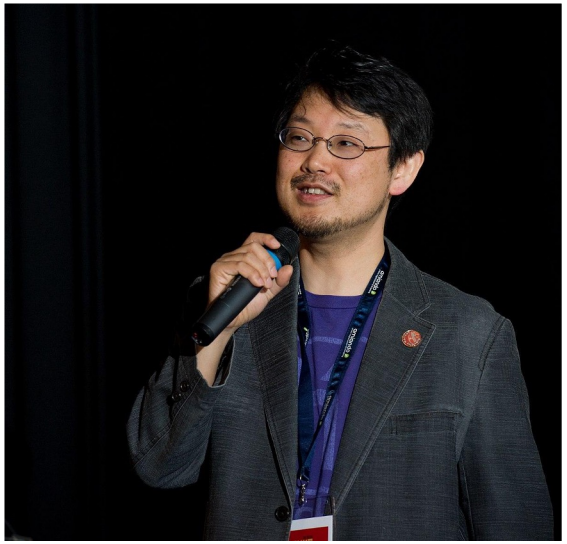
Ruby

By Colton, Kha, and Jonah

Background

- Developed by Yukihiro Matsumoto “Matz” in 1993
- Publicly released in 1995
- Gained popularity in 2006
- Resembles Perl and Python
- Many active communities, but somewhat in decline
  - Ruby-Talk
  - Discord and Reddit communities

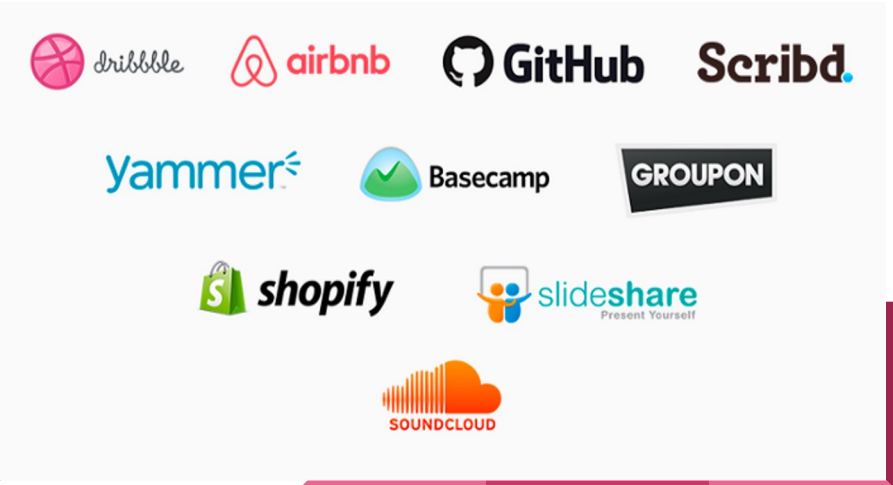


Design Philosophy

- Matsumoto designed the language for productivity and fun
  - “Often people, especially computer engineers, focus on the machines.”
  - Wanted Ruby to be focused on the person
  - Combines Scripting with Object Oriented languages
- Created a Language he wanted to code in
  - Matsumoto developed in Python, but disliked its inability to code like OOP
  - Minimizing work and confusion, making the human efficient even at the cost of the machine
- Easy to learn, Easy to use

Functionalities

- Build web applications:
  - Ruby: running a few commands, and you get a working application with a predefined directory structure, database configurations.
  - Python: Setting up a new Django project also requires running commands, but you may need to configure settings and directory structures manually or through boilerplate code
- Framework: Ruby on Rails.
- Library: RubyGems



