CSCI 2320: Principles of Programming Languages

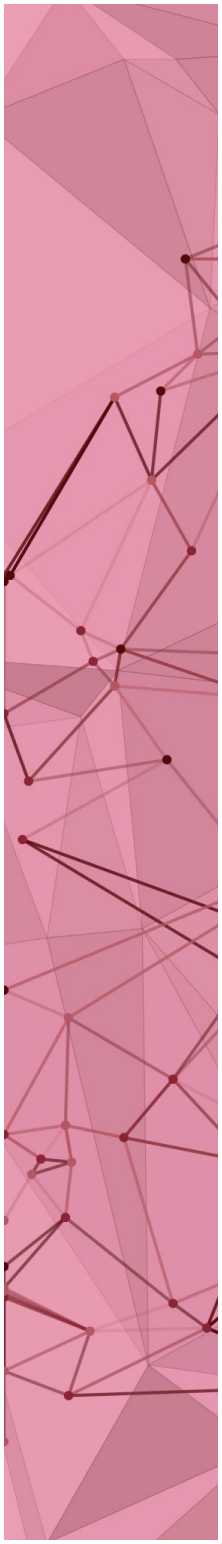
OOP Paradigm:

Ruby

Mohammad T. Irfan

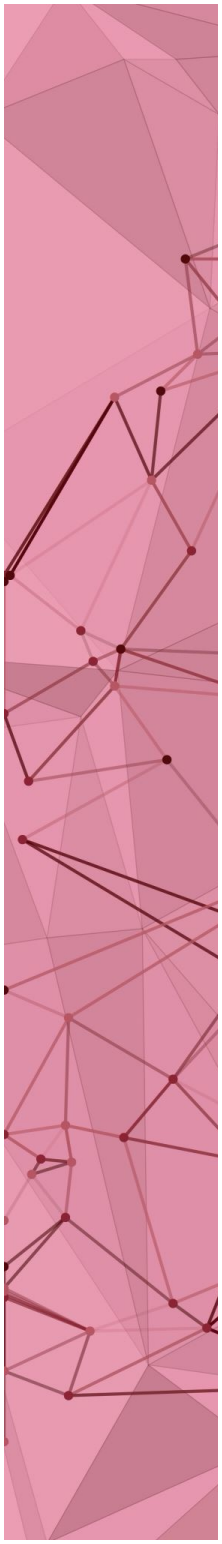
Ruby resources

- Installation– see class website
- Learning
  - http://ruby-doc.org/
  - English translation of the creator's user guide (by Mark Slagell)
    - https://ruby-doc.org/docs/ruby-doc-bundle/UsersGuide/rg/
  - Other good reference
    - http://www.tutorialspoint.com/ruby/
  - Interactive tutorial using only your web-browser
    - https://try.ruby-lang.org/



Origin

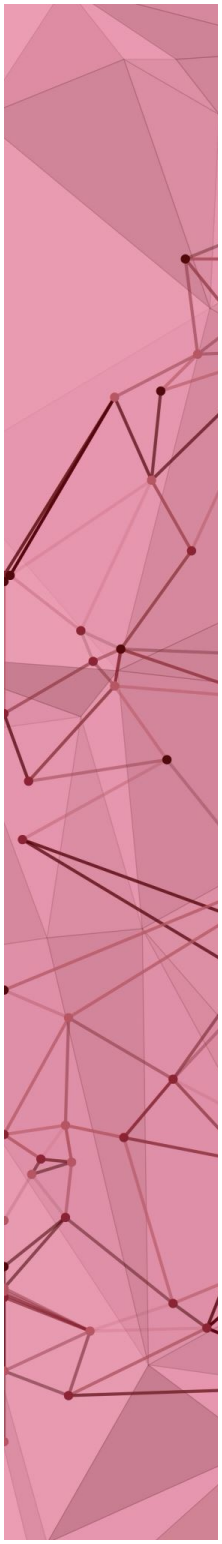
- Designed by Yukihiro Matsumoto (Matz) in early 1990s
- Inspired by Perl and Python
  - Less scripting than Perl
  - More object-oriented than Python
  - Happy experience!



Bruce Stewart (2001): Did you have a guiding philosophy when designing Ruby?

Matz: Yes, it's called the "principle of least surprise." I believe people want to express themselves when they program. They don't want to fight with the language. Programming languages must feel natural to programmers. I tried to make people enjoy programming and concentrate on the fun and creative part of programming when they use Ruby.

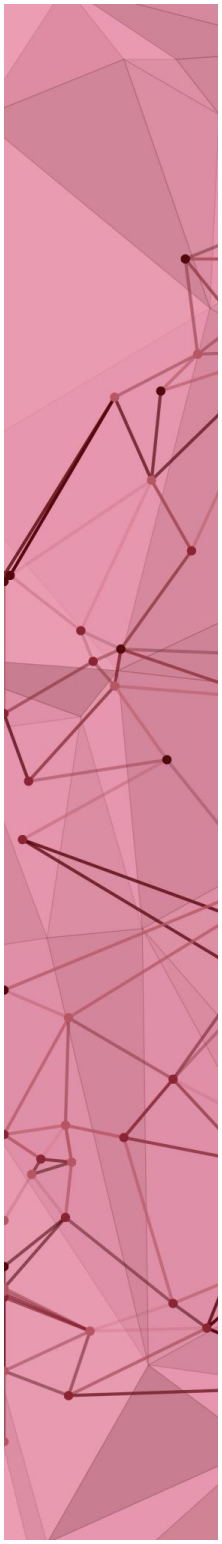
(https://ruby.fandom.com/wiki/Ruby)



Bill Venners (2003): In an introductory article on Ruby, you wrote, "For me the purpose of life is partly to have joy. Programmers often feel joy when they can concentrate on the creative side of programming, So Ruby is designed to make programmers happy." How can Ruby make programmers happy?

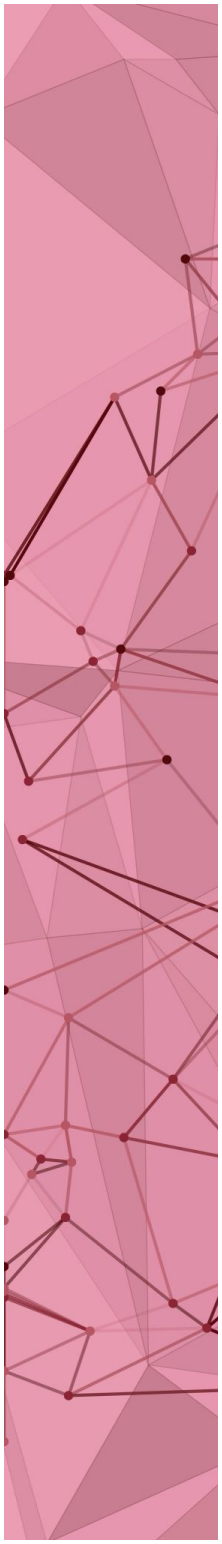
Matz: You want to enjoy life, don't you? If you get your job done quickly and your job is fun, that's good isn't it? That's the purpose of life, partly. Your life is better.

I want to solve problems I meet in the daily life by using computers, so I need to write programs. By using Ruby, I want to concentrate the things I do, not the magical rules of the language, like starting with public void something something something to say, "print hello world." I just want to say, "print this!" I don't want all the surrounding magic keywords. I just want to concentrate on the task. That's the basic idea. So I have tried to make Ruby code concise and succinct.
(http://www.artima.com/intv/ruby.html)



Interview of Matz

http://vimeo.com/52954702



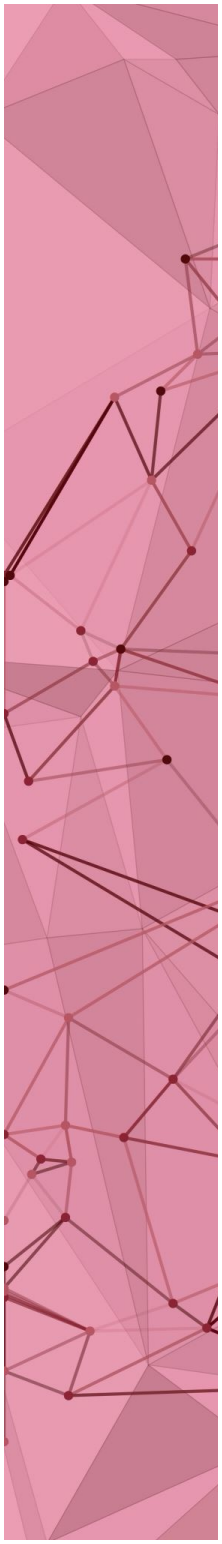
Features

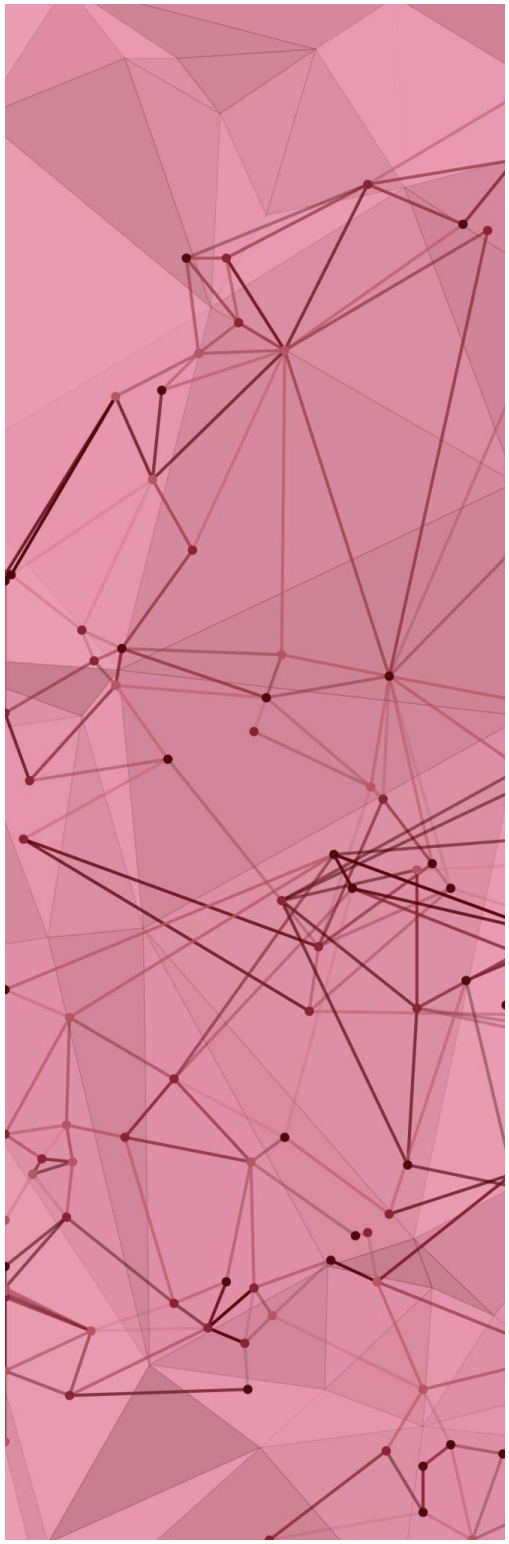
- **Purely** object oriented
  - Every data value is an object – no primitive type
  - Every subroutine is a method
  - Inheritance can be applied to any class
- Both classes and objects are **dynamic**!
  - Can add methods to classes and objects dynamically
  - Different objects of the same class can behave differently

- **Dynamically** typed
- **Static** scoping

You should be able to explain these!

