLParser

A showcase of some of your options

- An example of valid HTML (written by hand) (examples/parsing/)
  - Parsed with HTMLParser
- An example of invalid HTML (cooked by hand) (examples/parsing/invalid-html/)
  - Parsed with HTMLParser
- An example of valid XHTML (written by hand) (examples/parsing/valid-xhtml/)
  - Parsed with xml.dom.minidom
  - Parsed with HTMLParser
- An example of invalid XHTML <http://www.washington.edu/accessit/webdesign/student/unit5/inv (examples/parsing/invalid-xhtml/)

A showcase of some of your options

- An example of valid HTML (written by hand) (examples/parsing/)
  - Parsed with HTMLParser
- An example of invalid HTML (cooked by hand) (examples/parsing/invalid-html/)
  - Parsed with HTMLParser
- An example of valid XHTML (written by hand) (examples/parsing/valid-xhtml/)
  - Parsed with xml.dom.minidom
  - Parsed with HTMLParser
- An example of invalid XHTML <http://www.washington.edu/accessit/webdesign/student/unit5/inv (examples/parsing/invalid-xhtml/)
  - in Firefox

A showcase of some of your options

- An example of valid HTML (written by hand) (examples/parsing/)
  - Parsed with HTMLParser
- An example of invalid HTML (cooked by hand) (examples/parsing/invalid-html/)
  - Parsed with HTMLParser
- An example of valid XHTML (written by hand) (examples/parsing/valid-xhtml/)
  - Parsed with xml.dom.minidom
  - Parsed with HTMLParser
- An example of invalid XHTML <http://www.washington.edu/accessit/webdesign/student/unit5/inv (examples/parsing/invalid-xhtml/)
  - in Firefox
  - In xml.dom.minidom

A showcase of some of your options

- An example of valid HTML (written by hand) (examples/parsing/)
  - Parsed with HTMLParser
- An example of invalid HTML (cooked by hand) (examples/parsing/invalid-html/)
  - Parsed with HTMLParser
- An example of valid XHTML (written by hand) (examples/parsing/valid-xhtml/)
  - Parsed with xml.dom.minidom
  - Parsed with HTMLParser
- An example of invalid XHTML <http://www.washington.edu/accessit/webdesign/student/unit5/inv (examples/parsing/invalid-xhtml/)
  - in Firefox
  - In xml.dom.minidom
  - in HTMLParser

A showcase of some of your options

- An example of valid HTML (written by hand) (examples/parsing/)
  - Parsed with HTMLParser
- An example of invalid HTML (cooked by hand) (examples/parsing/invalid-html/)
  - Parsed with HTMLParser
- An example of valid XHTML (written by hand) (examples/parsing/valid-xhtml/)
  - Parsed with xml.dom.minidom
  - Parsed with HTMLParser
- An example of invalid XHTML <http://www.washington.edu/accessit/webdesign/student/unit5/inv (examples/parsing/invalid-xhtml/)
  - in Firefox
  - In xml.dom.minidom
  - in HTMLParser
- If web HTML is not always parseable, we need a different approach

Other ways to get information out of web pages?

Other ways to get information out of web pages?

- "squash" in page_contents.lower()

Other ways to get information out of web pages?

- “squash” in page_contents.lower()
- re.search(“squash”, page_contents, re.IGNORECASE)

Inspirational quote: JWZ

Some people, when confronted with a problem, think "I know, I'll use regular expressions." Now they have two problems.
– Jamie Zawinski

What's wrong with regular expressions for scraping

What's wrong with regular expressions for scraping

- <a href="/whatever/">

What's wrong with regular expressions for scraping

- <a href="/whatever/">
- <a href='whatever'>

What's wrong with regular expressions for scraping

- <a href="/whatever/">
- <a href='whatever'>
- <a href=‘whatever”>

What's wrong with regular expressions for scraping

- <a href="/whatever/">
- <a href='whatever'>
- <a href=‘whatever”>
- Okay for “Reviews 1-10 of 430”

What's wrong with regular expressions for scraping

- <a href="/whatever/">
- <a href='whatever'>
- <a href=‘whatever”>
- Okay for “Reviews 1-10 of 430”
- Kodos: Regular expression GUI (since redemo.py seems unmaintained)

Inspirational quote: Jon Postel

Robustness principle: "Be conservative in what you do, be liberal in what you accept from others."
– Jon Postel, *Transmission Control Protocol*, RFC 793

Inspirational quote: Leonard Richardson

"You didn't write that awful page. You're just trying to get some data out of it. Right now, you don't really care what HTML is supposed to look like."
– Leonard Richardson, author of BeautifulSoup

Back to curry

New goal for curry: Objectify

Map the menu to Python objects

New goal for curry: Objectify

Map the menu to Python objects

- play with the source in BeautifulSoup

New goal for curry: Objectify

Map the menu to Python objects

- play with the source in BeautifulSoup
- ...this is a text processing problem, not tag processing.

Model the data

examples/curry/menu.py
class Entree:

Model the data

examples/curry/menu.py
class Entree:

- index

Model the data

examples/curry/menu.py
class Entree:

- index
- name

Model the data

examples/curry/menu.py
class Entree:

- index
- name
- description

Model the data

examples/curry/menu.py
class Entree:

- index
- name
- description
- long_winded_description

Model the data

examples/curry/menu.py
class Entree:

- index
- name
- description
- long_winded_description
- price

Mini-lesson

Mini-lesson

- hand-written pages vs.

