Installation/Setup

- Miniconda: *https://conda.io/miniconda.html*
  - includes Python, conda and small number of other useful packages
- Pycharm(Community Edition): *https://www.jetbrains.com/pycharm*
  - Python IDE(Integrated Development Environment)
- Chrome: https://www.google.com/chrome/browser/desktop/
- FireAnt: http://www.laurenceanthony.net/software/fireant/
- (optional) EmEditor: *https://www.emeditor.com*
  - Text editor for Windows

Building a Web Corpus

언어정보연구원
강범일

How to Build a Corpus

- Transcribing Spoken Data
- Converting Paper to Electronic Documents
  - Keying in manually
  - Scanning using OCR software
- Getting Texts from the Web

Type of Text Data on the Web

- Newspaper Articles
- Social Media Texts
- Product Reviews
- Journal Papers
- True/Scanned PDF
- Full Text Database
- …

Tools to Extract Text from the Web

- There are plenty of web scraping software or tools.
  - http://www.hongkiat.com/blog/web-scraping-tools/

- In the Courpus Linguists Field,
  - WebBootCaT in Sketch Engine
  - Webgetter in WordSmith
  - FireAnt
  - …

But, It's better to develop your own tools for web scraping!

Why Programming?

"Programming for Non-Programmers"

Why Programming?

BAAL Corpus Symposium 2016

'Software and programming in corpus linguistics from novice to expert: to script or not to script? A BAAL SIG Symposium in honour of Adam Kilgarriff'.

Confirmed Speakers

- Jack Grieve (Aston University)
- Susan Hunston (University of Birmingham)
- Ramesh Krishnamurthy (Aston University)
- Garry Plappert (Aston University)
- Paul Rayson (Lancaster University)
- Mike Scott (Aston University)
- Paul Thompson (University of Birmingham)

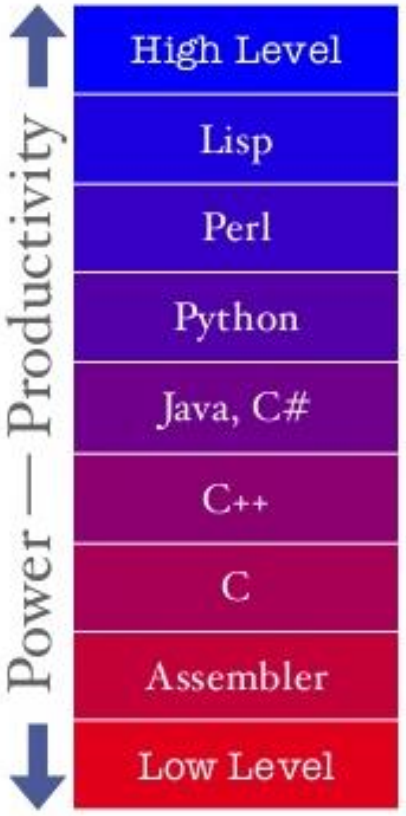
Does a corpus linguist need to be a computer programmer? What are the current range of options for the 'non-scripting' corpus linguist? How can the corpus linguistics community best support those with little or no programming skills? The BAAL Corpus Linguistics SIG is holding a one-day conference on Friday May 6th at Aston University, Birmingham, inhonour of Adam Kilgarriff. The programme will be comprised of a series of invited talks on the theme of software and programming in Corpus Linguistics, and is intended to provoke a lively and informative discussion around thesetopics. We welcome attendees at all levels of practice though it is anticipated that the day will be of most use to those actively engaged with corpus methods in their own work.