- 37 reasons to love Ruby!
  - https://www.bowdoin.edu/~mirfan/Ruby37.html

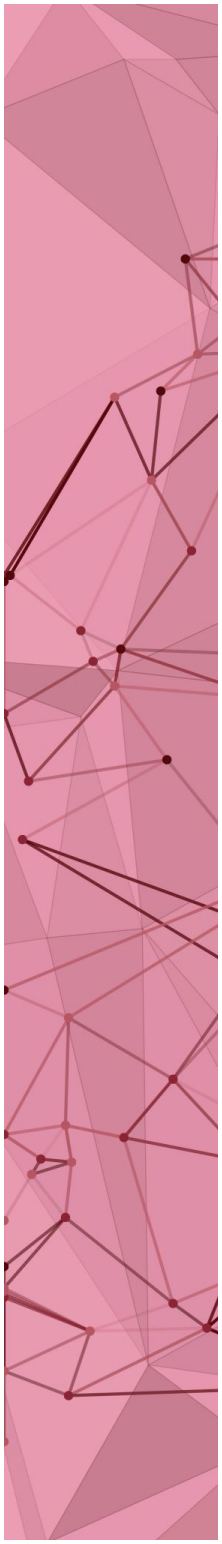




Let’s code in Ruby

Interactive Ruby in Terminal

- If you want to quickly check something without writing a program
  - Use the **irb** command in Terminal
- Examples in **irb**
  - x = 10
    if x % 2 == 0
      puts "Even"
    else puts "Odd"
    end
  - What does nil mean in the output? In Ruby, there is no statement. Everything is an expression returning a value, whether you explicitly say return or not.
  - x = ["NFL", "NBA", 100]
    x.class
    x.class.methods
    x.include? "NBA"
    x.include? 200

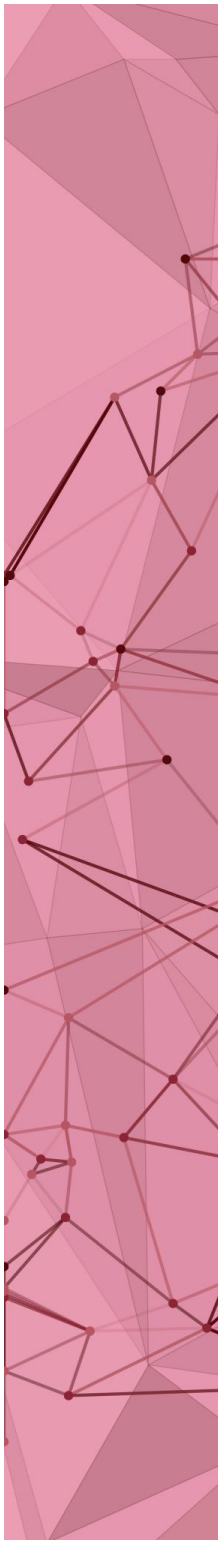


Variables

- Type is implicit
- Type can be changed dynamically
- Naming starts with:

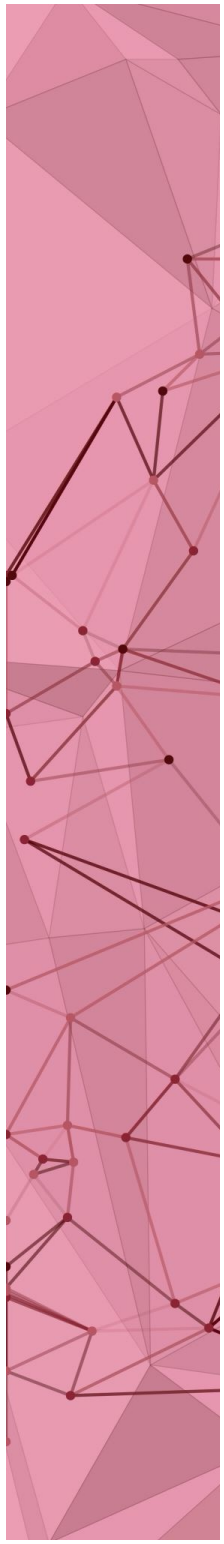
$	Global variable
@	Instance variable
[a-z] or _	Local variable
[A-Z]	Constant (only first letter needs to be uppercase)

- Examples (in irb)
  - x = 10.99
    x.class #prints Float
    x = "Hello Ruby!"
    x.class #prints String
  - Very rich **String** class
    - Examples: https://ruby-doc.org/core-2.6/String.html



Arrays (mutable): Creation, insertion, deletion

- myArray = ["NFL", "NBA", 100]
- myString = myArray.join(" ") #outputs "NFL NBA 100"
- left = myArray.shift #left has value "NFL"
- myArray #myArray is now ["NBA", 100]
- myArray.push("MLS") #myArray is now ["NBA", 100, "MLS"]
- myArray.unshift("NFL")
  #myArray is now ["NFL", "NBA", 100, "MLS"]
- **delete(element), delete_at(index), delete_if { |item| block }**
  - #block represents a condition for deleting item
    scores = [97, 42, 75]
    scores.delete_if {|score| score < 80 } **#=> [97]**

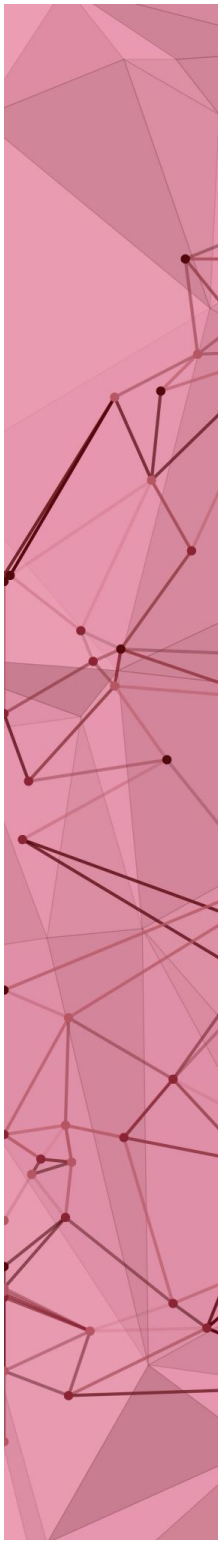


Arrays: accessing elements

- myArray[0] #"NFL"
- myArray[0..-1] #everything in the array
- myArray.each {|item| puts item} #iterate through array items
  - Equivalently:

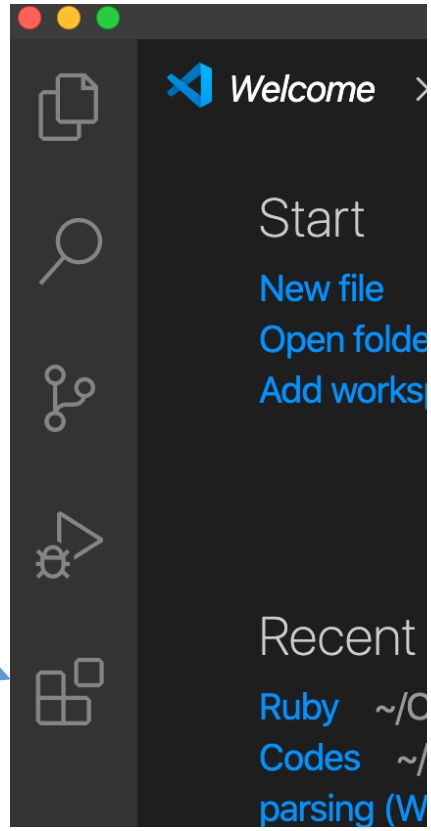
```
myArray.each do |item|
        puts(item)
end
```

- myArray.each_index {|i| print i, "->", myArray[i], "\n"}

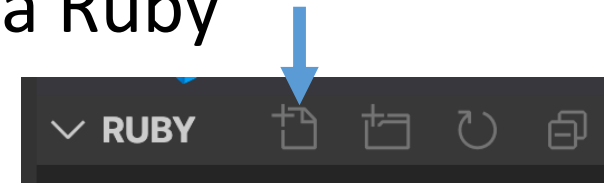


Coding in Ruby with VS Code

- Set up Ruby extension
  - Click on the Extension icon
  - Type Ruby in search box
  - Install Ruby by Shopify



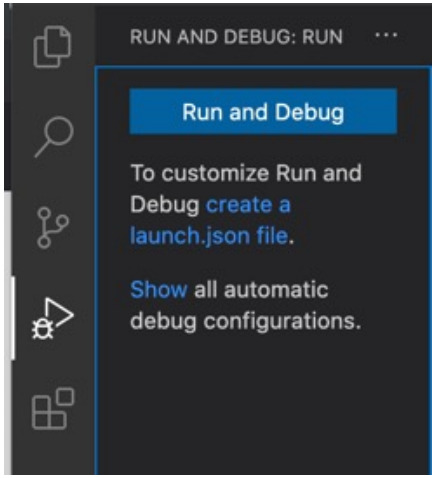
- Make a folder in your computer for Ruby codes
- VS Code → File → Open Folder, browse to the folder you just created and open it
- Click on the new file icon to create a Ruby source file with .rb extenstion



