Ruby

Syntax, Collections, Logic

J. Scott Johnson
AppData, LLC
scott@appdata.com

Ruby Overview

- Dynamic, late bound object oriented language
- Everything is an object
- Integers are object
- Everything is dynamic
- Want to make 2 + 2 = 5 ? redefine + in integer

Syntax

- object.method
- Example - take a string “scott” and replace o with a
- “scott”.sub(/o/,’a’)
- .sub is a substitution command
- (“scott”.methods - Object.methods).sort <== full introspection

Methods I

- def add(a,b)
- return a + b
- end

- Method is a procedure or function or routine; you've seen it before; stored functionality

Methods 2 Booleans

- def month_is_short?(month)
  return true if month.days.size == 28
  return false
- end
- A ? at the end of the method name indicates that it returns a boolean value
- **convention** over **configuration**

Methods 3 !

- ! <== indicates an in place operation
- “foo”.gsub(/o/,’a’) <== return faa
- “foo”.gsub!(/o/,’a’) <== changes it in place

Collections

- Classical languages use for loops, while or other iterative constructs
- Ruby is brilliantly simple -- you almost always only iterate with **.each** i.e.
- [1,2,3,4] <== an array or collection
- [1,2,3,4].each do |num|
  puts num
  end

Logic - if

- if Date.today == Date.new(2014,11,20)
- puts “Happy Birthday Scott”
- end

Logic - if

- if a
- elsif b
- end

Logic - if

- if a
- elsif b
- elsif c
- else
- end

Logic - Case

- case expression
- when something
- when something_else
- else
- end

Logic - if

- right hand ifs are allowed (and encouraged)
- multiple return statements per method enable exiting more quickly and aborting flow

To Return or Not to Return

- Default is to return the last evaluated expression
- This is an implicit return
- Not always clear what this is
- We generally opt for **explicit** return

Boolean Operators

- && <== logical and
- || <== logical or
- and <== logical and
- or <== logical or
- use && and || (higher order of evaluation precedence)

Sigil

- @ <== instance variable
- It has a broader scope than a local variable
- variously used in rails
  - makes variables from controller appear in view
  - makes test coverage share variables from higher up

Regular Expressions (regex)

- / & ^ $
- oh my
- no I'm not cursing
- Easy way to match / extract / parse data
- http://www.rubular.com/