

# How to Teach a Computer to Read the Internet



**An Open-Source  
Project on GitHub**



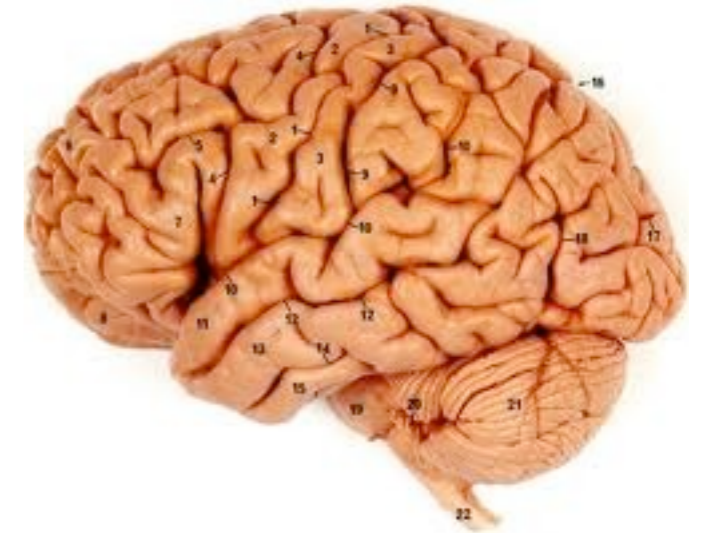
## Structured

RSS  
XML  
JSON



## Semi-Structured

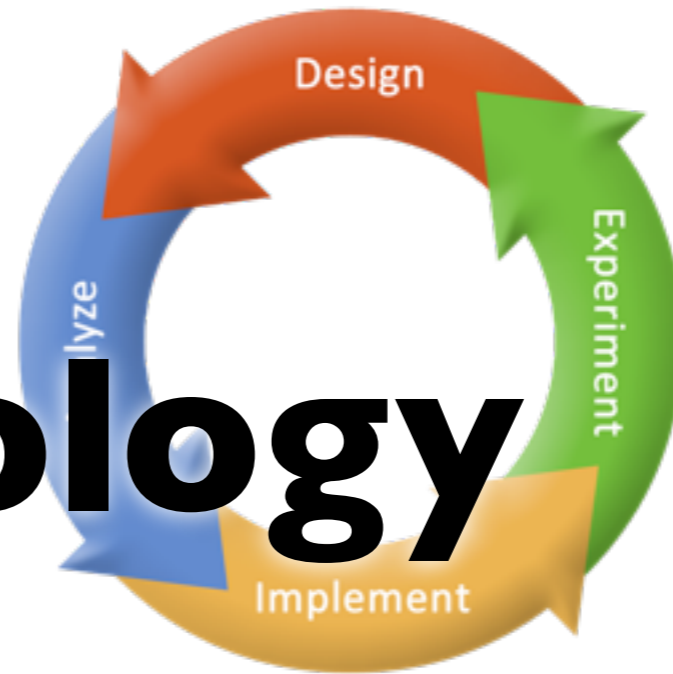
Wiki  
Twitter  
Meta Tags



## Unstructured

News  
Translations  
Poetry

# Methodology



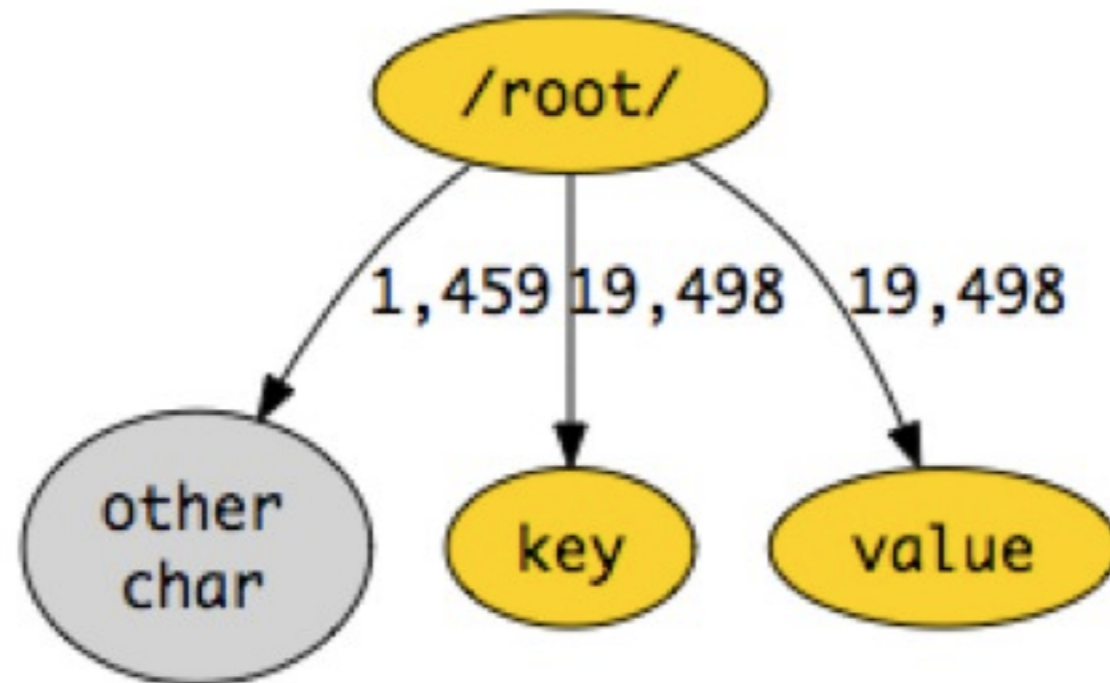
1. **parse** what you expect
2. **see** what else you get
3. **repeat** real fast