Mini-lesson

- hand-written pages vs.
- machine-written pages

New goal: Scrape Yahoo! finance

New goal: Scrape Yahoo! finance

- examples/tree-builders/beautifulsoup_yfinance.py

We're done!

Right?

Trees of tags

What defines how HTML gets parsed?

Web browsers

Surfing tag trees in FireBug

Surfing tag trees in FireBug

- Or Opera Dragonfly

Surfing tag trees in FireBug

- Or Opera Dragonfly
- Or Chrome's Inspector

Parsing trees and finding elements

Early history

Early history

- 1998: HTML::TokeParser for Perl

Early history

- 1998: HTML::TokeParser for Perl
  - $p->get_tag("title")

Early history

- 1998: HTML::TokeParser for Perl
  - $p->get_tag("title")
- 1999: W3C XPath standard

Early history

- 1998: HTML::TokeParser for Perl
  - $p->get_tag("title")
- 1999: W3C XPath standard
  - xmlDoc.selectNodes("//title")

Early history

- 1998: HTML::TokeParser for Perl
  - $p->get_tag("title")
- 1999: W3C XPath standard
  - xmlDoc.selectNodes("//title")
- 2004: BeautifulSoup for Python, Release 1.0, "So rich and green"

Early history

- 1998: HTML::TokeParser for Perl
  - $p->get_tag("title")
- 1999: W3C XPath standard
  - xmlDoc.selectNodes("//title")
- 2004: BeautifulSoup for Python, Release 1.0, "So rich and green"
  - soup("title")

Early history

- 1998: HTML::TokeParser for Perl
  - $p->get_tag("title")
- 1999: W3C XPath standard
  - xmlDoc.selectNodes("//title")
- 2004: BeautifulSoup for Python, Release 1.0, "So rich and green"
  - soup("title")
- 2006: scrAPI for Ruby

Early history

- 1998: HTML::TokeParser for Perl
  - $p->get_tag("title")
- 1999: W3C XPath standard
  - xmlDoc.selectNodes("//title")
- 2004: BeautifulSoup for Python, Release 1.0, "So rich and green"
  - soup("title")
- 2006: scrAPI for Ruby
  - CSS Selectors...

Early history

- 1998: HTML::TokeParser for Perl
  - $p->get_tag("title")
- 1999: W3C XPath standard
  - xmlDoc.selectNodes("//title")
- 2004: BeautifulSoup for Python, Release 1.0, "So rich and green"
  - soup("title")
- 2006: scrAPI for Ruby
  - CSS Selectors...
  - title

Early history

- 1998: HTML::TokeParser for Perl
  - $p->get_tag("title")
- 1999: W3C XPath standard
  - xmlDoc.selectNodes("//title")
- 2004: BeautifulSoup for Python, Release 1.0, "So rich and green"
  - soup("title")
- 2006: scrAPI for Ruby
  - CSS Selectors...
  - title
  - span.title

Recent history

Recent history

- 2007: lxml.html improved, publicized by Ian Bicking

Recent history

- 2007: lxml.html improved, publicized by Ian Bicking
  - CSS selectors for Pythonistas

Recent history

- 2007: lxml.html improved, publicized by Ian Bicking
  - CSS selectors for Pythonistas
- 2007: html5lib: Parse web pages like a browser

Recent history

- 2007: lxml.html improved, publicized by Ian Bicking
  - CSS selectors for Pythonistas
- 2007: html5lib: Parse web pages like a browser
- 2008: BeautifulSoup 3.1.0, the end of an era

Recent history

- 2007: lxml.html improved, publicized by Ian Bicking
  - CSS selectors for Pythonistas
- 2007: html5lib: Parse web pages like a browser
- 2008: BeautifulSoup 3.1.0, the end of an era
- 2010: html5lib deprecates BeautifulSoup

Recent history

- 2007: lxml.html improved, publicized by Ian Bicking
  - CSS selectors for Pythonistas
- 2007: html5lib: Parse web pages like a browser
- 2008: BeautifulSoup 3.1.0, the end of an era
- 2010: html5lib deprecates BeautifulSoup
  - "cannot correctly represent any HTML 5 tree (for lack of namespace support), and cannot represent at all any containing MathML or SVG"

Searching tag trees

Searching tag trees

- BeautifulSoup API (examples/tree-builders/beautifulsoup/search.py)

Searching tag trees

- BeautifulSoup API (examples/tree-builders/beautifulsoup/search.py)
- html5lib creates BeautifulSoup objects (or others) (examples/tree-builders/html5lib/search.py)

Searching tag trees

- BeautifulSoup API (examples/tree-builders/beautifulsoup/search.py)
- html5lib creates BeautifulSoup objects (or others) (examples/tree-builders/html5lib/search.py)
- lxml provides XPath (examples/tree-builders/lxml/search_xpath.py)

Searching tag trees

- BeautifulSoup API (examples/tree-builders/beautifulsoup/search.py)
- html5lib creates BeautifulSoup objects (or others) (examples/tree-builders/html5lib/search.py)
- lxml provides XPath (examples/tree-builders/lxml/search_xpath.py)
- "minimal stable XPath"

Searching tag trees

- BeautifulSoup API (examples/tree-builders/beautifulsoup/search.py)
- html5lib creates BeautifulSoup objects (or others) (examples/tree-builders/html5lib/search.py)
- lxml provides XPath (examples/tree-builders/lxml/search_xpath.py)
- "minimal stable XPath"
- lxml provides CSSSelect (examples/tree-builders/lxml/search_css.py)

Interacting with the web

Basic Yahoo! search (hard-coded)

examples/search/yahoo.py

Basic Google! search (hard-coded)

examples/search/google.py

Basic Google! search (hard-coded)

examples/search/google.py

- Great code, but broken due to ?