Running Ruby in VS Code

Two ways:

1. Open a new terminal *within* VS Code: Terminal → New Terminal
   - Then run ruby source.rb command in terminal, where source.rb is your source code
2. Alternatively, click on the debug (triangle) icon on the left and select "create a launch.json file", select Ruby for environment, and then choose "Debug Local File."
   - Replace main.rb by your source file name in launch.json
   - Run the code by: Run → Run Without Debugging
   - Further editing of launch.json required for debugging (need to supply environment info)
   - Note: VS Code Debugger currently cannot handle user input (like gets)



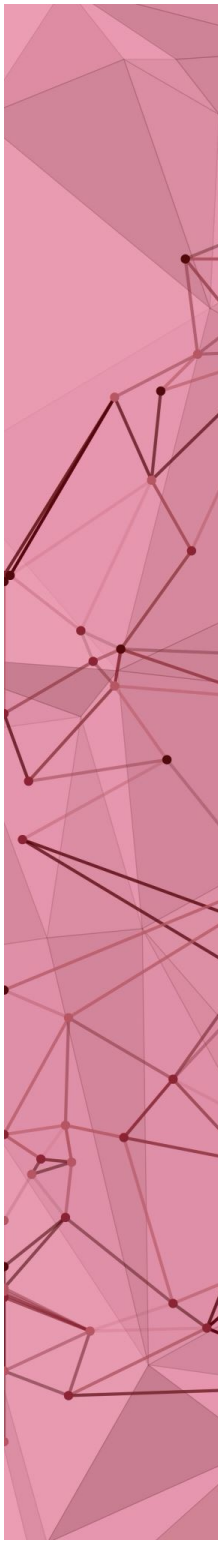
Sample program: factorial

- Save it as source.rb

```
def fact(n)
  if n == 0
    1
  else
    n * fact(n-1)
  end
end
```

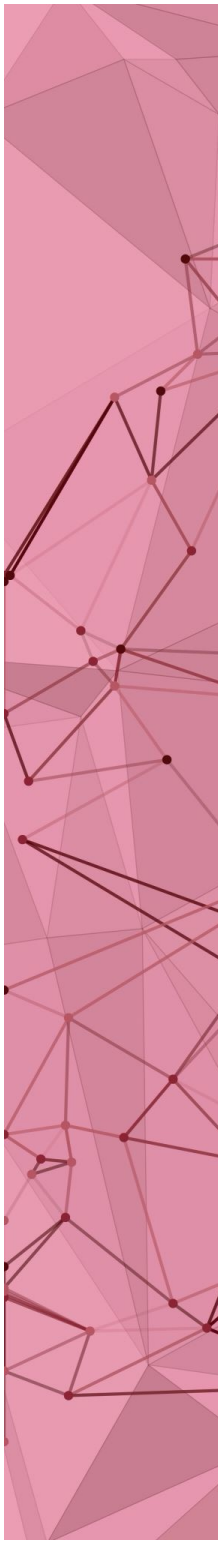
- Two ways to run in terminal
  1. Add puts fact(10) at the end of source.rb and run it using the ruby source.rb command.
  2. ruby -I ./ -r source.rb -e "puts fact(10)"

  Command line arguments are also supported

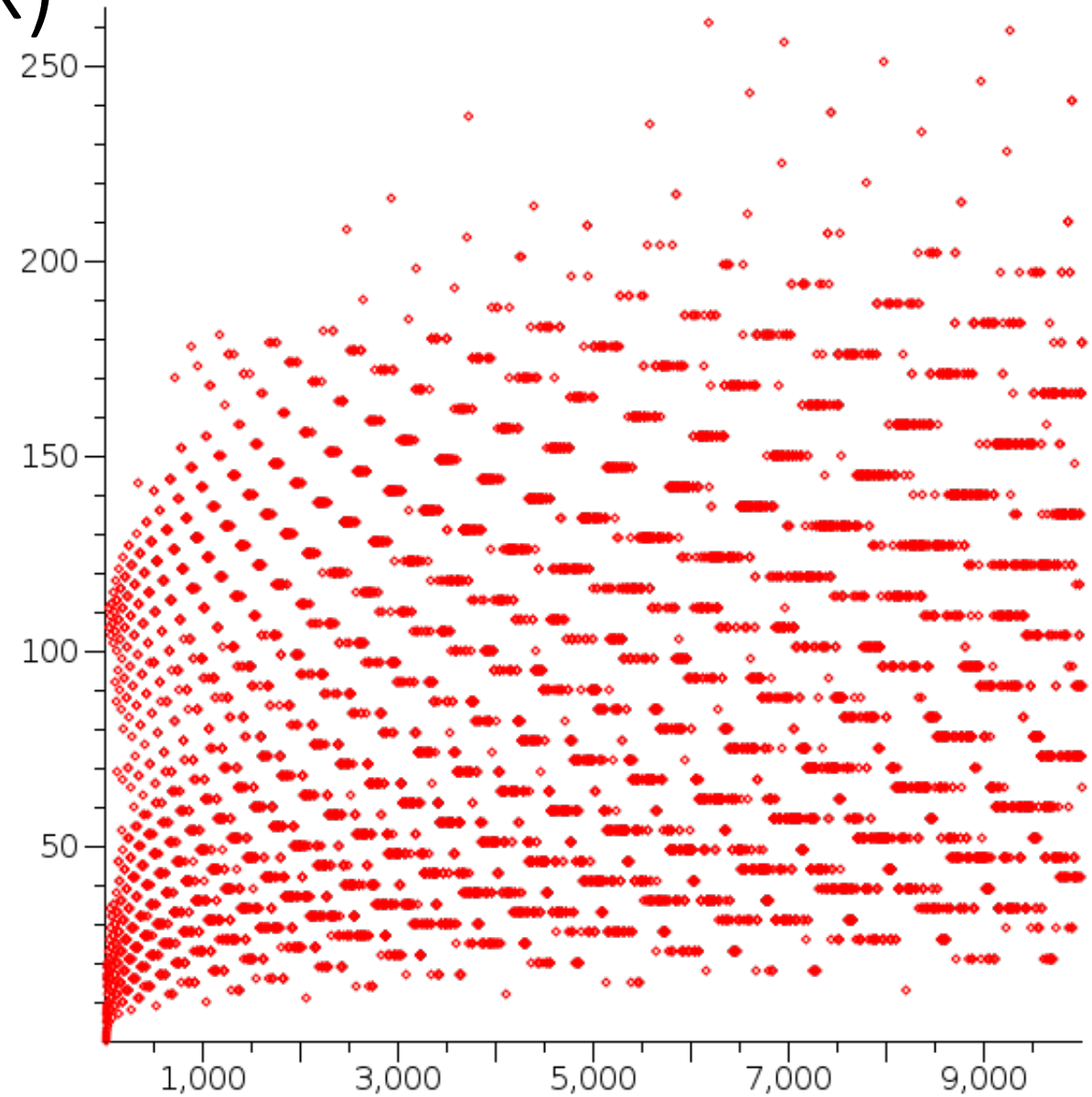


Problem: Collatz Conjecture

- From Wikipedia http://en.wikipedia.org/wiki/Collatz_conjecture
- Take any integer $n > 0$ as input. The conjecture is that no matter what n is, you will always eventually reach 1 if you follow this procedure:
- If n is even, assign $n = n/2$.
- If n is odd, assign $n = 3n + 1$.
- Repeat the process until you reach n = 1 (conditional statements and loops)
  - (Extra job) Print all values of n to a file
- The # of steps is called the cycle length of n
  - Output the cycle length (to standard output)
  - (Extra job) Also write it to the file

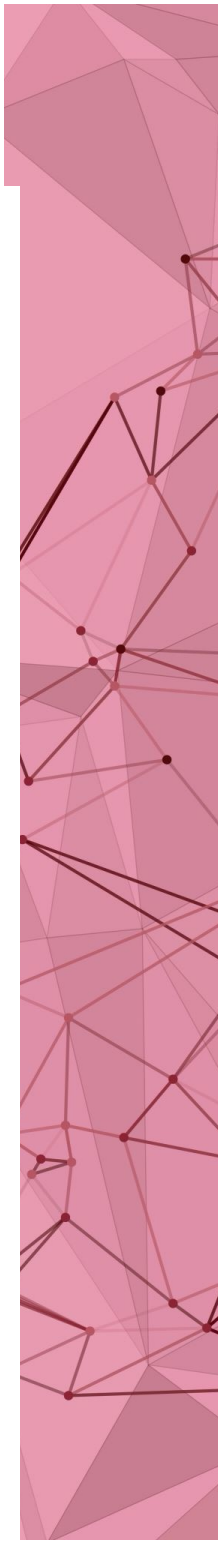


of steps (y) vs. input number (x)

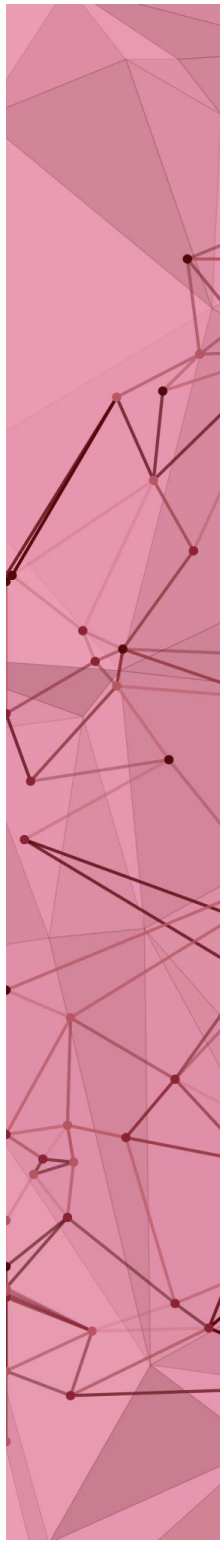


collatz.rb

```ruby
=begin
 Colltaz conjectore
 Print cycle length
=end #end of multiline comment
while true #get a valid input
  print "Input n > 0: " #No semicolon needed
  n = gets.chomp.to_i #get string -> delete \n -> int
  break if n > 0
end

puts "Input #{n} is #{n.class}" #print with \n

#Open an output file
out_file = File.new("collatz_#{n}.txt", "w")
if out_file
  out_file.write("#{n} -> ") #Write to file
else #Show error msg and exit
  raise "File error"
end
```