# I. **parse** what you expect

```
fact = key value | other_char
key = whitespace << word+ >> ':'
value = << ( !key . )+ >>
word = [A-Za-z]+ ' '*
whitespace = '\n' ' '*
other-char = << . >>
```

**there are facts**  
**that have keys made of words**  
**and values that aren't keys**

## 2. **see** what else you get

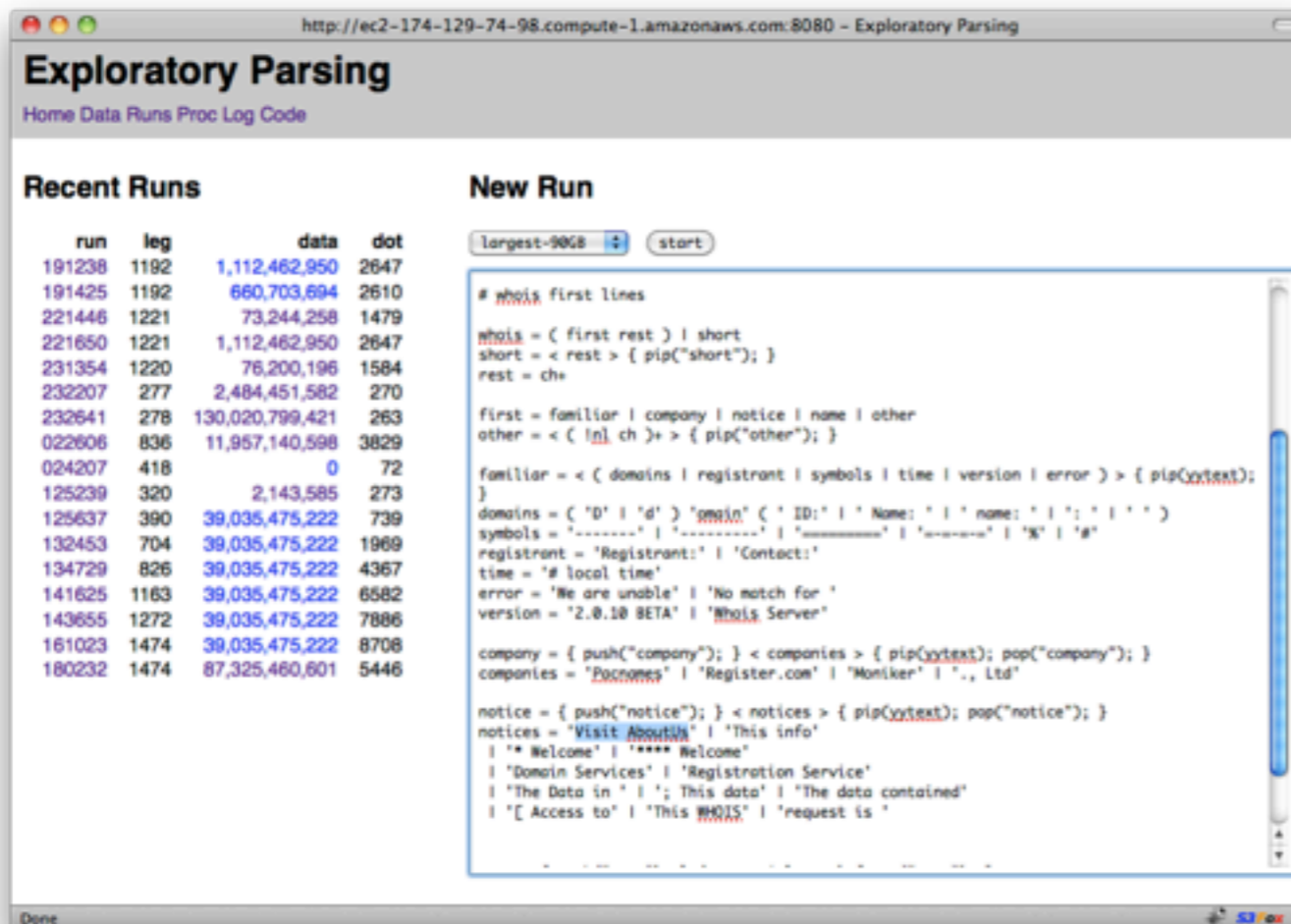


```
Imports:
  $331 million (f.o.b., 1989 est)
commodities:
  capital equipment, petroleum,
partners:
  US 37%, Netherlands 15%, Netho
  Brazil 5%, UK 3%, other 20%
External debt:
  $138 million (1990 est.)
Industrial production:
  growth rate NA; accounts for 2
```

**we found 19,498 keys**  
**one key was **partners****



# 3. **repeat** real fast



The screenshot shows the Exploratory Parsing web interface. On the left, there is a table titled "Recent Runs" with columns for run ID, leg, data size, and dot count. On the right, there is a "New Run" section with a dropdown menu set to "largest-9068" and a "start" button. Below the button is a text area containing a complex XPath query for parsing WHOIS data.