Something's wrong...

Outline

The web: HTTP and you

A network trace of an HTTP conversation

User-Agent, and other headers the client sends

Status codes

Status codes

- 2xx: Success

Status codes

- 2xx: Success
- 3xx: Redirection

Status codes

- 2xx: Success
- 3xx: Redirection
- 4xx: Error

Status codes

- 2xx: Success
- 3xx: Redirection
- 4xx: Error
- 402: Payment Required

Status codes

- 2xx: Success
- 3xx: Redirection
- 4xx: Error
- 402: Payment Required
- 404 Not Found

Status codes

- 2xx: Success
- 3xx: Redirection
- 4xx: Error
- 402: Payment Required
- 404 Not Found
- 410 Gone

Status codes

- 2xx: Success
- 3xx: Redirection
- 4xx: Error
- 402: Payment Required
- 404 Not Found
- 410 Gone
- 418 I'm a teapot

HTTP methods

HTTP methods

- GET

HTTP methods

- GET
- POST

HTTP methods

- GET
- POST
- PUT

HTTP methods

- GET
- POST
- PUT
- BREW

Once we set User-Agent, are we just like Firefox?

Once we set User-Agent, are we just like Firefox?

- JavaScript behavior

Once we set User-Agent, are we just like Firefox?

- JavaScript behavior
- Image download behavior

Once we set User-Agent, are we just like Firefox?

- JavaScript behavior
- Image download behavior
- Cookie behavior

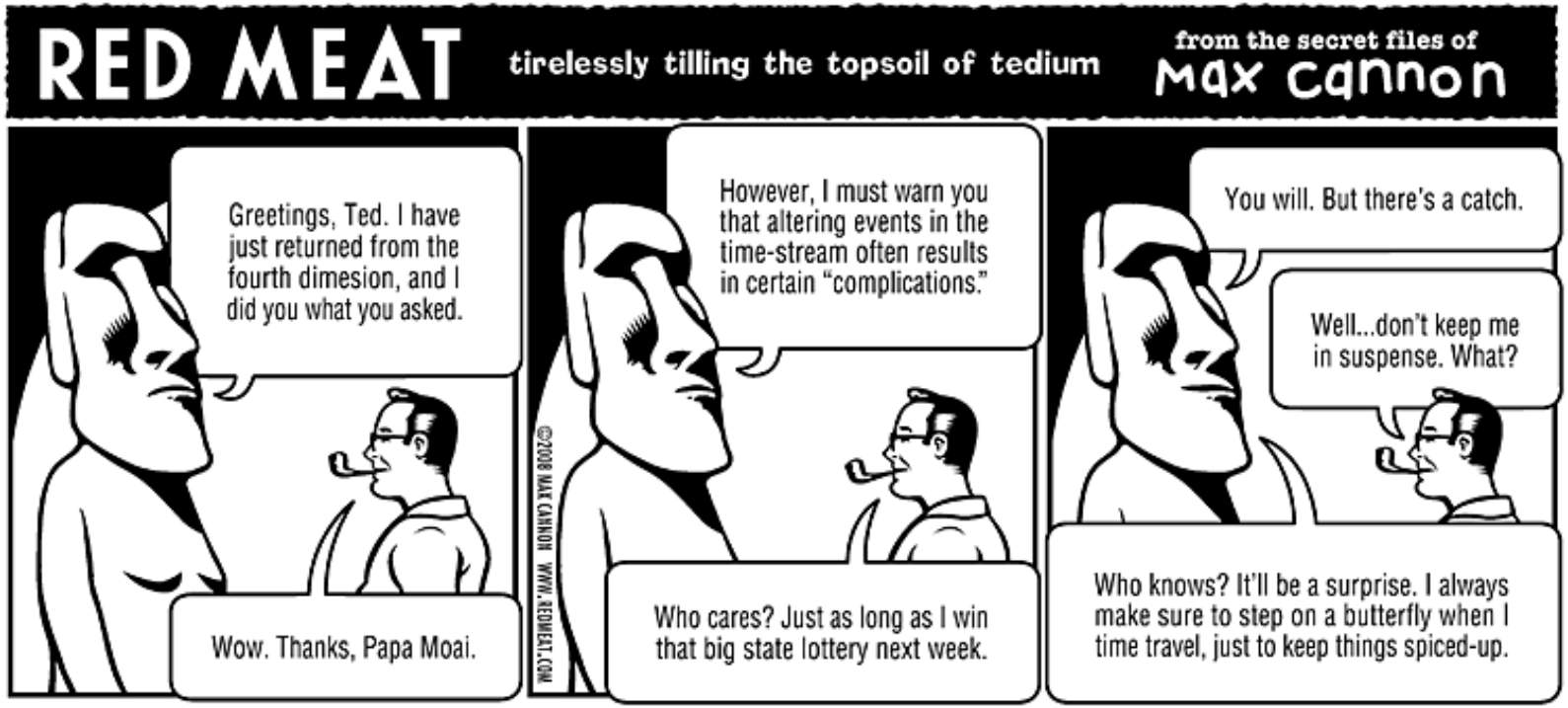
Once we set User-Agent, are we just like Firefox?

