LinuxAcademy.com

Ruby 101

Ruby Basics:Class Methods and Singletons

Class Methods:

Perform functions related to the class, but not necessarily a particular object of that class

Uses:

Operate on all members of a class, class variables, database cleanup methods.

Defining a class method:

```
class Tree
    def self.trim
        "All trees are trimmed now!"
    end
end
```

Calling a class method:

Class_Name.method_name

eg Tree.trim

Singleton Methods

A method that is defined on only one object.

eg

```
abc = "abc"

def abc.twice
	"#{self}#{self}"
end

abc.twice
```

Singleton Classes

Objects that exist in only one instance
Help to provide a "top" structure to a hierarchy

Multiple ways to create singletons:

- Create a blank class, then operate on it directly.
- Create an instance of class Object, then extend it.

Create a blank class, then operate on it directly.

```
class TableCorporation
end


class << TableCorporation
    ...
end
 -OR-
class TableCorporation
    class << self
        ...
    end
end
```

Create an instance of class Object, then extend it.

```
TableCorporation = Object.new
class << TableCorporation
    ...
end
```

Attritubes and methods

```
class TableCorporation
class << self
      attr_accessor :owner, :corporation_name

      def print_owner
            puts @owner
      end

      def print_name
            puts @corporation_name
      end
end
end
```

Try it yourself:

Write a class for a shipping company. This class will correspond to a box that will be shipped. The class should include:

- class variables for materials cost
- class methods for changing the cost of materials
- instance variables for size, weight, and travel distance
- a method for calculating the cost of the individual package

Try it yourself:

```
class Box
    attr_accessor :length, :width, :height, :weight, :distance
    def initialize
        @@materials_cost = 0.01 #1 cent per square inch
        @@rate = 0.01 # 1 cent per pound per mile
    end
    def self.rate= rate
        @@rate = rate
    end
    def self.materials_cost= cost
        @@materials_cost = cost
    end
    def package_cost
        (@length * @width * 2 + @length * @height * 2 +
            @width * height * 2) * @@materials_cost + @weight *

            @distance * @@rate
    end
```