Ruby on Rails

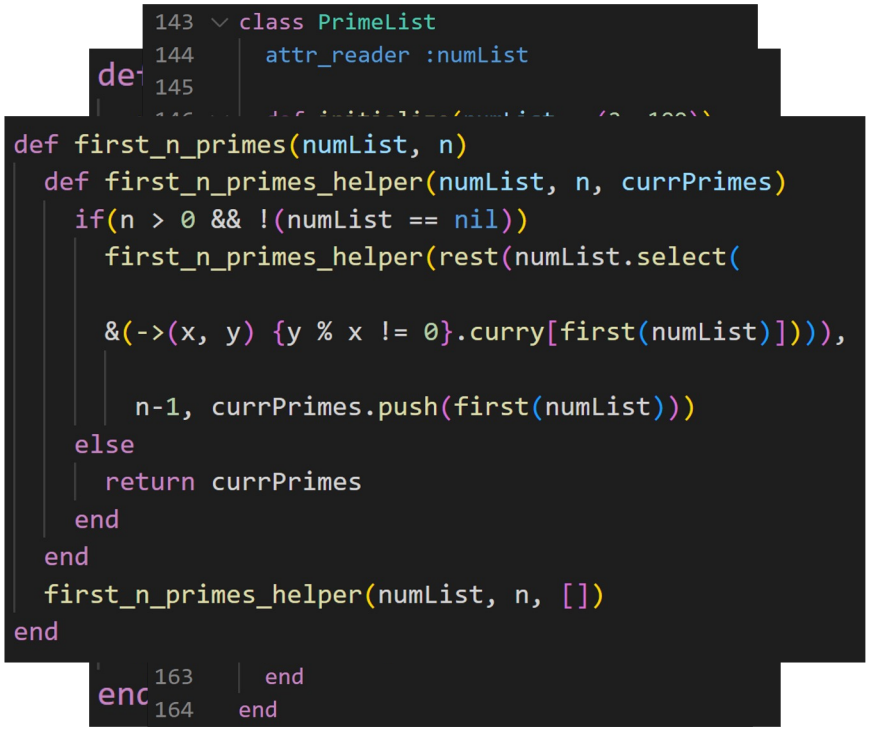
- Framework built on top of Ruby: simplicity and rapid development in web development. (Limited)
  - For those who prefers expressive and human-friendly syntax.
  - Convention over Configuration.
- Python is more extensive: web development, data analysis, AI.
  - Faster in execution time.
  - Framework: Django

RubyGems

- Gems:
  - Host repository of gems (libraries or packages).
- Dependency Management
  - Gemfile and Gemfile.lock.
- Version and Update
  - Semantic versioning, gem update

What makes Ruby Unique?

- Everything is an object
  - Everything has methods
- Supports multiple programming paradigms
  - Procedural
  - Object Oriented
  - Functional
- Inspired by simple LISP languages, with an Object system and blocks like higher order functions

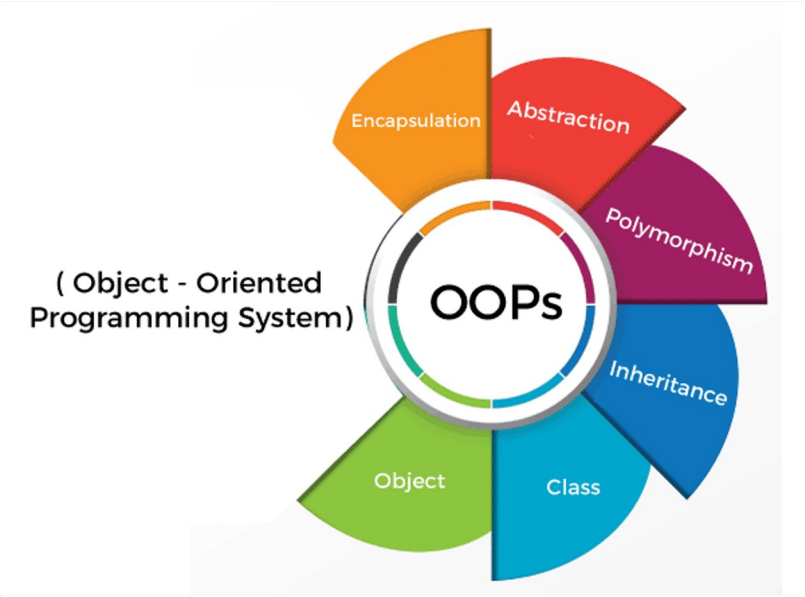


What does this mean?