Programming Languages

(Visual) FoxPro: FoxPro, Fox Pro, VFP, 4th Dimension/4D: 4D, 4th Dimension, ABAP, ABC: ABC , ActionScript: ActionScript, AS1, AS2, AS3, Ada, Agilent VEE, Algol, Alice: Alice , Angelscript, Apex, APL, Applescript, Arc, AspectJ, Assembly language: Assembly, Assembly language, ATLAS, AutoIt, AutoLISP, Automator, Avenue, Awk: Awk, Mawk, Gawk, Nawk, Bash, Basic: Basic , BBC BASIC, bc, BCPL, BETA: BETA , BlitzMax: BlitzMax, BlitzBasic, Blitz Basic, Boo, Bourne shell: Bourne shell, sh, C shell: Csh, C shell , C#: C#, C-Sharp, C Sharp, CSharp, CSharp.NET, C#.NET, C++, C++/CLI, C-Omega, C: C , Caml, Ceylon, CFML: CFML, ColdFusion, cg: cg , Ch: Ch , CHILL, CIL, CL (OS/400): CL , CLLE, Clarion, Clean: Clean , Clipper, Clojure, CLU, COBOL, Cobra, CoffeeScript, COMAL, Common Lisp, Crystal, cT, Curl, D: D , dlang, Dart, DCL, Delphi/Object Pascal: Delphi, Delphi.NET, DwScript, Object Pascal, Pascal , DiBOL: DBL, Synergy/DE, DIBOL, Dylan, E: E , ECMAScript, EGL, Eiffel, Elixir, Elm, Emacs Lisp: Emacs Lisp, Elips, Erlang, Etoys, Euphoria, EXEC, F#: F#, F-Sharp, FSharp, F Sharp, Factor, Falcon, Fantom, Felix: Felix , Forth, Fortran, Fortress, Gambas, GNU Octave, Go: Go, Golang, Gosu, Groovy: Groovy, GPATH, GSQL, Groovy++, Hack, Haskell, Haxe, Heron, HPL, HyperTalk, Icon: Icon , IDL: IDL , Inform, Informix-4GL, INTERCAL, Io, Ioke, J#, J: J , JADE, Java, JavaFX Script, JavaScript: JavaScript, JS, SSJS, JScript, JScript.NET, Julia, Korn shell: Korn shell, ksh, Kotlin, LabVIEW, Ladder Logic, Lasso, Limbo, Lingo, Lisp, LiveCode: Revolution, LiveCode, Logo: Logo , LotusScript, LPC, Lua, Lustre, M4, MAD: MAD , Magic: Magic , Magik, Malbolge, MANTIS, Maple, MATLAB, Max/MSP, MAXScript, MDX, MEL, Mercury, Miva, ML, Modula-2, Modula-3, Monkey, MOO, Moto, MQL4: MQL4, MQL5, MS-DOS batch, MUMPS, NATURAL, Nemerle, NQC, NSIS, NXT-G, Oberon, Object Rexx, Objective-C: Objective-C, objc, obj-c, OCaml: Objective Caml, OCaml, Occam, OpenCL, OpenEdge ABL: Progress, Progress 4GL, ABL, Advanced Business Language, OpenEdge, OPL, Oxygene, Oz, Paradox, Pascal: Pascal , Perl, PHP, Pike, PILOT: PILOT , PL/I: PL/1, PL/I, PL/SQL, Pliant, PostScript: PostScript, PS, POV-Ray, PowerBasic, PowerScript, PowerShell, Processing: Processing , Programming Without Coding Technology: Programming Without Coding Technology, PWCT, Prolog, Pure Data: Pure Data, PD, PureBasic, Python, Q, R: R , Racket, REBOL, REXX, RPG (OS/400): RPG , RPGLE, ILERPG, RPGIV, RPGIII, RPG400, RPGII, RPG4, Ruby, Rust, S-PLUS: S-PLUS , S: S , SAS, Sather, Scala, Scheme: Scheme , Scratch, sed, Seed7, SIGNAL: SIGNAL , Simula, Simulink, Slate: Slate , Smalltalk, Smarty, SPARK, SPSS, SQR, Squeak, Squirrel, Standard ML: Standard ML, SML, Stata, Suneido, SuperCollider: SuperCollider , Swift, TACL, Tcl: Tcl/Tk, Tcl, Tex, thinBasic, TOM: TOM , Transact-SQL: T-SQL, Transact-SQL, TSQL, TypeScript, Vala/Genie: Vala, Genie, VBScript, Verilog, VHDL, Visual Basic .NET: Visual Basic .NET, VB.NET, Visual Basic.NET, Visual Basic , VB , Visual Basic: Visual Basic , VB , VBA, VB6, WebDNA, Whitespace, Wolfram, X10, xBase, XBase++, Xen, Xojo: REALbasic, Xojo, XPL, XQuery, XSLT, Xtend, yacc, Yorick, Z shell: Z shell, zsh

Why Python?

- Open-Source language
- High-level language
  - Low level languages give you more power, but take very long to write code in.
  - High level languages emphasise developer productivity; get things done fast



Why Python?

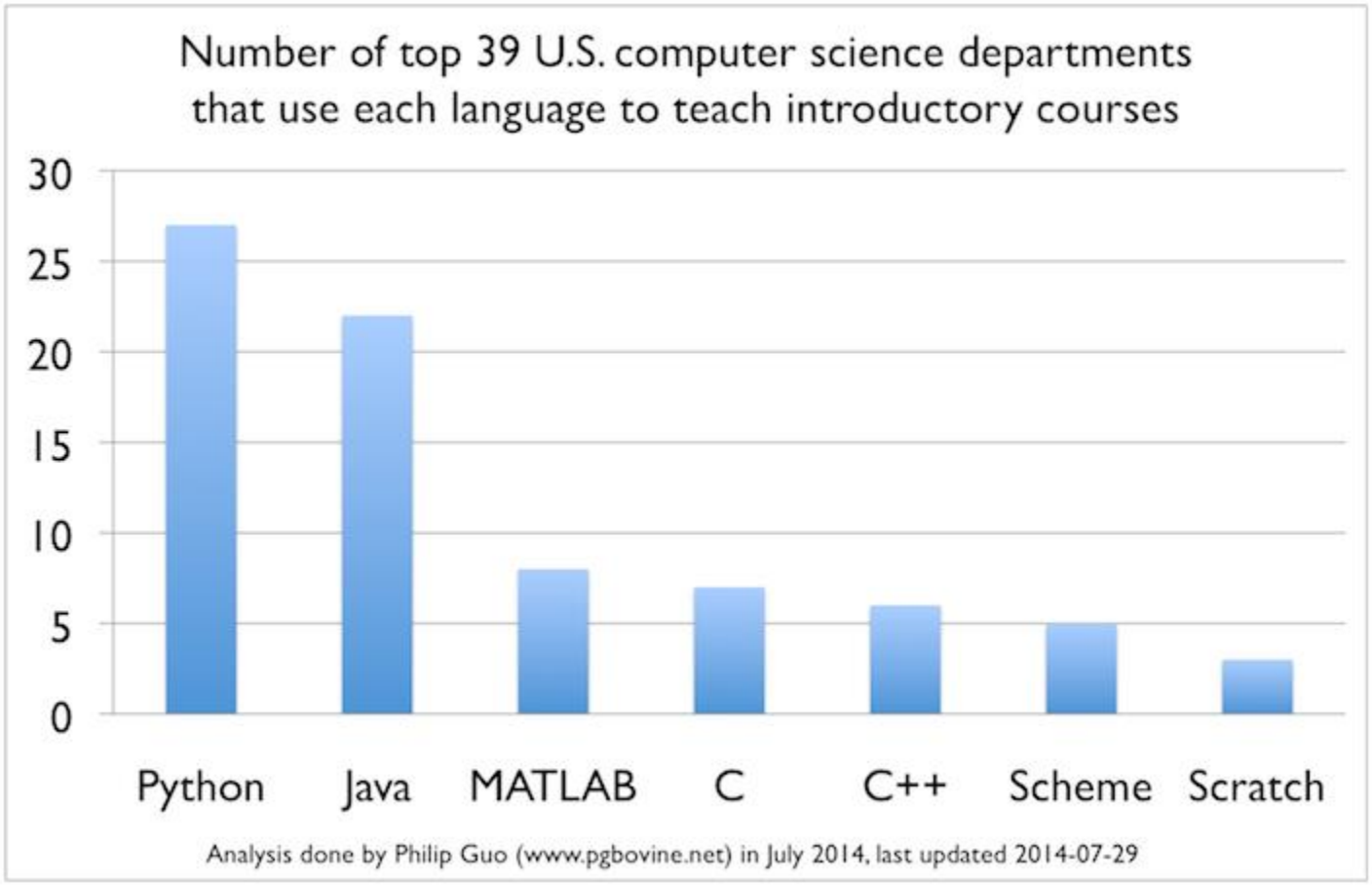
- Easy to learn
  - Clean, clear syntax
  - Very few keywords