- JavaScript behavior
- Image download behavior
- Cookie behavior
- Invalid HTML handling behavior (?)

Once we set User-Agent, are we just like Firefox?

- JavaScript behavior
- Image download behavior
- Cookie behavior
- Invalid HTML handling behavior (?)
- Accept: headers

What if we settle for approximate emulation?



Re-do of Google search with a cooked user-agent

examples/search/urllib2-user-agent/google_as_ie.py

Favorite User-Agent headers

Favorite User-Agent headers

- Mozilla/4.0 (compatible; MSIE 5.0; Windows 98;)

Favorite User-Agent headers

- Mozilla/4.0 (compatible; MSIE 5.0; Windows 98;)
- Mozilla/4.0 (compatible; MSIE 5.0; Windows 98; (compatible;))

Favorite User-Agent headers

- Mozilla/4.0 (compatible; MSIE 5.0; Windows 98;)
- Mozilla/4.0 (compatible; MSIE 5.0; Windows 98; (compatible;))
- I can't believe it's not Googlebot/2.1

HTTP: State via cookies

HTTP: State via cookies

- HTTP implements state on *top* of TCP

robots.txt

robots.txt

- User-agent: *

robots.txt

- User-agent: *
- Disallow: /

robots.txt

- User-agent: *
- Disallow: /
- Allow: /crawlme.html

robots.txt

- User-agent: *
- Disallow: /
- Allow: /crawlme.html
- http://www.robotstxt.org/

robots.txt and detectability

robots.txt and detectability

- “How does the server know you’re a robot?”

robots.txt and detectability

- "How does the server know you're a robot?"
- Well, if you GET /robots.txt...

Filling out more forms: POST and GET

(Be sure to pay attention to the clock; minute 90 is when snack break starts.)

POST: Cepstral Weather demo (by hand)

http://cepstral.com/cgi-bin/demos/weather

Note the URL we POST to

Note the URL we POST to

- from FireBug

Note the data we POST

Note the data we POST

- from FireBug

Write simple Python that also POSTs

examples/cepstral/just_post.py

Pull out the .wav file and play it with mplayer

examples/cepstral/play_wav.py

POST: Cepstral weather demo (via mechanize)

examples/cepstral/just_post_via_mechanize.py

Basic Yahoo! search (via mechanize)

examples/search/yahoo_mechanize.py

Basic Yahoo! search (via mechanize)

examples/search/yahoo_mechanize.py

- Great code, but broken due to robots.txt

Basic Yahoo! search (via mechanize, handle_robots=False)

examples/search/yahoo_mechanize_norobots.py

Basic Google! search (via mechanize, handle_robots=False, changeuser-agent)

examples/search/google_mechanize.py

Cookies

emusic: Log in and verify that we logged in successfully (with cookielib)(optional)

examples/cookies/emusic_login_byhand.py

emusic: Log in and verify that we logged in successfully (with mechanize)

examples/cookies/emusic_login_mechanize.py

emusic: Check how many downloads we have left (with mechanize)

examples/cookies/emusic_check_downloads.py

Now we're done, right?

Whew.

Outline

Recap and philosophy

Recap

We've seen:

Recap

We've seen:

- Loading web pages from the network with urllib2

Recap

We've seen:

- Loading web pages from the network with urllib2
- Parsing web pages (even broken ones)

Recap

We've seen:

- Loading web pages from the network with urllib2
- Parsing web pages (even broken ones)
- Scraping that page into a set of structured Python objects

Recap

We've seen:

- Loading web pages from the network with urllib2
- Parsing web pages (even broken ones)
- Scraping that page into a set of structured Python objects
- HTTP status codes

Recap

We've seen:

- Loading web pages from the network with urllib2
- Parsing web pages (even broken ones)
- Scraping that page into a set of structured Python objects
- HTTP status codes
- Faking the user agent header

Recap

We've seen:

- Loading web pages from the network with urllib2
- Parsing web pages (even broken ones)
- Scraping that page into a set of structured Python objects
- HTTP status codes
- Faking the user agent header
- Submitting forms

Recap

We've seen:

- Loading web pages from the network with urllib2
- Parsing web pages (even broken ones)
- Scraping that page into a set of structured Python objects
- HTTP status codes
- Faking the user agent header
- Submitting forms
- Keeping a session with cookies

“Play nice” on the web

“Play nice” on the web

- Ignore Terms of Service at your own peril

“Play nice” on the web

- Ignore Terms of Service at your own peril
- robots.txt

“Play nice” on the web

- Ignore Terms of Service at your own peril
- robots.txt
- DO NOT BECOME AN EVIL COMMENT SPAMMER

Why scrape the web?

Why scrape the web?

- Anger

Why scrape the web?

- Anger
- Interoperation with unmaintained systems

Why scrape the web?

- Anger
- Interoperation with unmaintained systems
- “Rogue interoperability”

Web APIs

Facebook uses standards!

Facebook uses standards!

- XMPP chat doesn't support:

Facebook uses standards!

- XMPP chat doesn't support:
  - support grouping contacts

Facebook uses standards!

- XMPP chat doesn't support:
  - support grouping contacts
  - status messages

Facebook uses standards!

- XMPP chat doesn't support:
  - support grouping contacts
  - status messages
  - large profile images

Facebook uses standards!