run	leg	data	dot
191238	1192	1,112,462,950	2647
191425	1192	660,703,694	2610
221446	1221	73,244,258	1479
221650	1221	1,112,462,950	2647
231354	1220	76,200,196	1584
232207	277	2,484,451,582	270
232641	278	130,020,799,421	263
022606	836	11,957,140,598	3829
024207	418	0	72
125239	320	2,143,585	273
125637	390	39,035,475,222	739
132453	704	39,035,475,222	1969
134729	826	39,035,475,222	4367
141625	1163	39,035,475,222	6582
143655	1272	39,035,475,222	7886
161023	1474	39,035,475,222	8708
180232	1474	87,325,460,601	5446

```
# whois first lines
whois = ( first rest ) | short
short = < rest > { pip("short"); }
rest = ch+

first = familiar | company | notice | name | other
other = < ( [n] ch )+ > { pip("other"); }

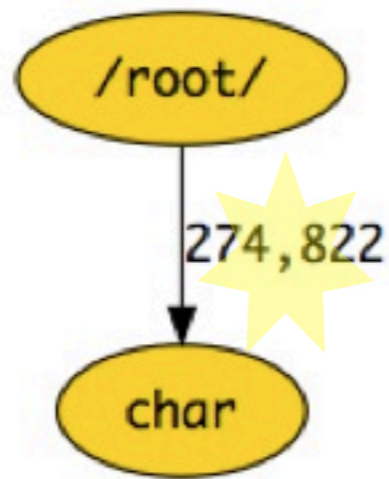
familiar = < ( domains | registrant | symbols | time | version | error ) > { pip("text"); }
}
domains = ( 'D' | 'd' ) 'main' ( ' ID:' | ' Name:' | ' name:' | ' ' | ' ' )
symbols = '-----' | '-----' | '-----' | '-----' | 'X' | 'g'
registrant = 'Registrant:' | 'Contact:'
time = '# local time'
error = 'We are unable' | 'No match for '
version = '2.0.10 BETA' | 'Whois Server'

company = { push("company"); } < companies > { pip("text"); pop("company"); }
companies = 'Dorcoms' | 'Register.com' | 'Moniker' | '., Ltd'

notice = { push("notice"); } < notices > { pip("text"); pop("notice"); }
notices = 'Visit AboutUs' | 'This info'
| '* Welcome' | '**** Welcome'
| 'Domain Services' | 'Registration Service'
| 'The Data in ' | ' '; This data' | 'The data contained'
| '[ Access to' | 'This WHOIS' | 'request is '
```

**the run started at 18:02:32**  
**parsed 87,325,460,601 bytes**

char = << ..... >>



Executive branch:

president, prime minister, Council of Ministers (cabinet)

Legislative branch:

unicameral National People's Assembly (Assembleia Popular Nacional)

Judicial branch:

Supreme Court

Leaders:

Chief of State:

President Miguel TROVOADA (since 4 April 1991)

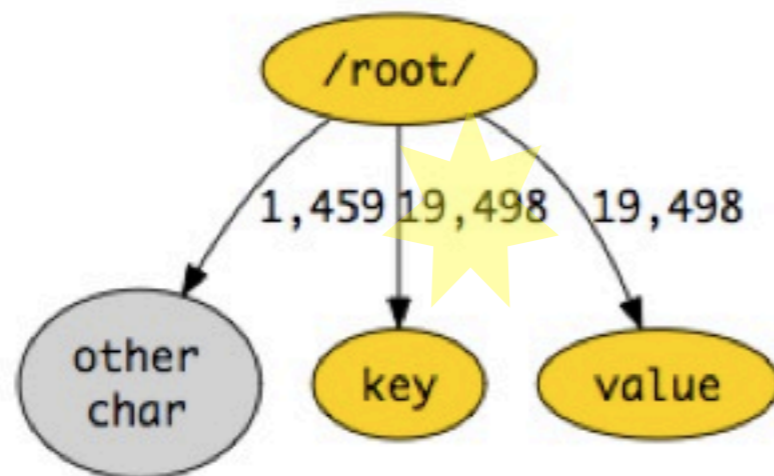
key

value

```

fact = key value | other_char
key = whitespace << word+ >> ':'
value = << ( !key . )+ >>
word = [A-Za-z]+ ' '*
whitespace = '\n' ' '*
other-char = << . >>

```



Imports:  
 \$331 million (f.o.b., 1989 est.)  
 commodities:  
 capital equipment, petroleum, foodstuffs, cotton, consumer goods  
 partners:  
 US 37%, Netherlands 15%, Netherlands Antilles 11%, Trinidad and Tobago 9%,  
 Brazil 5%, UK 3%, other 20%

External debt:  
 \$138 million (1990 est.)

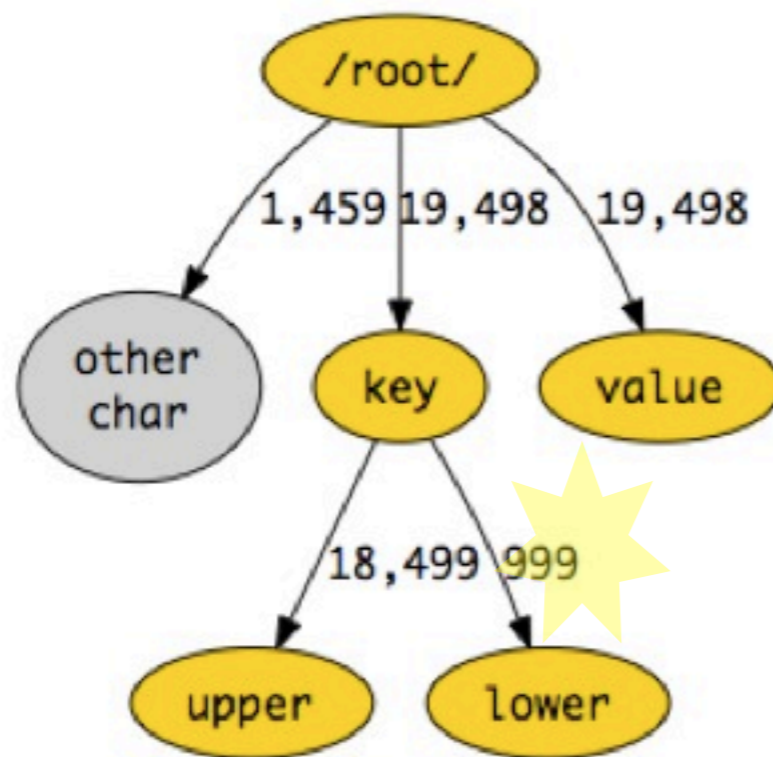
Industrial production:  
 growth rate NA; accounts for 22% of GDP

upper

lower



```
key = whitespace << ( upper | lower ) >> ':'  
upper = << [A-Z] word+ >>  
lower = << word+ >>
```