** Print "Hello, World!"*

C++	Java	Python

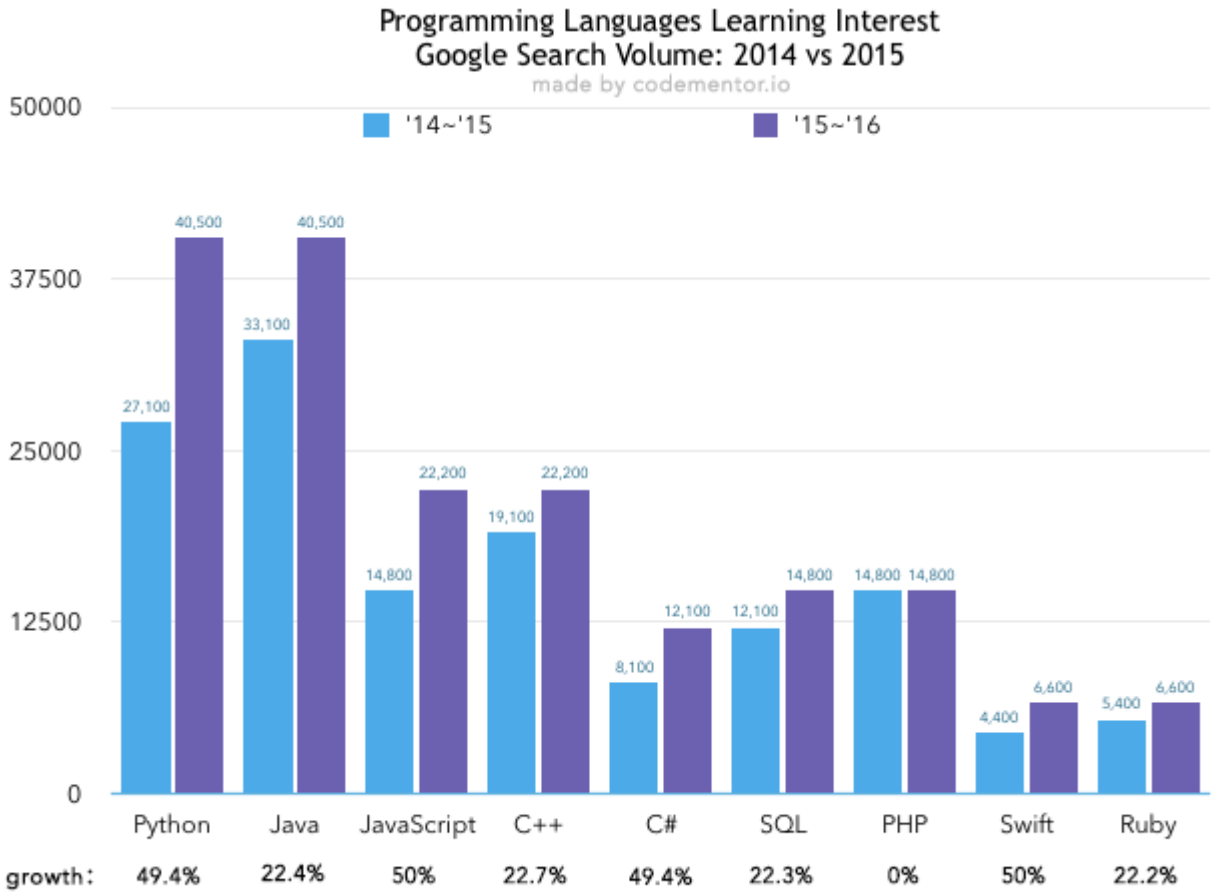
#include<iostream>
using namespace std;
 	
int main()
{
  cout<<"Hello World";
  return 0;
}
	public class Hello
	{
	  public static void main(String argv[])
	  {
	  System.out.println("Hello, World!");
	  }
	}
		print("Hello, World!")