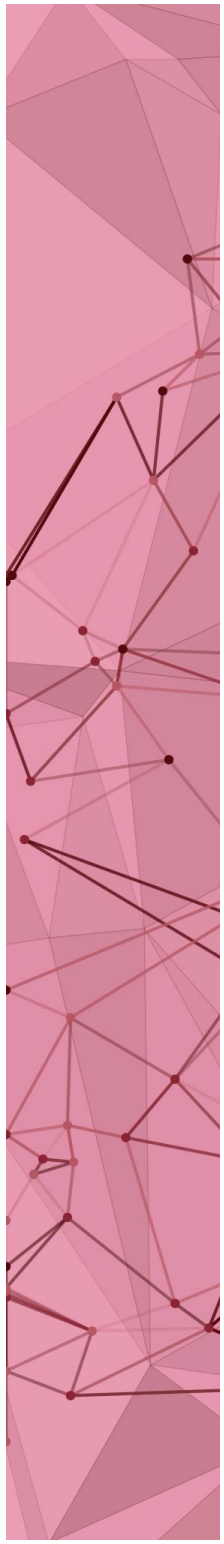
```
20 cycle_len = 0
21 while n != 1 #No parentheses needed
22   if n%2 == 0
23     n /= 2 #Indentation is not needed
24   else
25     n = 3*n + 1
26   end #end of if statement: ADA style
27   out_file.write("#{n} -> ")
28   cycle_len += 1
29 end #end of while
30
31 puts "Cycle length = #{cycle_len}"
32
33 out_file.close
```



Review:
What's new in Ruby? (vs. Java/C++)

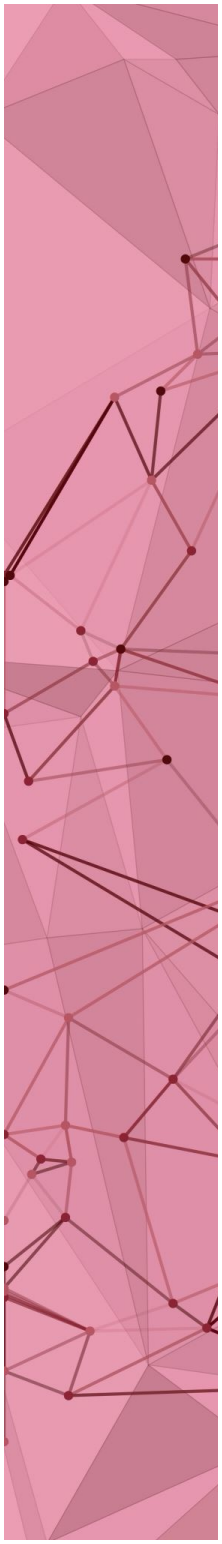
- **Purely** object oriented
- Classes and objects are **dynamic**
- Class can be defined **later, dynamically**

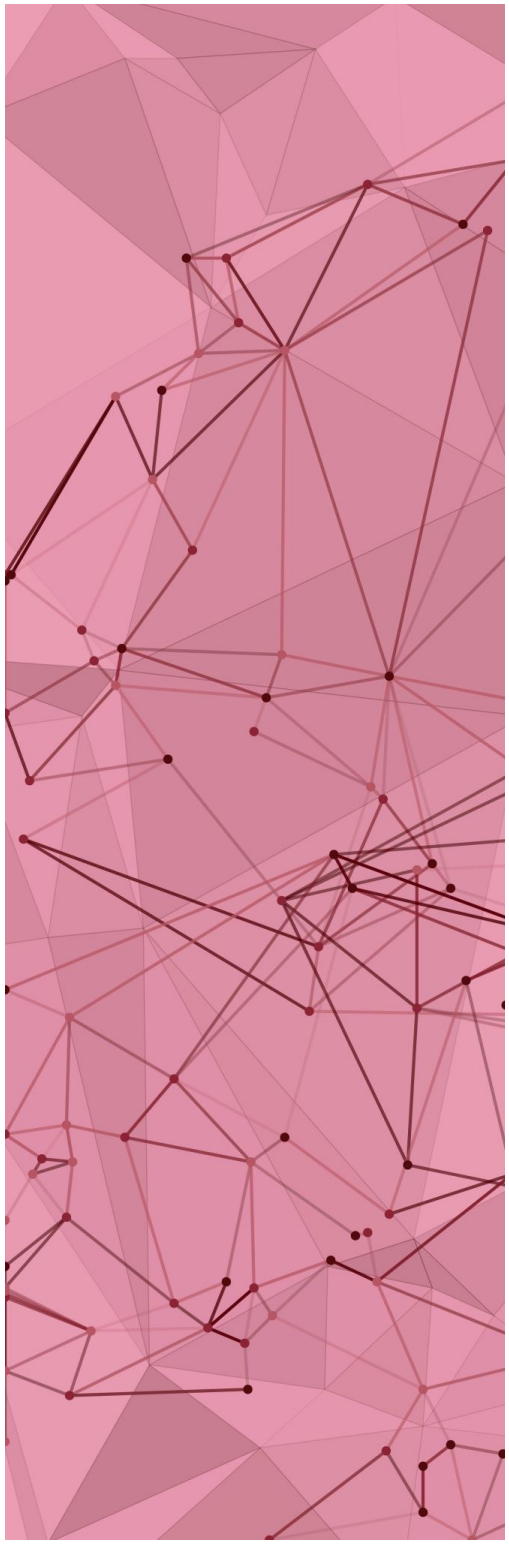
$	Global variable
@	Instance variable
[a-z] or _	Local variable
[A-Z]	Constant (only first letter needs to be uppercase)



Control structure

- Conditional
  - if – elsif – else – end
  - ---- if *condition*
- Iteration
  - Usual while loops
  - arrayName.each do |item|
    ...
    end
  - arrayName.each { |item| ...}
  - Other ways: for loop
    for i in 0..4
    ...
    end



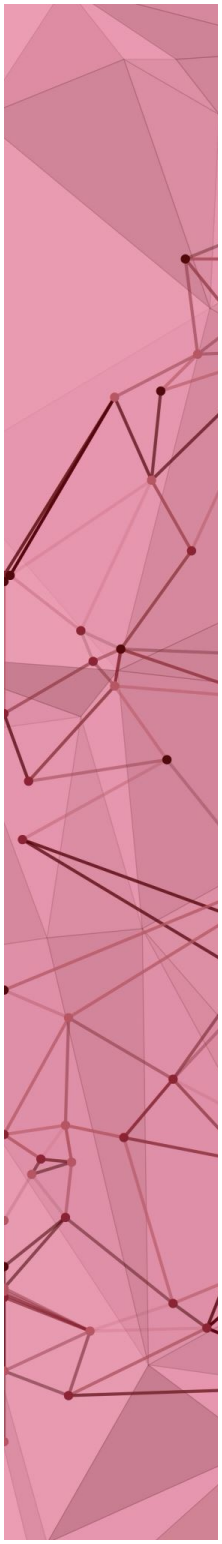


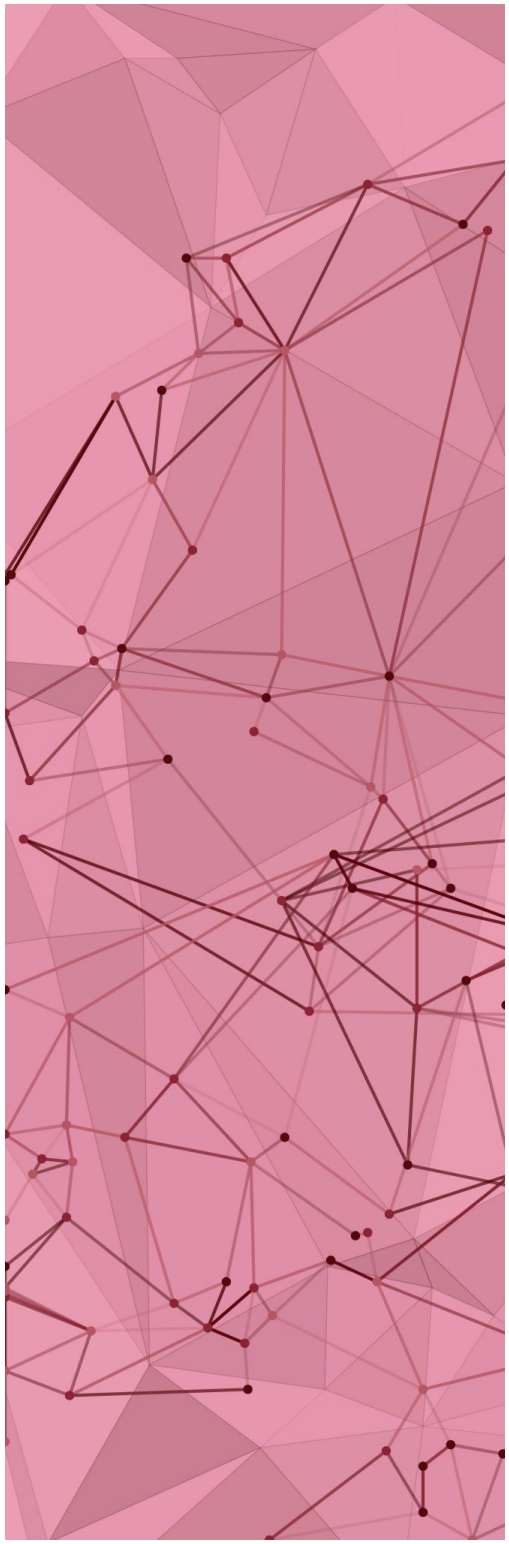
Cool stuff:
Reading a website

Reading a website

Alternative: HTTParty “gem”

```
1  load "open-uri.rb" #Standard library for URI
2
3  url = "http://www.bowdoin.edu"
4  begin
5    webPage = URI.open(url).read
6    #Error above? Termination model: control goes to rescue
7    puts webPage.class #String
8    puts "Here goes the lines from the web page"
9    webPage.each_line do |line| #Iterator provided by String
10     print line
11   end
12 rescue #Exception handler
13   puts "There's an error:"
14   puts "\t#{$!}"
15 end
16 puts "\nEnd of my program"
```





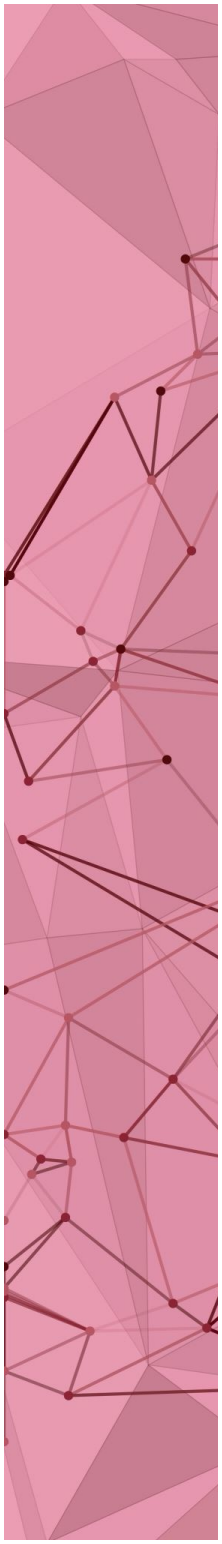
More fun:
Can we "crawl" the web?