FTP directly to the Project Gutenberg archives:

```
ftp mrcnext.cso.uiuc.edu
```

```
login: anonymous
```

```
password: your@login
```

```
cd etext/etext91
```

```
or cd etext92 [for new books] [now also cd etext/etext92]
```

```
or cd etext/articles [get suggest gut for more information]
```

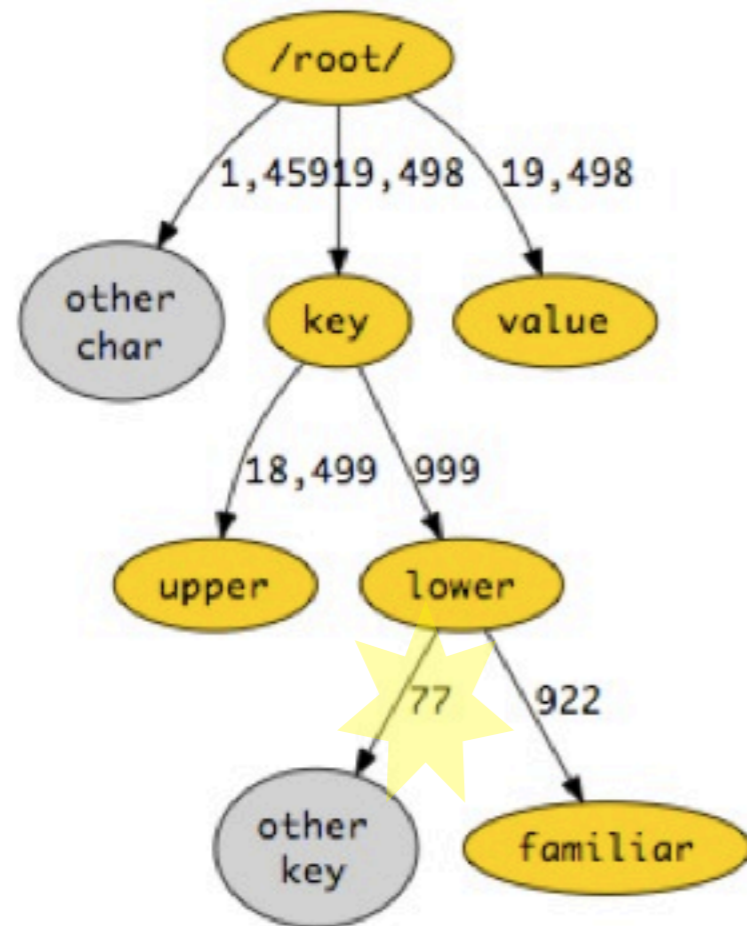
```
dir [to see files]
```

familiar

lower = << ( familiar | other-key ) >>

familiar = << ( 'commodities' | 'partners' ) >>

other-key = << word+ >>



#### Disputes:

Iran and Iraq restored diplomatic relations in 1990 but are still trying to work out written agreements settling outstanding disputes from their eight-year war concerning border demarcation, prisoners-of-war, and freedom of navigation and sovereignty over the Shatt-al-Arab waterway; Iran occupies two islands in the Persian Gulf claimed by the UAE: Tunb as Sughra (Arabic), Jazireh-ye Tonb-e Kuchek (Persian) or Lesser Tunb, and Tunb al Kubra (Arabic), Jazireh-ye Tonb-e Bozorg (Persian) or Greater Tunb; it jointly administers with the UAE an island in the Persian Gulf claimed by the UAE, Abu Musa (Arabic) or Jazireh-ye Abu Musa (Persian)

#### Climate:



# Real Fast Iterating

http://ec2-174-129-74-98.compute-1.amazonaws.com:8080 - Exploratory Parsing

## Exploratory Parsing

[Home](#) [Data](#) [Runs](#) [Proc](#) [Log](#) [Code](#)

### Recent Runs

run	leg	data	dot
191238	1192	1,112,462,950	2647
191425	1192	660,703,694	2610
221446	1221	73,244,258	1479
221650	1221	1,112,462,950	2647
231354	1220	76,200,196	1584
232207	277	2,484,451,582	270
232641	278	130,020,799,421	263
022606	836	11,957,140,598	3829
024207	418	0	72
125239	320	2,143,585	273
125637	390	39,035,475,222	739
132453	704	39,035,475,222	1969
134729	826	39,035,475,222	4367
141625	1163	39,035,475,222	6582
143655	1272	39,035,475,222	7886
161023	1474	39,035,475,222	8708
180232	1474	87,325,460,601	5446