Why Python?



http://cacm.acm.org/blogs/blog-cacm/176450-python-is-now-the-most-popular-introductory-teaching-language-at-top-u-s-universities

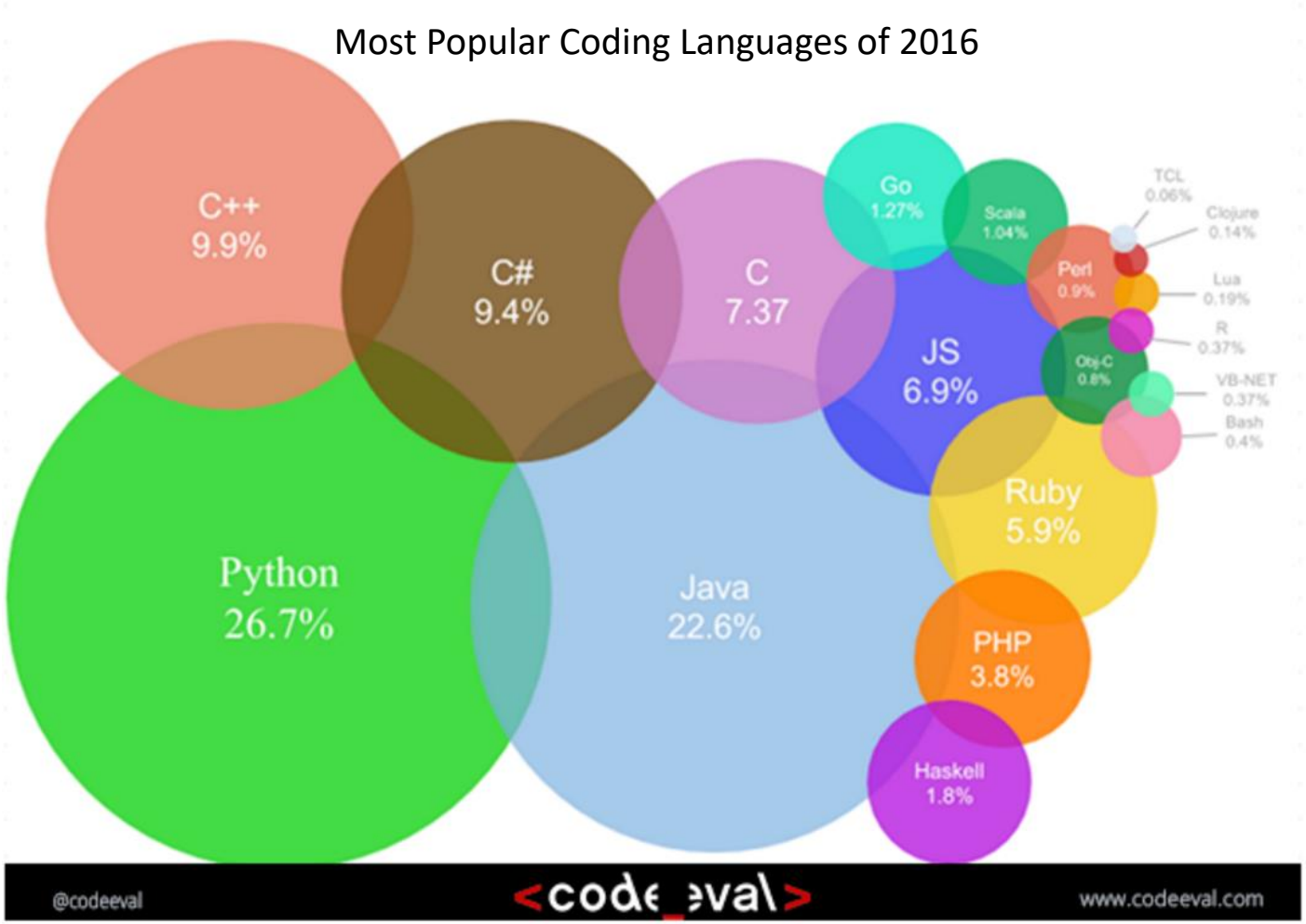
Why Python?



https://www.codementor.io/learn-programming/beginner-programming-language-job-salary-community

Why Python?