Extract all links from a web page

Crawling a website

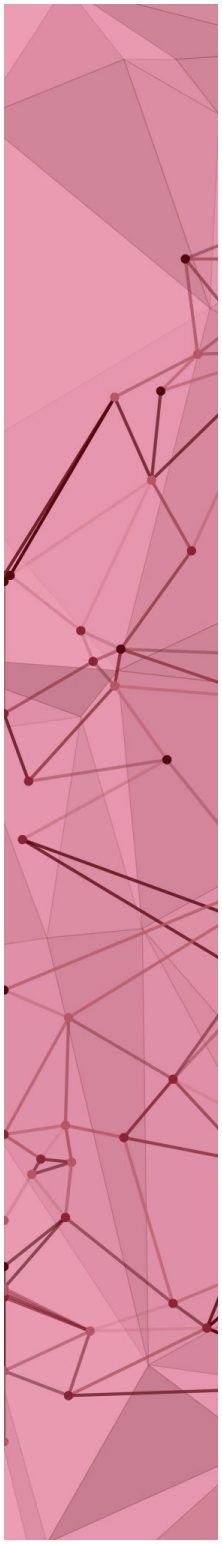
Check out: rubular.com

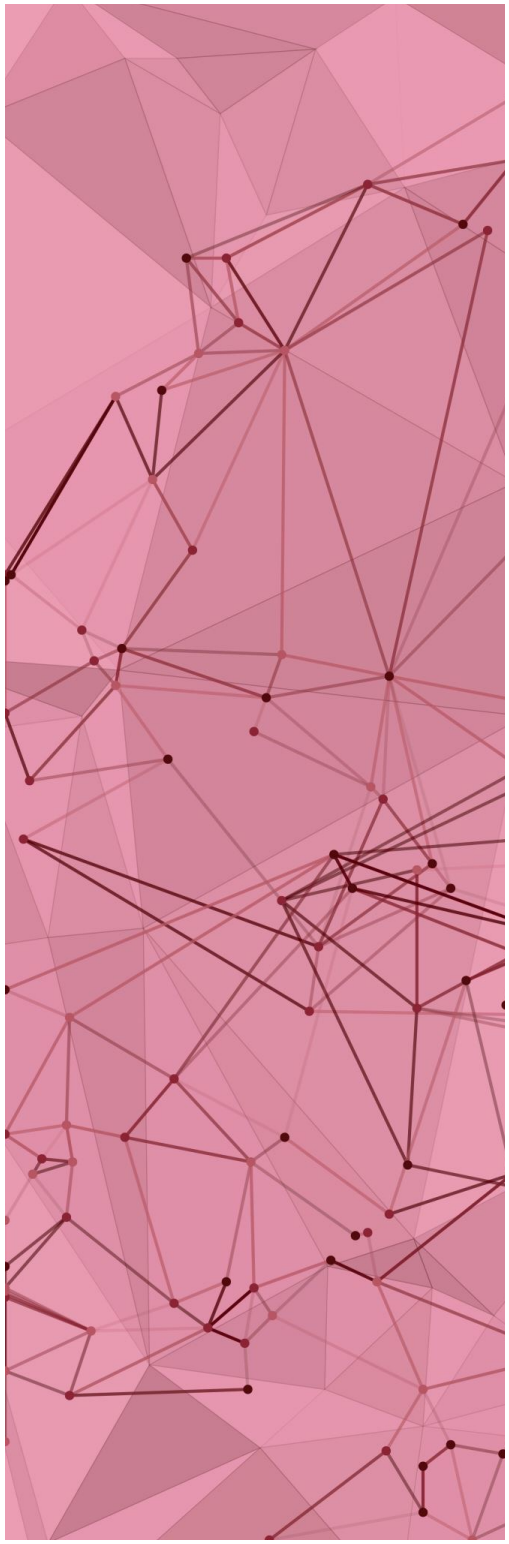
```ruby
load "open-uri.rb" #Standard library for URI

def getLinks(url, str)
  links = str.scan(/<a\s+href="(.+?)"/i).flatten
  links.each do |oneLink|
    oneLink.insert(0, url) if oneLink.start_with?("/")
  end
end

url = "http://www.bowdoin.edu"
begin
  webPage =  URI.open(url).read
  #Error above? Termination model: control goes to rescue
=begin #multiline comment
  puts webPage.class #String
  puts "Here goes the lines from the web page"
  webPage.each_line do |line| #Iterator provided by String
    print line
  end
=end
  links = getLinks(url, webPage)
  puts links
  puts "# of links = #{links.size}"
rescue #Exception handler
  puts "There's an error:"
  puts "\t#{$!}"
end
puts "\nEnd of my program"
```



"Gem" for crawling the web

- **nokogiri**: gem for parsing web pages
- Command line: $ gem install nokogiri
- Tutorials
  - https://oxylabs.io/blog/web-scraping-with-ruby
  - https://www.scrapingbee.com/blog/web-scraping-ruby/



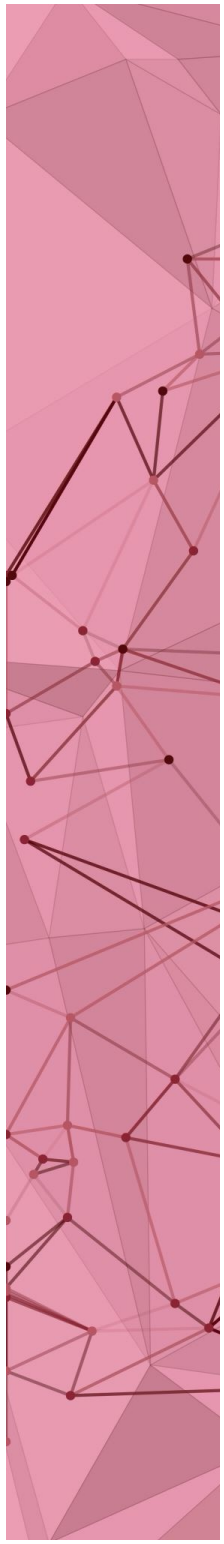


Object-oriented features

Open class

Can add a method to an existing class

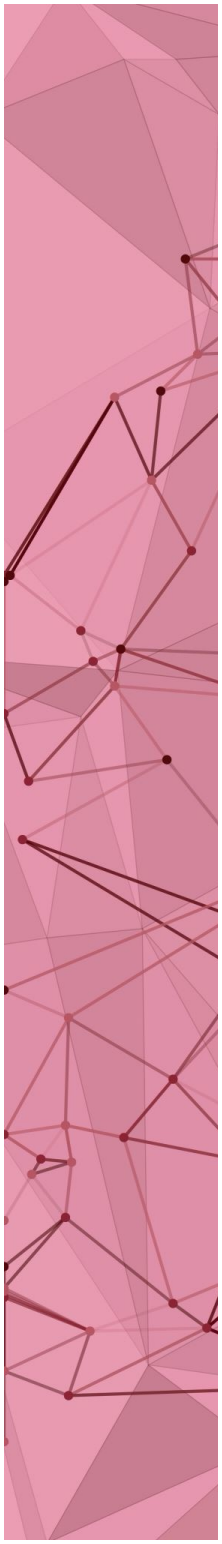
```
class Array
     def summarize
          self.each do |x|
               print x, " "
          end #iterator
          print "\n"
     end #def
end #class
```



Open class example

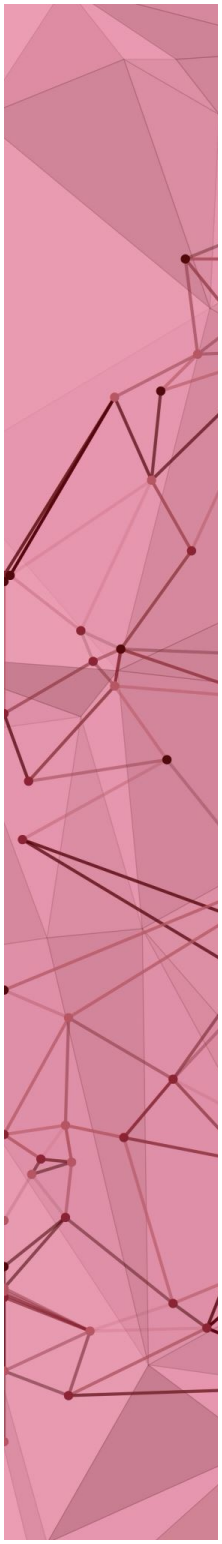
```
1  x = ["NFL", "NBA"]
2
3  class Array
4    def summarize
5          self.each do |x|
6                print x, " "
7          end
8          print "\n"
9    end
10 end
11
12 x.summarize
13
14
15 class Array
16   def addNumbers
17         total = 0
18         self.each do |x|
19               total += x
20         end
21         total #also: return total
22   end
23 end
24 y = [10, 20, 30]
25 puts y.addNumbers
```

```
NFL NBA
60
```