- Primitives in most languages are objects in Ruby
  - integers, booleans, nulls
- Multiple paradigms supports multiple language backgrounds
  - Can code without classes (python)
  - Fully object oriented (Java)
  - Use anonymous functions (Clojure)

Features

- Dynamically Bound
- Object-oriented
- Classes with Inheritance
- Mixins
- Iterators
- Exception Handling
- Garbage Collection
  - Mark and sweep



Variables, Data types (dynamic vs static)

```
84    # Java
85    int x = 10;
86    String message = "hello";
87    System.out.println(message);
88
89    # Ruby
90    x = 10
91    message = "hello"
92    puts message
```

```
94    # Python
95    x = 10
96    message = "hello"
97    print(message)
98
```

OOP

```ruby
# Ruby
class Person
    attr_reader :name

    def initialize(name)
      @name = name
    end
  end

  person = Person.new("John")
  puts person.name
```

```python
# Python
class Person:
    def __init__(self, name):
        self.name = name

person = Person("John")
print(person.name)
```

```java
// Java
public class Person {
    private String name;

    public Person(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public static void main(String[] args) {
        Person person = new Person("John");
        System.out.println(person.getName());
    }
}
```

Class Inheritance

```ruby
# Ruby
class Person
  include PrintSelf
  attr_reader :name

  def initialize(name = "John")
    @name = name
  end
end
```

```ruby
class Student < Person
  attr_reader :school
  def initialize(name = "John", school="Creighton")
    super(name)
    @school = school
  end
end
```

Modules and Mixins

```ruby
module PrintSelf
  def selfString
    return "My name is #{self.name}, I am a #{self.class.name.downcase}."
  end
end
```

```ruby
people = []
people.push(Person.new())
team = ["Colton", "Kha", "Jonah"]
team.each {|name|
  people.push(Student.new(name))
}
people.each {|person|
  puts person.selfString
}
```

```
My name is John, I am a person.
My name is Colton, I am a student.
My name is Kha, I am a student.
My name is Jonah, I am a student.
```

Iterators

```python
#Python
class MyIterator:
    def __init__(self, data):
        self.data = data
        self.index = 0

    def __iter__(self):
        return self

    def __next__(self):
        if self.index < len(self.data):
            result = self.data[self.index]
            self.index += 1
            return result
        else:
            raise StopIteration

my_iterator = MyIterator("hello")

for i in my_iterator:
    print(i)
```

```ruby
# Ruby
array = [1, 2, 3, 4, 5]

array.each do |i|
  puts i
end
```

Exception Handling

```ruby
# Ruby
begin

  result = 10 / 0
rescue ZeroDivisionError

  puts "Cannot divide by zero."
end
```

```java
// Java
public class ExceptionHandlingExample {
    Run | Debug
    public static void main(String[] args) {
        try {
            int[] myNumbers = {1, 2, 3};
            System.out.println(myNumbers[10]);
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("An error occurred: " + e.getMessage());
        }
        System.out.println(x:"This will still be executed.");
    }
}
```

```python
# Python
try:
    result = 10 / 0
except ZeroDivisionError:
    print("Cannot divide by zero.")

print("This will still be printed.")
```

Summary

- Ruby’s popularity was largely due to its novelty at the time of its release
- Modern developments in JavaScript and Python caused a paradigm shift

References

- https://www.ruby-lang.org/en/about/
- https://www.techotopia.com/index.php/Ruby_Essentials
- https://www.manning.com/books/the-well-grounded-rubyist
- https://www.rubyguides.com
- https://ruby-doc.org
- https://www.infoworld.com/article/3687219/whatever-happened-to-ruby.html