### New Run

largest-98GB

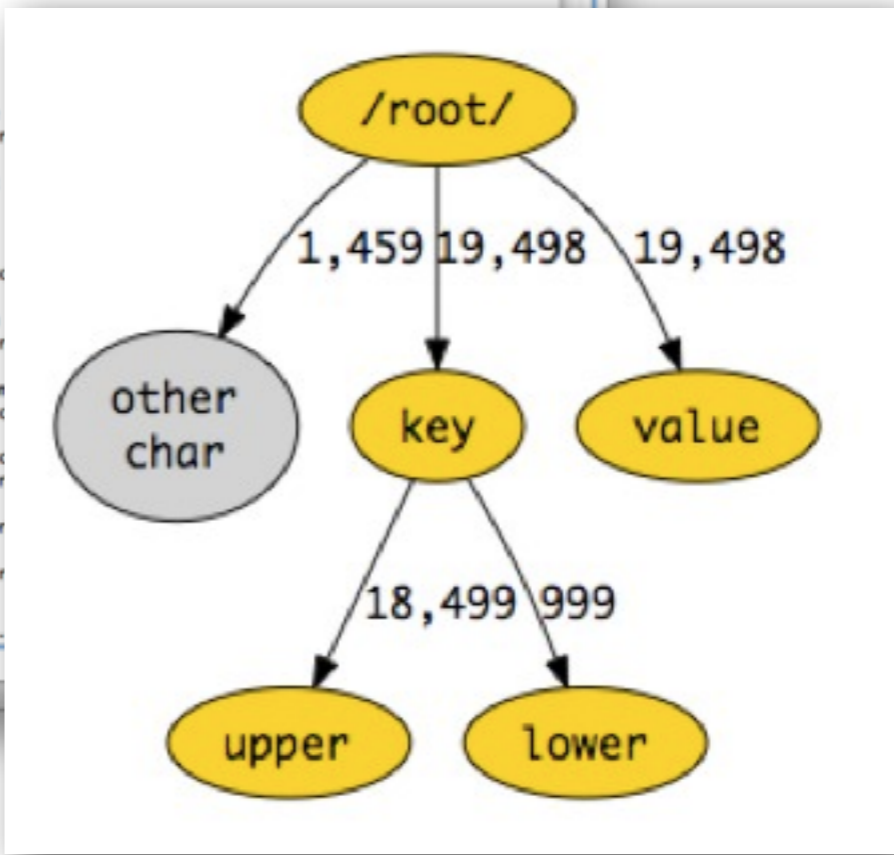
```
# whois first lines
whois = ( first rest ) | short
short = < rest > { pip("short"); }
rest = ch+

first = familiar | company | notice
other = < ( [n] ch )+ > { pip("other") }

familiar = < ( domains | registrant
)
domains = ( 'D' | 'd' ) 'omain' ( '
symbols = '-----' | '-----' |
registrant = 'Registrant:' | 'Contac
time = '# local time'
error = 'We are unable' | 'No match
version = '2.0.10 BETA' | 'Whois Ser

company = { push("company"); } < con
companies = 'Pocnames' | 'Register.c

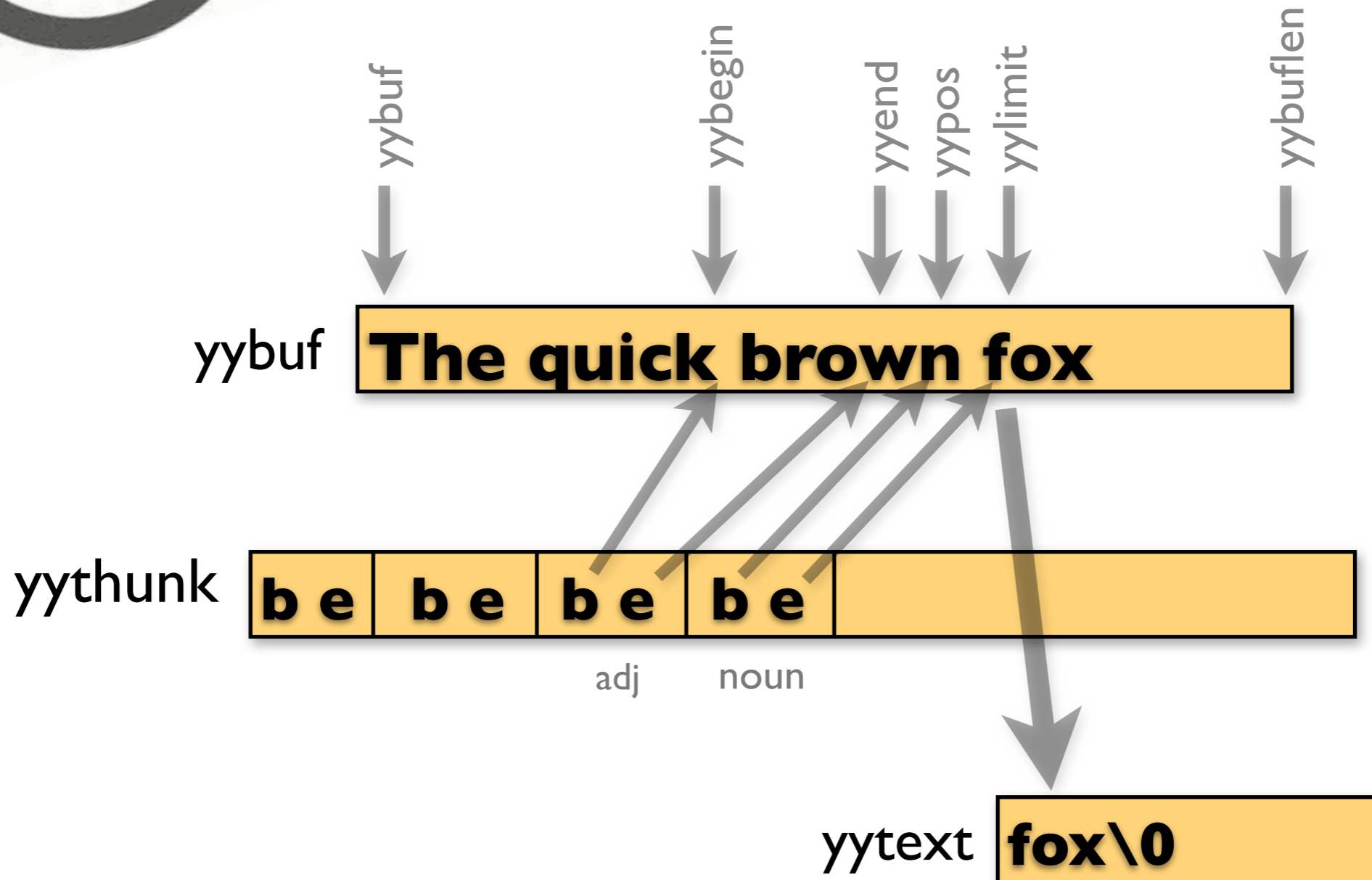
notice = { push("notice"); } < notic
notices = 'Visit AboutUs' | 'This ir
| '* Welcome' | '**** Welcome'
| 'Domain Services' | 'Registration
| 'The Data in ' | '; This data' |
| '[' Access to' | 'This WHOIS' | 'r
```