In Matz's words… [Artima]

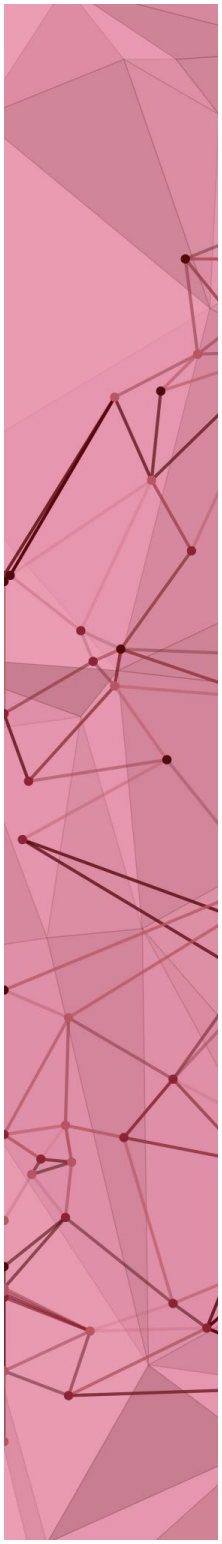
- **Bill Venners**: In Ruby, I can add methods and variables to objects at runtime. ... But in Java, for example, once a class is loaded or an object is instantiated, its interface stays the same. Allowing the interface to change at runtime seems a bit scary to me. ... What's the benefit of being able to add methods at runtime?
- **Yukihiro Matsumoto**: First of all, you don't have to use that feature. The most useful application of dynamic features, such as adding methods to objects, is meta-programming. Such features allow you to create a library that adapts to the environment, but they are not for casual uses.



Naming rules

Starts with	Category of variable
$	Global variable
@	Instance variable
@@	Class variable
[a-z] or _	Local variable
[A-Z]	Constant

Next: Classes in Ruby – the usual stuff



Instance variables and permissions

```
class Animal
  attr_reader :name #delete this and see
  def initialize(animal_name)
    @name = animal_name
  end
end

a = Animal.new("dog")
b = Animal.new("cat")

#How can we do the following?
#a.name = "horse"

puts a.name
puts b.name
```

- No declaration needed
- Dynamically appended to object when it's first referenced (even if the constructor doesn't reference it)
- Permission levels: attr_reader, attr_writer, attr_accessor
- Contrast: Java, Python

Website.
rb

```ruby
load "open-uri.rb"

#The Website class models a single website
class Website
  @@protocol = "http" #Class variable shared by all objects (unused here)
  attr_reader :url, :broken #Set read permission-- "getter methods"
  #Similar: attr_writer for "setters" and attr_accessor = reader & writer

  def initialize(url) #Constructor
    @url = url #Instance variable: different objects may have diff values
    @broken = nil #Another instance variable
  end

  def broken? #Is the website broken? Q: Where is the return value?
    if @broken == nil #Status unknown
      begin
        content = URI.open(@url).read
        @broken = false #When there's no exception
      rescue #Exception handler
        @broken = true
      end
    else
      @broken
    end #End of if
  end

  def showInfo
    print "Is ", @url, " broken? ", broken?, "\n"
  end
end #End of Website class

w = Website.new("http://www.bowdoin.edu/invalid")
w.showInfo
puts "URL: #{w.url}" #Can read it
#x.broken = false #Error: Can't write
```

Classes in Ruby

www.ruby-doc.org/core-2.0.0/

Classes filter...

- C Array
- C Bignum
- C BasicObject
- C Object
- C Module
- C Class
- C Complex
- C Complex::compatible
- C NilClass
- C Numeric
- C String
- C Float
- C Fiber
- C FiberError
- C Continuation

Methods filter...

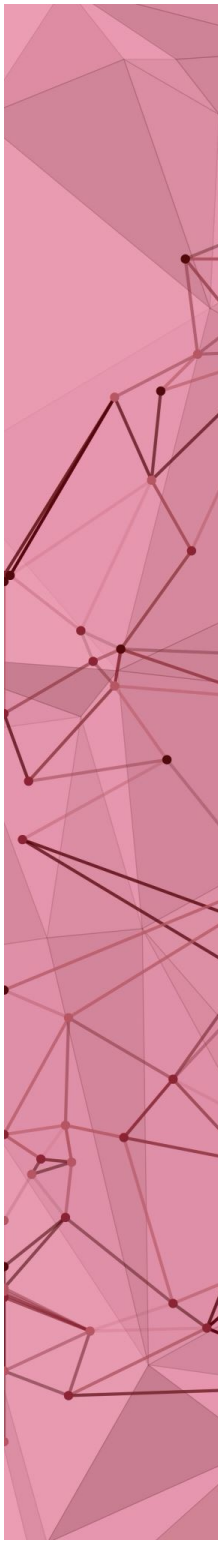
- :: === (SystemCallError)
- :: DEBUG (Thread)
- :: DEBUG= (Thread)
- :: [] (Array)
- :: [] (Dir)
- :: [] (ENV)
- :: [] (Hash)
- :: []= (ENV)
- :: _id2ref (ObjectSpace)
- :: abort (Process)
- :: abort_on_exception (Thread)
- :: abort_on_exception= (Thread)
- :: absolute_path (File)
- :: acos (Math)
- :: acosh (Math)

Classes in Ruby: Surprise!

"Classes in Ruby are first-class objects—each is an instance of class Class."

-- Ruby documentation

What does it mean?



```ruby
load "open-uri.rb"

#The Website class models a single website
Website = Class.new do
  @@protocol = "http" #Class variable shared by all objects (unused here)
  attr_reader :url, :broken #Set read permission-- "getter methods"
  #Similar: attr_writer for "setters" and attr_accessor = reader & writer

  def initialize(url) #Constructor
    @url = url #Instance variable: different objects may have diff values
    @broken = nil #Another instance variable
  end

  def broken? #Is the website broken? Q: Where is the return value?
    if @broken == nil #Status unknown
      begin
        content = URI.open(@url).read
        @broken = false #When there's no exception
      rescue #Exception handler
        @broken = true
      end
    else
      @broken
    end #End of if
  end

  def showInfo
    print "Is ", @url, " broken? ", broken?, "\n"
  end
end #End of Website class

```

Delete this line: class variable is static

We can create classes dynamically (just like any other object)

Modifying a class dynamically

Website.rb

After the previous code that defines the Website class and creates an object of it, copy the following code.

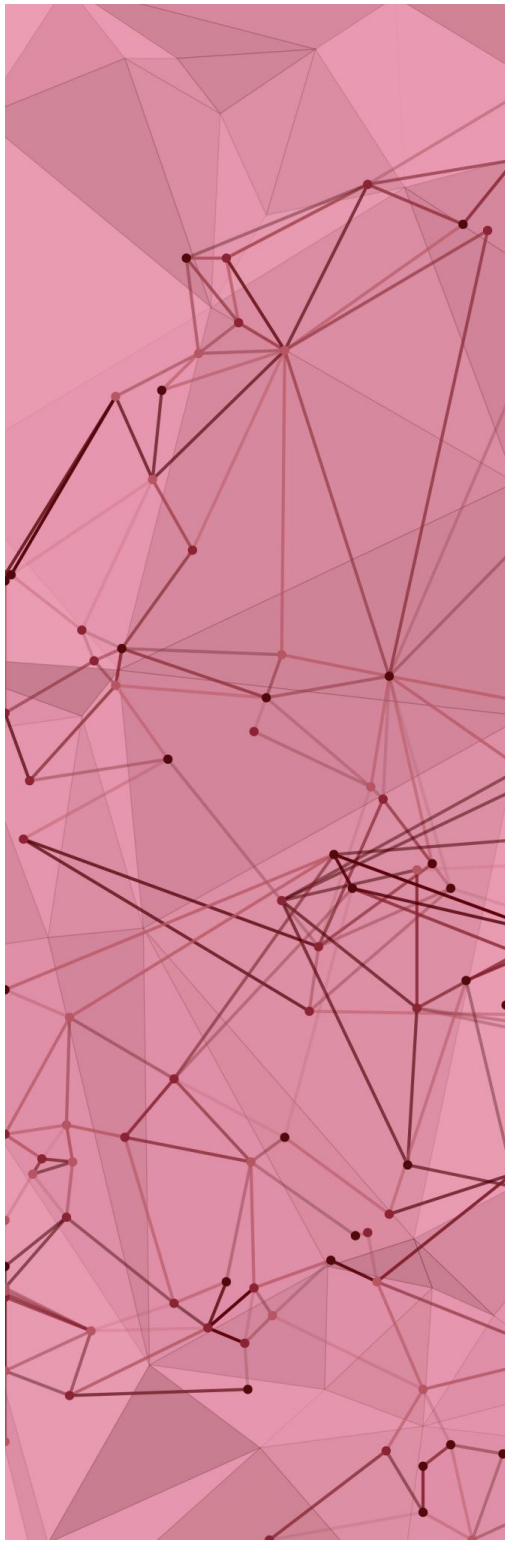
```
38 #Modify the previously defined Website class
39 class Website
40   attr_accessor :is_pdf
41
42   def pdf? #Sets and returns the value of is_pdf
43     if @url =~ /.+\.(pdf)$/i
44       @is_pdf = true
45     else
46       @is_pdf = false
47     end
48   end
49 end
50
51 puts "Is it pdf? #{w.pdf?}"
```

Q. How does @is_pdf work?
Q. What is =~?

Modify a specific object dynamically! (Not the whole class)

Singleton method

```
53  #Create a new object of the website class
54  w2 = Website.new("http://www.bowdoin.edu/president")
55
56  #Define a president method only for the object w2
57  def w2.president
58    @president_name = "Safa Zaki"
59  end
60
61  puts w2.president
62  #puts w.president #Error: No such method
```