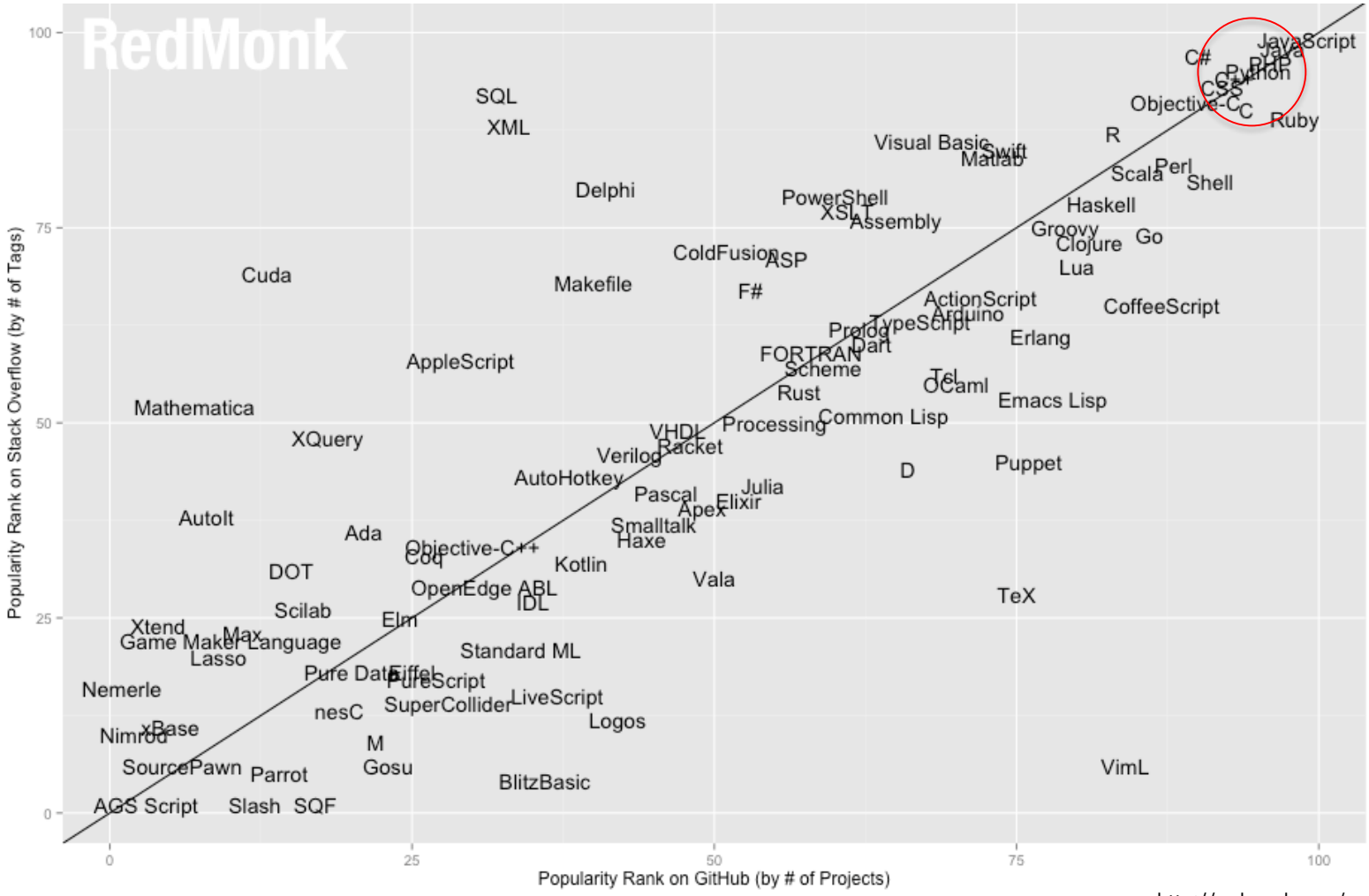
- Widely used



http://www.codeeval.com

Why Python?

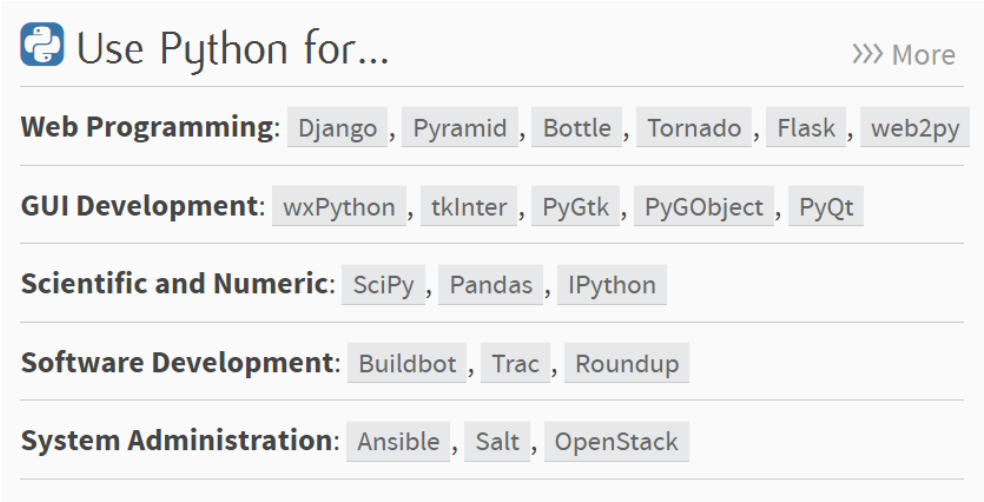
The RedMonk Programming Language Rankings: January 2016



http://redmonk.com/

Why Python?