# Real Fast Parsing








# Real Fast Data

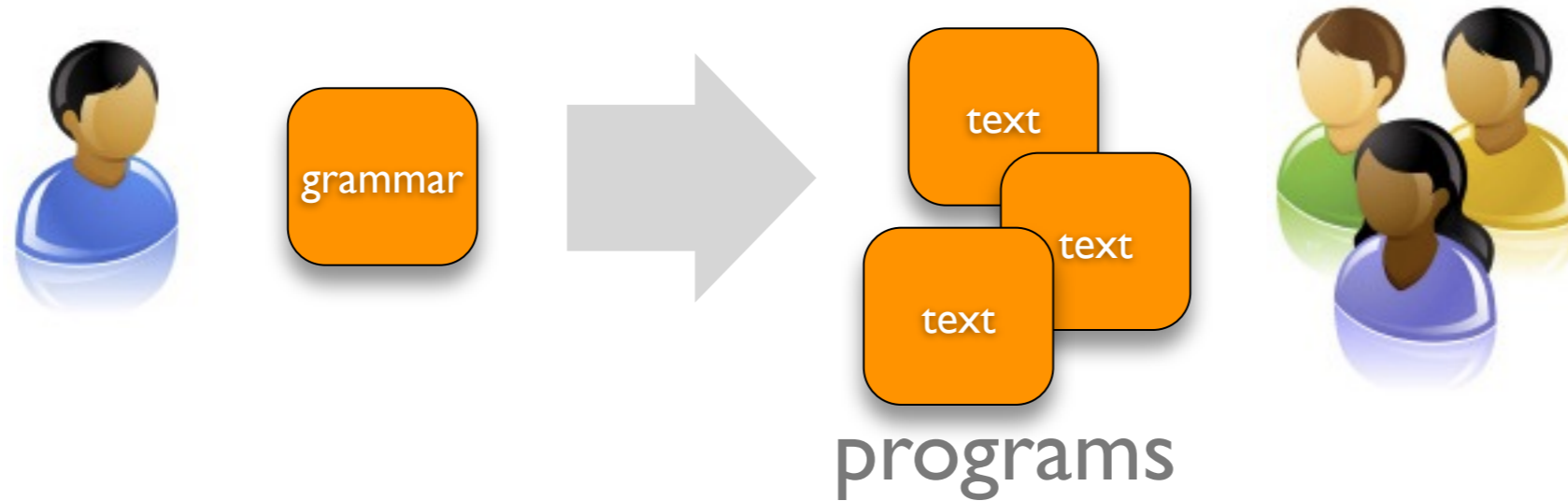
50 min for every byte  
5 min for useful prefix  
5 sec for last sampling

(30GB of wikitext)

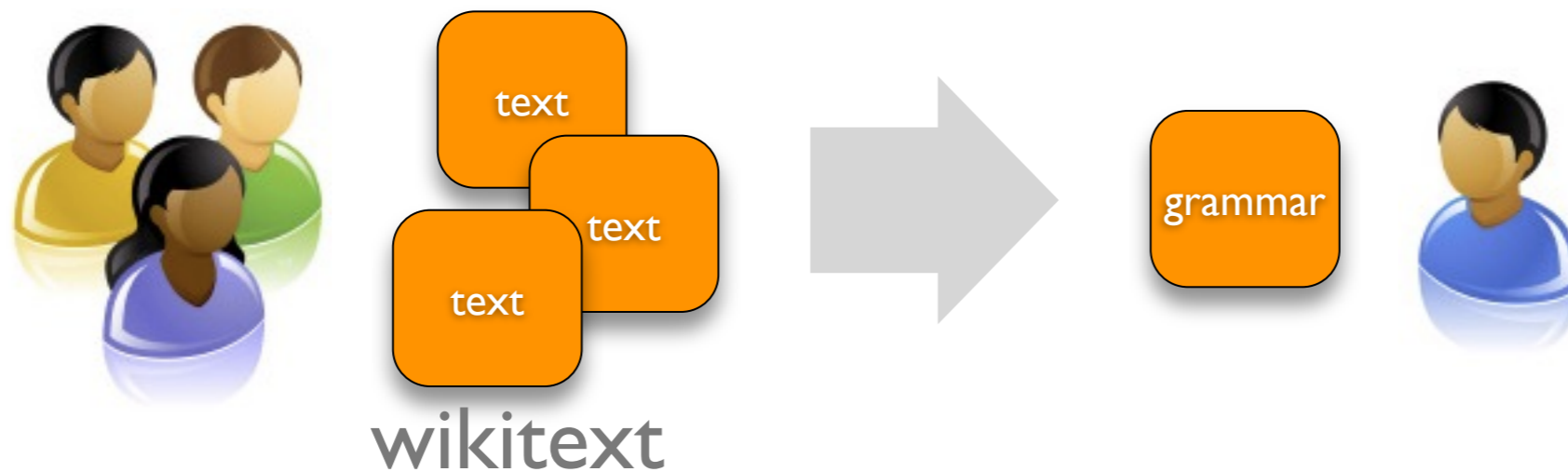
Nike, Inc.		Apple Inc.		Intel Corporation	
					