- XMPP chat doesn't support:
  - support grouping contacts
  - status messages
  - large profile images
  - notifications

Facebook uses standards!

- XMPP chat doesn't support:
  - support grouping contacts
  - status messages
  - large profile images
  - notifications
- What's the point?

“Sorry”

“Sorry”

- Ohloh: “Sorry, it is not currently possible to get the list of commits through the API.”

“Sorry”

- Ohloh: “Sorry, it is not currently possible to get the list of commits through the API.”
- Flickr: No way to get a user avatar via the API.

“Sorry”

- Ohloh: “Sorry, it is not currently possible to get the list of commits through the API.”
- Flickr: No way to get a user avatar via the API.
- API keys are evidence of submission.

“Sorry”

- Ohloh: “Sorry, it is not currently possible to get the list of commits through the API.”
- Flickr: No way to get a user avatar via the API.
- API keys are evidence of submission.
- Where is the love?

“Sorry”

- Ohloh: “Sorry, it is not currently possible to get the list of commits through the API.”
- Flickr: No way to get a user avatar via the API.
- API keys are evidence of submission.
- Where is the love?
- Why even play this game?

Outline

Parser redux

Choosing a parser

Choosing a parser

- Performance

Choosing a parser

- Performance
- Ease-of-use

Choosing a parser

- Performance
- Ease-of-use
- Quality

Choosing a parser

- Performance
- Ease-of-use
- Quality
  - Especially as relates to cleaning broken HTML