- Multi-purpose language



www.python.org

- Rich ecosystem
  - various packages
  - vast community

How the Python Works

Python at a glance

Variable

List

Function

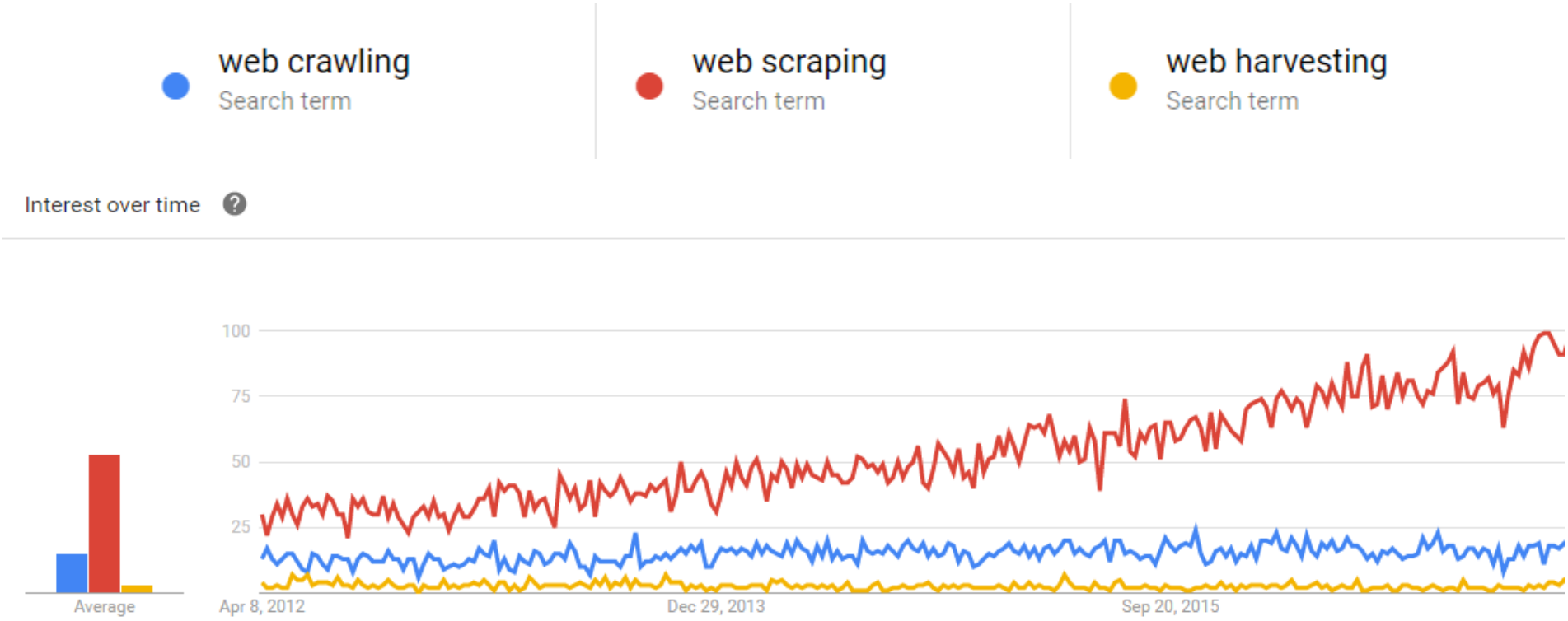
Conditional Statements

Loop Statements

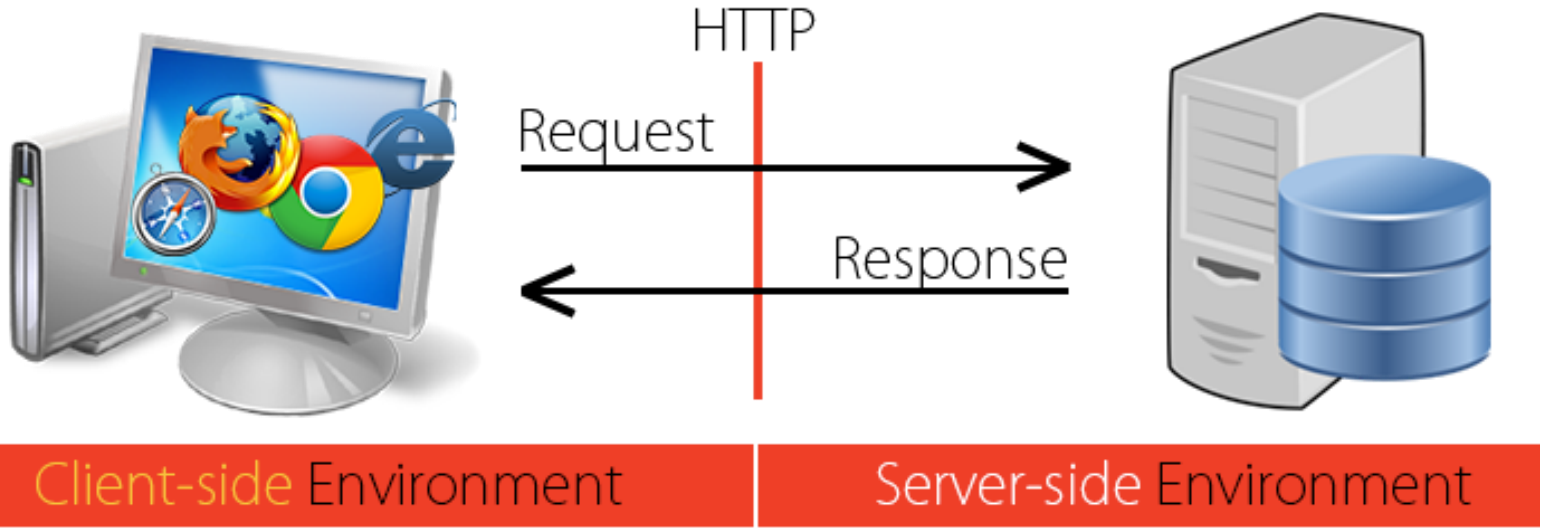
File I/O

Web Scrapping

Crawling? Scraping? Harvesting?