<b>Type</b>	Public	<b>Type</b>	Public	<b>Type</b>	Public company
<b>Traded as</b>	NYSE: NKE <a href="#">↗</a>	<b>Traded as</b>	NASDAQ: AAPL NASDAQ-100 S&P 500 Com	<b>Traded as</b>	NASDAQ: INTC <a href="#">↗</a> NYSE: INTC <a href="#">↗</a> Euronext: INCO <a href="#">↗</a> SEHK: 4335 <a href="#">↗</a> Dow Jones Component NASDAQ-100 Component
<b>Industry</b>	Clothing and Sport	<b>Industry</b>	Computer har Computer sof Consumer ele Digital distribu	<b>Industry</b>	Semiconductors
<b>Founded</b>	1964 (as Blue Ribb	<b>Founded</b>	April 1, 1976	<b>Founded</b>	Mountain View, California, U (July 18, 1968) <sup>[1]</sup>
<b>Founder(s)</b>	Bill Bowerman Philip Knight	<b>Founder(s)</b>	Steve Jobs Steve Woznia Ronald Wayn	<b>Founder(s)</b>	Gordon Moore, Robert Noy
<b>Headquarters</b>	Washington Count United States (Near Beaverton, C	<b>Headquarters</b>	Apple Campu: 1 Infinite Loop Cupertino, Ca	<b>Headquarters</b>	Santa Clara, California, U.S
<b>Area served</b>	Worldwide	<b>Area served</b>	Worldwide	<b>Area served</b>	Worldwide
<b>Key people</b>	Philip Knight (Chairman) Mark Parker (President and CEO)	<b>Key people</b>	Paul Otellini (President & CEO) Jane Shaw (Chairman)	<b>Key people</b>	Paul Otellini (President & CEO) Jane Shaw (Chairman)
<b>Products</b>	Athletic shoes Apparel Sports equipment Accessories	<b>Products</b>	Bluetooth chipsets, flash memory, microprocessors, motherboard chipsets, netw interface cards	<b>Products</b>	Bluetooth chipsets, flash memory, microprocessors, motherboard chipsets, netw interface cards
<b>Revenue</b>	▼ US\$ 19.014 billi	<b>Revenue</b>	▲ US\$ 43.623 billion (2010) <sup>[1]</sup>	<b>Revenue</b>	▲ US\$ 43.623 billion (2010) <sup>[1]</sup>
<b>Operating</b>	US\$ 2.517 billi	<b>Operating</b>	▲ US\$ 16.045 billion (2010) <sup>[1]</sup>	<b>Operating</b>	▲ US\$ 16.045 billion (2010) <sup>[1]</sup>



# Compiler Writer:



# Parsing Explorer:



regex text  
substitutions

```
$lines =~ s/\\n/ /g;
foreach (split(/\\n/, $lines)) {
  $InPlace=0;
  while (s/\\b(http|ftp|mailto|file|telnet|news):[\\s<>\\[\\]""'\\(\\)]*[\\s<>\\[\\]""'\\(\\)]\\,
    $InPlace[$InPlace++] = $&
  }
  s/&/&amp;/g;
  s/</&lt;/g;
  s/>/&gt;/g;
  s/"/&quot;/g;
  $code = "";
  s/^\\s*$/<p><\\p>/ && ($code = '...');
  s/^\\(\\t+)(.+)\\t/<dt>$2<dd>/ && &enter('DL', length $1);
  s/^\\(\\t+)\\*/<li>/ && &enter('UL', length $1);
  s/^\\(\\*+)/<li>/ && &enter('UL', length $1);
  s/^\\(\\t+)\\d+\\.?.?/<li>/ && &enter('OL', length $1);
  /^\\s/ && &enter('PRE', 1);
  s/''(.*)''/<strong>$1<\\strong>/g;
  s/''(.*)''/<em>$1<\\em>/g;
  s/^-----*/<hr>/;
  s/\\b$link\\b/&anchor($&)/geo;
  s/$SEP(\\d+)$SEP/&InPlace($1)/geo;
  s/^[Search\\]/<form action=$base><input type="text" size="40" name="search" value="
  s/^[Fullsearch\\]/<form action=fullSearch><input type="text" size="40" name="search
  s/^[?ISBN:?(\\[0-9- xX]{10,})\\]?/&BookLink($1)/igeo;
  &enter("", 0) unless $code;
  print; print "\\n";
}
&enter("", 0);
```

extra code  
for nesting

as in regex  
lookahead

primary = identifier ! '=' | '(' expression ')'

nesting  
as in yacc



# Piumarta 2007

## peg/leg — recursive-descent parser generators for C

`peg` and `leg` are tools for generating recursive-descent parsers: programs that perform pattern matching on text. They process a Parsing Expression Grammar (PEG) [Ford 2004] to produce a program that recognises legal sentences of that grammar. `peg` processes PEGs written using the original syntax described by Ford; `leg` processes PEGs written using slightly different syntax and conventions that are intended to make it an attractive replacement for parsers built with `lex` and `yacc`. Unlike `lex` and `yacc`, `peg` and `leg` support unlimited backtracking, provide ordered choice as a means for disambiguation, and can combine scanning (lexical analysis) and parsing (syntactic analysis) into a single activity.