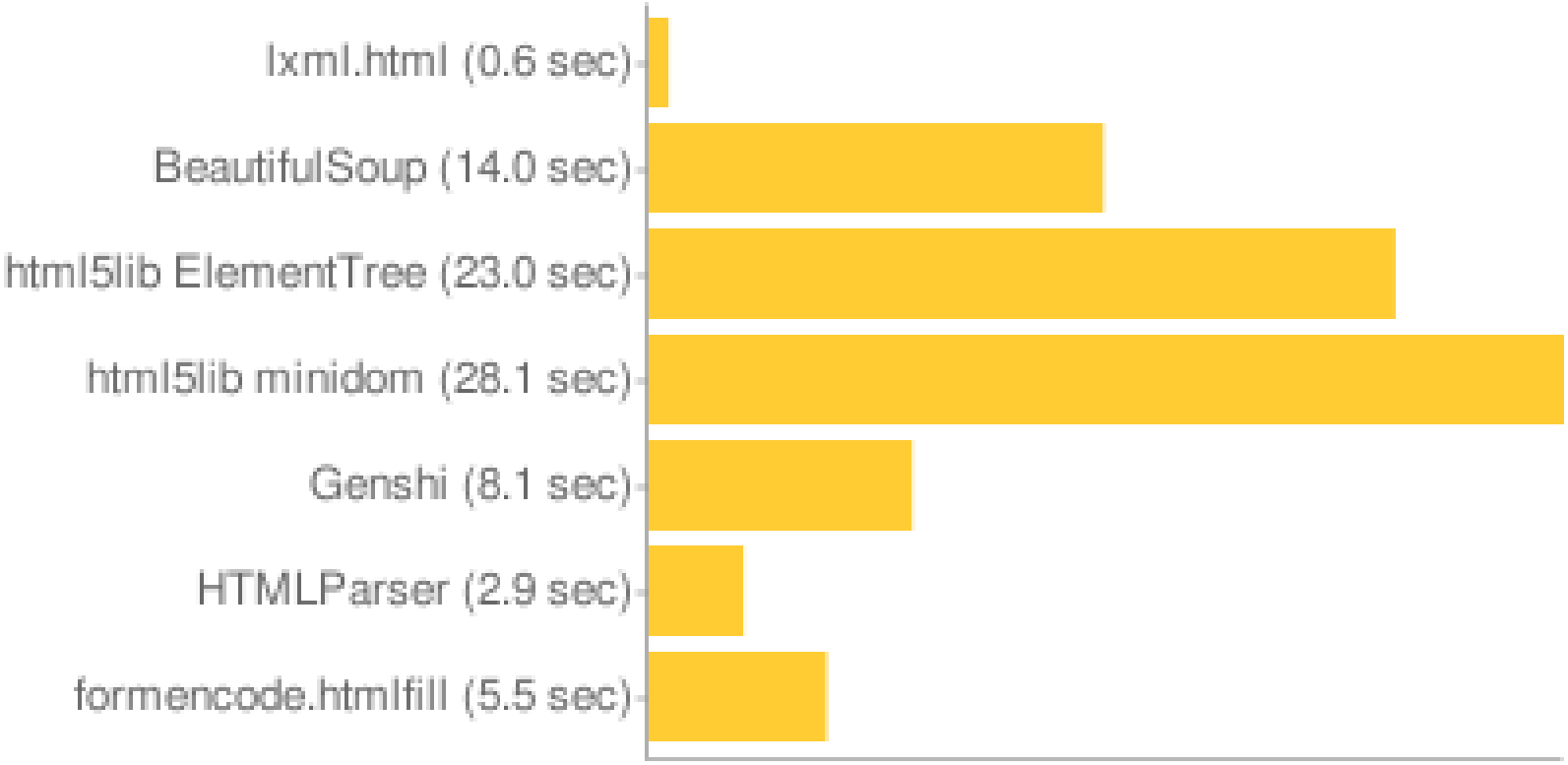
Choosing a parser

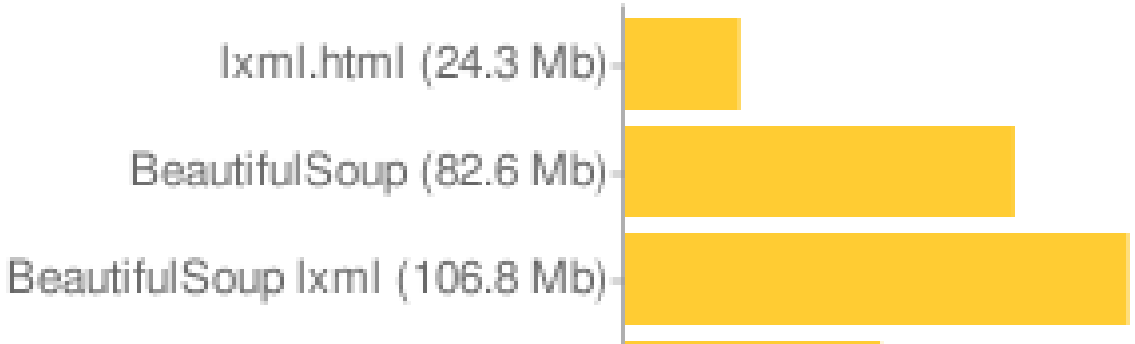
- Performance
- Ease-of-use
- Quality
  - Especially as relates to cleaning broken HTML
  - HTML: 1998-style, or 2003-style?