Basic Architecture of a Web Application



URL(Uniform Resource Locator)

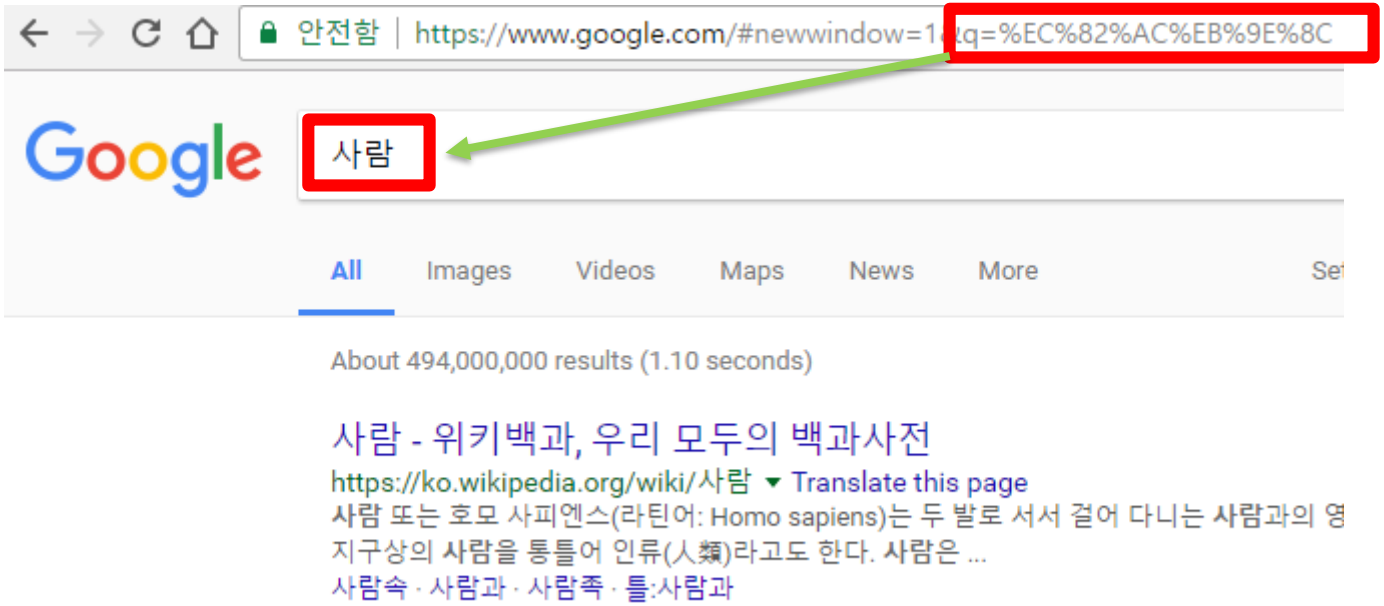
```
scheme:[//[user:password@]host[:port]][/]path[?query][#fragment]
```

https://www.google.com/search?q=human&start=30

- host: domain name or IP
- scheme: http, https, ftp, mailto, file, data
- path: contains data, usually organized in hierarchical form
- query: attribute-value pairs

Query delimiter	Example
Ampersand (&)	key1=value1&key2=value2

URL Encoding



Chrome DevTools

- A set of web authoring and debugging tools built into Google Chrome

- Select More Tools > Developer Tools from the Chrome Menu or press F12

- “Network” Panel
  - provides insights into resources that are requested and downloaded over the network in real time

HOW to parse HTML in Python

- BeautifulSoup
  - Python Library for pulling data out of HTML and XML files.
- Regular Expression

