**Part 1 - Installation**

**RVM & Ruby Setup**

<https://www.digitalocean.com/community/tutorials/how-to-install-ruby-on-rails-with-rvm-on-ubuntu-18-04>

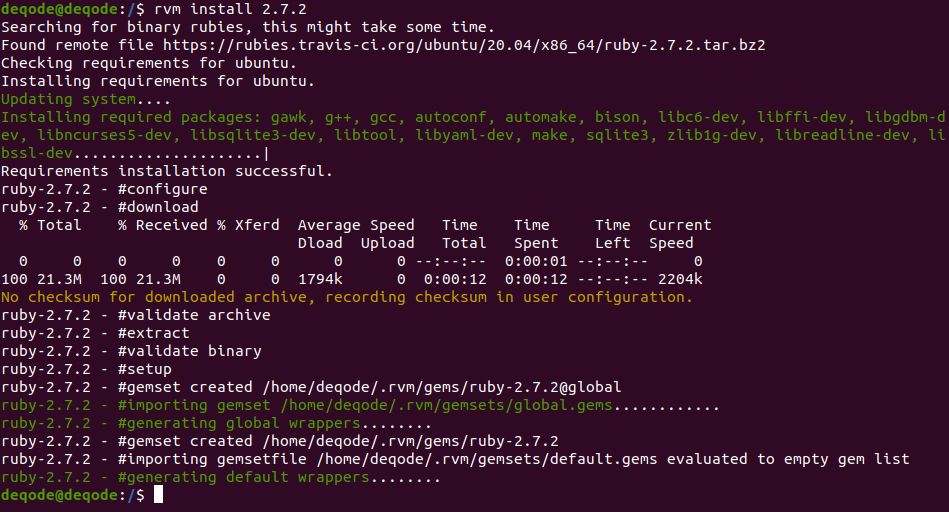
<https://rvm.io/rvm/basics>

The command-line tool **RVM** (**R**uby **V**ersion **M**anager) provides you with a solid development environment. RVM will let you manage and work with multiple Ruby environments and allow you to switch between them.

rvm list known

Then, install the specific version of Ruby that you need through RVM, where ruby\_version can be typed as ruby-2.4.0, for instance, or just 2.4.0:

rvm install ruby\_version



After the installation, we can list the available Ruby versions we have installed by typing:

rvm list

We can switch between the Ruby versions by typing:

rvm use ruby\_version

**Gemsets**

<https://www.rubyguides.com/2018/09/ruby-gems-gemfiles-bundler/#:~:text=A%20gem%20is%20a%20package,feature%20to%20your%20Rails%20app>

<https://guides.rubygems.org/what-is-a-gem/>

A gem is a package that you can download & install. When you require an installed gem you’re adding extra functionality to your Ruby program.

The RubyGems software allows you to easily download, install, and use ruby software packages on your system. The software package is called a “gem” which contains a packaged Ruby application or library.

Gems can be used to extend or modify functionality in Ruby applications. Commonly they’re used to distribute reusable functionality that is shared with other Rubyists for use in their applications and libraries. Some gems provide command line utilities to help automate tasks and speed up your work.

**Inside gems are the following components:**

Code (including tests and supporting utilities)

Documentation

gemspec .gemspec file is where you’ll find all the information about the gem.

------------------------------------------------------------------------------------------------------------

<https://riptutorial.com/ruby-on-rails/example/25818/gemsets#:~:text=A%20gemset%20is%20just%20a,worry%20about%20its%20own%20gems>.

**GEMSETS**

A gemset is just a container you can use to keep gems separate from each other. Creating a gemset per project allows you to change gems (and gem versions) for one project without breaking all your other projects. Each project need only worry about its own gems.

**Sample Query: Why use RVM? Can you define a few sets of commands that you have come across?**

<http://medium.com/@moulayjam/whats-an-rvm-and-why-you-should-use-it-for-rails-apps-b5cced2469a8#:~:text=RVM%20is%20a%20command-line,interpreters%20to%20sets%20of%20gems>.

Because RVM can manage multiple **ruby** and **rails** versions easily

rvm list known

rvm install 2.4.1

rvm list

rvm use ruby\_version

rvm use 2.4.1@gemSet\_name

**Part 2 - Literals**

**Data Types - string, boolean, number, symbol**

[**https://www.javatpoint.com/ruby-data-types**](https://www.javatpoint.com/ruby-data-types)A string is made up of multiple characters. They are defined by enclosing a set of characters within single (‘x’) or double (“x”) quotes.

Name =” Ajay Agrawal”

A number is a series of digits that use a dot as a decimal mark (where one is required). Integers and floats are the two main kinds of numbers; Ruby can handle them both.

Num = 33.45

The Boolean data type represents only one bit of information that says whether the value is true or false. A value of this data type is returned when two values are compared.

Check =true

Symbols are like strings. A symbol is preceded by a colon (:)

:abcd

They do not contain spaces. Symbols containing multiple words are written with (\_).

**Diff btw symbol and string**

One difference between string and symbol is that, if text is a data then it is a string but if it is a code it is a symbol.

Symbols are unique identifiers and represent static values, while string represent values that change.

**Array**

<https://next.tech/lessons/6b5f418c-4621-488f-8f1e-10aea3920413?access_token=A2620273D3C8C9A4FBC0E529BD508E88&step_id=b0372d24-9d68-454c-87f0-ade6d7a1491b>

Arrays are a common data structure used to store a collection of data types, including integers, floats, strings, and even other arrays.