Benchmarks by Ian Bicking

Benchmarks by Ian Bicking

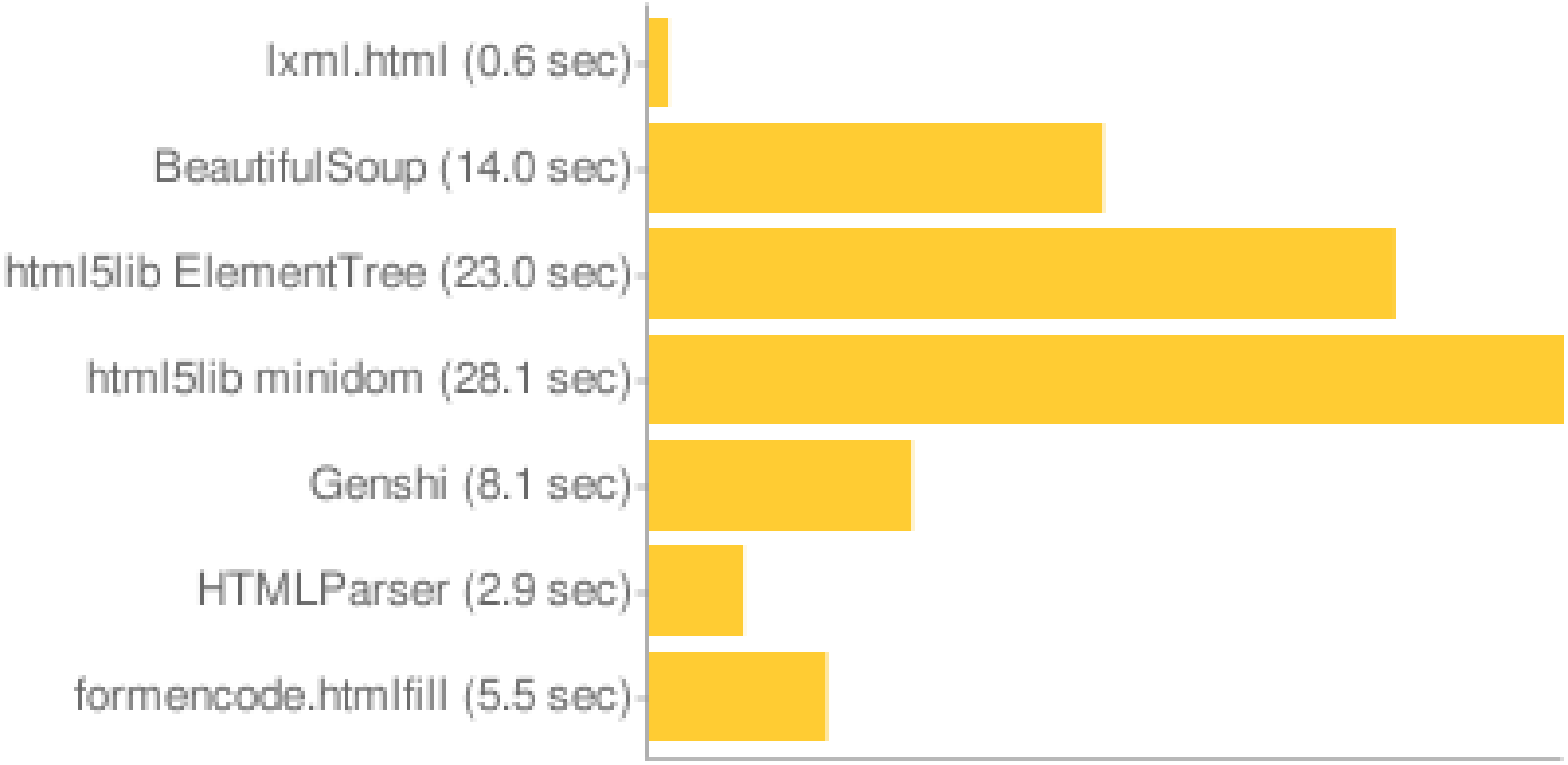
- Benchmarks run by me this morning

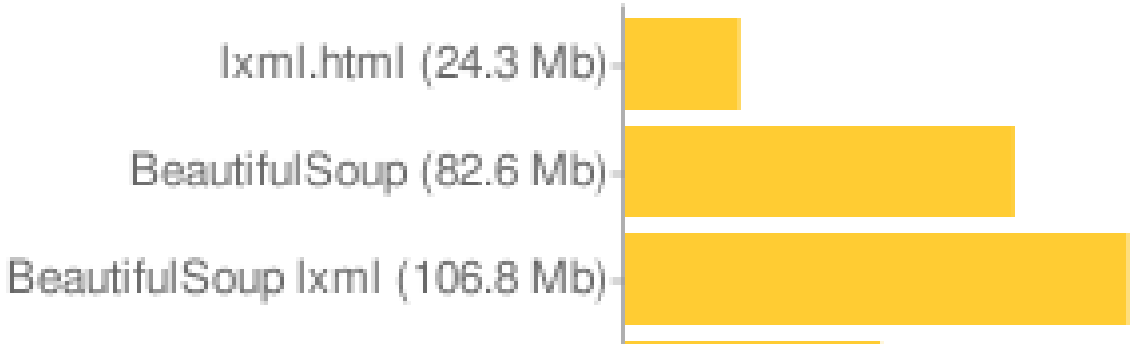




Benchmarks by Ian Bicking

- Benchmarks run by me this morning





Ease of use

Tree fixups

Tree fixups

- lxml $\approx$ BeautifulSoup

Tree fixups

- lxml $\approx$ BeautifulSoup
- lxml $\approx$ html5lib

Tree fixups

- lxml $\approx$ BeautifulSoup
- lxml $\approx$ html5lib
- BeautifulSoup 3.0.7 $>$ BeautifulSoup 3.1.0

A winner

A winner

- lxml!

A winner

- lxml!
- ...?

More about CSS selectors

More about CSS selectors

- FireQuark

More about CSS selectors

- FireQuark
- http://www.imdb.com/title/tt0111161/

More about CSS selectors

- FireQuark
- http://www.imdb.com/title/tt0111161/
  - h5:contains("Release")

More about CSS selectors

- FireQuark
- http://www.imdb.com/title/tt0111161/
  - h5:contains("Release")
- CSS...

Outline

Countermeasures

Easy

Imagine a really stupid bot

Check Referer header

Check Referer header

- mechanize solves this

Extra hidden form fields

Extra hidden form fields

- mechanize solves this

Requiring cookies

Requiring cookies

- mechanize solves this

Countermeasures: hard

Per-IP address query limits

Example: Yahoo web search API

Per-IP address query limits

Example: Yahoo web search API

- Use more IPs

Per-IP address query limits

Example: Yahoo web search API

- Use more IPs
  - Tor, or

Per-IP address query limits

Example: Yahoo web search API

- Use more IPs
  - Tor, or
  - your own machines

Per-IP address query limits

Example: Yahoo web search API

- Use more IPs
  - Tor, or
  - your own machines
- Use SOCKS (plus SSH) to make this easy

CAPTCHAs

Example: Google web search (when you exceed undeclared query limits).

CAPTCHAs

Example: Google web search (when you exceed undeclared query limits).

- uh-oh

JavaScript

Example: “Hash cash” system for avoiding comment spam.

JavaScript

Example: “Hash cash” system for avoiding comment spam.

- uh-oh

Invisible countermeasures

Behavior profiling

Behavior profiling

- Time-based?

Inserting false link visible only to bots

Inserting false link visible only to bots

- “Tarpits”

robots.txt access

robots.txt access

- As soon as you access it, you lose.

Getting around IP address limits

Understand

Understand

- We still have to stay within the limits. We can just take advantage of IPs we do have.

ssh -D

ssh -D

- Borrow the IP of any machine you can log in to

ssh -D

- Borrow the IP of any machine you can log in to
- ssh -D 1080 asheesh.org

socks_monkey

socks_monkey

- SOCKSify Python from within Python

socks_monkey

- SOCKSify Python from within Python
- examples/ip-limits/socks_monkey.py

tsocks

tsocks

- SOCKSify Python via LD_PRELOAD

tsocks

- SOCKSify Python via LD_PRELOAD
- examples/ip-limits/tsocks/

tor

"The onion router"

tor

"The onion router"

- SOCKSify but borrow someone else's IP

tor

"The onion router"

- SOCKSify but borrow someone else's IP
- (play nice...)

Cycling strategies

Cycling strategies

- Drain it dry

Cycling strategies

- Drain it dry
  - easy to implement first

Cycling strategies

- Drain it dry
  - easy to implement first
- Round-robin

Cycling strategies

- Drain it dry
  - easy to implement first
- Round-robin
  - generally preferable

Return to JavaScript: breaking Hash Cash

Detecting its presence

Detecting its presence

- Attempt to submit a comment with JS disabled

Detecting its presence

- Attempt to submit a comment with JS disabled
- Attempt to submit a comment with JS enabled

Detecting its presence

- Attempt to submit a comment with JS disabled
- Attempt to submit a comment with JS enabled
- Trace the second in FireBug

Rewriting the JavaScript as Python

Rewriting the JavaScript as Python

- You may think I'm joking, but this is a common strategy.