HTML

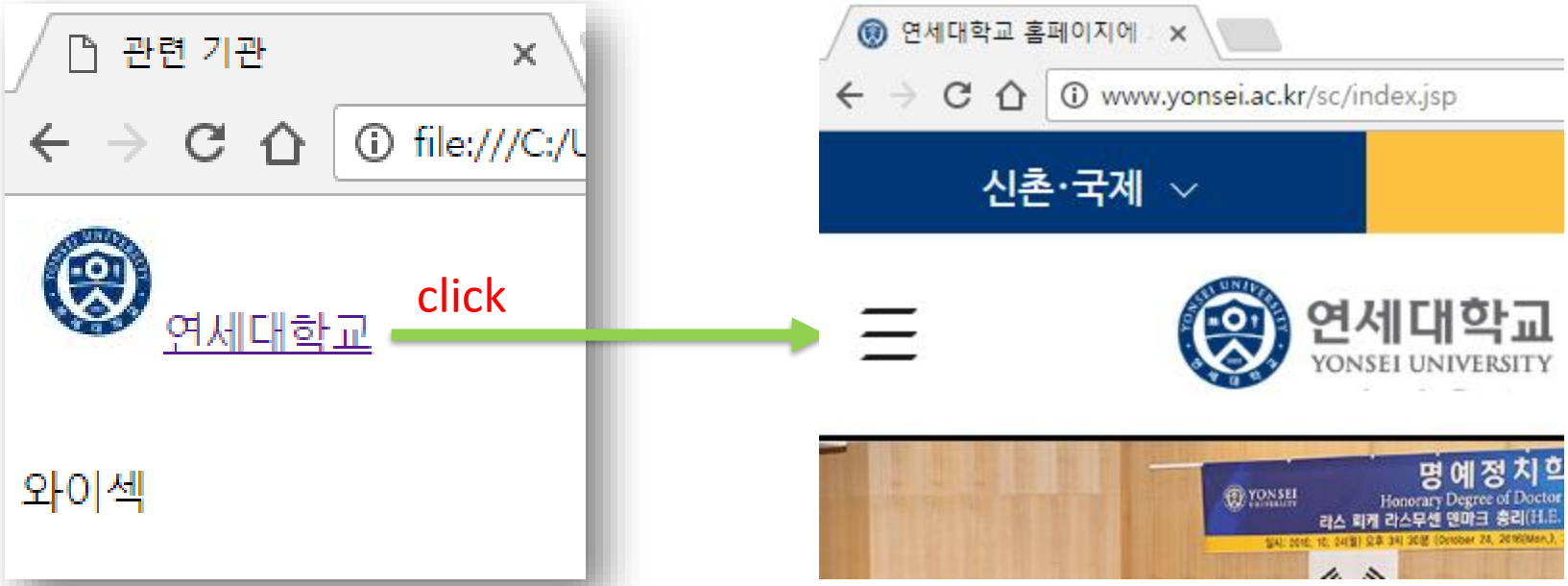
- Language for creating web pages and web applications.
- Tags and their attributes are key components.
  - Tags most commonly come in pairs like <title> and </title>
    - <title>Hello world!</title>
  - Some tags are unpaired, for example <img>
    - <img src = “image.gif”>
  - Most of the attributes of an element are name-value pairs.
    - Wikipedia

http://www.w3schools.com/

HTML

sample/web/sample1.html

```
<html>
  <head>
    <title>관련 기관</title>
  </head>
  <body>
    <a href="http://yonsei.ac.kr"><img src="img/yonsei.jpg">연세대학교</a><br><br><br>
    와이셋
  </body>
</html>
```



sample/web/sample2.html

```
<html>
  <head>
    <title>The Dormouse's story</title>
  </head>
  <body>
    <p class="title"><b>The Dormouse's story</b></p>
    <p class="story">Once upon a time there were three little sisters; and their names were
      <a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
      <a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and
      <a href="http://example.com/tillie" class="sister" id="link3">Tillie</a>;
      and they lived at the bottom of a well.</p>
  </body>
</html>
```

```
from bs4 import BeautifulSoup

soup = BeautifulSoup(sample2Html, 'html.parser')

soup.title
# <title>The Dormouse's story</title>

soup.title.string
# 'The Dormouse's story‘

soup.title.name
# 'title'
```

sample/web/sample2.html

```
<html>
  <head>
    <title>The Dormouse's story</title>
  </head>
  <body>
    <p class="title"><b>The Dormouse's story</b></p>
    <p class="story">Once upon a time there were three little sisters; and their names were
      <a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
      <a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and
      <a href="http://example.com/tillie" class="sister" id="link3">Tillie</a>;
      and they lived at the bottom of a well.</p>
  </body>
</html>
```

```
soup.title.parent.name
# 'head'

soup.p
# <p class="title"><b>The Dormouse's story</b></p>

soup.p['class']
# 'title'

soup.a
# <a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>
```

sample/web/sample2.html

```
<html>
  <head>
    <title>The Dormouse's story</title>
  </head>
  <body>
    <p class="title"><b>The Dormouse's story</b></p>
    <p class="story">Once upon a time there were three little sisters; and their names were
      <a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
      <a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and
      <a href="http://example.com/tillie" class="sister" id="link3">Tillie</a>;
      and they lived at the bottom of a well.</p>
  </body>
</html>
```

```
soup.find_all('a')
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]

soup.find(id="link3")
# <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>

for link in soup.find_all('a'):
    link.get('href')
# http://example.com/elsie
# http://example.com/lacie
# http://example.com/tillie
```

sample/web/sample2.html

```
<html>
  <head>
    <title>The Dormouse's story</title>
  </head>
  <body>
    <p class="title"><b>The Dormouse's story</b></p>
    <p class="story">Once upon a time there were three little sisters; and their names were
      <a href="http://example.com/elsie" class="sister" id="link1">Elsie</a>,
      <a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and
      <a href="http://example.com/tillie" class="sister" id="link3">Tillie</a>;
      and they lived at the bottom of a well.</p>
  </body>
</html>
```

```
soup.find("a", id="link3").get_text()
# Tillie

soup.find_all("a", class_="sister")
# [<a class="sister" href="http://example.com/elsie" id="link1">Elsie</a>,
#  <a class="sister" href="http://example.com/lacie" id="link2">Lacie</a>,
#  <a class="sister" href="http://example.com/tillie" id="link3">Tillie</a>]

soup.find_all(id=re.compile("[0-9]"))
```