Intro to Web Design and Development, Class 8

Ruby View, Gems, Sinatra

Schedule

Part 1 - Homework Question, Review, String Manipulation

1. Going over the homework
    1. Create the following methods:
        a. Adds five to argument given
        ```
        def addFive(n)
                n+5
        end
        ```
        b. Multiplies argument given by 15
        ```
        def mult15(n)
                n*15
        end
        ```
        c. Performs a mathematical operation using four arguments
        ```
        def mathOp(a,b,x,y)
                a+b+x*y
        end
        ```
        d. Prints the argument given four times
        ```
        def printArg(myAwesomeString)
                myAwesomeString*4
        end
        ```
        e. Prints an uppercase version of the argument given
        ```
        def printUpperCase(argg)
                argg.upcase
        end
        ```
    2. Create an object with two attributes and one method. Don’t copy the object from the slideshow exactly, please.
    ```
    class Pen
            attr_accessor :brand, :length

            def draw_me
                    puts “Pen is drawing”
            end
    end
    ```
    3. Use comments (denoted with a #) in your file to specify the different methods and what they should do.
        a. Comments are denoted using the # sign.

2. Programming Basics
    a. What is programming?
        i. Defining commands
        ii. Issuing them
        iii. Ensuring they get executed
    b. Variables
        i. Variables are containers for values
            1. x = 5
            2. y = “Jonathan”
            3. q = x + r
        ii. Variables have many types
            1. Float - numbers with decimal points - 10.32, 65.323, .32
            2. Integer - natural numbers - 11, 53, 3
            3. Strings - a line of text - “Jon”, “Elephants are awesome”
            4. Boolean - either true or false
            5. Arrays - a collection of values - [5,3,12,”omega”]
            6. Objects - a representation of something in the real world, with properties and methods. A Car object would have brakes, and a method to make the car move.
    c. Conditions
        i. A condition is a test for something
        ii. if “this” then “that”
    d. Loops
        i. Repetitive conditions where one variable in the loop changes
    e. Functions
        i. A function is a shortcut to a block of code
        ii. It can take arguments
        iii. It typically returns a value
3. Fun with strings
    a. Is the string empty?
        i. awesomeString.empty?
    b. String length
        i. awesomeString.length
    c. Concatenation - bringing two strings together
        i. awesomeString = “Hello” + “World”
        ii. awesomeString = “Hello” “World” **(only works with two strings)**
        iii. awesomeString = “Hello” << “World” **(chains)**
        iv. awesomeString = “Hello”.concat(“World”) **(concat method)**
    d. String interpolation
        i. awe = “Hello”
        ii. run = “World”
        iii. u = “Well #{awe}, #{run}!”
    e. Freezing

            i. awesomeString.freeze
            ii. Once frozen, a string is immutable
    f. Searching
        i. awesomeString[“Hello”] **(finds the string hello inside of awesomeString, if it exists)**
    g. Accessing parts of the string
        i. awesomeString = “Hello World”
        ii. awesomeString[0] = “H”
    h. String manipulation
        i. Substitution 1
            1. yourString = “Hello World”
            2. yourString[“World”] = “Universe”
            3. yourString = “Hello Universe”
        ii. Substitution 2
            1. yourString.gsub “Universe”, “World”
        iii. Repeating
            1. yourString * 3
            2. **returns** Hello World!Hello World!Hello World!
        iv. Inserting text
            1. yourString = “Hello World”
            2. yourString.insert 5, “ to the”
            3. **returns** “Helloto the world!”
        v. Chomp and chop
            1. yourString = “Hello World! H”
            2. yourString.chop
            3. **returns** “Hello World! “
            4. yourString.chomp
            5. **returns** “Hello World!” **(removes white space)**
        vi. Reversing strings
            1. yourString.reverse
        vii. Make uppercase, lowercase
            1. yourString.upcase
            2. yourString.downcase
        viii. Swapcase - yourString.swapcase - swaps the case of a string
        ix. Capitalize - yourString.capitalize - Capitalizes the first letter of a string
    i. Heredocs - free-format strings
        i. yourText = <<DOC
        Hello Sir,
        I know you’re enjoying learning about programming.
        DOC

Part 2 - Gems, Bundler, and Sinatra

1. Libraries and RubyGems

   a. A library is a collection of methods and classes
   b. There is typically a well-defined way for these to be accessed
   c. Libraries are available for many programming languages including Ruby
   d. Ruby’s library system is known as the RubyGem system, or “Gems” for short
2. Using Gems
   a. Installation: **gem install mygem**
   b. Uninstallation: **gem uninstall mygem**
   c. List all installed gems **gem list --local**
   d. When using RVM - the “sudo” command is typically not necessary
3. Bundler
   a. Bundler is a gem that makes installing other gems as part of your file super easy
   b. To install: **gem install bundler**
   c. Once bundler is installed, use a Gemfile to describe which gems you’d like
   d. Great resource for gems - [The Ruby Toolbox]
4. Gemfile
   a. A gemfile contains a source at the top:
      i. source ‘http://rubygems.org’
   b. Then it contains all the gems you’ll use
      i. gem ‘sinatra’
      ii. gem ‘will_paginate’
      iii. gem ‘data_mapper’
   c. Groups are used for different development environments
      i. Development
      ii. Test
      iii. (Staging)
      iv. Production
      v. group :development do | gem ‘pg’ | end
   d. Run bundle install to install all the gems at once!
5. Sinatra
   a. Sinatra is a DSL “Domain Specific Language”
   b. A DSL is used to solve a very specific problem
   c. Sinatra is a web application framework, similar to RoR
6. Using Sinatra
   a. First install the Gem - gem install sinatra
   b. Then create a main file, yourapp.rb
   c. Use the following code:

```
require ‘sinatra’

get ‘/hi’ do
        “Hello World!”
end
```

   d. Run the file - ruby yourapp.rb
7. Make your own!

    a. Create a Sinatra site with several different routes
    b. Use methods to encapsulate functionality
    c. Sample: sin_class.rb

Homework

Goals

1. Continue to solidify your understanding of basic programming concepts
2. Become familiar with the Sinatra DSL

Assignment

1. Write 3 new methods that take a string and transform it in some way
    a. **Example:**
    ```
    def makeUpper(inputString)
            puts “Your string uppercase is: #{inputString.upcase}”
    end
    ```
    b. Put this in a Git repository and push it to Github
2. Continue working on your Sinatra web app and experimenting with the functionality. Make sure it is in a git repository and push it to Github.

Recommended Activity

Continue with the “Introduction to Ruby” track on Codecademy
http://www.codecademy.com/courses/ruby-beginner-en-d1Ylq?curriculum_id=5059f8619189a5000201fbcb

Explore the Ruby Toolbox website to see what kind of gems are available. Come in with any questions on gem functionality for discussion.
https://www.ruby-toolbox.com/

Check out the Sinatra documentation:
http://www.sinatrarb.com/

eHandout

1. Homework
2. Session outline