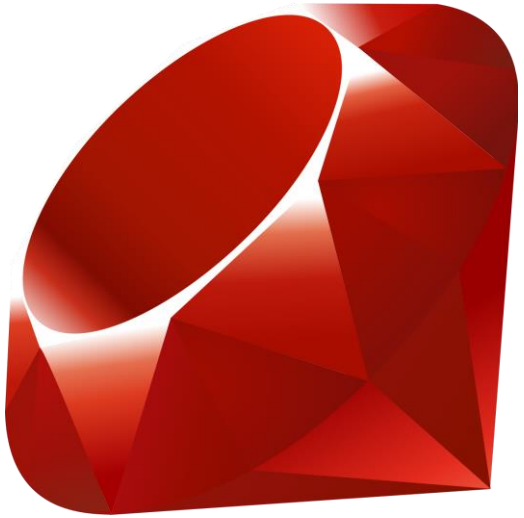
Introduction to Ruby and its programming environment

León Jaramillo

softserve

What is Ruby?

- **Ruby** is a programming language which is:
  - General purpose
  - High level
  - Interpreted (JIT compilation)
  - Multiparadigm
  - Dynamically typed



- It was created by **Yukihiro “Matz” Matsumoto** in Japan, being released in 1995.

- Ruby aims at **beautiful** and **artful** code.

More about Ruby

- Ruby was **influenced** by other languages such as Perl, Smalltalk, Eiffel, Ada and Lisp.
- It gained widespread **popularity** due to the Ruby on Rails web framework and its strength for implementing *Domain Specific Languages* (DSL).
- Ruby is **free** to use, copy, modify and distribute.
- In Ruby, everything is an **object**.

How to get Ruby?

- On Windows, just use the installer that you can find here: https://rubyinstaller.org/ (initially, with DevKit)

- On Linux/UNIX, just use the package manager which is available:
  - On Ubuntu and Debian-based distros: `apt install ruby-full`
  - Using yum: `yum install ruby`

- On Macos, you can use Homebrew, installing it first, and using the following command: brew `install ruby`

- Or you can try it without installing it, here: https://try.ruby-lang.org/

What IDE or code editor should I use?

- JetBrains offers **RubyMine** as the IDE of choice (https://www.jetbrains.com/ruby/). However, it is paid. You can check if your university offers educational licenses.

- **Visual Studio Code** features different extensions which helps on Ruby programming.
  - There is **Ruby by Shopify**, with not much configuration needed (but opinionated).
  - There is, also, Ruby by **Peng Lv**. Which could need further configuration after installing.

Hello, World!

- Let's create a typical "Hello, World!" program. For this, we'll just use the `puts` instruction:

```
puts "Hello, World!"
```

- Above statement adds a new line at the end of the printed text. To prevent this, use the `print` instruction:

```
print "Hello, World!"
```

- We can use **escape characters** in Ruby.

Comments in Ruby

- Comments are used (and important) in every language out there. These won't be executed (or compiled) and are used, mainly, informational purposes.

- You can write one-line comments:

```
#I just need one line to write this comment
```

- And also, multi-line comments:

```
= begin
Instead, I need more than one line
to write this comment
= end
```

Getting input from the user

- You can get user input using the `gets` instruction:

```
print "Ingrese su nombre: "
name = gets
print "Hola, " + name + "!"
```

- This instruction will include the new line from our input in its results, so you might use `chomp` method to remove it:

```
name = name.chomp
```

Getting numbers from user input

- In the following example, we will get two integer numbers and multiply them. We will need to chomp the input and to convert it:

```
print "Ingrese un número: "
num1 = gets.chomp.to_i
print "Ingrese otro número: "
num2 = gets.chomp.to_i
res = num1 * num2
print "El resultado es #{res}"
```

Converting to other data types

- Ruby does not support implicit data type conversions in its variables. So, we need to perform it explicitly.

- We can convert strings to integer numbers

```
variable.to_i
```

- We can convert strings to float numbers

```
variable.to_f
```

- We can convert variables of any type to strings

```
variable.to_s
```

Ruby Data Types

- Numbers
  - Integer
  - Float
- Boolean
- Strings
- Hashes
- Arrays
- Symbols

Understanding variables

- We can declare variables just assigning them a value:

```
my_variable = 45
```

- We declare constants capitalizing variable names:

```
MYCONSTANT = 3.1416
```

- Changing a variable's name, might change its type too.

- We can figure out the type of a variable in two ways:
  - `my_var.class`
  - `y.kind_of? Integer`

Homework

- What is the difference between `chomp` and `chop`?
- Which are the different variables' scopes and what are them used for?
- Write different programs that, based on user input, calculate the area of:
  - A circle
  - A triangle
  - A square
  - A rectangle
  - A trapezium
- A string is an object from the `String` class. What are the methods `reverse`, `capitalize`, and `length` for?

Useful Resources

- What is Ruby?: https://www.ruby-lang.org/en/about/
- Installing Ruby's programming environment: https://www.ruby-lang.org/en/downloads/
- Trying Ruby in the browser: https://try.ruby-lang.org/
- A first contact with Ruby: https://www.geeksforgeeks.org/ruby-for-beginners/


COLOMBIA
DC
Thanks! Any question?
softserve