Download the source code: [peg-0.1.4.tar.gz](#)

Browse the source code: [peg-0.1.4](#)

Read the manual page: [peg.1.html](#)

`peg` is distributed under  
modify it

# Ford 2004

## Parsing Expression Grammars: A Recognition-Based Syntactic Foundation

Bryan Ford  
Massachusetts Institute of Technology  
Cambridge, MA  
baford@mit.edu

### Abstract

For decades we have been using Chomsky's generative system of grammars, particularly context-free grammars (CFGs) and regular expressions (REs), to express the syntax of programming languages. This paper introduces a new system of generative grammars, called parsing expression grammars (PEGs), which is designed to be a syntactic foundation for recognition-based parsing.

### 1 Introduction

Most language syntax theory and practice is based on generative systems, such as regular expressions and context-free grammars, in which a language is defined formally by a set of rules applied recursively to generate strings of the language. A recognition-based system, in contrast, defines a language in terms of rules or predicates that decide whether or not a given string is in the language. For

# dev.AboutUs.org

## AboutUs.org Developer Blog

EXPERIENCES IN SOFTWARE DEVELOPMENT AT ABOUTUS



Sam Goldstein  
22 Aug 2011

### [Cassandra Truncate Means Slow Test Suites](#)

How using Cassandra's truncate (or clear\_keyspace) command made our tests super slow, and how we sped them up.



Sam Goldstein  
18 Jul 2011

### [Managing Cassandra's Schema from Rails](#)

Our approach to managing our cassandra cluster's schema using rails, rake, and capistrano.



Ward Cunningham  
03 Jul 2011

### [Getting Started Exploratory Parsing](#)

A parser reads text to discover structure and meaning. For example, a C language parser can read a C program and understand in a real sense everything that the program has to say. Contrast this to a pattern matcher...



Sam Goldstein  
17 May 2011

### [Agile vs. Open Source: They're Actually a Little Different](#)

We usually think of Agile and Open Source going hand in hand but there are some significant differences between an Agile organization and an Open Source one.



Matt Youell  
05 May 2011

### [Rodents Of Unusual Size](#)

Communication via Github is a great way to share knowledge

read this

### [Learning by Pairing](#)

is one of





**<http://c2.com/~ward/sao/TechIgnite-v1/>**

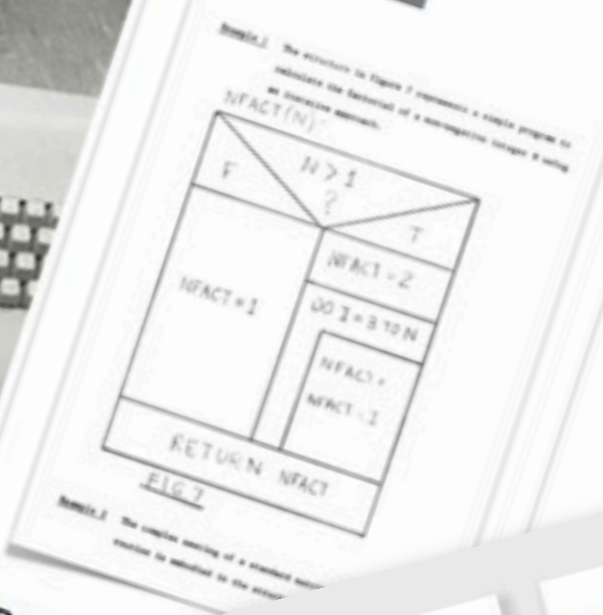
# COMING ATTRACTIONS

1967



Doug Engelbart invents innovative control devices like the mouse and chord keyboard. His intention: augmenting the human intellect. ▲

1972



Ben Sh...

1976



2011



## ■ Welcome Visitors

Welcome to the federated wiki. This page was first drafted Sunday, June 26th, 2011, at [indie-web-camp](#). You are welcome to copy this page to any server you own and revise its welcoming message as you see fit. You can assume this has happened many times already.

## Test Deployments

code here first. Look for: ... data feed

## ■ Indie Web Camp

Rather than posting content on many third-party silos of data, we should all begin owning the data we're creating. Publish short status updates on your own domain, and syndicate to Twitter. Publish photos on your own domain, syndicate to Flickr, etc, etc.

This is the basis of the "Indie Web" movement (see [why-indie-web](#) for more). We'll get together for a weekend to talk about what has been done in the field, and what still needs to be done. There will be workshops and breakout sessions.



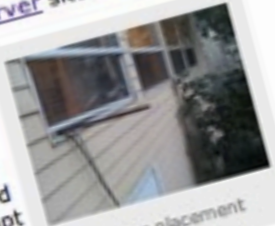
Ward's Lighted Electric Bike at [Pedalpalooza](#)

## ■ Air Temperature

Here we report the air temperature measured in Ward Cunningham's backyard. The thermometer is placed 18 inches from the east-facing lower-level of the house. Trees shade the sensor from morning sun except for an hour or two mid-morning. More elaborate reporting can be found in Ward's [SensorServer](#) site.

**59.8**  
Degrees Fahrenheit  
updated hourly

This page includes a chart item that contains a time series of data samples, the most recent of which is shown in the readout. The time series is updated by a cron script. The script transmits a [Txtzyme](#) program over USB to an attached Teensy micro-controller. This program signals a [Maxium DS18B20](#) digital thermometer over their one-wire protocol. Return data is translated to Fahrenheit and then edited into the flat-file format of this page.



Sensor placement

erated Wiki, it sites, it