Retorno

¡Comencemos!

¿Qué aprenderás?

- Entender la importancia del retorno de un método.
- Crear métodos con retorno.
- Saber que los retornos en Ruby son, por defecto, implícitos

Introducción

En muchas situaciones necesitaremos utilizar el resultado de un método para seguir trabajando con él.

En escasas ocasiones los métodos muestran información directa en pantalla, normalmente retorna el valor para que puedas seguir trabajando.

También será muy frecuente que, en nuestro trabajo, nos soliciten un método que reciba cierta información, la procese y devuelva algo.

¡Vamos con todo!

Creando nuestro primer método con retorno

```
def transformar_a_fahrenheit(f)
 celsius = (f + 40) / 1.8 - 40
 return celsius
end
```

De esta forma si queremos mostrar el resultado simplemente escribiremos:

```
puts transformar_a_fahrenheit(123)
```

Al separar el imprimir del método lo hacemos mucho más flexible.

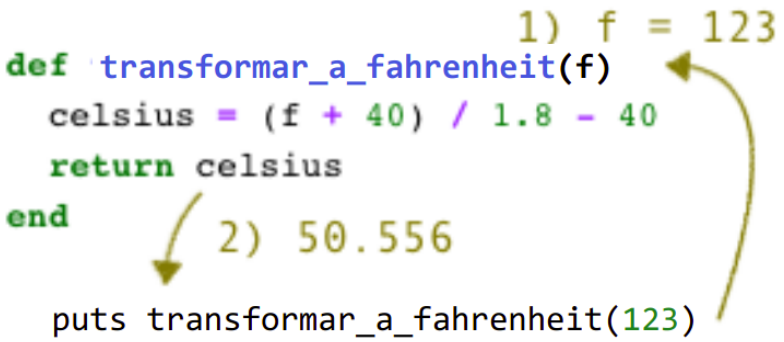

Imagen 1. Método `transformar_a_fahrenheit()`.
Fuente: Desafío Latam.

El retorno

Los métodos pueden recibir parámetros y pueden devolver un valor. A este valor se le conoce como retorno. Todos los métodos tienen un retorno, en algunos casos es implícito, y en otros, explícito.

Retorno explícito

```
def fahrenheit(f)
 celsius = (f + 40) / 1.8 - 40
 return celsius
end
```

Retorno implícito

```ruby
def fahrenheit(f)
 celsius = (f + 40) / 1.8 - 40
end
```

El retorno implícito es siempre sobre la última línea

Return sale del método

Todo lo que viene después de la instrucción return es ignorado.

```ruby
def prueba
 x = 'mensaje que no se muestra'
 return # Punto de salida
 puts x
end

prueba # => nil
```

Recordemos que `nil` es la ausencia de un valor.

¿Implícito o explícito?

La instrucción `return` no se usa frecuentemente en Ruby. En la mayoría de los casos basta con un retorno implícito.

Cuidado con los puts y prints

En los métodos, el retorno es implícito, y esto quiere decir que el return se hace sobre la última línea.

```ruby
def fahrenheit(f)
 celsius = (f + 40) / 1.8 - 40
 puts celcius # puts celcius devuelve nil
end

# fahrenheit(45) #=> nil
```

La instrucción `puts` siempre retorna `nil`. Es por esto que debemos tener cuidado con utilizar la instrucción `puts` en la última línea de un método.

Resumen

- Retorno, devolver, salida son sinónimos para el valor que devuelve un método.
- Podemos hacer que un método devuelva un valor, de manera explícita, utilizando la instrucción `return`.
- La última línea de un método tiene un `return` implícito.
- Retornar **NO ES LO MISMO** que mostrar en pantalla.
- `puts` devuelve `nil`, por lo que agregar un `puts` en la última línea de un método hace que este devuelva `nil`.
- nil es un objeto que representa la ausencia de un valor.