Basic Coding Concepts

Modern coding languages, and just enough Ruby

Time to learn Ruby

- Learning:
  - Enough Ruby and programming concepts to get you started
- Not learning:
  - All of Ruby
  - All of programming language theory

Prerequisites

- Ruby installed
- Terminal window
- Text editor
- Directory to put code into
- Git installed

Strings, Printing & your first program

- Create a directory to put your Ruby code in.
- Open a terminal window, and use “cd” to move to that directory.
- Open your editor, and type this in the editor:

puts “Hello World!”

- Save to file “helloworld.rb” in your Ruby code directory
- Go into your terminal window again, and type:

ruby helloworld.rb

That was your first program…

- Ruby filenames end in ".rb"
- **"Hello World!"** is a "string"
- **puts "Hello World!"** prints "Hello World!" to your terminal
- **print "Hello World!"** also prints to your terminal - but doesn't add a "newline"

Try this…

puts “String1”

puts “String2”

print “String1”

print “String2”

Comments

- Add a line starting with "#" to your helloworld.rb file, e.g.

#This is a comment

- And add text starting with "#" to the end of your "puts" line, e.g.

puts "Hello World!" #English

Magic Comments

#!/usr/bin/ruby

#encoding: utf-8

- “Shebang” at start of every Ruby program
- See http://en.wikibooks.org/wiki/Ruby_Programming/Encoding

Variables

```
my_string = “Hello World!”

puts my_string

my_boolean = true

my_number = 10

puts my_number

my_number = 15.5

puts “My number is #{my_number}”
```

- Note: If you cut and paste the above code, you might see this error message:

```
variables.rb:3:in `<main>': undefined local variable or method `”' for main:Object (NameError)
```

- This happens because the symbols “ and ” above aren’t the same as the ones in your code editor. It’s annoying, but easily fixed: just delete them and type “ and ” in the right places in the editor. And if you’re in sublimetext, notice how the code changes color when you do this…

String formatting

puts “My number is #{my_number}”

IRB (Interactive Ruby)

- Type “irb” in your terminal window
- Try out some Ruby commands
- Type “exit” to leave IRB

2.1.3 :001 > **print “Give me text”**

Getting Input from the User

```
print “Give me some text”

user_text = gets.chomp

lower_text  = user_text.downcase

text_length = user_text.length

puts “Your text is #{user_text}, its length is #{text_length}”

puts “In lowercase, that’s #{lower_text}”

user_text.downcase!

puts “Your text is now #{user_text}”
```

Collections : Hash

- You'll meet two types of collection in Ruby: Hash and Array
- Hash:

iso3166 = {

SLE: "Sierra Leone",

NGA: "Nigeria",

LBR: "Liberia"

}

iso3166[:LBR]

iso3166.keys

Collections: Array

- Array:

rowvals = [1, 3, 5, 6, 4, 7, 3, 1, 3]

rowvals[3]

rowvals.max

rowvals.sort!

rowcols = [[‘a','b','c'], ['d','e',1]]

rowcols[0][2]

Getting input from a file

- Libraries are your friends!

require ‘csv’

rows = CSV.read(‘ebola-data-db-format.csv’)

- You just read in a CSV file. Now take a look at it:

puts rows.length

puts rows[0].inspect

puts rows[1].inspect

- You used the csv library here; function File.read is also useful, e.g.

myfilecontents = File.read(‘myfile.txt’)

Iterators

- Iterators allow you to use every item in a list, in turn

```
iso3.each do |code, name|
    puts “the code for #{name} is #{code}”
end
```

Conditionals

```
#!/usr/bin/ruby

require 'csv'

rows = CSV.read('ebola-data-db-format.csv')

#Find all the rows about Liberia

rows.each do |row|
        if row[1] == "Liberia"
                puts row
        end
end
```

- You might need to convert from strings to other data types. Conversion methods include to_i, to_s and to_a

Objects and Classes

```
class Vehicle
    attr_reader :colour, :model

    def initialize(colour, model)
        @colour = colour
        @model = model
    end

    def repaint (newcolour)
        @colour = newcolour
    end
end
```

Objects and Classes

```
mytruck = Vehicle.new("Black", "Ram3000")

puts "My #{mytruck.model} truck is #{mytruck.colour}"

mytruck.repaint("Red")

puts "My #{mytruck.model} truck is #{mytruck.colour}"
```

Class Inheritance

```
class Rig < Vehicle
  def add_load (load)
    @load = load
  end
end
```

Libraries

- Libraries are pieces of code that somebody else wrote to do something you need
- In Ruby, they're called "gems"
- You already used one when you typed **require 'csv'**
- Places to look for gems include:
  - http://blog.teamtreehouse.com/10-must-have-ruby-gems
  - https://www.ruby-toolbox.com/categories/by_name
  - https://rubygems.org/gems

NB: Rails is a Ruby gem

That's enough Ruby for now

- For more, see sites including:
  - tryruby.org
  - http://ruby.learncodethehardway.org/book/ex0.html
  - http://en.wikibooks.org/wiki/Ruby_Programming
  - ruby-doc.org
  - https://rubygems.org/gems
  - Codecademy Ruby course