DOMForm

DOMForm

- Good news

"DOMForm is a Python module for web scraping and web testing. It knows how to evaluate embedded JavaScript code in response to appropriate events."
– John J. Lee of mechanize

DOMForm

- Good news

"DOMForm is a Python module for web scraping and web testing. It knows how to evaluate embedded JavaScript code in response to appropriate events."
– John J. Lee of mechanize

- Bad news

"This module is unmaintained. Maybe someday..."
Also, it does not execute page-global JavaScript, which is where HashCash is implemented.

python-spidermonkey

python-spidermonkey

- Good news

python-spidermonkey

- Good news
  - "Python/JavaScript bridge module, making use of Mozilla's spidermonkey JavaScript implementation."

python-spidermonkey

- Good news
  - "Python/JavaScript bridge module, making use of Mozilla's spidermonkey JavaScript implementation."
- Bad news

python-spidermonkey

- Good news
  - "Python/JavaScript bridge module, making use of Mozilla's spidermonkey JavaScript implementation."
- Bad news
  - ...do you really want to parse the web page for JavaScript and execute it?

python-spidermonkey

- Good news
  - "Python/JavaScript bridge module, making use of Mozilla's spidermonkey JavaScript implementation."
- Bad news
  - ...do you really want to parse the web page for JavaScript and execute it?
  - examples/javascript/hashcash.py

lck

Ick

- None of this is as clean and automated as mechanize.

“Breaking” CAPTCHAs

Fallback: yourself

Fallback: yourself

- Can always just prompt the operator to figure it out and enter it

Mailinator: “Enter these words to delete the email”

Mailinator: “Enter these words to delete the email”

- Only so many different images

Mailinator: “Enter these words to delete the email”

- Only so many different images
- So build a look-up table

Mailinator: “Enter these words to delete the email”

- Only so many different images
- So build a look-up table
- ...indexed by URL?

Mailinator: “Enter these words to delete the email”

- Only so many different images
- So build a look-up table
- ...indexed by URL?
- ...indexed by image contents?

Mailinator: “Enter these words to delete the email”

- Only so many different images
- So build a look-up table
- ...indexed by URL?
- ...indexed by image contents?
- ...indexed by fuzzy image contents?

(I don’t have a good tool for the last one.)

Audio captchas: “Simple” signal analysis

Audio captchas: “Simple” signal analysis

- Should be doable in pylab/matplotlib with fast Fourier transforms

JavaScript CAPTCHAs (like reCAPTCHA)

JavaScript CAPTCHAs (like reCAPTCHA)

- re-implement CAPTCHA-downloading logic in Python

JavaScript CAPTCHAs (like reCAPTCHA)

- re-implement CAPTCHA-downloading logic in Python
- ...or execute the JavaScript with spidermonkey

...JDownloader

...JDownloader

- "Again, our captcha team did a great job and implemented many new captcha methods."

The website from Hell: US PTO Public PAIR

http://portal.uspto.gov/external/portal/pair

Start with a CAPTCHA

Solve it and move on to...

Solve it and move on to...

- document.write()

The page is invisible.

Outline

Automating the web browser

Selenium Remote Control

examples/seleniumrc/start.py

Selenium IDE

Selenium IDE

- Our friend, XPath

Selenium IDE

- Our friend, XPath
- FireBug

Why don't we just do this all the time?

Why don't we just do this all the time?

- Firefox memory footprint

Why don't we just do this all the time?

- Firefox memory footprint
- Flexibility

Outline

Other tricks

Your parser may fail

Text encoding

Text encoding

- Look in the HTTP header!

Text encoding

- Look in the HTTP header!
- Try UTF-8!

Text encoding

- Look in the HTTP header!
- Try UTF-8!
- ...chardet, if you must

Automatically reverse-engineer templates

Automatically reverse-engineer templates

- templatemaker by Adrian Holovaty

Automatically reverse-engineer templates

- templatemaker by Adrian Holovaty
- everyblock templatemaker

table2dict

table2dict

- Python bug tracker

Outline

Conclusions

Scaling and stability

Scaling and stability

- Choosing reliable queries from web pages

Scaling and stability

- Choosing reliable queries from web pages
- Expanding to more IP addresses when necessary using SSH (and Python 2.6 multiprocessing for a plausible model of how to rotate SOCKS proxies)

Scaling and stability

- Choosing reliable queries from web pages
- Expanding to more IP addresses when necessary using SSH (and Python 2.6 multiprocessing for a plausible model of how to rotate SOCKS proxies)
- Tor (and other proxy considerations)

Scaling and stability

- Choosing reliable queries from web pages
- Expanding to more IP addresses when necessary using SSH (and Python 2.6 multiprocessing for a plausible model of how to rotate SOCKS proxies)
- Tor (and other proxy considerations)
- registrar.py: was seven years stable...

Summary

Summary

- If it's on a web page, you can scrape it out.

Summary

- If it's on a web page, you can scrape it out.
- "Now you have an API for everything."

Future directions

Future directions

- More automation

Future directions

- More automation
- Using cssselect everywhere, geez it's cool

Bonus time

If we have time:

Bonus time

If we have time:

- Greasemonkey demo: scraping in the browser

Bonus time

If we have time:

- Greasemonkey demo: scraping in the browser
- Audience-suggested scraping lab

Bonus time

If we have time:

- Greasemonkey demo: scraping in the browser
- Audience-suggested scraping lab
- Workshopping on queries or regular expressions