An array can store multiple data items of all types. Items in an array are separated by a comma in-between them and enclosed within square brackets. The first item of the array has an index of 0.

x=[1,2,3]

y= Array.new

Y[0] = 5

y.push(4,5,6)

Y.pop # last item removed

y.delete(4) #all 4 are deleted from array

y.delete\_at(2) #3

**Hash**

<https://next.tech/lessons/6b5f418c-4621-488f-8f1e-10aea3920413?access_token=A2620273D3C8C9A4FBC0E529BD508E88&step_id=6184a529-42a2-4e10-a829-5523a346ad90>

The hash data structure is a powerful tool in Ruby programs. Hashes are key/value-based collections. Sometimes, the value itself can be another collection like an array or a hash. They let you access data elements with more than a pure index, such as with arrays.

languages = {

USA: "English",

France: "French",

China: "Chinese"

}

P language[:France]

Language.each\_key

Language.each\_value

to convert a hash to an array you could use either Array(languages) or languages.to\_a

**Part 3 - Ruby Syntax**

**Loops**

i=0

While i<5

p i

i +=1

end

----------------------------------------------------------

until i==5

p i

i +=1

End

-----------------------------------------------------------------

for i in range

p i

end

------------

The **next** keyword is used within a loop to pass over certain elements and skip to the following iteration. It is useful for omitting elements that you do not wish to have iterated. next is followed by an if statement which defines which elements are to be skipped.

for i in 1..10

next if i % 2 == 0

puts i

end

---------------------------------------------------------

Break : Terminates the most internal loop.

----------------------------------------------------------

Iterators:

To iterate over an array in Ruby, use the .each method. It is preferred over a for loop as it is guaranteed to iterate through each element of an array.

(1..7).each{|x| p x}

-----------------------------------------------------------------

.times {

}

------------------------------------

The select method is a powerful method that automatically iterates through a collection in a Ruby program and extracts the values you want to retrieve.

evens = (1...10).to\_a.select do |x|

x.even?

end

p evens # [2,4,6,8]

----------------------------------------------------------

.select : when we want to check for a certain condition then retrive

.map : when we want to perform certain actions before retrive

**Conditional**

If cond

stat1

end

-------------------------------------------------------------

If cond

stat1

else

stat2

End

-------------------------------------------------  
If

st1

Elsif

st2

Else  
 st3

End

------------------------------------------------------

Unless (reverse of if/else)

unless 1>2

P “1 is smaller”

Else  
 P “1 is greater”

end

-------------------------------------------------------

Case expression

When exp1

When exp2

Else

End

-------------------------------------------------------

**Operators**

<https://www.javatpoint.com/ruby-operators>

Assignment operators in Ruby are used to assign or update values to variables. The most common assignment operator is = but others also exist, like +=, -=, \*= and /=.

Math/Arithmatic operator + , \*, \*\*, /, %, -

Logical operators &&, ||

Ternary Operator ?: condition? True : false

Comparision Operators <, >, ==, !=, >=, <=

Range Operator

|  |  |
| --- | --- |
| .. | Range is inclusive of the last term |
| ... | Range is exclusive of the last term |

**Comments**

# single line   
-----------------------------------------

=begin

Multiple line

=end

**Methods**

<https://www.geeksforgeeks.org/ruby-methods/?ref=lbp>

Method is a collection of statements that perform some specific task and return the result. Methods allow the user to reuse the code without retyping the code. Methods are time savers and help the user to reuse the code without retyping the code.

def method\_name(var1, var2, var3)

# Statement 1

# Statement 2

.

.

end

Methods have a specific syntax. They begin with the def word followed by the name of your method, which should reflect its functionality or behavior for easy readability. All the words should be in lowercase and joined by an underscore. Methods always end with the end word, with the logic in the area between the name and end.

**file I/O**

[**https://www.javatpoint.com/ruby-file-io**](https://www.javatpoint.com/ruby-file-io)

**Open/ create a file object:**

There are two methods to open a file in Ruby:

File.new method : Using this method a new file can be created for reading, writing or

both.

File.open method : Using this method a new file object is created. That file object is

assigned to a file.

Difference between both the methods is that File.open method can be associated with a block while File.new method can't.

aFile = File.new("filename", "mode")

# ... process the file

aFile.close

Or

File.open("filename", "mode") do |aFile|

# ... process the file

end

-------------------------------------------------------------------------------

**Modes:**

"r": read-only mode is the default mode starts at beginning of file.

"r+": read-write mode, starts at beginning of file.

"w": write-only mode, either creates a new file or truncates an existing file for writing.

"w+": read-write mode, either creates a new file or truncates an existing file for reading and writing.

"a": write-only mode, if file exists it will append the file othrwise a new file will be created for writing only.

"a+": read and write mode, if file exists it will append the file othrwise a new file will be created for writing and reading.

Msg =”hello”

file = File.open('log.txt','a')

file.puts msg

file.write "writing in the file \n"

file << "hello something top write \n"

file.syswrite(“msg to write”)

------------------------------------------------------------------------------------------------

P file.sysread(40)

T**here are three different methods to read a file.**

To return a single line, following syntax is used.

f.gets

code...

To return the whole file after the current position, following syntax is used.

f.read

code...

To return file as an array of lines, following syntax is used.

f.readlines

[code...]

----------------------------------------------------

File.rename("log.txt", "msg\_log.txt")

File.delete("msg\_log.txt")

**Sample Query: Write down some of the IMPORTANT reserved keywords that you came across while exploring?**

def

end

=begin

=end

While , until, unless,

do , if , case , when

P , print, puts