Ruby

A first pass look at the Ruby programming language

Comments

- First and foremost
  - how can I put a note in a program so I can understand it next semester
  - # is the line comment character
    - like in python
    - same as // in java or C
  - block comments are strongly discouraged but still available
    - begin and end block comments with
      - =begin
      - =end

Ruby: Dynamically typed

- what does that mean for programming
  - no need to declare and declare type of var before using them
  - if you want a var that holds a string to hold a FixNum (like and int) fine do it
    - x = “this”
    - x = 10
    - will work fine for ruby
    - note single equals assignment like in java/C
    - note that = is used as assignment in Ruby

Identifiers

- Variable Names
  - same names allowed as Java
    - letters numbers and underscore
      - no numbers to start identifier
  - variable identifiers beginning with Capital letter are treated as constant in Ruby
    - though like Python and lisp, constants are more recommendations than enforced by interpreter
      - get warning when changing value
      - but interpreter changes value

types

- In Ruby everything is an object
  - same as in python
  - numbers are all objects
    - have methods already defined on them
  - Strings are objects
    - no C-strings
    - like Java

Numbers

- Fixnum
  - default integer type
  - range is implementation specific
    - but if you follow Java/C int ranges probably safe
      - range is native machine word minus one bit
    - if number goes out of range auto converts to BigNum
      - what does this imply for code safety?
- Float
  - uses native double precision floating point representation
  - like java/c double

Math

- Numbers aren't so useful without Math
  - usual arithmetic is available: most operators similar to Java
    - +, - , /, *, %
      - note that / will return a value of the type to the left of the operator like in Java
        - 10/3 -> 3
        - 10.0/3 -> 3.333333333333333
    - ** is new x**y raises x to the yth power
      - 2**4 ->16

Strings

- Objects like in Java/Python
  - like python, different ways of deliminating strings
    - use the one that won't appear in a string
      - x = 'she said “this is it!” yesterday'
      - y = “I won't go back!”
      - use %q to make up your own deliminator
        - a = %q# this is %^*&^%$* I tell you#
  - string concatenation
    - use '+' as in Java
    - will concatenate two strings into one
      - but not other types as in Java

Some simple string operations

- Strings support several operators that look like logic/math
  - string * number
    - returns string repeated number times
    - “this “ * 3 returns “this this this “
  - comparisons
    - string1 > string2
    - returns true iff string1 comes alphabetically after string2
      -

Objects and Methods

- Now we've covered the “basic types” from c/Java (sorta)
  - But since everything in Ruby is an object, you should be able to call a method on everything right?
  - yup and to call a method
    - use object.methodname notation as with Java or c++
      - but there the analogy breaks down