Inheritance

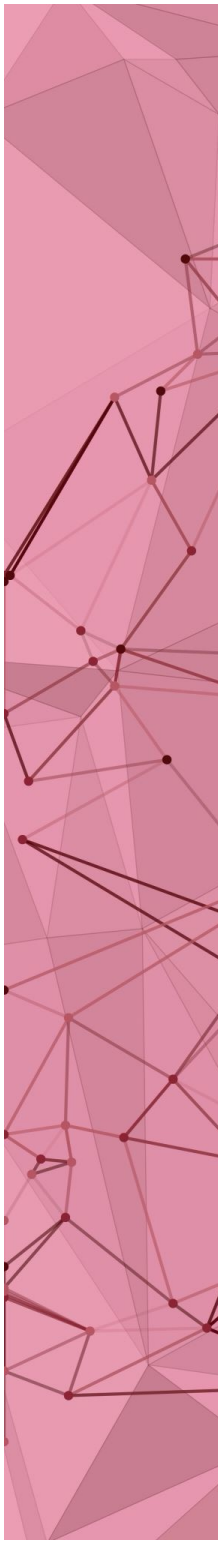
Inheritance: the usual stuff

Website.rb

```ruby
#PersonalWebsite class inherits Website
class PersonalWebsite < Website
  attr_reader :owner #New attribute of PersonalWebsite (not Website)

  def initialize(url)
    super(url) #Call the base class's constructor method

    if url =~ /~(.+)$/i || url =~ /~(.+)\//i
      @owner = $1
    else
      @owner = nil
    end
  end
end

#Create a PersonalWebsite object. Show new and inherited attributes
myWebsite = PersonalWebsite.new("http://www.bowdoin.edu/~mirfan")
puts "Owner: #{myWebsite.owner}, URL: #{myWebsite.url}"
```

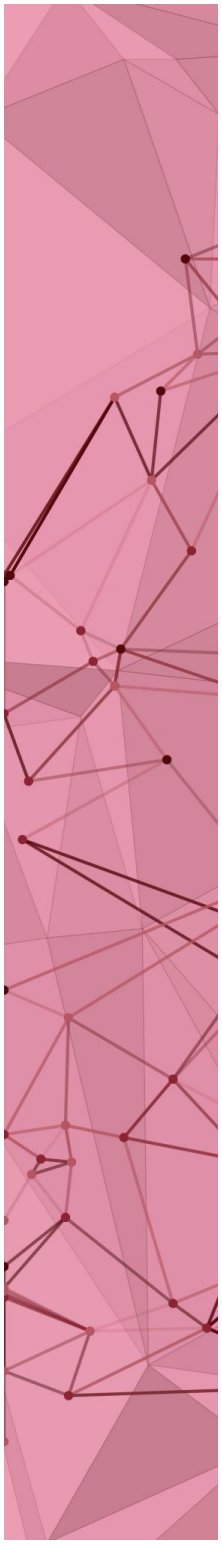
No multiple inheritance

Matz: "Single inheritance is good because the whole class inheritance structure forms a single tree with a single root, named Object, and that is very easy to understand. In languages with multiple inheritance, the classes form a network, which is harder to understand."



Inheritance: cool stuff!

- **Mix-in**: multiple inheritance in some sense
  - Share the *behavior, not data*
- Building block: **module**
  - Collection of methods and constants
  - Unlike a class, modules cannot be instantiated
  - Example: Math



Mix-in example

Mixin.rb

```
module A
        PI = 3.14
        E = 2.718
        def printPI
                puts PI
        end
        def printE
                puts E
        end
end
```

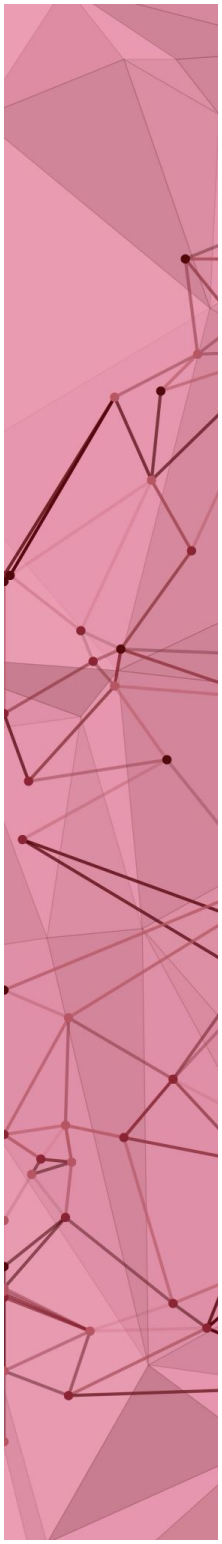
```
module B
      PI = 3.14159
      def printPI
            puts PI
      end
end
```

```
class C #mix-in class
      include A
      include B
end
c = C.new
c.printPI #=> 3.14159
c.printE #=> 2.718
```

Collision!

Matz on Mix-ins

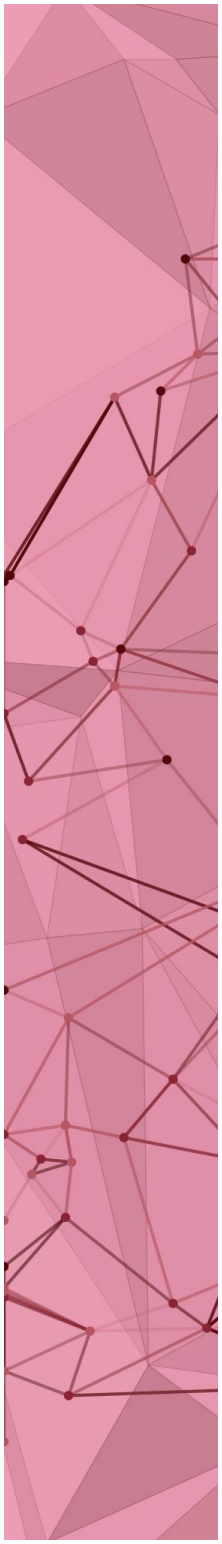
Matz: "[...] approach of plugging in modules is called Mix-ins. Mix-ins originally started in LISP culture as a usage of multiple inheritance. In fact, a mix-in is actually a strict way of using multiple inheritance. So, in LISP, it's a style of using multiple inheritance. In Ruby, we force you to use mix-ins by supporting classes and modules."



Another example of modules: Singleton design pattern

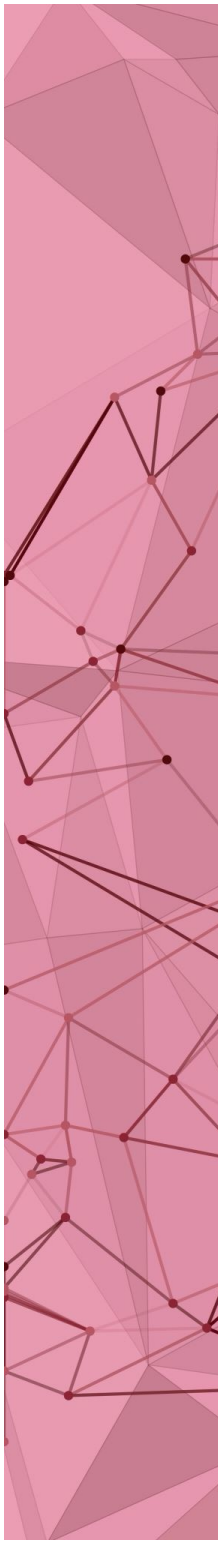
- What's singleton design pattern? (popular job interview Q) https://en.wikipedia.org/wiki/Singleton_pattern
- Use predefined Singleton module

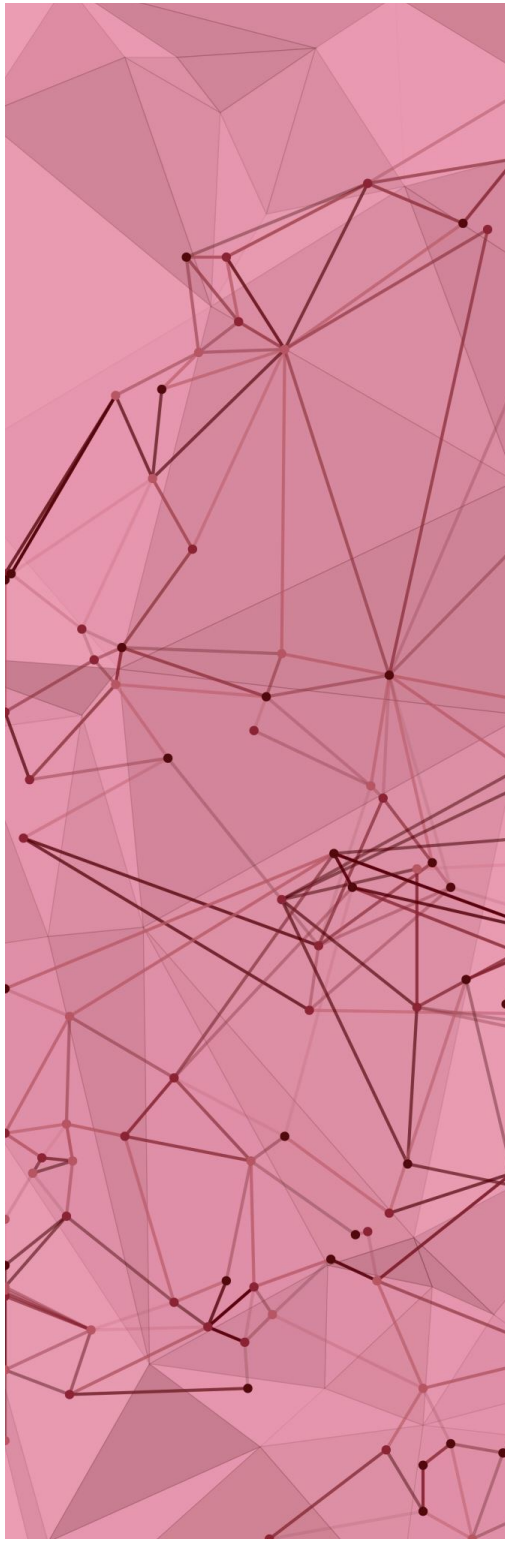
```
require 'Singleton'

class SingletonClass
    include Singleton #include module
    # ...
end

a = SingletonClass.instance
b = SingletonClass.instance

a == b #=> true

c = SingletonClass.new    #=> NoMethodError
                          # new is private
```



load vs. include vs. require

- load 'open-uri.rb'
  - Must mention .rb
  - Can load the same library files multiple times
- require 'open-uri'
  - No .rb
  - Loads a library only once– prevents multiple loading
- include
  - Used for including modules within a class
  - Like copying and pasting code (not file)—within a class or module





