Core Ruby 41st Batch

Navigation

Home

- My home
  - Site pages
  - My profile
  - Current course
    - PORPC101-41C
      - Participants
      - General
      - 5 April - 11 April
      - 12 April - 18 April
      - 19 April - 25 April
      - **Week 3: Tutorial**
      - Week 3: Exercises
      - Week 3: Forum
      - FizzBuzz Forum (Special Discussion Area)
      - Some useful Ruby methods
      - Recipe
      - *PARALLEL TRACK* Create a simple webpage
        - Week 3: Quiz
      - 26 April - 2 May
      - 3 May - 9 May
      - 10 May - 16 May
      - 17 May - 23 May
      - 24 May - 30 May
      - 31 May - 6 June
      - 7 June - 13 June

Week 3: Tutorial

Before you begin:

First complete: Week 1 to Week 2.

Week 3

- You need to read through the following pages, in the given order. While doing so, please make a note of all your doubts, queries, questions, clarifications and after you have completed all the pages, post these on the forum here. There may be some questions that relate to something that has not been mentioned or discussed by me here; you could post the same too.
  - Ruby Arrays: Parallel Assignment; Environment Variables; Command line arguments; Library GetoptLong
  - Ranges
  - Ruby Symbols
  - Hashes: Using Symbols as Hash Keys
  - Random Numbers
  - Read/Write files: Traversing Directory Trees; Random Access
- Some of the **important points to remember** after you have read through the above pages are:
  - An **Array** is just a list of items in order. Every slot in the list acts like a variable: you can see what object a particular slot points to, and you can make it point to a different object. You can make an array by using square brackets.
  - Arrays are indexed by integers and the index starts from 0.
  - A trailing comma in an array declaration is ignored.
  - You can access an array beyond its boundary limits; it will return **nil**.
  - We can add more elements to an existing array.
  - Refer to the **Array** documentation for a list of methods.
  - The method **each** (for any object) is an iterator that extracts each element of the array. The method each allows us to do something (whatever we want) to each object the array points to.
  - The variable inside the "goalposts" ie. | | refers to each item in the array as it goes through the loop. You can give this any name you want.
  - Sequences have a start point, an end point, and a way to produce successive values in the sequence. In Ruby, these sequences are created using the ".." and "..." range

My courses

Settings

Course administration

My profile settings

operators.

- The two dot form creates an inclusive range, and the three-dot form creates a range that excludes the specified high value.
- In Ruby, the sequence 1..100000 is held as a **Range** object containing references to two **Fixnum** objects.
- The **.to_a** method converts a **Range** to an **Array**.
- Another use of the versatile range is as an interval test: seeing if some value falls within the interval represented by the range. We do this using **===**, the case equality operator.
- Ranges are not limited to integers or numbers. The beginning and end of a range may be any Ruby object.
- A symbol looks like a variable name but it's prefixed with a colon.
- You can think of :id as meaning the name of the variable id, and plain id as meaning the value of the variable.
- Symbols are useful because a given symbol name refers to the same object throughout a Ruby program.
- Symbols can be considered constants without values.
- Symbols are more efficient than strings. Two strings with the same contents are two different objects, but for any given name there is only one **Symbol** object. This can save both time and memory.
- When do we use a string versus a symbol? a. If the contents (the sequence of characters) of the object are important, use a string
  b. If the identity of the object is important, use a symbol.
- A **Symbol** object is created by prefixing an operator, string, variable, constant, method, class, module name with a colon.
- If Fred is a constant in one context, a method in another, and a class in a third, the **Symbol** :Fred will be the same object in all three contexts.
- Symbols ARE used like this:

```
mystring = :satishtalim
```

Or this:

```
mystring = :satishtalim.to_s
```

Or this:

```
myint = :satishtalim.to_i
```

Or this:

```
attr_reader :satishtalim
```

You'll never see this:

```
:myname = "satish"
```

- Hashes are similar to arrays in that they are indexed collection of object references. However, while you index arrays with integers, you can index a hash with objects of any types: strings, regular expressions, and so on.
- When you store a value in a hash, you actually supply two objects - the index (normally called the key) and the value.
- **nil** is returned when an attempt is made to access keys that do not exist in the hash.
- The method to get a randomly chosen number in Ruby is rand.
- If you call **rand**, you'll get a float greater than or equal to 0.0 and less than 1.0. If you give it an integer parameter (by calling **rand(5)**), you will get an integer value greater than or equal to 0 and less than 5.
- The **File.open** method can open a file in different modes like 'r' Read-only, starts at beginning of file (default); 'r+' Read/Write, starts at beginning of file; 'w' Write-only, truncates existing file to zero length or creates a new file for writing.
- **File.open** opens a new **File** if there is no associated block. If the optional block is given, it will be passed file as an argument, and the file will automatically be closed when the block terminates.
- Always close a file that you open. In the case of a file open for writing, this is very important and can actually prevent lost data.
- The **seek** method of class **IO**, seeks to a given offset an **Integer** (first parameter of method) in the stream according to the value of second parameter in the method. The second parameter can be **IO::SEEK_CUR** - Seeks to first integer number parameter plus current position; **IO::SEEK_END** - Seeks to first integer number parameter plus end of stream (you probably want a negative value for first integer number parameter); **IO::SEEK_SET** - Seeks to the absolute location given by first integer number parameter.

Exercises

Please complete the Week 3 exercises and discuss in the Week 3 forum.

Quiz

Please attempt the quiz

Last modified: Saturday, 9 June 2012, 1:43 AM

You are logged in as C. O'Keefe (Logout)