Intro To Web Scraping with Ruby

Jake Sorce / Dave Jungst - Devpoint Labs

What is Web Scraping?

Web scraping is an approach for extracting data from websites that don’t have an API.

Pros & Cons of Web Scraping

Pros	Cons
Only a good approach if you do not have an API to work with	Very brittle
Only a good approach if the API you are working with is limited or broken	Against a lot of websites terms of use
	Could be very inaccurate

Well known ways to Web Scrape with Ruby

- Mechanize - https://github.com/sparklemotion/mechanize
- Nokogiri - http://www.nokogiri.org/
- Pismo - https://github.com/peterc/pismo
- Wombat - https://github.com/felipecsl/wombat

Scraping with Mechanize - Setup

- gem install mechanize
  - to install the machanize gem on your machine
- gem install pry
- create a new text file with a .rb extension

Scraping with Mechanize - Setup

- require 'mechanize'
- require ‘pry’

- agent = Mechanize.new

- page = agent.get('http://google.com/')

Scraping with Mechanize - Filling Out Forms

- form = page.forms.first
  - gets the first form on the page and sets it to the variable form

- form['q'] = 'funny cat videos'
  - finds the form input with the name of ‘q’ and fills in the value with ‘funny cat videos’

- page = form.submit
  - submits the form just like a user would do when searching for something by either clicking on the search button or pressing the enter key

- page.search('a').each do |a|
- puts a.text.strip
- end
  - searches the pages for all of the ‘a’ tags and outputs the text of those tags

Scraping with Mechanize - Extracting Data

- Basic
  - page.title
    - get the title of the webpage
  - page.forms.first.buttons.last.value.strip
    - gets the first form on the page
    - gets the last buttons value in that form and strips the whitespace
  - page.at('#hpplink').text.strip
    - gets the div with the id of ‘hpplink’
    - gets the text of that div and strips the whitespace

Scraping with Mechanize - Following Links

- link = page.link_with(text: 'Images')
  - gets the link on the page with the text of “Images” and sets it to the variable link
    - will get the first link on the page if there are multiple links with the text of images

- page = link.click
  - clicks on the link
  - sets your page variable to the new page content

Scraping with Mechanize - Resources

- Finding links
  - page.links
  - page.link_with(:text => 'News')
  - page.links_with(:text => 'News')[1]
  - page.link_with(:href => '/something')
  - page.link_with(:text => 'News', :href => '/something')

Scraping with Mechanize - Resources

- Advanced Form Techniques
  - Select option selection
    - form.field_with(:name => 'list').options[0].select
  - Checkbox selection
    - form.checkbox_with(:name => 'box').check
  - Specific radio selection
    - form.radiobuttons_with(:name => 'box')[1].check

Scraping with Mechanize - Resources

- Scraping Data using Nokogiri
  - Mechanize uses Nokogiri behind the covers. This means that you can call Nokogiri methods on your Mechanize pages.

Scraping with Mechanize - Resources

- Nokogiri Examples:
  - agent = Mechanize.new
  - page = agent.get('http://google.com/')
  - page.search("p.posted")
    - finds all the p tags with the posted class
  - page.css('title')
    - gets the nokogiri element of the title
  - page.css('div')
    - all div elements

Scraping with Mechanize - Resources

- Mechanize examples:
  http://docs.seattlerb.org/mechanize/EXAMPLES_rdoc.html

- Mechanize Github: https://github.com/sparklemotion/mechanize

- Nokogiri:
  http://www.nokogiri.org/
  http://ruby.bastardsbook.com/chapters/html-parsing/

Scraping with Mechanize - Exercises

1. Write a script to count the number of links in a web page (use google.com)

2. Write a script to simulate a user logging in to your local Rails app (http://localhost:3000)

Amazon Follow Along

- Create a new rails project
- Use Mechanize to scrape Amazon
- We will scrape Amazon for the search term cell phones
- We will create a rake task to find all the cellphones from the search response and store them in our database.
- We will then write a view to display all the scraped cellphones

Mini Project

- Use Mechanize to scrape www.southwest.com
  - Southwest is one of the only airlines that isn't searched by the big travel search sites like kayak, hotwire, cheap o' air, ect…
  - Southwest isn't searched because they don't have an open API like most other airlines.

Mini Project Cont...

1. use Mechanize to scrape www.southwest.com
2. scrape southwest for prices on flights leaving from SLC and going to MCO (Orlando) leaving today and coming back a week from now
3. scrape southwest for prices on flights leaving from SLC going to DEN a month from now for a week
4. If you are having SSL errors when add this before your get: mechanize.agent.http.verify_mode = OpenSSL::SSL::VERIFY_NONE
    a. this is far from ideal but a workaround - please see http://stackoverflow.com/questions/8567973/why-does-accessing-a-ssl-site-with-mechanize-on-windows-fail-but-on-mac-work for full details on a better solution

Mini Project Bonus

1. Make a script that will take arguments for the going from and going to airports as well as the dates and return the lowest price that southwest gives back
   a. hint: you'll have to use script args which come in as an array. http://stackoverflow.com/questions/4244611/pass-variables-to-ruby-script-via-command-line