Ruby Gems

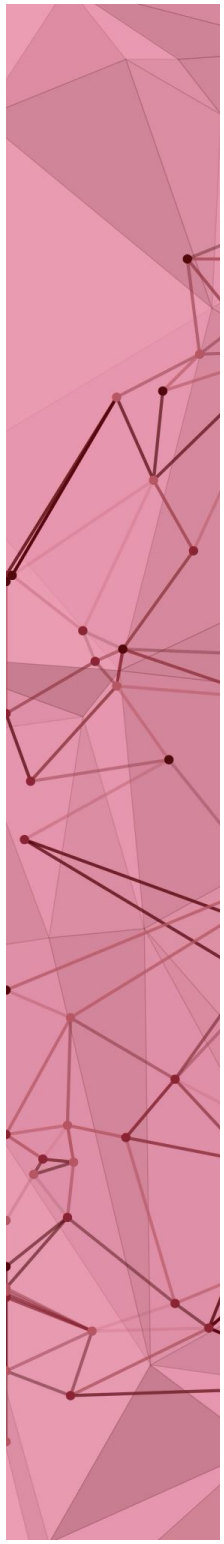
SQLite3 Database

Rails

X (Twitter)

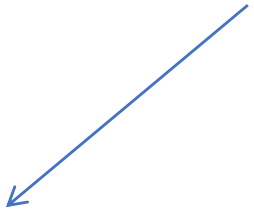
Installing new gems

- Command
  - gem install x
  - gem install mail
- Other gems we will need later:
  - gem install sqlite3
  - gem install rails
- Useful commands:
  gem uninstall ..., gem list ..., etc.
  - http://guides.rubygems.org/command-reference/

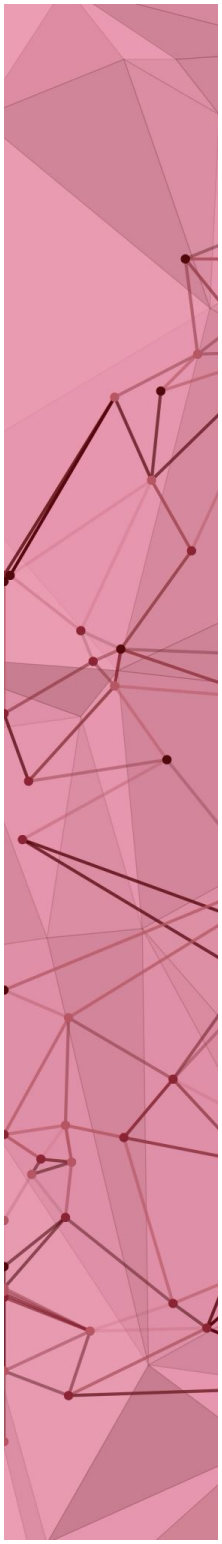


Creating Sqlite3 database using Ruby

Creates database Salary.db



```ruby
require 'sqlite3'

myDB = SQLite3::Database.new("Salary.db")
myDB.execute("create table if not exists SalaryTable
              (id integer primary key, name text, salary integer);")

myDB.execute("insert into SalaryTable (name, salary) values ('Alice', 5000);")
myDB.execute("insert into SalaryTable (name, salary) values ('David', 1000);")

rows = myDB.execute("select * from SalaryTable;")
puts rows
```



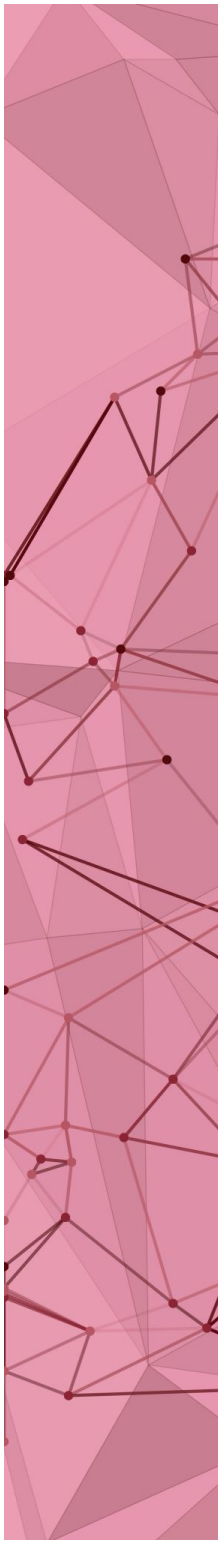
Accessing Salary.db from terminal (SQLite3 shell)

Assumption: Ruby folder contains salary.db

Blank file will be created if it doesn't exist

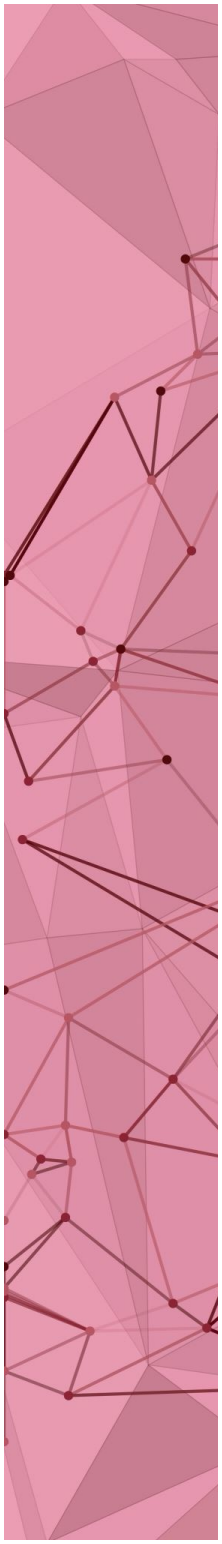
```
(base) mirfan@JG4P06CTV7 Ruby % sqlite3 salary.db
SQLite version 3.41.2 2023-03-22 11:56:21
Enter ".help" for usage hints.
sqlite> .tables
SalaryTable
sqlite> .schema
CREATE TABLE SalaryTable
                (id integer primary key, name text, salary
integer);
sqlite> select * from SalaryTable;
1|Alice|5000
2|David|1000
sqlite> .quit
```

SQL commands in blue



Some useful SQL commands

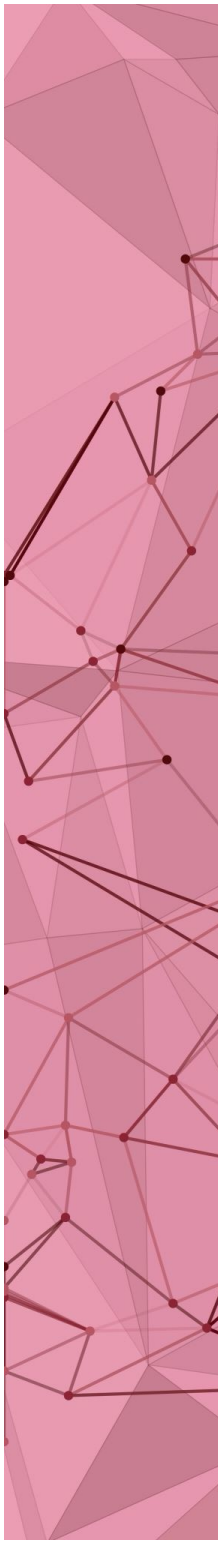
- delete from SalaryTable where name="Alice";
  - Deletes only those rows where the name is Alice
  - You can specify complex conditions using and/or
- delete from SalaryTable;
  - Deletes all rows from that table
- drop table SalaryTable;
  - Deletes the SalaryTable itself
  - Both the schema and the contents are gone
- Best tutorial
  - http://www.w3schools.com/sql/default.asp



Working with the X gem

Documentation with examples

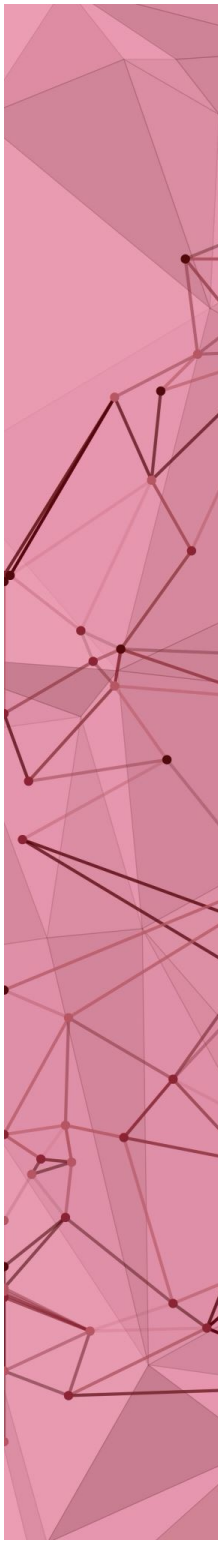
https://sferik.github.io/x-ruby/



Typical workflow

First, sign up for an X developer account

https://developer.x.com



Documentation

```ruby
require "x"

x_credentials = {
  api_key:              "INSERT YOUR X API KEY HERE",
  api_key_secret:       "INSERT YOUR X API KEY SECRET HERE",
  access_token:         "INSERT YOUR X ACCESS TOKEN HERE",
  access_token_secret: "INSERT YOUR X ACCESS TOKEN SECRET HERE",
}

# Initialize an X API client with your OAuth credentials
x_client = X::Client.new(**x_credentials)
```

```
# Get data about yourself
x_client.get("users/me")
# {"data"=>{"id"=>"7505382", "name"=>"Erik Berlin",
"username"=>"sferik"}}

# Post
post = x_client.post("tweets", '{"text":"Hello, World! (from
@gem)"}')
#{"data"=>{"edit_history_tweet_ids"=>["1234567890123456789"],
"id"=>"1234567890123456789", "text"=>"Hello, World! (from
@gem)"}}

# Delete the post
x_client.delete("tweets/#{post["data"]["id"]}")
# {"data"=>{"deleted"=>true}}
```

Coming up next

- Ruby on Rails
  - Web programming platform built using Ruby
  - Model
    - DB and constraints on data
  - View
    - Generates what users see
  - Controller
    - Takes user input
    - Consults with model
    - Directs the view