# JMS 302: HACKING THE MEDIA

CLASS 5: 9/2/2014

# TODAY'S TASKS

- Beginning Ruby

# REVIEW

- HTML Tags



We’re going to focus on Ruby today and maybe Thursday as well.

# RUBY PROGRAM BASICS

- Programs generally execute from top to bottom, left to right.
- methods, conditionals (e.g., if), parenthesis change the order
- # this is a comment.
- 1+2 #everything from # until end of line is ignored
- # 1+2

C



A/FYI: You can do multiline comments with =begin and =end (but they must be at the very beginning of their lines.

http://ruby.bastardsbook.com/chapters/conventions/ (until debugging)

# RUBY BASICS

- White space & indentation
- Objects & methods (nouns and verbs)
- every verb needs a noun
- dot notation

C



http://ruby.bastardsbook.com/chapters/conventions/ (until debugging)

# NUMBERS

B • integers (Fixnum) & decimals (Float)

C • + - / *

B • % **

B • order of operations



% = modulo or the remainder operation. very useful, in grouping for example.
** = to the power of
order of operations, main things to remember:

- as a matter of practice, use parenthesis to make it obvious (and to override)
- * and / take precedence over + and - (otherwise, left to right)
- FYI, the order (highest to lowest):
    - **
    - * / %
    - + -
    - More info: http://www.techotopia.com/index.php/Ruby_Operator_Precedence
    -

# NUMBERS

- methods (beyond those above)

B
- abs

C
- rand(n) [always 0 to n-1]

A
- Math object
- Math:: PI
- Math.sqrt(9)

# LETTERS (STRINGS)

- What are these?

C

- ‘Hello’, 12, “world”, ‘’, ’12’

C

- What do we use print something to the screen (on its own line)? When we want to accept input from keyboard?

B

- What do these commands do?
- ‘pig’ * 5 ; 5 * ‘pig’; s.chomp;



- string, integer, string, empty string, string (there’s a small difference between single and double quotes; we’ll get to that later)
- puts, gets
- 1st one prints “pigpigpigpigpig.” The second throws a TypeError. The third cuts off a trailing newline character from s, if there is one.

# LETTERS (STRINGS)

C
- arithmetic (adding strings)
- What happens?

B
- '2' * '5', '2' * 5, '12' + 12, 12 + 12, '12' + '12'

C
- Escape quotes with \



escape:
- 'this single quote, \', needs to be escaped'
- "this one, ', doesn't"
- "this double quote, \", needs to be escaped"
- 'this one, ", doesn\'t'

# STRING METHODS

C • Most methods don't change the object.

C • reverse, length, upcase, downcase

FYI • swapcase, capitalize, center, ljust, rjust



If I said, spell your name backwards, that doesn't change the spelling of your name, just how you spell it that one time (backwards). Similarly for methods.

Generally, in Ruby, methods that change the object end with an exclamation point (e.g., s.reverse vs. s.reverse!).

# BOOLEANS

- true
- false
- vs. ‘true’ and ‘false’

C



not explicitly in book

# VARIABLES

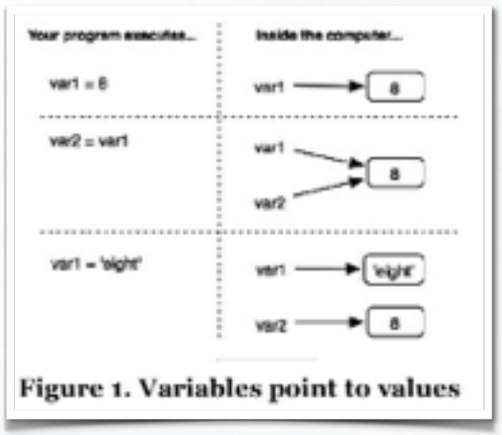

Figure 1. Variables point to values

- use to store things and reference later
- start with lowercase letter
- our convention: use underscores between "words," all lowercase

B

# VARIABLE CONVERSION

- problems
- var_1 = 2; var_2 = ‘5’; var_1 + var_2;
- x = 9/5 and you want decimal result
- to_i, to_f, to_s

C

# PREDICT THE OUTPUT

- 12 - 1 * 4
- 4.2 * 5
- 4.2.to_i * 5
- 10 + 1.to_f
- 14 % 4
- 1 % 100
- 6 % 3
- 9**.05
- 2 ** 3 / 2 A+/FYI
- 2 / 2 ** 3 A+/FYI
- 20.9.to_i / 15 % 3 A+/FYI
- (-2*4.5).abs

B



Test in irb

Note: As a matter of practice, use parenthesis to make the order of operations clear.

I only expect you to memorize order of operations for + - * /

# GETTING AROUND

- irb
- ctl-c, ctl-d, quit, cmd-n, cmd-t



irb is an interactive Ruby shell. It's a great way to quickly test code. Simply type "irb" on the command line. This should work on any system with ruby installed.

- B: ctl-c, ctl-d, quit are useful for getting around. cmd-n and cmd-t get you new windows and tabs.

See https://www.ruby-lang.org/en/documentation/quickstart/ .

```
"Hello, world"
puts "Hello, world"
3+2
3*2 (edit via tab and back space)
3**2
Math.sqrt(9)
```

# EXERCISES

- Write a program that gets user's first, middle and last names (separately), prints the name out backwards, and tells the user the number of characters in the full name.
- Write a program that asks a user for 5 grades and calculates a final grade, with the first two grades counting for 1/2 of the final grade.



Create a "class_5" directory under "exercises" in GitHub and add the directory. Commit your work.