Computer Science Seminar

Whats the next big thing?
Ruby? Python? Neither?

Introduction

- Seminar Style course
  - unlike many computer science courses
    - discussion important, encouraged and part of your grade
  - each student will be responsible for presenting one (or more) of the topics through out the semester.
    - There is a list of topic spaces to sign up on my office door right now.
    - 26 class meetings: at current enrollments, 18 of the presentations will be made by students (with some doubling up)
    - I'll be doing the first couple of weeks.

Grading (Assessment)

- Since this is a seminar course grading distribution is a little different
  - Programming projects: 35%
  - Papers: 35%
  - Presentation: 10%
  - Exam: 10%
  - Everything else (mostly participation and demonstrating that you understand the material regularly): 10%

More Administravia

- No room for slackers
  - I don't take attendance but:
    - Joining the conversation matters.
    - You may be called on, have a reasonable answer
    - In my classes doing well is highly correlated with showing up ***and paying attention***
    - If you show up and spend the time checking scores or surfing gaming wikis, you are not likely to do well.
  - when someone asks if there is anything to do to bring their grade up, my first answer is turn in all the assignments.
- Now lets look at the syllabus to make sure we are all on the same page.

So why this seminar

- For the last 10 years, Java has been a/the major programming language
  - in combined education/industry.
  - lots its no good for, but largest share of many languages
  - C and C++ round out the top three
  - large application development languages.
  - Each of these others had their own time of dominance.
  - perl, php and visual basic also had their time as specialty languages.
  - now big application languages seem to be in decline

So what's next

- Are the big application languages really in decline? or is is a brief outlier?
- If so, what takes their place
  - second tier specialty languages also seem to be declining
  - a couple of possibilities
    - everything becomes sort of specialty: lots of languages, none dominant.
    - a new language rises to become dominant
      - We will look at two promising candidates
      - one more possible 'D'
      - D is outside the scope of this course

The contenders

- Python and Ruby
  - share several similarities
    - Both dynamically typed
    - both interpreted
    - both include garbage collection
    - both have a difference syntax from C-like languages
      - no more curly braces to annoy newer programmers
  - both can be full fledges OO languages
    - or used as lightweight scripting languages.

Why Python

- Why might python be a big thing?
  - http://www.tiobe.com/tpci.htm
  - tiobe/tcpi programming index shows python to have reached high single digits after years of steady growth
  - Starting to show up in computer science education conferences as
    - language to teach first programming course
  - used as support language in industry
    - NASA to civilization IV game
    - wide specialized use of a general purpose programming language

Why Python II

- Jobs out there now that want python as main skill
  - http://www.python.org/community/jobs/
- Python, like Java and others has libraries for everything
  - web, gui, gaming, networking etc.

Why Ruby?

- Spent 10 years in obscurity before becoming
  - overnight sensation
    - well in 1.5-2 years has nearly doubled its share.
    - tiobe/tcpi list shows it in low double digits.
      - has moved up fast in last year or so.
  - Started to get some of the sort of buzz that Java did 10 years or so ago.
    - lots of buzz on forums
    - lots of books by excited people
  - not nearly as widely used
    - european academics at upper level classes
    - no heavy industry presence
  - much smaller set of libraries.
  - one ace in the hole: Rails

Ruby on Rails

- Rails is a ruby framework for slapping together a web-based database application.
  - supposed to be fast, scalable and easy
  - data bases on web are big and growing field
  - like what?

Where did Python come from

- Guido Van Rossum
  - worked on a language called ABC in early mid-80s
  - 1989: Guido starts work on Python during the Christmas holidays to create a better way for system administration than C programs or Bourne shell scripts.
  - 1991 announces python via usenet and makes it available to the world.
  - still “Benevolent dictator for life”

Where did ruby come from

- Yukihiro Matsumoto ("Matz")
  - 1993, February 24: Yukihiro Matsumoto starts to work on Ruby.
  - 1993, Summer: First "Hello, world!" program works.
  - 1995, December: First release 0.95.
  - 1996, December: 1.0 is released.
  - 1999: Supposedly overtakes Python in Japan.
  - 2000: The first official newsgroup.
  - 2004 July: ruby on rails released

Ruby fans: very passionate

- Notice when reading much of the Ruby literature
  - fans are passionate.
    - similar to Mac fans
    - Saturn owners circa 1994
    - lisp fans for 30+ years
  - utterly convinced of their products superiority
    - more emotional than intellectual
  - many similarities to lisp
    - no parentheses though

How are they doing today

- You've seen the Tiobe/tcpi list
- Sourceforge
  - As of July
    - shows 528 ruby projects in various stages
    - shows 3450 python projects again in all stages
  - sourceforge and its place
  - vs. commercial software
- Python by far more widely used,
  - but increase in share has slowed
- Ruby
  - very obscure till a year or so ago
    - outside of European Universities
    - now seeing big upswing.

Running Python at BSC

- python has one executable from command line
  - python
    - with no args will bring up interactive prompt
    - with an argument will execute

Running Ruby at BSC

- ruby has two executables
  - irb to run interactively
  - or ruby to run ruby files as programs